UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Facultad 6

TÍTULO: Propuesta de métricas para evaluar software orientado al procesamiento y gestión de videos e imágenes digitales en el Departamento de Señales Digitales del centro GEySED.

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

AUTOR: **Ormanis de la Cruz Díaz**

TUTOR: **Ing. Anay Iyenis Chapman Hernández.**

La Habana 28 de junio 2011
"Año 53 de la Revolución"

PENSAMIENTO

"Seamos realistas y hagamos lo imposible."

CHE

DEDICATORIA

Dedico este trabajo de diploma a toda mi familia, a mi querida madre María Eladia Díaz, a mi papá Orlys de la Cruz, a mi hermana Lisbet de la Cruz Díaz, a mi tía Elsy de la Cruz, a mis primos Adrian Guerra de la Cruz y Alexander Guerra de la Cruz.

A mis seres queridos que hoy no se encuentran presente físicamente pero que siempre han estado en mi corazón, en mi pensamiento y que me han servido de ejemplo y guía durante toda mi vida, desde el lugar donde estén siempre me han estado mirando e indicando el camino correcto y se que hoy están muy orgullosos de mi, Redomil de la Cruz mi tío, a mi abuelo Juan Bautista Boss de la Cruz, a mi abuela Angélica Guilarte.

Y a todos mis amigos que de una forma u otra siempre me han apoyado, a ellos sin excepción de ninguno, les dedico también este trabajo.

De todo corazón a todas las personas que de una forma u otra han influido en mi vida durante 24 años.

AGRADECIMIENTOS

Agradecer a toda mi familia, en especial a mi madre por estar a mi lado durante todo este tiempo y apoyarme en todo momento, incluso en los más difíciles, a mi papá, a mi querida hermana, a la tía que más quiero en el mundo, a mis primos que aunque están lejos de aquí para mí siempre han estado presentes, a mi otra familia aquí en La Habana que de no ser por ellos quien sabe que hubiera sido de mí, a Juan Carlos, Karina, Rubén, mi primita Natalí, mi prima Karelia, que a pesar de estar lejos también nunca se ha olvidado de mí.

A todos mis amigos y amigas de la Universidad y fuera de esta porque han estado cerca de mí y me han servido algunos de experiencia, otros de consuelo y la mayoría, casi todos, han estado a mi lado en todas las batallas libradas dentro y fuera de la Universidad. Sería egoísta de mi parte mencionarlos a todos porque son muchos pero trataré de mencionar algunos para evitar discordias y celos, los que no están aquí no son menos queridos: Yolexy, Harol, Oscarito, Agner, Pedro, los jimaguas Frank y Fredy, Enrique, Franklin, Rosel, el Chino, Alex Izaguirre, a mis pollos, Chuchi, Oscarito, Alberto, Rancés, Jorgito, a Yasser, a otros que por no ser los primeros en mencionar no se quieren menos, Keisey, Sujail, el Moro.

A mis queridos profesores Anay, Yuneiry, Karolai, Dayani, Igor, Burgos, Osmel, Sandy, Frank Alain.

No puedo dejar de agradecer a las niñas del apartamento 101 106 por soportarme durante todo un año Listley, Arianna, Alexaidy, Noly, Jessica.

A mis queridas amigas Liudmila, Arianna, Karelia, Yailín, Margelis, Yusleimy.

A una persona especial que estuvo conmigo durante tres años, por aguantarme durante esos tres años y ser una escuela para mí: Tahimí Ramírez.

A la persona que más quiero en estos momentos y quisiera que nunca me faltara en mi vida, por ser paciente, comprenderme, perdonarme y estar conmigo en todo momento durante todo este año y por darme el amor, el cariño y el afecto que día a día me entrega: Yeinery Rebollo Vera.

A la Revolución cubana que me ha dado la posibilidad de estudiar en esta magnífica universidad.

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este Trabajo de Diploma y autorizo a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

. _____ _____

Autor: Ormanis de la Cruz Díaz

Tutor: Ing. Anay Iyenis Chapman Hernández

RESUMEN

Durante el proceso de desarrollo de software es importante el uso de métricas para la evaluación del producto en cada etapa, para lograr depurarlo desde sus inicios, y que el mismo llegue a las manos del cliente cumpliendo todos los requerimientos y funcionalidades, quedando lo más libre posible de defectos. El presente trabajo, titulado “Propuesta de métricas para evaluar software orientado al procesamiento y gestión de videos e imágenes digitales en el Departamento de Señales Digitales del centro GEySED” se realizó con el objetivo de proponer métricas para la evaluación de software orientados al procesamiento y gestión de videos y señales digitales para el Departamento de Señales Digitales del centro GEySED. Para dar cumplimiento a este objetivo se hizo indispensable llevar a cabo todo un proceso de investigación, con el fin de detectar métricas aplicables a todos los proyectos del departamento. Para ello fue necesario realizar una fundamentación teórica en la que se abordan los diferentes conceptos emitidos por varios autores referentes a aspectos relacionados con la calidad de software, las normas de calidad existentes y las métricas. Además fue indispensable realizar una entrevista que permitió conocer que los miembros de los proyectos del departamento no tienen conocimiento sobre las métricas, por tanto estas no se aplican en los proyectos; además facilitó identificar las funcionalidades comunes de los proyectos que se realizan en el departamento. Teniendo en cuenta el resultado antes obtenido se escogieron las métricas para evaluar los proyectos del departamento Señales Digitales del centro GEySED y los parámetros que fueron objeto de evaluación en los mismos. Una vez aplicadas las métricas propuestas para la evaluación del software a 2 versiones del proyecto PRIMICIA del departamento, se recopilan los principales resultados obtenidos en la misma, permitiendo hacer una comparación entre ellos.

Palabras claves: **métricas, modelos de calidad, procesamiento de imágenes, procesamiento de video.**

Índice

Índice de Tablas

Índice de Figuras

INTRODUCCIÓN

El siglo XX fue el escenario propicio y la Segunda Guerra Mundial el suceso más trascendental para el inicio de importantes estudios y grandes descubrimientos científicos y tecnológicos que favorecieron el surgimiento y desarrollo de la computación, tecnología desconocida hasta ese momento y que posteriormente se convertiría en uno de los recursos más importantes para el desarrollo económico y social a nivel mundial. Muchos han sido los retos, desafíos y obstáculos encontrados a lo largo del camino, sin embargo ha sido asombrosa la velocidad con que se han desarrollado las Tecnologías de la Información y las Comunicaciones (TIC). Una de las aristas de las TIC es la informática, que no es más que el procesamiento automático de grandes cantidades de datos, en otras palabras, la automatización de procesos para la gestión y el control de la información. En la actualidad esta rama se ha convertido en una poderosa herramienta, y como consecuencia, su acelerado desarrollo ha provocado una revolución tecnológica a gran escala.

Desde sus inicios ha sido posible apreciar en distintos campos el avance que representa hacer uso de un software que permita automatizar procesos. Hoy día existen industrias que se dedican netamente a la producción de software, y como parte de la rapidez con la que esta se ha ido desarrollando, surgen diversos estilos para distintas aplicaciones lo que ha traído consigo que la calidad del producto final varíe entre un producto y otro. En este sentido se hace necesario establecer un estándar que permita a los consumidores de software decidir si el producto es de calidad y cumple con ciertos requisitos de funcionalidad.

“Para obtener un software de calidad es preciso medir el proceso de desarrollo, cuantificar lo que se ha hecho y lo que falta por hacer, estimar el tamaño del programa, costos, tiempo de desarrollo y otros parámetros. La medición de un producto se realiza mediante las métricas, para caracterizar numéricamente los distintos aspectos del desarrollo del software. Las métricas de software tienen un papel decisivo en la obtención de un producto de alta calidad, porque determinan mediante estadísticas basadas en la experiencia, el avance del software y el cumplimiento de parámetros requeridos. Siempre existirán elementos cualitativos para la creación de software. El problema estriba en que la valoración cualitativa puede no ser suficiente. Un ingeniero del software necesita criterios objetivos para guiarse en el diseño de datos, de la arquitectura, de las interfaces y de los componentes. El verificador necesita una referencia cuantitativa que le ayude en la selección de los casos de prueba y de sus objetivos.

Las métricas técnicas facilitan una base para que el análisis, diseño, codificación y prueba puedan ser conducidos más objetivamente y valorados más cuantitativamente". (Pressman, 2005)

Cuba no está exenta del avance tecnológico actual, y consciente de la facilidad que proporcionan las TIC para el desarrollo económico y social de la nación, dedica gran parte de sus esfuerzos en esta dirección. Fomentar y promover el desarrollo de la informática, se ha convertido en una tarea de prioridad para el país, con el propósito de insertarse en el mercado del software mundial. La Industria Cubana del Software está llamada a convertirse en una significativa fuente de ingresos nacionales.

La Universidad de las Ciencias Informáticas (UCI) surge en el 2002 con la misión de formar profesionales en el campo de la informática a partir de un modelo pedagógico, que vincula el estudio con la producción y la investigación, de acuerdo a las necesidades sociales del país. Para dar cumplimiento a este objetivo la Universidad cuenta con una serie de centros, en los que se desarrollan diferentes proyectos de Investigación, Desarrollo e Innovación (I+D+i); los cuales tienen que lograr el nivel de calidad que el cliente necesita. Es por ello que por cada software que se produce es necesario realizar una exhaustiva evaluación, para garantizar que estos cumplan con las expectativas deseadas por el cliente.

El Departamento de Señales Digitales se encuentra en el GEySED, en el mismo se llevan a cabo determinadas aplicaciones destinadas al procesamiento y gestión de videos y señales digitales. El desarrollo de este tipo de software, implica gran responsabilidad por parte de cada uno de los miembros de los proyectos, con el fin de que los productos queden con la calidad que el cliente espera. Vale la pena señalar que los productos que se elaboran en este departamento tienen características diferentes a otros productos que se desarrollan en el centro, incluso a otros proyectos de Investigación, Desarrollo e Innovación (I+D+i) de la Universidad.

Estas aplicaciones están destinadas al trabajo con medias (audio, imágenes y video) y se ejecutan diferentes procesos en las mismas como: la catalogación, procesamiento y gestión de medias así como la trasmisión de señales digitales. Con el empleo de efectivos métodos de evaluación del software, en cada etapa del proceso desarrollo, se garantiza un gran por ciento de la calidad de los productos que se realizan en el departamento. Según estudios realizados, se ha llegado a la conclusión que en la mayoría de los proyectos desarrollados no se aplica de

forma correcta ningún mecanismo de evaluación del proyecto, de los procesos definidos en el mismo y del producto a desarrollar.

Por tal motivo se ha identificado que el **problema a resolver** consiste en: la inexistencia de mecanismos para la evaluación de software orientados al procesamiento y gestión de videos e imágenes digitales en el Departamento de Señales Digitales del centro GEySED.
Por tanto el **objeto de estudio** de la investigación es: el proceso de evaluación de la calidad del software, enmarcando el **campo de acción**: proceso de evaluación de la calidad del software orientado al procesamiento y gestión de videos e imágenes en el Departamento de Señales Digitales del centro GEySED, basado en lo anteriormente mencionado el **objetivo general** de este trabajo es: proponer métricas para la evaluación de software orientados al procesamiento y gestión de videos y señales digitales para el Departamento de Señales Digitales del centro GEySED.

Teniendo en cuenta lo antes mencionado se tiene como **Idea a defender:** con la selección adecuada de las métricas que permitan evaluar un software destinado al procesamiento y gestión de videos y señales digitales en el departamento Señales Digitales del centro GEySED, se logrará mejorar considerablemente la calidad de los productos desarrollados en el mismo.

Para el cumplimiento del objetivo especificado se trazaron las siguientes tareas de investigación:

1. Sintetizar conocimientos relacionados con la calidad y evaluación de software.
2. Clasificar métricas existentes para la evaluación de software.
3. Describir modelos de calidad que incluyan métricas para evaluar diferentes atributos de calidad del producto.
4. Identificar líneas comunes de los proyectos de Investigación Desarrollo e Innovación (I+D+i) que existen en el departamento Señales Digitales del centro GEySED.
5. Seleccionar parámetros que serán objeto de evaluación en los proyectos del departamento Señales Digitales del centro GEySED.
6. Determinar métricas para evaluar los proyectos del departamento Señales Digitales del centro GEySED.
7. Aplicar las métricas propuestas para la evaluación del software de procesamiento y gestión de videos y señales digitales al departamento Señales Digitales del centro GEySED.

Una vez concluida la investigación se espera obtener el siguiente resultado:

- ✓ Métricas para evaluar software orientado al procesamiento y gestión de videos e imágenes digitales en el departamento Señales Digitales del centro GEySED.

Con el propósito de darle solución al objetivo trazado durante la investigación, se usaron varios métodos científicos:

- ✓ **Métodos teóricos:** con la utilización de estos es posible estudiar las características del flujo de trabajo para la evaluación de un software que no son observadas directamente. A continuación se detallan los que se utilizaron en la presente investigación.

- ✓ **Método histórico - lógico**: para investigar la información que se tiene hasta el momento del estado del arte de las métricas utilizadas en la realización de los proyectos informáticos, para determinar los principales conceptos relacionados con la calidad de software y para identificar rasgos comunes de los proyectos desarrollados en el Departamento de Señales Digitales del centro GEySED.

- ✓ **Método analítico sintético:** este método se utilizó con el objetivo de realizar un estudio detallado de las métricas propuestas para la evaluación del software de procesamiento y gestión de videos y señales digitales y para caracterizar los proyectos del departamento.

- ✓ **Métodos empíricos:** estos métodos son los que permiten hacer una retroalimentación de trabajos realizados con anterioridad sobre un tema en específico, en este caso para las métricas propuestas para la evaluación del software de procesamiento y gestión de videos y señales digitales y mediante estos se pueden analizar diferentes variantes. De ellos se usaron los que siguientes.
- ✓ **Método de Observación:** se utilizó en toda la investigación, pues se recoge información de los aspectos tratados en la tesis desde el punto de vista de otros autores así como sus definiciones y resultados para permitir realizar un análisis del estado del arte en ese campo.

- ✓ **Entrevista:** se utilizó para entrevistar a los líderes de proyectos del Departamento de Señales Digitales del centro GEySED con el fin de sintetizar las principales funcionalidades de los mismos. En este caso se seleccionó como población a 7 proyectos del Departamento de Señales Digitales. De ellos como muestra se han elegido 4 líderes de proyecto para entrevistar lo que representa el 57% de la población.

El contenido de este trabajo se encuentra estructurado en tres capítulos, los que se definen de la siguiente manera:

Capítulo 1. En este capítulo se definieron algunos conceptos generales importantes para la investigación como: calidad de software, procesamiento y gestión de imágenes y videos digitales, mediciones y métricas. Además se realizó un análisis de todos los aspectos fundamentales que están relacionados con las métricas de software, así como su clasificación. Se describieron los modelos de calidad que incluyen métricas para evaluar diferentes atributos de calidad del producto.

Capítulo 2. En este capítulo se identificaron las líneas comunes de los proyectos existentes en el departamento y se seleccionaron los parámetros que serán objeto de evaluación en estos. Finalmente se escogieron las métricas para evaluar los proyectos del departamento Señales Digitales del centro GEySED.

Capítulo 3. En este capítulo se recopilan los principales resultados luego de haber aplicado las métricas propuestas para la evaluación del software de procesamiento y gestión de videos y señales digitales al departamento Señales Digitales del centro GEySED, realizándose la validación de la investigación.

CAPÍTULO 1

Fundamentación Teórica.

1.1. Introducción

Durante el proceso de desarrollo de software es importante el uso de métricas para la evaluación del producto en cada etapa, pues con esta se logra depurar un software desde sus inicios, para que el mismo llegue a las manos del cliente cumpliendo todos los requerimientos y funcionalidades, quedando lo más libre posible de defectos. Para asegurar que un proceso o los productos resultantes son de calidad o poder compararlos, es necesario asignar valores, descriptores, indicadores o algún otro mecanismo mediante el cual se pueda llevar a cabo dicha comparación, como parte de un proceso de retroalimentación, que permite que los errores encontrados se resuelvan de manera efectiva en iteraciones posteriores de forma tal que no afecten el tiempo de duración del proyecto y el costo del mismo. Este capítulo tiene como objetivo principal abordar los temas fundamentales referidos a las métricas para la evaluación del software, así como los conceptos y las definiciones expuestos por varios autores. Haciendo énfasis en algunos aspectos importantes relacionados con la calidad de software, tales como: los factores que miden la calidad del software y las métricas. Así como los modelos de calidad más difundidos a nivel mundial.

1.2 Calidad de Software. Conceptos y definiciones.

Actualmente, la calidad de software es una tarea en la que se empeñan muchos esfuerzos. En la informática cuando se habla de este tema lo primero que piensan las personas es en un producto que funcione bien, sin embargo la calidad va mucho más allá de esto y hay que tener en cuenta que para lograrla se deben realizar una serie de pasos durante todo el proceso de desarrollo del software, para que al finalizar se pueda decir que se cuenta con un producto de buena calidad. Es importante señalar que un software casi nunca es perfecto. Todo proyecto tiene como objetivo producir software de la mejor calidad posible, sin dejar de cumplir con las expectativas de los usuarios.

Aun cuando se habla de la calidad de software, para muchos es desconocido su verdadero significado. Pudiéndose decir que frecuentemente gran número de personas se refieren a este término, sin saber cuál es su magnitud. Antes de brindar un concepto formal de calidad

de software se expondrán algunos conceptos de que se entiende por calidad, de forma general.

Calidad: Según el Diccionario de la Real Académica Española es: "propiedad o conjunto de propiedades inherentes a un objeto que permiten apreciarlo como mejor, igual o peor que otros objetos de su especie".

En cuanto al usuario, la calidad implica satisfacer sus necesidades y deseos. Esto quiere decir que la calidad de un producto depende de la forma en que este responda a las preferencias del cliente. También puede decirse que la calidad significa aportar valor al cliente, consumidor o usuario.

Según William Edwards Deming[1] la calidad, significa desarrollar, diseñar, producir y mantener un producto que sea el más económico, el más útil y siempre satisfactorio para el consumidor. Es la aplicación de los principios y técnicas estadísticas en todas las fases de la producción, dirigida a la fabricación más económica de un producto (servicio) que es útil en grado máximo y que tiene mercado.

Se puede decir que existen varias definiciones de calidad, pero básicamente para que un producto o servicio tenga calidad, debe satisfacer o superar las expectativas esperadas por el cliente. La definición de calidad nunca será precisa, ya que se trata de la apreciación subjetiva de cada persona y siempre será entendido de diferentes maneras, ya que para unos la calidad residirá en un producto y para otros en el servicio de postventa del mismo.

La buena calidad de un producto o servicio se encuentra determinada por tres cuestiones básicas:

- Dimensión técnica (que abarca los aspectos científicos y tecnológicos que afectan al producto).
- Dimensión humana (cuida las buenas relaciones entre clientes y empresas).
- Dimensión económica (que busca minimizar los costos, tanto para la empresa como para el cliente).

[1] William Edwards Deming estadístico estadounidense, profesor universitario, autor de textos, consultor y difusor del concepto de calidad total.

Teniendo en cuenta esta definición muchos autores han creado conceptos sobre la Calidad de Software. De ellos se escogieron los siguientes:

La calidad del software es el conjunto de cualidades que lo caracterizan y que determinan su utilidad y existencia. La calidad es sinónimo de eficiencia, flexibilidad, corrección, confiabilidad, mantenibilidad, portabilidad, usabilidad, seguridad e integridad. (Carrasco, 1995)

Teniendo en cuenta el estándar 610-1900 de la IEEE, la calidad del software es el grado con el que un sistema, componente o proceso cumple los requerimientos especificados y las necesidades o expectativas del cliente o usuario.

Otra definición es: concordancia del software producido con los requerimientos explícitamente establecidos, con los estándares de desarrollo prefijados y con los requerimientos implícitos no establecidos formalmente, que desea el usuario. (Presman, 1998)

En cierta medida todos los conceptos expuestos hacen referencia a lo que es la calidad de software por lo que se puede definir de forma general que la calidad de software no es más que: el grado con el que un componente o un software cumplen con las necesidades del cliente, que cuenta con una serie de parámetros definidos y lo dotan de cualidades únicas, cumpliendo la totalidad de características de un producto de software, posibilitando satisfacer las necesidades explícitas o implícitas del usuario. (Mc-Graw-Hill, 2002)

Otras definiciones a tener en cuanta son: qué se entiende por una medida y qué es la medición de un software. A continuación se detallan estos aspectos:

Medida: proporciona una indicación cuantitativa de la cantidad, dimensiones o tamaño de algunos atributos de un producto.

Medición: acto de determinar una medida.

Luego de haber realizado el análisis de estos conceptos vale la pena señalar que para garantizar la calidad del software es necesario evaluar el producto en cada etapa, durante todo el proceso de desarrollo, para detectar y eliminar los errores encontrados antes de pasar a una nueva fase y de esta forma garantizar que el resultado final sea un producto con calidad. Para esto se hace necesario el uso de **métricas**, que no son más que medidas cuantitativas del grado en que un sistema, componente o proceso posee un atributo dado.

Teniendo en cuenta lo antes definido de forma general se puede decir que una **métrica de software** es cualquier medida o conjunto de medidas destinadas a conocer o estimar el tamaño u otra característica de un software o un sistema de información, generalmente para realizar comparativas o para la planificación de proyectos de desarrollo.

Es importante agregar que existen métricas aplicables a todo tipo de software, pero como resultado de la gran diversidad de aplicaciones que se desarrollan actualmente, existen métricas que solo se aplican a productos con determinadas características en el proceso de evaluación del software. Es por eso que a continuación se tienen en cuenta funciones o procesos asociados a las aplicaciones que se desarrollan en el Departamento de Señales Digitales del centro GEySED.

Procesamiento y gestión de imágenes.

El procesamiento de imágenes está dado por un conjunto de operaciones llevadas a cabo sobre la imagen a fin de realizar mediciones cuantitativas para poder describirlas; es decir extraer ciertas características que permitan mejorar, perfeccionar o detallar la imagen. (Asencio, 2002)

Algunas de las operaciones principales en el procesamiento de imágenes son:

Escalamiento: agranda o reduce el tamaño físico de la imagen cambiando el número de píxeles que contiene.

Codificación: buscar el código que represente la imagen original sin distorsión con el menor número de bits o simplemente cambiar el formato de la imagen.

1- Codificación sin pérdidas: la imagen reconstruida es pixel a pixel idéntica al original.
2- Codificación con pérdidas: la imagen reconstruida no es pixel a pixel idéntica a la original

Extracción de características: técnica para reducir la dimensión de los modelos o clases de los objetos, se realiza mediante el reconocimiento de objetos.

Reconocimiento de patrones: consiste en el reconocimiento de patrones de señales a partir de los procesos de segmentación, extracción de características y descripción dónde cada objeto queda representado por una colección de descriptores.

Transformación: es el tratamiento o edición de imágenes digitales, mediante herramientas que permiten manipular el contenido de una selección o de una capa como si fuesen objetos.

Compresión de datos: es la reducción del volumen de datos tratables para representar una determinada información empleando la menor cantidad de espacio posible

1- Compresión sin pérdida: los datos antes y después de comprimirlos son exactos en la.

2- Compresión con pérdida: puede eliminar datos para reducir aún más el tamaño, con lo que se suele reducir la calidad de la imagen.

Cuando se utiliza el término gestión de imágenes, se hace mención a algunas de las funciones que se muestran a continuación:

- Organizar y facilitar el acceso a imágenes mediante un sistema de categorización, indexación y búsqueda.
- Proporcionar sistemas de visualización: vistas en grupo, individuales, creación de presentaciones y vista de metadatos.
- Proporcionar opciones básicas de edición: color, dimensión, transformaciones, rotaciones, filtros y ajustes.
- Facilitar operaciones de mantenimiento: cambio y exportación de formatos, creación de miniaturas, creación de hojas de contacto y realización de cambios globales.

Procesamiento y gestión de video.

El procesamiento de video puede incluir un amplio rango de efectos, que van desde la eliminación de componentes de color hasta los complejos que incluyen introducir parte de un video dentro de otro video. Para el procesamiento de videos los receptores han de ser capaces de convertir el formato recibido en el formato del dispositivo de reproducción, a ser posible en tiempo real. (Contreras, 2005)

Un sistema de gestión de video puede admitir muchas características diferentes. Dentro de ellas se enumeran algunas de las más comunes:

- Visualización simultánea de video desde varias cámaras.
- Grabación de video y audio.

- Funciones de gestión de eventos con video inteligente, como detección de movimiento de video.
- Administración y gestión de cámaras.
- Opciones de búsqueda y reproducción.
- Control de acceso de usuarios y registro de actividades.

Unos de los principales problemas existentes es que muchas veces las personas tienden a confundir los términos medición y métricas de software. Es por esto que a continuación se hace una descripción detallada de cada una de estos aspectos.

1.3 Comparar medición del software con métricas de software.

Para entender mejor el concepto de métrica es necesario aclarar que los términos, métricas y medición no tienen el mismo significado y no se utilizan de igual manera aunque tienen como finalidad un mismo propósito. A partir de aquí en la tabla 1 se establece una comparación entre la medición del software y las métricas de software.

Tabla 1. Comparación entre medición del software y métricas de software

Medición del software	Métricas de software
La medición del software, implica las mediciones de las actividades relacionadas con el software siendo algunos de sus atributos típicos el esfuerzo, el coste y los defectos encontrados.	Las métricas de software no son más que la aplicación continua de mediciones basadas en técnicas para el proceso de desarrollo del software y sus productos.
Se puede analizar el trabajo desarrollado, conocer si se ha mejorado o no con respecto a proyectos anteriores.	Las métricas para la evaluación del software, se recopilan de todos los proyectos y durante un largo período de tiempo.
Posibilita la detección de áreas con problemas para poder remediarlos a tiempo y a realizar mejores estimaciones.	Se obtiene información relevante a tiempo, así con el empleo de estas técnicas mejorará el proceso y sus productos.
En el software lo que se mide son atributos propios del mismo, se descompone un atributo general en otros más simples de	Su intento es proporcionar indicadores que lleven a mejoras de los procesos de software a largo plazo. Un indicador es una métrica o una combinación de

medir, a veces se mide bien o mal ya que la descomposición del atributo genérico de calidad en otros sub-atributos se torna irreal, se mide con datos estadísticos no avalados.	métricas que proporcionan una visión profunda del proceso del software, del proyecto de software o del producto como tal.

1.4 Factores que miden la calidad del software.

Aspectos importantes del conocimiento acerca de la calidad del software, son los factores por los cuales se puede medir la calidad de un producto. Basándose en los estudios de McCall los factores que afectan a la calidad del software se centran en tres aspectos importantes de un producto software: sus características operativas, su capacidad de soportar los cambios (revisión del producto) y su adaptabilidad a nuevos entornos (o transición del producto). En la Tabla 2 se muestra los factores de calidad definidos por McCall. (Pressman, 1995)

Tabla 2. Factores de calidad de McCall

FACTORES DE CALIDAD DE McCall		
Aspecto que trata	Factor Calidad	Relacionado con …
Operación del Producto	Corrección	Grado en que un programa consigue los objetivos de la misión encomendada por el cliente.
	Fiabilidad	Probabilidad de operación libre de fallos de un programa de computadora en un entorno determinado durante un tiempo específico.
	Eficiencia	Cantidad de recursos de computadora y de código requeridos por un programa para llevar a cabo sus funciones.
	Integridad	Grado en que puede controlarse el acceso al software o a los datos, por personal no autorizado.
	Facilidad de uso	Esfuerzo requerido para aprender un programa, trabajar con él, preparar su entrada e interpretar su salida.
Revisión del producto	Facilidad de Mantenimiento	Esfuerzo requerido para localizar y arreglar un error en un programa.
	Flexibilidad	Esfuerzo requerido para modificar un programa operativo.

	Facilidad de prueba	Esfuerzo requerido para probar un programa de forma que se asegure que realiza la función requerida.
Transición del producto	Portabilidad	Esfuerzo requerido para transferir el programa desde un hardware y/o entorno de sistemas de software a otro
	Reusabilidad	Grado en que un programa (o partes de un programa) puede ser usado en otras aplicaciones.
	Facilidad de Interoperación	Esfuerzo requerido para acoplar un sistema a otro.

Estos factores de calidad de McCall, a pesar de haberse definido en el año 1977, conservan en gran medida su vigencia. Es difícil desarrollar medidas directas de los anteriores factores de calidad. Por tanto, se define un conjunto de métricas usadas para evaluar los factores de calidad. Teniendo en cuenta de qué factor se trate, se utilizarán determinadas métricas ponderadas para determinar el mismo. Las métricas que utiliza McCall para evaluar los distintos factores de calidad se muestran en la Tabla 3. [2]

Tabla 3. Métricas propuestas por McCall

Métricas propuestas por McCall	
Métrica	Relacionado con …
Facilidad de auditoría	La facilidad con que se puede comprobar la conformidad con los estándares.
Exactitud	La precisión de los cálculos y del control.
Normalización de las comunicaciones	El grado en que se usa el ancho de banda, los protocolos y las interfaces estándar.
Completitud	El grado en que se ha conseguido la total implantación de las funciones requeridas.
Concisión	Lo compacto que es el programa en términos de líneas de código.
Consistencia	El uso de un diseño uniforme y de técnicas de documentación a lo largo de todo el programa.
Estandarización en los datos	El uso de estructuras de datos y de tipos estándar a lo largo de todo el

[2] Jim McCall realizador del modelo de calidad de McCall, desarrollado inicialmente para la Fuerza Aérea de los EE.UU en 1977 y que actualmente es uno de los más renombrados.

	programa.
Tolerancia de errores	El daño que se produce cuando el programa detecta una situación errónea.
Eficiencia en la ejecución	El rendimiento en tiempo de ejecución de un programa.
Facilidad de expansión	El grado en que se puede ampliar el diseño arquitectónico, de datos o procedimental.
Generalidad	La amplitud de aplicación potencial de los componentes del programa.
Independencia del hardware	El grado en que el software es independiente del hardware sobre el que opera.
Instrumentación	El grado en que el programa muestra su propio funcionamiento e identifica errores que aparecen.
Modularidad	La independencia funcional de los componentes del programa.
Facilidad de operación	La facilidad de utilización de un programa.
Seguridad	La disponibilidad de mecanismos que controlen o protejan los programas o los datos.
Auto documentación	El grado en que el código fuente proporciona documentación significativa.
Simplicidad	El grado en que un programa puede ser entendido sin dificultad.
Independencia del sistema de software	El grado en que el programa es independiente de características no estándar del lenguaje de programación, de las características del sistema operativo y de otras restricciones del entorno.
Facilidad de traza	La posibilidad de seguir la pista a la representación del diseño o de los componentes reales del programa hacia atrás, hacia los requisitos.
Formación	El grado en que el software ayuda a permitir que nuevos usuarios empleen el sistema.

1.5Clasificación de las Métricas

Existen innumerables métricas con propósitos diferentes que reflejan o describen la conducta del software, estas pueden medir entre otros aspectos la competencia, calidad, desempeño y

la complejidad del software contribuyendo a establecer de una manera sistemática y objetiva una visión interna del trabajo mejorando así la calidad del producto.(Hernández, 2007)

A continuación se muestra la clasificación de las mismas:

Métricas de complejidad: son todas las métricas de software que definen de una u otra forma la medición de la complejidad; tales como volumen, tamaño, anidaciones, costo (estimación) y configuración. Estas son los puntos críticos de la concepción, viabilidad, análisis, y diseño de software.

Métricas de calidad: son todas las métricas de software que definen de una u otra forma la calidad del software; tales como exactitud, estructuración o modularidad, pruebas, mantenimiento y reusabilidad. Estas son los puntos críticos en el diseño, codificación, pruebas y mantenimiento.

Métricas de competencia: son todas las métricas que intentan valorar o medir las actividades de productividad de los programadores o practicantes con respecto a su certeza, rapidez, eficiencia y competencia.

Métricas de desempeño: corresponden a las métricas que miden la conducta de módulos y sistemas de un software, bajo la supervisión del sistema operativo o hardware. Generalmente tienen que ver con la eficiencia de ejecución, tiempo, almacenamiento, complejidad de algoritmos computacionales.

Métricas estilizadas: son las métricas de experimentación y de preferencia; por ejemplo: estilo de código y limitaciones.

Luego de haber clasificado las métricas, es conveniente realizar una descripción de las mismas, teniendo en cuenta que la evaluación de una aplicación está dirigida a medir aspectos relacionados con el proceso, el proyecto y el producto. Por esta razón existen métricas destinadas a cada uno de estos fines. En el siguiente epígrafe se hace una descripción detallada de estas métricas.

1.5.1 Métricas del proceso de software, métricas del proyecto de software y métricas del producto de software.

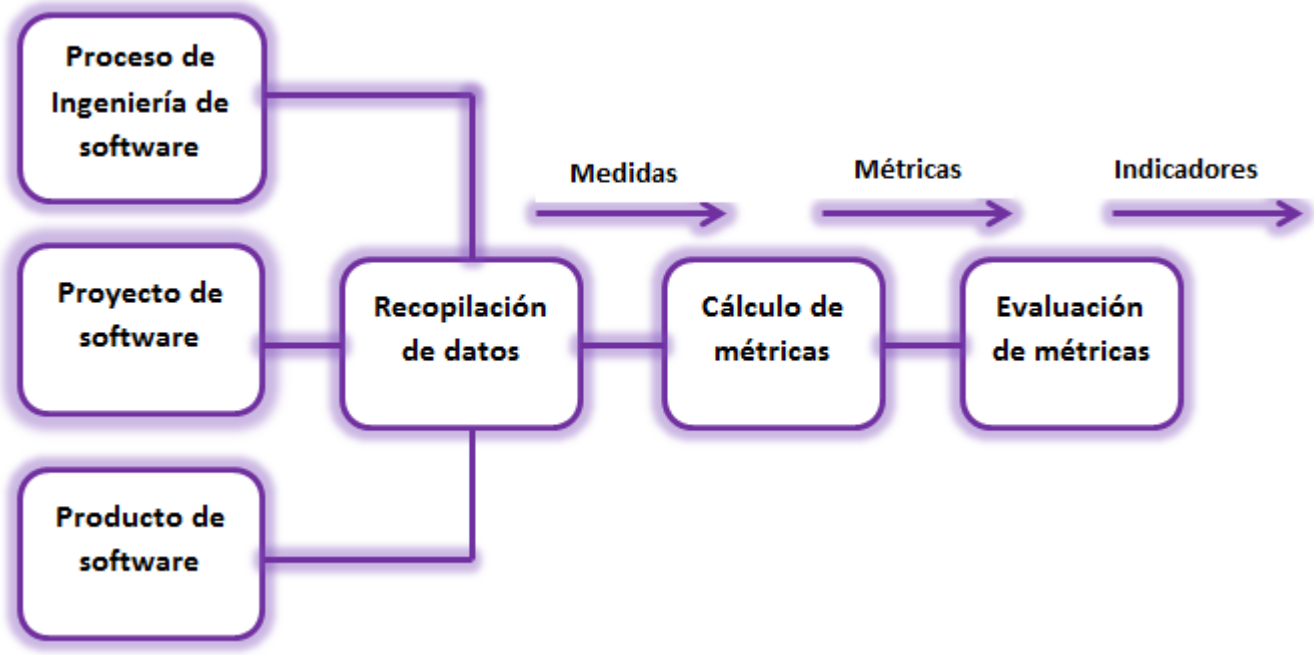

Figura 1. Proceso de evaluación del software

Métricas de proceso: las métricas del proceso de software deben medir los atributos del mismo, y se utilizan para proporcionar indicadores que conduzcan a la mejora del proceso. Los errores detectados antes de la entrega del software, la productividad, recursos, tiempo consumido y ajuste con la planificación son algunos de los resultados que pueden medirse en el proceso, así como las tareas específicas de la ingeniería del software. (Hernández, 2007)

- Se recopilan de todos los proyectos, durante un largo período de tiempo.
- Caracterizados por el control y ejecución del proyecto.
- Medición de tiempos de las fases.

Métricas de proyecto: las métricas del proyecto son medidas cuantitativas que proporcionan a los ingenieros de software una amplia visión del proceso y un conocimiento detallado acerca del proyecto que se lleva a cabo utilizando el proceso como marco de trabajo. La medición permite destacar las tendencias (ya sean buenas o malas) y hacer mejores estimaciones que conducirán a un proyecto exitoso; comienza definiendo un conjunto limitado de medidas del proceso y del proyecto, las cuales por lo general se normalizan empleando métricas

orientadas al tamaño o la función, el resultado se analiza y compara con promedios pasados, luego se valoran las tendencias y se generan conclusiones. (Hernández, 2007)

- Permiten evaluar el estado del proyecto.
- Permiten seguir la pista de los riesgos.

Métricas de producto: las métricas del producto se centran en las características del software y no en cómo fue producido. Un producto no es solo el software o sistema funcionando sino también los artefactos, documentos, modelos, módulos, y componentes que lo conforman, por tanto, las métricas del producto deben hacerse sobre la base de medir cada uno de estos. En los artefactos del producto se mide, entre otras cosas, el tamaño, la calidad (teniendo en cuenta los defectos y la complejidad), la totalidad, rastreabilidad, volatilidad, esfuerzo.(Hernández, 2007)

- Se centran en las características del software, no en cómo fue producido.
- También son productos los artefactos, documentos, modelos, y componentes que conforman el software.
- Se miden el tamaño, la calidad, la totalidad, la volatilidad, y el esfuerzo.

1.5.2. Caracterización de las métricas del proceso de software y métrica del proyecto software y del producto de software.

1.5.2.1 Métricas del Proceso de Software

Las métricas del proceso se caracterizan por:

- El control y ejecución del proyecto.
- Medición de tiempos del análisis, diseño, implementación y despliegue.
- Medición de las pruebas (errores, cubrimiento, resultado en número de defectos y número de éxito).
- Medición de la transformación o evolución del producto.

Los indicadores del proceso permiten:

1. Evaluar el estado del proyecto en curso.
2. Seguir la pista de riesgos potenciales.
3. Detectar áreas problemáticas antes de que se conviertan en críticas.
4. Ajustar el flujo y las tareas de trabajo.

5. Evaluar la habilidad del equipo del proyecto en controlar la calidad de los productos de trabajo de la ingeniería de software. (Sicilia, 2009)

1.5.2.2 Métricas del Proyecto de Software

Las métricas del proyecto se caracterizan por:

- Tener un propósito táctico.
- Poseer una doble finalidad.
- Minimizar tiempos de desarrollo, reducción de problemas y riesgos.
- Valorar calidad del producto y reducción de reelaboración.

Los indicadores del proyecto permiten:

1. Evaluar el estado del proyecto en curso.
2. Seguir la pista de riesgos potenciales.
3. Detectar áreas problemáticas antes de que se conviertan en críticas.
4. Ajustar el flujo y las tareas de trabajo.
5. Evaluar la habilidad del equipo del proyecto en controlar la calidad de los productos de trabajo de la IS.(Sicilia, 2009)

1.5.2.3 Métricas del Producto de Software

Estas métricas describen las características del producto que de alguna forma determinan la mantenibilidad, por ejemplo el tamaño, complejidad o características del diseño.
Las métricas orientadas al producto son:

- La densidad de comentarios en el código.
- Métricas de Complejidad.
- El índice de madurez del software (IMS).
- Métricas en Orientación a Objetos: Chidamber & Kemerer.

Los indicadores del producto permiten:

1. Evaluar el estado de un producto que se está realizando o se terminó.
2. Seguir la pista de los riesgos potenciales de la insatisfacción de los requisitos acordados con el cliente.
3. Detectar los procesos y áreas con problemas antes de que se conviertan en críticas.

4. Ajustar el proceso de realización de software para lograr productos eficientes, eficaces y rentables.
5. Evaluar la habilidad del equipo de proyecto y de las áreas productoras.(Sicilia, 2009)

1.6 Modelos de calidad

Desde el principio de la ingeniería de software, se observó que la calidad está definida por una composición de muchas características. Un modelo de calidad describe estas características y sus relaciones, muchos modelos hacen difusa la distinción entre atributos internos y externos, lo que dificulta la comprensión del concepto de calidad. En la práctica, los modelos de calidad resultan de utilidad para la predicción de confiabilidad y en la gerencia de calidad durante el proceso de desarrollo, así como para efectuar la medición del nivel de complejidad de un sistema de software.

Es interesante destacar que la organización y descomposición de los atributos de calidad han permitido el establecimiento de modelos específicos para efectos de la evaluación de la calidad arquitectónica. Pressman, indica que los factores que afectan a la calidad del software no cambian, por lo que resulta útil el estudio de los modelos de calidad que han sido propuestos en este sentido desde los años 70. Dado que los factores de calidad presentados para ese entonces siguen siendo válidos, se estudiaron los modelos más importantes propuestos hasta la fecha: FURPS (1987), CMM (1993), Dromey (1996), ISO/IEC 9126(2004) y CMMI que es el modelo que se aplica actualmente en la Universidad.

1.6.1 Modelo FURPS

El modelo de McCall ha servido de base para modelos de calidad posteriores, este es el caso del modelo FURPS, producto del desarrollo de Hewlett-Packard. El modelo FURPS incluye, además de los factores de calidad y los atributos, restricciones de diseño y requerimientos de implementación, físicos y de interfaz. Las restricciones de diseño especifican o restringen el diseño del sistema. Los requerimientos de implementación especifican o restringen la codificación o construcción de un sistema. Por su parte, los requerimientos de interfaz especifican el comportamiento de los elementos externos con los que el sistema debe interactuar. Por último, los requerimientos físicos especifican ciertas propiedades que el

sistema debe poseer, en términos de materiales, forma, peso, tamaño (por ejemplo, requisitos de hardware, configuración de red).

En este modelo se desarrollan un conjunto de factores de calidad de Software, bajo el acrónimo de FURPS: funcionalidad (Functionality), usabilidad (Usability), confiabilidad (Reliability), desempeño (Performance) y capacidad de soporte (Supportability). La tabla 4 presenta la clasificación de los atributos de calidad que se incluyen en el modelo, junto con las características asociadas a cada uno. (Pressman, 2002)

Tabla 4. Atributos de Calidad – Modelo FURPS

Factor de Calidad	Atributos
Funcionalidad	Características y capacidades del programa Generalidad de las funciones Seguridad del sistema
Facilidad de uso	Factores humanos Factores estéticos Consistencia de la interfaz Documentación
Confiabilidad	Frecuencia y severidad de fallas Exactitud de las salidas Tiempo medio de fallos Capacidad de recuperación ante fallas Capacidad de predicción
Rendimiento	Velocidad del procesamiento Tiempo de respuesta Consumo de recursos Rendimiento efectivo total Eficacia
Capacidad de Soporte	Extensibilidad Adaptabilidad Capacidad de pruebas Capacidad de configuración Compatibilidad

	Requisitos de instalación

1.6.2 Modelo CMM

Este modelo establece un conjunto de prácticas o procesos claves agrupados en Áreas Clave de Proceso, (*Key Process Area - KPA* por sus siglas en inglés). Para cada área de proceso define un conjunto de buenas prácticas que habrán de ser:

- Definidas en un procedimiento documentado.
- Provistas (la organización) de los medios y formación necesarios.
- Ejecutadas de un modo sistemático, universal y uniforme (institucionalizadas).
- Medidas.
- Verificadas.

Cada *KPA* identifica un conjunto de actividades y prácticas interrelacionadas, las cuales cuando son realizadas en forma colectiva permiten alcanzar las metas fundamentales del proceso. Las *KPAs* pueden clasificarse en 3 tipos de proceso: gestión, organizacional e ingeniería.

Las prácticas que deben ser realizadas por cada Área Clave de Proceso están organizadas en 5 características comunes, las cuales constituyen propiedades que indican si la implementación y la institucionalización de un proceso clave es efectivo, repetible y duradero. Estas 5 características son:

- Compromiso de la realización.
- La capacidad de realización.
- Las actividades realizadas.
- Las mediciones y el análisis.
- La verificación de la implementación.

Este modelo establece una medida del progreso, conforme al avance en niveles de madurez. Cada nivel a su vez cuenta con un número de áreas de proceso que deben lograrse. El alcanzar estas áreas o estadios se detecta mediante la satisfacción o insatisfacción de varias metas claras y cuantificables. Con la excepción del primer nivel, cada uno de los restantes Niveles de Madurez está compuesto por un cierto número de Áreas Claves de Procesos, conocidas a través de la documentación del CMM por su sigla en inglés: *KPA.*

A su vez estas Áreas de Procesos se agrupan en cinco "niveles de madurez", de modo que una organización que tenga institucionalizadas todas las prácticas incluidas en un nivel y sus inferiores, se considera que ha alcanzado ese nivel de madurez.

Los niveles son:

Inicial: las organizaciones en este nivel no disponen de un ambiente estable para el desarrollo y mantenimiento de software. Aunque se utilicen técnicas correctas de ingeniería, los esfuerzos se ven minados por falta de planificación. El éxito de los proyectos se basa la mayoría de las veces en el esfuerzo personal, aunque a menudo se producen fracasos y casi siempre retrasos y sobrecostes. El resultado de los proyectos es impredecible.

Repetible: en este nivel las organizaciones disponen de prácticas institucionalizadas de gestión de proyectos, existen métricas básicas y un razonable seguimiento de la calidad. La relación con subcontratistas y clientes está gestionada sistemáticamente.

Definido: además de una buena gestión de proyectos, a este nivel las organizaciones disponen de correctos procedimientos de coordinación entre grupos, formación del personal, técnicas de ingeniería más detallada y un nivel más avanzado de métricas en los procesos. Se implementan técnicas de revisión por pares (peer reviews).

Gestionado: se caracteriza porque las organizaciones disponen de un conjunto de métricas significativas de calidad y productividad, que se usan de modo sistemático para la toma de decisiones y la gestión de riesgos. El software resultante es de alta calidad.

Optimizado: la organización completa está volcada en la mejora continua de los procesos. Se hace uso intensivo de las métricas y se gestiona el proceso de innovación.

1.6.3. Modelo de Dromey

Dromey[3] (1996)propuso un marco de referencia o metamodelo para la construcción de modelos de calidad, basado en cómo las propiedades medibles de un producto de software pueden afectar los atributos de calidad generales, como por ejemplo, confiabilidad y mantenibilidad. El problema que se plantea es cómo conectar tales propiedades del producto con los atributos de calidad de alto nivel. Para solventar esta situación, Dromey sugiere el uso de cuatro categorías que implican propiedades de calidad, que son: correctitud, internas, contextuales y descriptivas.

[3]George Dromey creador del modelo de calidad Dromey en 1996, explica que la importancia de este modelo es que, a diferencia de los anteriores, plantea que no todos los productos de software pueden evaluar su calidad de la misma manera.

El proceso de construcción de modelos de calidad propuesto por Dromey consta de 5 pasos, basados en las propiedades mencionadas. Los pasos del marco de referencia propuesto son:

- Especificación de los atributos de calidad de alto nivel (por ejemplo, confiabilidad, mantenibilidad).
- Determinación de los distintos componentes del producto a un apropiado nivel de detalle (por ejemplo, paquetes, subrutinas, declaraciones).
- Determinación y categorización de las implicaciones más importantes de calidad de cada componente.
- Proposición de enlaces que relacionan las propiedades implícitas a los atributos de calidad, o alternativamente, uso de enlaces de las cuatro categorías de atributos propuestas.
- Iteración sobre los pasos anteriores, utilizando un proceso de evaluación y refinamiento.

La tabla 5 presenta la relación que establece Dromey entre las propiedades de calidad del producto y los atributos de calidad de alto nivel.

Tabla 5. Relación entre propiedades del producto y atributos de calidad – Modelo de Dromey

Propiedad del producto	Atributos de Calidad
Correctitud	Funcionalidad Confiabilidad
Internas	Mantenibilidad Eficacia Confiabilidad
Contextuales	Mantenibilidad Reusabilidad Portabilidad Confiabilidad
Descriptivas	Mantenibilidad Reusabilidad Portabilidad Usabilidad

1.6.4 ISO/IEC 9126

El modelo ISO/IEC 9126 de calidad de software para efectos de la evaluación de arquitecturas de software, se basa en los atributos de calidad que se relacionan directamente con la arquitectura: funcionalidad, confiabilidad, eficiencia, mantenibilidad y portabilidad. Los autores del modelo plantean que la característica usabilidad propuesta por el modelo ISO/IEC 9126 puede ser refinada para obtener atributos que se relacionan con los componentes de la interfaz con el usuario. Dado que estos componentes son independientes de la arquitectura, no son considerados en la adaptación del modelo. La tabla 6 presenta los atributos de calidad planteados por Losavio et al. (2003), que poseen subcaracterísticas asociadas con elementos de tipo arquitectónico.

Tabla 6. Atributos de calidad planteados por Losavio et al. (2004)

Características	Subcaracterísticas	Elementos de tipo arquitectónico
Funcionalidad	Adecuación	Refinamiento de los diagramas de secuencia.
	Exactitud	Identificación de los componentes con las funciones responsables de los cálculos.
	Interoperabilidad	Identificación de conectores de comunicación con sistemas externos.
	Seguridad	Mecanismos o dispositivos que realizan explícitamente las tareas.
Confiabilidad	Tolerancia a fallas	Existencia de mecanismos o dispositivos de software para manejar excepciones.
	Recuperabilidad	Existencia de mecanismos o dispositivos de software para restablecer el nivel de desempeño y recuperar datos.
Eficiencia	Desempeño	Componentes involucrados en el flujo de ejecución para una funcionalidad.
	Utlización de recursos	Relación de los componentes en términos de espacio y tiempo.
Mantenibilidad	Acoplamiento	Interacciones entre components.
	Modularidad	Número de componentes que dependen de un

		componente.
Portabilidad	Adaptabilidad	Presencia de mecanismos de adaptación.
	Instalabilidad	Presencia de mecanismos de instalación.
	Coexistencia	Presencia de mecanismos que faciliten la coexistencia.
	Reemplazabilidad	Lista de componentes reemplazables para cada componente.

1.6.5 Modelo CMMI

CMMI es un modelo para la mejora de procesos que proporciona a las organizaciones los elementos esenciales para procesos eficaces, además muestra la madurez de una organización basándose en la capacidad de sus procesos. Surge como la integración del CMM (Capability Maturity Model) v.2.0 y de la ISO 15504 Draft Standar v.1.00. Las mejores prácticas CMMI se publican en los documentos llamados modelos. En la actualidad hay dos áreas de interés cubiertas por los modelos de CMMI: Desarrollo y Adquisición. La versión actual de CMMI es la versión 1.2. Hay dos modelos de la versión 1.2 disponible:

- CMMI para el Desarrollo (DEV-CMMI), Versión 1.2 fue liberado en agosto de 2006. En él se tratan procesos de desarrollo de productos y servicios.
- CMMI para la adquisición (ACQ-CMMI), Versión 1.2 fue liberado en noviembre de 2007. En él se tratan la gestión de la cadena de suministro, adquisición y contratación externa en los procesos del gobierno y la industria.

Este modelo de procesos tiene dos representaciones: continua y por etapas, siendo la diferencia entre éstas la evaluación por niveles de la capacidad de procesos o de la madurez de la organización, respectivamente. Las áreas de procesos (AP) en este modelo se agrupan en cuatro categorías: la Gestión de Proyectos, Soporte, la Gestión de Procesos, y de Ingeniería.

La representación del modelo CMMI que se utiliza en los proyectos es por etapas, el cual define cinco niveles de madurez dentro de los cuales se puede encontrar una organización. Un nivel de madurez representa un indicador evolutivo que permite alcanzar la madurez del proceso de software. Estos niveles pretenden alcanzar objetivos de acuerdo con la capacidad del proceso de software, los cuales una vez cumplidos, permitirán evolucionar al siguiente nivel. A continuación se especifican en la siguiente figura los cinco niveles de madurez de CMMI.

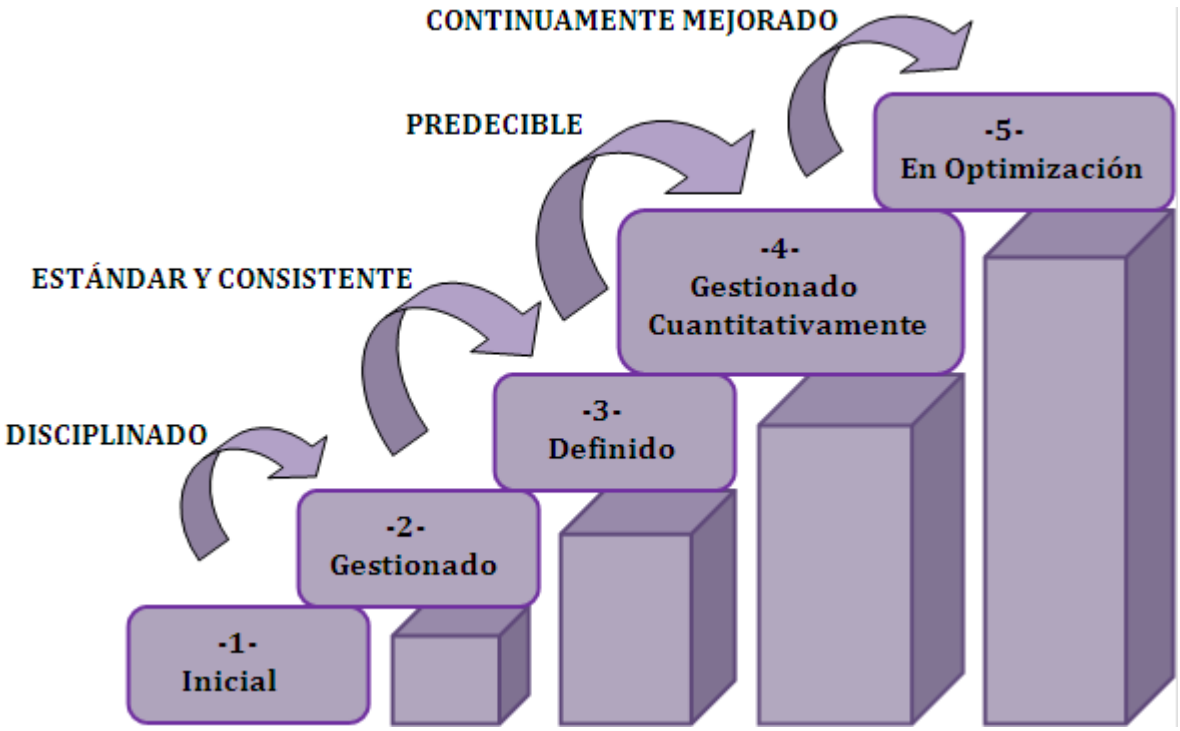

Figura 2. Niveles de madurez de CMMI

NIVEL 1 – Inicial. El proceso de software es impredecible, sin control y reactivo. El éxito de los proyectos depende del talento de las personas involucradas.

NIVEL 2 –Gestionado. Existen procesos básicos de gestión en los proyectos (costo, calendario, funcionalidad). Los procesos existentes hacen que se puedan repetir éxitos en proyectos de similares características.

NIVEL 3– Definido. Existe un proceso de software documentado y estandarizado dentro de la organización. Todos los proyectos utilizan una versión a medida del proceso.

NIVEL 4– Gestionado Cuantitativamente. La organización recolecta métricas del proceso software y de los productos desarrollados. Tanto el proceso como los productos se entienden y controlan cuantitativamente.

NIVEL 5– En Optimización. Existe una mejora continua del proceso software, basada en la realimentación cuantitativa del proceso y en la puesta en práctica de ideas y tecnologías innovadoras.

1.6.6 ISO 25000

La norma ISO 25000 proporciona una guía para el uso de nuevos estándares internacionales, llamados Requisitos y Evaluación de Calidad de Productos de Software (SQuaRE). Lo cual constituye una serie de normas basadas en la ISO 9126 y en la ISO 14598(Evaluación del Software), y su objetivo principal es guiar el desarrollo de los productos de software con la especificación y evaluación de requisitos de calidad. (LAURA)

Este estándar propone un modelo de calidad que se divide en tres vistas: interior, exterior y en uso.

- Vista interna: esta vista se ocupa de las propiedades del software como: el tamaño, la complejidad o la conformidad con las normas de orientación a objetos.
- Vista externa: analiza el comportamiento del software en producción y estudia sus atributos, por ejemplo: el rendimiento de un software en una máquina determinada, el uso de memoria de un programa o el tiempo de funcionamiento entre fallos.
- Vista en uso: mide la productividad y efectividad del usuario final al utilizar el software.

En la siguiente figura se representa el diagrama de las vistas internas y externas, descritas anteriormente.

Calidad externa e interna

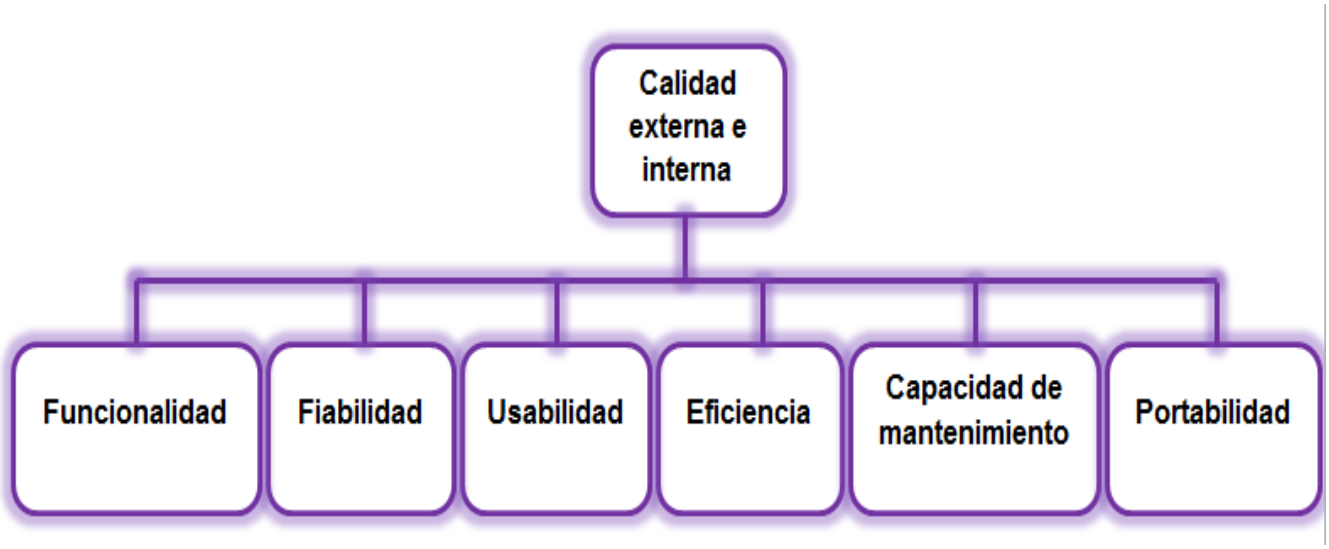

Figura 3. Calidad externa/ interna

A continuación se definen los aspectos que se evidencian en la figura anterior:

- Funcionalidad: capacidad del software de proveer los servicios necesarios para cumplir con los requisitos funcionales.

- Fiabilidad: capacidad del software de mantener las prestaciones requeridas del sistema, durante un tiempo establecido y bajo un conjunto de condiciones definidas.
- Usabilidad: esfuerzo requerido por el usuario para utilizar el producto satisfactoriamente.
- Eficiencia: relación entre las prestaciones del software y los requisitos necesarios para su utilización.
- Capacidad de mantenimiento: esfuerzo necesario para adaptarse a las nuevas especificaciones y requisitos del software.
- Portabilidad: capacidad del software ser transferido de un entorno a otro.

Posteriormente se muestra una figura que relaciona los aspectos a tener en cuenta en la vista calidad en uso.

Calidad en uso

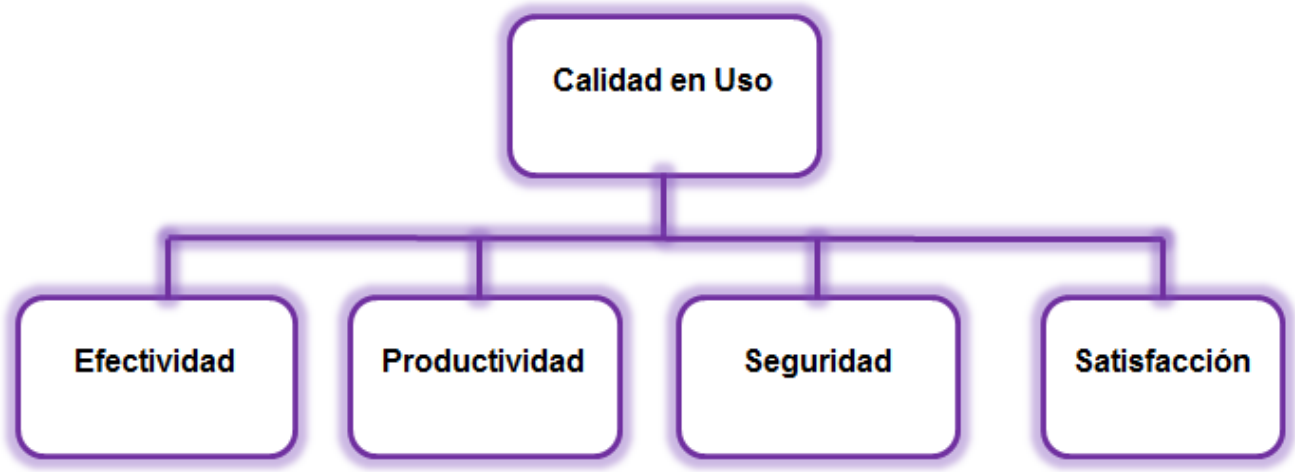

Figura 4. Calidad en uso

- Efectividad: capacidad del software de facilitar al usuario alcanzar objetivos con precisión y completitud.
- Productividad: capacidad del software de permitir a los usuarios gastar la cantidad apropiada de recursos en relación a la efectividad obtenida.
- Seguridad: capacidad del software para cumplir con los niveles de riesgo permitidos tanto para posibles daños físicos como para posibles riesgos de datos.
- Satisfacción: capacidad del software de cumplir con las expectativas de los usuarios en un contexto determinado.

Una vez analizado estos modelos de calidad, teniendo en cuenta que en la UCI actualmente se aplica el modelo CMMI y que se pretende alcanzar el nivel 2 del mismo; se puede apreciar que las prácticas de CMMI deben adaptarse a cada organización en función de sus objetivos de negocio. En este nivel no existen métricas definidas para los proyectos que se desarrollan en la universidad. Pues según lo que establece CMMI, cuando estén declaradas las métricas del proceso software y de los productos desarrollados la organización estará en el nivel 4 del modelo. Por tal motivo se decidió utilizar para la propuesta la norma ISO 25000, esta proporciona una guía para el uso de nuevos estándares internacionales, y su objetivo principal es guiar el desarrollo de los productos de software con la especificación y evaluación de requisitos de calidad. (LAURA)

1.7 Conclusiones

El análisis realizado en este capítulo ha demostrado de manera clara, la importancia que tiene la evaluación del software durante todo el ciclo de desarrollo, de forma tal que al culminar, el producto tenga la mayor calidad posible y cumpla con las expectativas del cliente. Se analizan aspectos relacionados con las métricas para evaluar un software, los factores por los que se ve afectada la calidad de un producto, así como algunos modelos de calidad propuestos por diferentes autores para evaluar atributos de calidad del producto**,** lo que permite tener una mejor visión ante los posibles fallos que puede tener el mismo. Finalmente se escogió la norma ISO 25000 para realizar la propuesta de métricas que arrojó la presente investigación.

CAPÍTULO 2

Selección de Métricas

2.1 Introducción

Para lograr que los productos que se realizan en el centro, sean de calidad, se debe garantizar la realización de una buena evaluación del software, desde la etapa de inicio y durante todo el proceso de desarrollo, de forma tal que en cada etapa se identifiquen y eliminen los defectos que tiene el producto, de modo que al finalizar el producto, tenga la menor cantidad de inconformidades posibles. Este capítulo tiene como objetivo principal abordar los temas fundamentales referidos a los proyectos de Investigación, Desarrollo e Innovación (I+D+i) que existen en el Departamento Señales Digitales del centro GEySED. Para ello se realizó un estudio detallado que permitió caracterizar cada uno de los proyectos, identificar rasgos comunes entre ellos, seleccionar los parámetros que fueron objeto de evaluación en estos, así como establecer comparaciones entre sí. De este modo se pudo recopilar toda la información necesaria para escoger las métricas adecuadas para evaluar los proyectos del departamento.

2.2 Líneas comunes de los proyectos de Investigación Desarrollo e Innovación (I+D+i) que existen en el Departamento Señales Digitales del centro GEySED.

Actualmente en el Departamento de Señales Digitales del centro GEySED se desarrollan 6 proyectos de Investigación Desarrollo e Innovación (I+D+i) estos son Sistema de Captura y Catalogación de Medias (SCCM), Plataforma de Transmisión Abierta para Radio y Televisión (PTAR - TV), Plataforma de Televisión Informativa (Primicia), Plataforma Video Web (VideoWeb), Video Vigilancia, Componente y Tecnología (CT), lo que representa el total de la población. De ellos se escogió de forma aleatoria una muestra de 4 proyectos, que representan el 57 % de la población. Estos son: Sistema de Captura y Catalogación de Medias (SCCM), Plataforma de Transmisión Abierta para Radio y Televisión (PTAR - TV), Plataforma de Televisión Informativa (Primicia) y Plataforma Video Web (VideoWeb). Mediante una entrevista (Ver Anexo 1) realizada a cada uno de los líderes de los proyectos

seleccionados se escogieron las líneas comunes de los proyectos del departamento. A continuación se caracterizan cada uno de los proyectos escogidos.

2. 2.1 Características de los proyectos del departamento Señales Digitales del centro GEySED.

2.2.1.1Sistema de Captura y Catalogación de Medias (SCCM)

Objetivo General: desarrollar una solución de software que se adapte a las características de los clientes para la grabación, monitoreo y análisis de las señales de radio y televisión.
El proyecto cuenta con 5 subsistemas que a su vez tienen otros subsistemas y módulos.

- Subsistema de Administración: cuenta con el subsistema de Catalogación, Recuperación y Préstamos, Gestión de Procesos de Medias y el subsistema de Ingesta.
- Subsistema de gestión de Procesos de Medias: cuenta con el módulo de Indexación y Codificación.

En la tabla 7 se especifican las funciones de cada uno de los subsistemas o módulos mencionados con anterioridad.

Tabla 7. Funciones de los subsistemas o módulos del proyecto SCCM

Subsistema	**Módulo**	**Función**
Administración	-	Gestión de procesos.
Catalogación	-	Descripción de materiales
Recuperación y Préstamo.	-	Obtención de materiales (flujo de préstamo)
Gestión de Procesos de Medias.	-	Control, supervisión y organización de los flujos de medias
Ingesta	-	Gestionar la organización de la empresa e inicia el proceso.
	Indexación y Codificación	Codificación e identificación de patrones.

2.2.1.2 Plataforma de Transmisión abierta para radio y televisión (PTAR - TV)

Objetivo General: integrar y organizar los procesos que se realizan en la Dirección de la Televisión Universitaria. Es capaz de difundir las informaciones mediante un sitio web o de un televisor, haciendo uso de las nuevas tecnologías, logrando que se pueda acceder al contenido audiovisual que pudiera emitirse por estos medios en todo momento.

El proyecto cuenta con 12 subsistemas:

- Subsistema de Transmisión
- Administración de la Transmisión
- Monitoreo
- Transferencia
- Reporte
- Equipamiento
- Producción
- Web
- Gestión de Medias
- Programación
- Seguridad
- Radial.

La tabla 8 muestra las funciones de cada uno de los subsistemas del proyecto PTAR -TV.

Tabla 8. Funciones de los subsistemas del proyecto PTAR -TV

Subsistema	Módulo	Función
Transmisión	-	Transmisión de la televisión.
Administración de la Transmisión	-	Gestionar el proceso de transmisión de la televisión.
S Monitoreo	-	Monitorear la trasmisión de la televisión.
Transferencia	-	Tiene dos funciones específicas: Conversión. Copia.
Reporte	-	Realizar los reportes del sistema.
Equipamiento	-	Equipamiento que se utiliza dentro del proyecto para la transmisión de radio y televisión.

Web	-	Prestar servicios mediante la web para la transmisión de radio y televisión.
Gestión de Medias	-	Organización de servicios medias: subir media, registrar media.
Programación	-	Programación de la transmisión de radio y televisión.
Producción	-	Producción.
Radial	-	Transmisión de radio.
Seguridad		Seguridad del sistema.

2.2.1.3 Plataforma de Televisión Informativa (Primicia)

Objetivo General: proveer un canal de Televisión Informativa.

El proyecto cuenta con 2 subsistemas que a su vez tienen módulos. Estos son el Subsistema de Administración y el subsistema de Transmisión. El subsistema de Administración cuenta con 6 módulos: Corrección, Editorial, Medias, Redacción, Reporte y Seguridad. El subsistema de Transmisión: cuenta con el módulo Transmisión.

En la tabla 9 que se muestra a continuación se especifican las funciones de cada uno de los subsistemas y módulos del proyecto Primicia.

Tabla 9. Funciones de los subsistemas y módulos del proyecto Primicia

Subsistema	Módulo	Función
Administración	-	Gestión y la Administración de noticias y medias del canal.
Transmisión	-	Transmisión y la visualización del canal.
-	Corrección	Corrección de las noticias que se van a publicar en el canal.
-	Medias	Gestión de Medias.
-	Redacción	Redacción de la noticia que se publicará en el canal.
-	Reporte	Registrar los reportes de las temáticas, y la música.
-	Seguridad	Seguridad del sistema.

2.2.1.4 Plataforma Video Web (VideoWeb)

Objetivo General: obtener una solución de software que permita la gestión y transmisión de contenidos multimedia a través de la red de datos por medio de un flujo constante de manera que los usuarios no tengan que descargar los materiales hacia la PC terminal donde se quieren reproducir.

El proyecto cuenta con 2 subsistemas y 10 módulos. El Subsistema de Administración que cuenta con 4 módulos: Tipología Archivo Multimedia, filehtml, ftphtml, y Almacén. El Subsistema de Gestión y Presentación de Contenido que cuenta con 6 módulos: Archivo Multimedia, Archivo Multimedia en vivo, Archivo Multimedia lista, Reproductor de Archivo Multimedia, Video Web, Video Web Bloque.

En la tabla 10 que se muestra a continuación se especifican las funciones de cada uno de los subsistemas y módulos del proyecto VideoWeb.

Tabla 10. Funciones de los subsistemas y módulos del proyecto VideoWeb

Subsistema	Módulo	Función
-	Tipología Archivo Multimedia.	Manipular información asociada a los archivos multimedia según su clasificación tipológica
-	Almacén	Capa de abstracción para la transferencia y almacenamiento de los Archivos Multimedia.
-	Archivo Multimedia	Gestión de los archivos multimedia que se publican en la plataforma video web, además almacena los archivos multimedia empleando “Almacén” y la información asociada a los archivos multimedia empleando el modulo “Tipología”.
-	Archivo Multimedia en vivo	Gestionar la transmisión de señales en vivo.
	Archivo Multimedia lista	Gestión de la lista de reproducción de los archivos multimedia.
-	Video Web Bloque	Gestionar bloques dinámicos.
	Video Web	Gestión de los archivos multimedia mediante la web.
-	Reproductor de Archivo Multimedia	Reproducción de archivos Multimedia (audio y video).

Además de caracterizar los proyectos escogidos, como resultado de la entrevista realizada se determinaron las funcionalidades esenciales que en cada uno de ellos se realizan. Entre ellas se encuentran: la captura de medias, la indexación de medias, la transmisión de medias, la ingesta de medias, la gestión de medias, la tipología y la conversión de medias. Es por esto que en la tabla 11 se puede apreciar en qué proyecto de los seleccionados se realiza la funcionalidad determinada. La columna representa el proyecto y la fila representa cada una de las funciones.

Tabla 11. Comparación entre los proyectos del Departamento Señales Digitales del centro GEySED

Funcionalidad / Proyecto	Captura de medias	Indexación de medias	Transmisión de medias	Ingesta de medias	Gestión de medias	Tipología	Conversión de medias
SCCM	**Sí**	**Sí**	**Sí**	**Sí**	**Sí**	**Sí**	**Sí**
PTAR - TV	**Sí**	**No**	**Sí**	**No**	**Sí**	**Sí**	**Sí**
Primicia	**No**	**No**	**Sí**	**No**	**Sí**	**Sí**	**No**
VideoWeb	**Sí**	**Sí**	**Sí**	**No**	**Sí**	**Sí**	**Sí**

De forma general se puede decir que los proyectos seleccionados desarrollan aplicaciones vinculadas con la radio y la televisión. Además teniendo en cuenta la información arrojada en la tabla anterior se puede llegar a las siguientes conclusiones: todos los proyectos realizan gestión y transmisión de medias, y todos clasifican dichas medias en tipologías. Sin embargo se ve como un caso particular que solo en el proyecto SCCM, que no es una solución web se realiza la ingesta de medias. La conversión de medias la realizan todos los proyectos menos Primicia y la indexación solo se hace en SCCM y VideoWeb.

Teniendo en cuenta las principales características y funcionalidades de cada uno de los proyectos del departamento. Así como los resultados de la entrevista realizada a los líderes de proyecto, se evidenció que en estos no se usa hoy día ninguna métrica para evaluar la calidad de los productos que se realizan en el mismo. Por tal motivo se decidió que antes de hacer una propuesta de métricas relacionadas con cada una de las funcionalidades mostradas anteriormente, se debe hacer una propuesta de métricas que defina el estándar ISO 25 000 que puedan aplicarse a todos los proyectos del departamento. Pues estas funcionalidades son bastante complejas para aplicarle métricas específicas basadas en algoritmos matemáticos que arrojan un resultado real. Partiendo que el objetivo de la investigación es proponer métricas que sean aplicables a cada uno de los proyectos del departamento en el siguiente epígrafe se describen las métricas que define la norma ISO 25000 de la cual se hará la propuesta final.

2.3 Métricas que define la norma ISO 25 000.

Las métricas de software deben ser un instrumento que ayude a mejorar el proceso, producto o proyecto de software, no tiene mucho sentido aplicar métricas que lejos de ayudar a los desarrolladores constituyan un problema; bien por ser demasiado complejas, porque no se entiendan correctamente los objetivos que persiguen o porque arrojen resultados imprecisos que no puedan ser interpretados por los ingenieros de software. Es importante entonces que una métrica pueda obtenerse fácilmente, que se entienda por qué y para qué se utiliza, que los cálculos no produzcan resultados ambiguos o en los que existan extrañas combinaciones de unidades, y que la interpretación de valores obtenidos esté acorde a las nociones intuitivas del ingeniero de software.

Los atributos obtienen sus valores tras realizar mediciones sobre el software. Estas mediciones dan como resultado una serie de métricas que se clasifican según la norma ISO 25 000 en tres categorías según sea su naturaleza:

- **Métricas básicas:** que se obtienen directamente de analizar el código o la ejecución del software.
- **Métricas de agregación:** que consisten en la composición de una métrica a partir de un conjunto definido de métricas básicas, generalmente mediante una suma ponderada.
- **Métricas derivadas:** que son una función matemática que utiliza como entrada el valor de otras métricas.

En la tabla que se muestra a continuación, se muestran las características y subcaracterísticas de calidad conforme al estándar ISO/IEC 9126.

Tabla 12. Características y Subcaracterísticas de calidad con una descripción sintética conforme al estándar ISO/IEC 9126(OLSINA)

Característica	Pregunta Central	Subcaracterística	Pregunta Central
Funcionalidad	¿Las funciones y propiedades satisfacen las necesidades	Adecuación	¿Tiene el conjunto de funciones apropiadas para las tareas especificadas?
		Exactitud	¿Hace lo que fue acordado en

	explícitas e implícitas?		forma esperada y correcta?
		Interoperabilidad	¿Interactúa con otros sistemas especificados?
		Conformidad	¿Está de acuerdo con las leyes o normas y estándares, u otras prescripciones?
		Seguridad de acceso	¿Previene accesos no autorizados a los datos y programas?
Confiabilidad	¿Pueden mantener el nivel de rendimiento, bajo ciertas condiciones y por cierto tiempo?	Nivel de Madurez	¿Con qué frecuencia presenta fallas por defectos o errores?
		Tolerancia a fallas	¿Si suceden fallas, cómo se comporta en cuanto al rendimiento especificada?
		Recuperabilidad	¿Es capaz de recuperar datos en caso de fallas?
Usabilidad	¿El software es fácil de usar y de aprender?	Comprensibilidad	¿Es fácil de entender y reconocer la estructura y la lógica y su aplicabilidad?
		Facilidad de aprender	¿Es fácil de aprender a usar?
		Operabilidad	¿Es fácil de operar y controlar?
Eficiencia	¿Es rápido y minimalista en cuanto al uso de recursos, bajo ciertas condiciones?	Comportamiento con respecto al Tiempo	¿Cuál es el tiempo de respuesta y rendimiento en la ejecución de la función?
		Comportamiento con respecto a Recursos	¿Cuántos recursos usa y durante cuánto tiempo?
Mantenibilidad	¿Es fácil de modificar y testear?	Analisabilidad	¿Es fácil diagnosticar una falla o identificar partes a modificar?

		Modificabilidad	¿Es fácil de modificar y adaptar?
		Estabilidad	¿Hay riesgos o efectos inesperados cuando se realizan cambios?
		Testeabilidad	¿Son fáciles de validar las modificaciones?
Portabilidad	¿Es fácil de transferir de un ambiente a otro?	Adaptabilidad	¿Es fácil de adaptar a otros entornos con lo provisto?
		Instalabilidad	¿Es fácil de instalar en el ambiente especificado?
		Conformidad	¿Adhiere a los estándares y convenciones de portabilidad?
		Reemplazabilidad	¿Es fácil de usarlo en lugar de otro software para ese ambiente?

En lo que sigue se especifican cada una de las características y subcaracterísticas de calidad conforme al estándar ISO/IEC 9126, mostrado en la tabla anterior:

2.3.1 Métricas de Funcionalidad

Adecuidad: capacidad del producto software para proporcionar un conjunto apropiado de funciones para tareas y objetivos de los usuarios especificados.

Exactitud: capacidad del producto software para proporcionar los resultados o efectos correctos o acordados, con el grado necesario de precisión.

Interoperabilidad: capacidad del producto software para interactuar con uno o más sistemas especificados.

Seguridad: capacidad del producto software para proteger información y datos de manera que las personas o sistemas no autorizados no puedan leerlos o modificarlos, al tiempo que no se deniega el acceso a las personas o sistemas autorizados.

Cumplimiento funcional: capacidad del producto software para adherirse a normas, convenciones o regulaciones en leyes y prescripciones similares relacionadas con funcionalidad.

2.3.2 Métricas de Fiabilidad

Madurez: capacidad del producto software para evitar fallar como resultado de fallos en el software.

Recuperabilidad: capacidad del producto software para restablecer un nivel de prestaciones especificado y de recuperar los datos directamente afectados en caso de fallo.

Cumplimiento de la fiabilidad: capacidad del producto software para adherirse a normas, convenciones o regulaciones relacionadas con la fiabilidad.

2.3.3 Métricas de Usabilidad

Entendimiento: capacidad del producto software que permite al usuario entender si el software es adecuado y cómo puede ser usado para unas tareas o condiciones de uso particulares.

Aprendizaje: capacidad del producto software que permite al usuario aprender sobre su aplicación.

Operabilidad: capacidad del producto software que permite al usuario operarlo y controlarlo.

Atracción: capacidad del producto software para ser atractivo al usuario.

Cumplimiento de la usabilidad: capacidad del producto software para adherirse a normas, convenciones, guías de estilo o regulaciones relacionadas con la usabilidad.

2.3.4 Métricas de Eficiencia

Comportamiento temporal: capacidad del producto software para proporcionar tiempos de respuesta, tiempos de proceso y potencia apropiados bajo condiciones determinadas.

Utilización de recursos: capacidad del producto software para usar las cantidades y tipos de recursos adecuados cuando el software lleva a cabo su función bajo condiciones determinadas.

Cumplimiento de la eficiencia: capacidad del producto software para adherirse a normas o convenciones relacionadas con la eficiencia.

2.3.5 Métricas de Mantenibilidad

Analizabilidad: capacidad del producto software para serle diagnosticadas deficiencias o causas de los fallos en el software, o para identificar las partes que han de ser modificadas.

Cambiabilidad: capacidad del producto software que permite que una determinada modificación sea implementada.

Estabilidad: capacidad del producto software para evitar efectos inesperados debidos a modificaciones del software.

Examinabilidad: capacidad del producto software que permite que el software modificado sea validado.

Cumplimiento de la mantenibilidad: capacidad del producto software para adherirse a normas o convenciones relacionadas con la mantenibilidad.

2.3.6 Métricas de Portabilidad

Adaptabilidad: capacidad del producto software para ser adaptado a diferentes entornos especificados, sin aplicar acciones o mecanismos distintos de aquellos proporcionados para este propósito por el propio software considerado.

Instalabilidad: capacidad del producto software para ser instalado en un entorno especificado.

Coexistencia: capacidad del producto software para coexistir con otro software independiente, en un entorno común, compartiendo recursos comunes.

Reemplazabilidad: capacidad del producto software para ser usado en lugar de otro producto software, para el mismo propósito, en el mismo entorno.

Cumplimiento de la portabilidad: capacidad del producto software para adherirse a normas o convenciones relacionadas con la portabilidad.

El siguiente epígrafe describe las métricas que constituyen la propuesta para aplicar en los proyectos del departamento Señales Digitales del centro GEySED, teniendo en cuenta las características de las métricas que propone la ISO 25 000. Además se tuvo presente para la selección de estas, que a pesar de no ser específicas para cada una de las funcionalidades que se realizan en los proyectos del departamento, estas se evalúan de forma general. Siguiendo el principio que deben ser métricas entendibles y relativamente fáciles de aplicar, pues como se demostró con la entrevista realizada, en los proyectos actualmente no se tiene conocimiento de que son las métricas, por consiguiente no se aplican. A continuación se describen cada una de las métricas propuestas a partir de sus indicadores, variables y algoritmos matemáticos.

2.7 Propuesta de métricas

2.7.1 Métrica estabilidad de los requerimientos

El objetivo de esta métrica es medir la estabilidad de los requerimientos para asegurar su adecuación antes de pasar al próximo flujo de trabajo. Se considera que los requerimientos son estables cuando no existen adiciones o supresiones en ellos que impliquen modificaciones en las funcionalidades principales de la aplicación. La estabilidad de los requerimientos se calcula como:

ETR = [[RT – RM] / RT] * 100

Donde:

ETR: valor de la estabilidad de los requerimientos.

RT: total de requerimientos definidos.

RM: número de requerimientos modificados, que se obtienen como la sumatoria de los requerimientos insertados, modificados y eliminados.

Esta métrica ofrece valores entre 0 y 100. El mejor valor de ETR es el más cercano a 100 porque mostrará que no se están realizando cambios sobre los requerimientos.

2.7.2 Grado de validación de los requerimientos

Los requerimientos deben ser posibles de validar. La validación de los requerimientos se realiza en consenso del equipo de desarrollo al contrastar lo que desea el cliente con la posibilidad real de implementarlo. El grado de validación de los requerimientos mide la corrección en la definición de los requerimientos. Este valor se calcula como:

VR = nc/ (nc + nnv)

Donde:

VR: grado de validación de los requerimientos.

nc: número de requerimientos que se han validado como correctos.

nnv: número de requisitos no validados aún.

El resultado de esta métrica está siempre entre 0 y 1. El valor óptimo de esta métrica es el más cercano a 1 e indica un alto nivel de corrección en la definición de los requerimientos.

2.7.3 Métricas orientadas a la función

Son medidas indirectas del software y del proceso por el cual se desarrolla. Las métricas orientadas a la función se centran en la funcionalidad o utilidad del programa. Los puntos de función se obtienen utilizando una función empírica basándose en medidas cuantitativas del dominio de información del software y valoraciones subjetivas de la complejidad del software.

Para el cálculo de métricas de punto de función se determinan 5 características del ámbito de la información y los cálculos aparecen en la posición apropiada de la tabla 13. Los valores del ámbito de información están definidos de la siguiente manera:

1. **Número de entrada de usuario**: se cuenta cada entrada del usuario que proporcione al software diferentes datos orientados a la aplicación. Las entradas deben ser distinguidas de las peticiones que se contabilizan por separado.
2. **Número de salida del usuario**: se cuenta cada salida que proporciona al usuario información orientada a la aplicación. En este contexto las salidas se refieren a informes, pantalla, mensajes de error. Los elementos de datos individuales dentro de un informe se encuentran por separado.
3. **Número de peticiones al usuario**: una petición está definida como una entrada interactiva que resulta de la generación de algún tipo de respuesta en forma de salida interactiva. Se cuenta cada petición por separado.

4. **Número de archivos**: se cuenta cada archivo maestro lógico, o sea una agrupación lógica de datos que puede ser una parte en una gran base de datos o un archivo independiente.
5. **Número de interfaces externas**: se cuentan todas las interfaces legibles por la máquina por ejemplo: archivos de datos, en cinta o discos que son utilizados para transmitir información a otro sistema.

Los puntos de función se calculan rellenando la tabla como se muestra en la tabla 13:

Tabla 13. Factores de peso de las medidas en caso de los límites de complejidad superiores e inferiores. (Tomado de Manual de Cálculo de Puntos de Función de la IPFUG, versión 4.1)

Medida	Cuenta	Complejidad Baja	Complejidad Media	Complejidad Alta
Número de entradas del usuario		3	4	6
Número de salidas del usuario		4	5	7
Número de archivos lógicos internos		3	4	6
Número de archivos de interfaz externa		7	10	15
Número de peticiones del usuario		5	7	10

Las Tablas 14, 15 y 16 indican como asignar la complejidad de las medidas descritas anteriormente.

Tabla 14. Evaluación de complejidad de las entradas del usuario. (Tomado de Manual de Cálculo de Puntos de Función de la IPFUG, versión 4.1)

Número de archivos actualizados o referenciados	Número de entradas		
	1-4	5-15	>15
0-1	Baja	Baja	Media
2-3	Baja	Media	Alta
4 o más	Media	Alta	Alta

Tabla 15. Evaluación de la complejidad de las salidas y las consultas del usuario. (Tomado de Manual de Cálculo de Puntos de Función de la IPFUG, versión 4.1)

Número de archivos actualizados o referenciados	Número de salidas o consultas del usuario		
	1-5	6-19	>19
0-1	Baja	Baja	Media
2-3	Baja	Media	Alta
4 o más	Media	Alta	Alta

Tabla 16. Evaluación de la complejidad de los archivos lógicos internos y de los archivos de interfaz externa. (Tomado de Manual de Cálculo de Puntos de Función de la IPFUG, versión 4.1)

Tipos de elementos de registro	Número de archivos internos lógicos y de interfaz externa		
	1-19	20-50	>50
1	Baja	Baja	Media
2- 5	Baja	Media	Alta
6 o más	Media	Alta	Alta

Para calcular los puntos de función sin ajustar se utiliza la siguiente relación.

PF = SUM(wij) * SUM(xij)

Donde:

PF: puntos de función sin ajustar.

wij: factores de peso de los 5 componentes por nivel de complejidad (alto, medio, bajo).

xij: valores de cada componente en la aplicación.

Las características del software que deben tenerse en cuenta para evaluar esta métrica son: la comunicación de los datos, las funciones distribuidas, el desempeño del programa, la configuración utilizada en exceso, la tasa de transacción, la entrada de datos en línea, la eficiencia para el usuario final, la actualización en línea, la complejidad de procesamiento, la reusabilidad, la facilidad de instalación, la facilidad de operación, la existencia de sitios

múltiples y la facilidad de cambios. La descripción de estas características se muestra en la Tabla 17.

Tabla 17. Características del software a evaluar para calcular los puntos de función

Característica	Descripción
Comunicación de los datos	¿Qué influencia tendrán los medios de comunicación que se necesitará para la transacción de datos entre la aplicación y otros sistemas?
Procesamiento distribuido	¿Cómo se manipulan los datos y las funciones distribuidas?
Desempeño	¿Es requerido por el usuario un tiempo de respuesta determinado?
Configuración utilizada en exceso	¿Cuán dependiente es la aplicación de la plataforma de hardware sobre la que se ejecuta?
Tasa de transacción	¿Cuál es la frecuencia de ejecución de las transacciones en la aplicación?
Entrada de datos en línea	¿Qué porcentaje de la información se introduce en línea?
Eficiencia para el usuario	¿La aplicación fue diseñada con para proveer la máxima eficiencia al usuario final?
Actualización en línea	¿Cuántos archivos internos se actualizan mediante transacciones en línea?
Procesamiento complejo	¿La aplicación tiene procesamiento lógico o matemático complejo?
Reusabilidad	¿El código del sistema puede utilizarse en otras aplicaciones?
Facilidad de instalación	¿Cuánta dificultad implica la instalación y la adaptación a diferentes plataformas de software?
Facilidad de operación	¿Es intuitivo el uso de la aplicación para el usuario final?
Sitios múltiples	¿Es posible utilizar la aplicación en diferentes

	organizaciones con cambios mínimos?
Facilidad de cambios	¿Es fácil realizar cambios sobre la aplicación?

Los valores de ajuste de complejidad son obtenidos de la asignación de pesos, donde se otorga un puntaje de 0 a 5 a las características del sistema mencionadas anteriormente. Estos valores se evalúan de acuerdo a la Tabla 18.

Tabla 18. Valores de ajuste de complejidad

.**Valor**	0	1	2	3	4	5
Evaluación	Sin influencia	Incidental	Moderado	Medio	Significativo	Esencial

El factor de ajuste del valor (FAV) indica la funcionalidad general que la aplicación provee al usuario final. Además permite ajustar los puntos de función con una aproximación de +/- 35%:

FAV = [0.65 +0.01 * SUM(fi)]

Donde:

FAV: factor de ajuste del valor.

fi: donde i puede ser de uno hasta 14 los valores de ajuste de complejidad basados en las respuestas a las cuestiones señaladas de la siguiente tabla.

Por último, el valor de los puntos de función ajustados (PFA) se obtiene multiplicando los puntos de fusión sin ajustar y el factor de ajuste de valores:

PFA = PF * FAV

Donde:

PFA: puntos de función ajustados.

Una vez calculado los puntos de función ajustados se usan de forma analógica como medida de la productividad, calidad y otros productos del software.

Productividad = PFA / persona-mes

Calidad = Errores / PFA

Costo = Dólares / PFA

Documentación = Págs. Doc / PFA

2.7.4 Métrica para el control de pruebas de unidad

La métrica para el control de pruebas de unidad ofrece una medida de los componentes que se han probado individualmente antes de la integración con respecto al total de

componentes que fueron implementados. La métrica para el control de pruebas de unidad se calcula como:

CPU =NCP / Clmp

Donde:

CPU: valor de la métrica para el control de pruebas de unidad.

NCP: número de componentes probados individualmente antes de integrarlos.

Clmp: número de componentes implementados.

Esta métrica ofrece valores entre 0 y 1. Si CPU = 1 significa que todos los componentes implementados fueron probados individualmente antes de la integración.

2.7.5 Densidad de defectos

La densidad de defectos ofrece una medida sobre la proporción de defectos con respecto a la cantidad de elementos de especificación. Esta métrica permite realizar análisis estadísticos al finalizar las pruebas para valorar la integridad y madurez del software analizado. La densidad de defectos se calcula como:

DD = TD / CER

Donde:

DD: densidad de defectos.

TD: número total de defectos encontrados durante las pruebas.

CER: número de elementos de especificación revisados.

Es recomendable para una alta calidad del software que la densidad de defectos tenga un valor mínimo.

2.7.6Tasa de propagación de defectos

La corrección de defectos en el software puede originar nuevos defectos, este fenómeno es conocido como propagación de defectos. La tasa de propagación de defectos indica el comportamiento de la propagación de defectos en la realización de las pruebas. La tasa de propagación de defectos se calcula como:

TPD = 1 – Dn / Dc

Donde:

TPD: tasa de propagación de defectos

Dn: número de defectos ocasionados al corregir defectos anteriores.

Dc: número de defectos corregidos.

El valor de TPD ofrece mejores resultados cuando está más cerca de 1. Al calcular esta métrica pueden obtenerse valores negativos, cuando el número de defectos ocasionados al corregir defectos es mayor que el número de defectos corregidos, en este caso dichos valores no se tendrán en cuenta al trabajar con el resultado de la métrica, sino que se considerarán como cero.

2.7.6Métrica de madurez del software

El Índice de Madurez del Software (IMS), propuesto por el estándar IEEE 982.1-1988, proporciona un indicador de la estabilidad del software basado en los cambios que ocurren en cada versión del producto, este es un indicador de la facilidad de mantenimiento del software. El IMS se calcula como:

IMS = [Mt –(Fc + Fa +Fe)] / Mt

Donde:

IMS: índice de madurez del software.

Mt: número de módulos en la versión actual.

Fc: número de módulos en la versión actual que han sido modificados.

Fa: número de módulos en la versión actual que han sido añadidos.

Fe: número de módulos en la versión actual que han sido eliminados.

El valor del IMS está siempre entre 0 y 1. Mientras más cerca esté el valor del IMS de 1, más estable será el producto software.

2.7.7 Métrica de éxito del software

Esta métrica se basa en registrar el porcentaje de usuarios que lograron realizar exitosamente la prueba que se les solicitó. A cada funcionalidad se le asigna un peso según su estado de realización como se indica en la Tabla 19.

Tabla 19. Estado de realización de la funcionalidad

Estado de realización de la funcionalidad	Peso
Funcionalidad no realizada	0
Funcionalidad a medio realizar	0.5

Funcionalidad realizada	1

La métrica de éxito se calcula como:

ME= [CFT + [CFM * 0.5]] * 100 / CTF

Donde:

ME: valor de la métrica de éxito.

CFT: cantidad de funcionalidades terminadas.

CFM: cantidad de funcionalidades a medio terminar.

CTF: número total de funcionalidades.

Mientras mayor sea el valor de ME, mayor usabilidad tendrá la aplicación web. Esto implica una mayor aceptación por parte del usuario.

2.8 Conclusiones

En el capítulo anterior se realizó un análisis de las métricas existentes y su clasificación, lo cual fue de suma importancia a la hora de escoger las métricas a aplicar. Se escogió el estándar ISO 2500, el cual rigió a partir de este momento la investigación. Finalmente se escogieron las métricas: estabilidad de requerimientos (ETR), grado de validación de requerimientos, métrica basada en puntos de función (PF), métrica para el control de pruebas de unidad, densidad de defectos, tasa de propagación de defectos, métrica de éxito, índice de madurez, como propuestas para aplicar en los proyectos del departamento Señales Digitales del centro GEySED.

CAPÍTULO 3

Resultados

3.1 Introducción

En el siguiente capítulo se expone el resultado de la aplicación de las métricas escogidas en el capítulo 2 al proyecto Primicia. Para la selección de este proyecto se tuvo en cuenta que este proyecto ya liberó una primera versión que es "Señal ACN", a esta versión se le aplicaron las métricas escogidas. Además se escogió otra versión del mismo proyecto que aún está en desarrollo, con el objetivo de comparar el resultado obtenido al aplicar las métricas a cada versión.

3.2 Aplicación de la guía de métricas a PRIMICIA

La validación de la guía de métricas propuestas se realiza mediante esta primera aplicación en el entorno de desarrollo del proyecto PRIMICIA del departamento de Señales Digitales del centro GEySED. Por tanto, la propuesta estará sujeta a los cambios que se estimen convenientes al concluir su aplicación.

3.2.1Versión Señal ACN

3.2.1.1 Recolección de los datos de Señal ACN

El proceso de recolección de datos se realizó mediante entrevistas al líder del proyecto PRIMICIA para obtener las variables independientes que permiten calcular las métricas necesarias para la evaluación del proyecto PRIMICIA del Departamento Señales Digitales del centro GEySED. Algunas de las variables independientes necesarias para calcular las métricas formuladas se presentan en la Tabla 20. La elaboración de esta guía de métricas se realizó una vez liberada esta versión del producto.

Tabla 20. Medidas directas tomadas de la versión de Señal ACN del proyecto PRIMICIA

Indicadores	Valores
Número total de requerimientos (RT)	37

Número de requerimientos modificados (RM)	32
Número de requerimientos validados correctos (n_c)	37
Número de requerimientos aun sin validar (n_{nv})	0
Número de entradas del usuario	12
Número de salidas del usuario	3
Número de archivos lógicos internos	8
Número de archivos de interfaz externa	6
Número de peticiones del usuario	28
Número de componentes implementados	6
Número de componentes probados	5
Número de defectos encontrados	22
Número de elementos revisados	41
Número de defectos corregidos	22
Número de defectos encontrados al corregir defectos anteriores	0
Número de módulos actuales	6
Número de módulos modificados	6
Número de módulos añadidos	5
Número de módulos eliminados	0
Número de funcionalidades terminadas (CFT)	41
Número de funcionalidades a medio terminar (CFM)	0
Número total de funcionalidades (CTF)	41

Luego de haber identificado y recolectado el valor de los indicadores necesarios en la versión “Señal ACN” del proyecto PRIMICIA se procede a aplicar las métricas

seleccionadas. A continuación se pueden apreciar los resultados que se obtuvieron de cada una de las métricas propuestas.

Métrica 1. Estabilidad de los requerimientos

ETR = [[RT – RM] / RT] * 100

ETR = [[37 – 32] / 37] * 100

ETR = [5 / 37] * 100

ETR = 0.14 * 100

ETR = 14

Métrica 2. Grado de validación de los requerimientos

VR = nc/ (nc + nnv)

VR = 37 / (37 + 0)

VR = 37 / 37

VR = 1

Métrica 3. Métricas orientadas a la función

La siguiente tabla recoge los indicadores necesarios para calcular los puntos de función.

Tabla 21. Factores de peso de las medidas en caso de los límites de complejidad superiores e inferiores

Medida	Cuenta	Complejidad Baja	Complejidad Media	Complejidad Alta
Número de entradas del usuario	12	3	4	6
Número de salidas del usuario	3	4	5	7
Número de archivos lógicos internos	8	3	4	6
Número de archivos de interfaz externa	6	7	10	15
Número de peticiones del usuario	28	5	7	10

Para poder hallar los puntos de función ajustados de una aplicación se deben calcular los puntos de función sin ajustar, que se realiza como se muestra en la siguiente fórmula:

PF = SUM(wij) * SUM(xij)

PF = (12 + 3 + 8 + 6 + 28) * (3 + 4 + 3 + 7 + 5)

PF = 57 * 22

PF = 1254.

Otro aspecto a tener en cuenta es el factor de ajuste del valor, por consiguiente se calcula según se indica a continuación.

FAV = [0.65 +0.01 * SUM(fi)]

Luego se dio valor de ajuste de complejidad a las características señaladas en la siguiente tabla.

Tabla 22. Valores de ajuste de complejidad asignados a cada característica

Característica	Valor
Comunicación de los datos	4
Procesamiento distribuido	4
Desempeño	3
Configuración utilizada en exceso	5
Tasa de transacción	3
Entrada de datos en línea	4
Eficiencia para el usuario	4
Actualización en línea	4
Procesamiento complejo	3
Reusabilidad	4
Facilidad de instalación	3
Facilidad de operación	4
Sitios múltiples	3
Facilidad de cambios	3
Suma de valores de la columna 2	51

Al aplicar la fórmula se obtiene:

FAV = [0.65 + 0.01 * 51]

FAV = [0.65 + 0.51]

FAV = 1.16

Finalmente se puede calcular los puntos de función ajustados (PFA) mediante la fórmula siguiente:

PFA = PF * FVA

PFA = 1254 * 1.16

PFA = 1454.64

Métrica 4. Control de pruebas de unidad

Se tuvo como resultado que los componentes probados individualmente antes de integrarlos fueron 5 y que en total se implementaron 6 componentes. Por esta razón se obtiene que:

CPU =NCP / Clmp

CPU = 5 / 6

CPU = 0.83

Métrica 5. Densidad de defectos

Se tiene que el total de defectos encontrados durante las pruebas fueron 22 y que el número de elementos de especificación revisados es 41. Dando lugar a que se obtengan los siguientes resultados:

DD = TD / CER

DD = 22 / 41

DD = 0.54

Métrica 6. Tasa de Propagación de defectos

Al verificar estos aspectos se obtuvo lo siguiente: no hubo defectos ocasionados al corregir los 22 defectos detectados.

TPD = 1 – Dn / Dc

TPD = 1 – 0 / 22

TPD = 1 – 0

TPD = 1

Métrica 7. Índice de Madurez del Software (IMS)

Según los valores de la tabla 20, se obtiene el siguiente resultado:

IMS = [Mt – (Fc + Fa +Fe)] / Mt

IMS = [6 – (6 + 0 + 0)] / 6

IMS = [6 – 6] / 6

IMS = 0 / 6

IMS = 0

Métrica8. Métrica de éxito

ME= [CFT + [CFM * 0.5]] * 100 / CTF

ME = [41 + [0 * 0.5]] * 100 / 41

ME = [41 + 0] * 100 / 13

ME = 41* 100 / 13

ME = 4100 / 41

ME = 100

3.2.1.2 Recolección de los datos del Subsistema de Configuración de PRIMICIA.

El proceso de recolección de datos se realizó mediante entrevistas al líder del proyecto para obtener las variables independientes que permiten calcular las métricas necesarias para la evaluación del proyecto PRIMICIA del departamento Señales Digitales del centro GEySED. Algunas de las variables independientes necesarias para calcular las métricas formuladas se presentan en la Tabla 23. La elaboración de esta guía de métricas se realizó sin haber sido liberada aún esta versión del producto, por tanto los datos de los requerimientos del análisis y diseño se obtuvieron durante el flujo de trabajo de implementación. Además esto implica que por el límite de tiempo para entregar los resultados de la investigación no es posible registrar los datos necesarios para aplicar las métricas durante la etapa de pruebas.

Tabla 23. Medidas directas tomadas del Subsistema de configuración de PRIMICIA

Indicadores	Valores
Número total de requerimientos (RT)	11
Número de requerimientos modificados (RM)	3
Número de requerimientos validados correctos (n_c)	11
Número de requerimientos aun sin validar (n_{nv})	0

Número de entradas del usuario	14
Número de salidas del usuario	8
Número de archivos lógicos internos	12
Número de archivos de interfaz externa	7
Número de peticiones del usuario	32
Número de componentes implementados	6
Número de componentes probados	-
Número de defectos encontrados	-
Número de elementos revisados	-
Número de defectos corregidos	-
Número de defectos encontrados al corregir defectos anteriores	-
Número de módulos actuales	6
Número de módulos modificados	0
Número de módulos añadidos	1
Número de módulos eliminados	1
Número de funcionalidades terminadas (CFT)	11
Número de funcionalidades a medio terminar (CFM)	2
Número total de funcionalidades (CTF)	13

Luego de haber identificado y recolectado el valor de los indicadores necesarios en la versión “Señal ACN” del proyecto Primicia se procede a aplicar las métricas seleccionadas. A continuación se pueden apreciar los resultados que se obtuvieron de cada una de las métricas propuestas.

Métrica 1. Estabilidad de los requerimientos

ETR = [[RT – RM] / RT] * 100

ETR = [[11 – 3] / 11] * 100

ETR = [8 / 11] * 100

ETR = 0.72 * 100

ETR = 72

Métrica 2. Grado de validación de los requerimientos

VR = nc/ (nc + nnv)

VR = 11 / (11 + 0)

VR = 11 / 11

VR = 1

Métrica 3. Métricas orientadas a la función

La siguiente tabla recoge los indicadores necesarios para calcular los puntos de función.

Tabla 24. Factores de peso de las medidas en caso de los límites de complejidad superiores e inferiores

Medida	Cuenta	Complejidad Baja	Complejidad Media	Complejidad Alta
Número de entradas del usuario	14	3	4	6
Número de salidas del usuario	8	4	5	7
Número de archivos lógicos internos	12	3	4	6
Número de archivos de interfaz externa	7	7	10	15
Número de peticiones del usuario	32	5	7	10

Para poder hallar los puntos de función ajustados de una aplicación se deben calcular los puntos de función sin ajustar, que se realiza como se muestra en la siguiente fórmula:

PF = SUM(wij) * SUM(xij)

PF = (14 + 8 + 12 + 7 + 32) * (3 + 4 + 3 + 7 + 5)

PF = 71 * 22

PF = 1562

Otro aspecto a tener en cuenta es el factor de ajuste del valor, por consiguiente se calcula según se indica a continuación.

FAV = [0.65 +0.01 * SUM(fi)]

Tabla 25. Valores de ajuste de complejidad asignados a cada característica

Característica	Valor
Comunicación de los datos	4
Procesamiento distribuido	3
Desempeño	4
Configuración utilizada en exceso	3
Tasa de transacción	3
Entrada de datos en línea	4
Eficiencia para el usuario	3
Actualización en línea	3
Procesamiento complejo	3
Reusabilidad	3
Facilidad de instalación	4
Facilidad de operación	3
Sitios múltiples	3
Facilidad de cambios	3
Suma de valores de la columna 2	46

Al aplicar la fórmula se obtiene:

FAV = [0.65 + 0.01 * 46]

FAV = [0.65 + 0.46]

FAV = 1.11

Finalmente se puede calcular los puntos de función ajustados (PFA) mediante la fórmula siguiente:

PFA = PF * FVA

PFA = 1562 * 1.11

PFA = 1733.82

Métrica 7. Índice de Madurez del Software (IMS)

Según los valores de la tabla 20, se obtiene el siguiente resultado:

IMS = [Mt – (Fc + Fa +Fe)] / Mt

IMS = [6 – (0 + 1 + 1)] / 6

IMS = [6 – 2] / 6

IMS = 4 / 6

IMS = 0.66

Métrica 8. Métrica de éxito

ME= [CFT + [CFM * 0.5]] * 100 / CTF

ME = [11 + [2 * 0.5]] * 100 / 13

ME = [11 + 1] * 100 / 13

ME = 12 * 100 / 13

ME = 1200 / 13

ME = 92.3

Teniendo en cuenta que el Subsistema de Configuración del proyecto PRIMICIA aún está en fase de desarrollo, y que actualmente no cuenta con una documentación que arroje los resultados de las revisiones realizadas hasta el momento para determinar indicadores como el número de componentes probados individualmente antes de integrarlos, el número de defectos encontrados, el número de elementos revisados, el número de defectos corregidos y el número de defectos encontrados al corregir defectos anteriores, las métricas de control de pruebas de unidad, densidad de defectos y tasa de propagación de defectos, no se pudieron aplicar a esta versión.

En la siguiente tabla se muestran agrupados los resultados de la aplicación de las métricas de ambas versiones.

Tabla 26. Resultados de las métricas aplicadas al Subsistema de Configuración y a Señal ACN

Métrica	Resultados ACN	Resultados Sub.Configuración
Estabilidad de requerimientos (ETR)	14 %	72%
Grado de validación de requerimientos (Q_3)	1	1
Métrica basada en puntos de función (PF)	1454.64	1733.82

Control de prueba	0.83	-
Densidad de defectos	0.54	-
Propagación de defectos	1	-
Métrica de éxito	100 %	92.3%
Índice de madurez	0 %	0.66%

La Tabla 27 representa la escala según la cual se evaluaron los resultados de las métricas aplicadas al Subsistema de Configuración y a Señal ACN del proyecto PRIMICIA.

Tabla 27. Escala de evaluación de los resultados de las métricas propuestas

Nombre de la métrica	Rango	Evaluación de los resultados
Estabilidad de los requerimientos	0 ≤ ETR ≤ 100	Caso peor (0 ≤ ETR < 50): si el resultado de la métrica se encuentra en este rango no se puede empezar el análisis y diseño hasta que no se estabilicen los requerimientos y se obtenga un mejor valor en esta métrica porque implica mayor costo de reparación de errores.
		Caso medio (50 ≤ ETR < 75): si el valor está en este intervalo se puede trabajar el análisis y diseño sobre los requerimientos, pero no son confiables y se debe prestar especial atención en caso de aparición de riesgos posteriores.
		Caso óptimo (75 ≤ ETR < 100): este valor indica que los requerimientos son estables, por tanto se puede iniciar el análisis y diseño sobre ellos, sin riesgo de cambios mayores para el desarrollo posterior del producto.
Grado de validación de los requerimientos	0 ≤ VR ≤ 1	Caso peor (0 ≤ VR < 0. 50): si el resultado está en este rango significa que deben modificarse o levantarse nuevos requerimientos porque los existentes no son posibles de implementar o no cumplen lo deseado por el cliente.

		Caso medio (0. 50 ≤ VR < 0.75): un resultado de este tipo indica que aún existen requerimientos que deben ser redefinidos, pero la mayoría se pueden aplicar al análisis y diseño.
		Caso óptimo (0.75 ≤ VR < 1): este valor indica que todos los requerimientos están correctamente validados y listos para emplear en el análisis y diseño de la aplicación.
Métrica basada en puntos de función	PF > 0	Los resultados obtenidos por esta métrica no tienen definido un límite superior, no obstante es recomendable que el valor de esta métrica sea el más alto posible porque indica un mayor nivel de funcionalidad entregada al usuario final.
Métrica para el control de pruebas de unidad	0 ≤ CPU ≤ 1	Caso peor (0 ≤ CPU < 1): un valor dentro de este rango indica que faltan componentes por probar antes de realizar la integración del software, por tanto es preciso diseñar estas pruebas antes de realizar la integración.
		Caso óptimo (CPU = 1): este valor indica que todos los componentes fueron probados antes de su integración.
Densidad de defectos	DD > 0	El valor de esta métrica no posee un límite superior definido, pero es recomendable que su valor sea el menor posible.
Tasa de propagación de defectos 0 ≤ TPD ≤ 1		Caso peor (0 ≤ TPD < 0.5): un resultado dentro de este rango indica que la tasa de propagación de defectos es bastante alta por tanto el software posee una pobre capacidad de recuperación e implica un mayor esfuerzo de pruebas.
		Caso medio (0.5 ≤ TPD < 0.75): un valor en este rango indica que la tasa de propagación de defectos es aceptable, por tanto no requerirá de un esfuerzo de pruebas adicional muy grande.

	Caso óptimo (0.75 ≤ TPD ≤ 1): este es el mejor resultado de la métrica pues no implica esfuerzo de pruebas adicional para los ingenieros de software.
Índice de Madurez del Software 0 ≤ IMS ≤1	Caso peor (0 ≤ IMS < 0.5): Un valor dentro de este rango indica una pobre madurez del software y por tanto afecta negativamente la facilidad de mantenimiento del mismo.
	Caso medio (0.5 ≤ IMS < 0.75): Un resultado dentro de este intervalo indica que la madurez del software es aceptable por tanto la facilidad de mantenimiento es relativamente simple.
	Caso óptimo (0.75 ≤ IMS ≤ 1): Este valor indica que la madurez del software excelente lo que implica que el software es fácil de mantener.
Métrica de éxito ME > 0	Es recomendable que el resultado de esta métrica sea el más alto posible para garantizar la usabilidad del programa.

3.2.2Resultados obtenidos.

A continuación se expone la interpretación de los resultados obtenidos de la aplicación de la propuesta de métricas al Subsistema de Configuración y a Señal ACN.

Tabla 28. Interpretación de los resultados de la aplicación de la propuesta de métricas a Señal ACN y al Subsistema de Configuración

Proyecto	Interpretación de resultados
Señal ACN	1. La estabilidad de los requerimientos se encuentra en la escala de 0 al 50 % con un valor del 14 %, lo cual indica que no se puede trabajar en el análisis hasta que no se estabilicen los requerimientos y se obtenga un mejor valor en esta métrica porque implica mayor costo de reparación

	de errores. 2. El resultado del grado de validación de requerimientos es 1 lo que indica que todos los requerimientos están correctamente validados y listos para emplear en el análisis y diseño de la aplicación. 3. Métrica basada en puntos de función el resultado es 1454.64indica un alto nivel de funcionalidad para el usuario, sin embargo este valor puede ser mayor debido a que esta métrica no tienen definido un límite superior, por lo que es recomendable que el valor de esta métrica sea el más alto posible. 4. Métrica para el control de pruebas de unidad el resultado es 0.83 se encuentra en el rango de (0 ≤ CPU < 1) indica que faltan componentes por probar antes de realizar la integración del software, por tanto es preciso diseñar estas pruebas antes de realizar la integración. 5. Métrica densidad de defecto con un valor de 0.54, esta métrica no posee un límite superior definido, pero es recomendable que su valor sea el menor posible. 6. Métrica tasa de propagación de defectos con un resultado de 1 se encuentra en el rango de (0.75 ≤ TPD ≤ 1): este es el mejor resultado de la métrica pues no implica esfuerzo de pruebas adicional para los ingenieros de software. 7. Métrica de éxito con un valor de un 92,3 % indica que nivel de usabilidad de la aplicación es bastante alto, pero puede ser mejor. 8. Índice de madurez con un resultado de 0.66 % se encuentra en el rango del 50 al 75% indica que la madurez del software es aceptable, por tanto la facilidad de mantenimiento es relativamente simple.
Subsistema de Configuración	1. La estabilidad de los requerimientos se encuentra en la escala de 50 al 75 % con un valor del 72 %, lo cual indica

	que se puede trabajar en el análisis y diseño sobre los requerimientos, pero no son del todo confiables y se debe prestar especial atención en caso de aparición de riesgos posteriores. 2. El resultado del grado de validación de requerimientos es 1 lo que indica que todos los requerimientos están correctamente validados y listos para emplear en el análisis y diseño de la aplicación. 3. Métrica basada en puntos de función el resultado es 1733.82indica un alto nivel de funcionalidad para el usuario, sin embargo este valor puede ser mayor debido a que esta métrica no tienen definido un límite superior, por lo que es recomendable que el valor de esta métrica sea el más alto posible. 4. Métrica de éxito con un valor de un 92,3 % indica que nivel de usabilidad de la aplicación es bastante alto, pero puede ser mejor. 5. Índice de madurez con un resultado de 0.66 % se encuentra en el rango del 50 al 75% indica que la madurez del software es aceptable, por tanto la facilidad de mantenimiento es relativamente simple.

3.3 Valoración de las métricas y los resultados obtenidos.

Los resultados de la aplicación de las métricas propuestas para la evaluación del software orientado al procesamiento y gestión de videos e imágenes digitales del Departamento de Señales Digitales del centro GEySED, al Subsistema de Configuración de PRIMICIA permitió identificar que las métricas aplicadas garantizan el perfeccionamiento y la calidad del producto, ya que reflejan resultados satisfactorios cercanos a la realidad del estado actual en que se encuentra el proyecto, estos resultados indican que el nivel de calidad del producto es bastante bueno pero aún no está al 100 %. Las métricas aplicadas no son complejas, se entiende correctamente por qué y para qué se utilizan, así como los objetivos que persiguen, los cálculos no producen resultados ambiguos, arrojan resultados precisos, y

la interpretación de los valores obtenidos está acorde a las nociones intuitivas del ingeniero de software. Lo que permite corregir los problemas encontrados para evitar costos adicionales en tiempo y esfuerzo del equipo de desarrollo.

3.4 Conclusiones

La aplicación de la guía de métricas al Subsistema de Configuración y a Señal ACN del proyecto PRIMICIA permitió valorar la funcionalidad, la facilidad de uso, la confiabilidad, la facilidad de mantenimiento y la facilidad de prueba del software mostrando que ambas aplicaciones hasta el momento tienen un alto grado de funcionalidad. El análisis de los resultados obtenidos mediante la aplicación de la guía de métricas a al Subsistema de Configuración y a Señal ACN, permitió detectar algunos problemas en su desarrollo como, errores en el análisis y diseño de la aplicación Señal ACN debido al grado de inestabilidad de los requerimientos y una pobre madures del software como resultado del número de módulos modificados, por otro lado el Subsistema de Configuración, no cuenta con una documentación que arroje los resultados de las revisiones realizadas hasta el momento para determinar algunos indicadores como el número de componentes probados individualmente antes de integrarlos, el número de defectos encontrados, el número de elementos revisados, el número de defectos corregidos y el número de defectos encontrados al corregir defectos anteriores, por lo que no se pudo aplicar las métricas de control de pruebas de unidad, densidad de defectos y tasa de propagación de defectos.

Conclusiones

Al finalizar esta investigación se concluye que:

- A pesar de que en la universidad se está aplicando actualmente el modelo de CMMI y se quiere alcanzar el nivel 2 del mismo, en este nivel solo se indica que se deben aplicar métricas para la evaluación del producto, por tanto los proyectos del departamento actualmente no cuentan con métricas definidas.

- Las métricas que se seleccionaron para la propuesta están basadas en los factores de calidad, tomados de la norma ISO 25000. Esta proporciona una guía para el uso de nuevos estándares internacionales, llamados Requisitos y Evaluación de Calidad de Productos de Software (SQuaRE). Lo cual constituye una serie de normas basadas en la ISO 9126 y en la ISO 14598 (Evaluación del Software), y su objetivo principal es guiar el desarrollo de los productos de software con la especificación y evaluación de requisitos de calidad.

- Se identificó un total desconocimiento sobre las métricas para la evaluación de software por los miembros del proyectos del departamento Señales Digitales del centro GEySED, producto a esto se detectó que no se aplican métricas en ningún proyecto del departamento.

- Se definió una guía de métricas para la evaluación del software orientado al procesamiento y gestión de videos e imágenes digitales del Departamento de Señales Digitales del centro GEySED, en ella se incluyen las métricas de estabilidad, y grado de validación de los requerimientos, puntos de función, métricas de control de pruebas de unidad, métrica de éxito y métrica de madurez del software.

- Al aplicar las métricas en el Subsistema de Configuración de PRIMICIA se obtuvieron resultados que indican que el proyecto se encuentra en buen estado, pero deben seguir perfeccionado el producto para alcanzar la máxima calidad del mismo.

- Las recomendaciones para la solución de los problemas detectados fueron transmitidas al equipo de desarrollo para ser efectuadas en próximas iteraciones.

Recomendaciones

- Aplicar la propuesta de métricas durante cada iteración de las fases de desarrollo de los proyectos que se desarrollan en el Departamento de Señales Digitales del centro GEySED.
- Capacitar al personal del Departamento de Señales Digitales del centro GEySED respecto al trabajo con métricas de calidad para garantizar su aplicación en los proyectos de Investigación, Desarrollo e Innovación (I+D+i) que se desarrollan en el departamento.
- Realizar registros de los resultados de la aplicación de las métricas en el repositorio del proyecto.
- Continuar la investigación para incluir nuevas métricas de proyecto y proceso en la propuesta realizada, para mejorar la evaluación de los proyectos que se desarrollan en el Departamento de Señales Digitales del centro GEySED, además de valorar otros factores de calidad del producto que sea necesario medir.
- Informatizar la recolección de los datos y el cálculo de las métricas propuestas.

Bibliografía

Bieman, James M., et al. 1996. Fundamental Issues in Software Measurements. [Online] 1996http://www.cs.colostate.edu/~bieman/Pubs/fundIssues96.pdf.

Briand, Lionel, Morasca, Sandro and Basili, Victor R. 1994. Goal-driven definition of product metrics based on properties. [Online] 1994. http://www2.umassd.edu:16080/SWPI/eseg/cs-tr-3346.pdf.

Burby, Jason and Brown, Angie. 2007. Web Analytics Definitions – Version 4.0. [Online] 08 16, 2007. http://www.dcc.ufla.br/infocomp/artigos/v4.3/art02.pdf.

Choque, Guillermo. 2002. *Instituto Blaise Pascal.* [Online] 12 2002. http://www.institutopascal.edu.ar/Alumnos/AsignaturasAnalistasDeSistemas/1ano/ArquitecturaDelComputador/IngenieriaSoft.pdf.

Classifying web metrics using the web quality model. **Calero, Coral, Ruiz, Julián and Piattini, Mario. 2005.** 3, Ciudad Real, España : Emerald Group Publishing Limited, 04 06, 2005, Vol. 29. 1468-4527.

CMMI v2. **Carnegie Mellon University. 2006.** EEUU : s.n., 2006. 082506.

2007. Cognitive Limits of Software Cost Estimation. [Online] 2007. http://paginaspersonales.deusto.es/cortazar/doctorado/articulos/2007/4.ESEM07CostEstimation_PsycoCognitiveLimits.pdf.

Coz, Norbeys. 2008. Propuesta de un Estándar y Métricas para el control del Proyecto Productivo Informatización del Convenio Integral Cuba-Venezuela. *Biblioteca UCI.* [Online] 06 2008. [Cited: 02 19, 2009.] http://bibliodoc.uci.cu/TD/TD_1341_08.pdf . MIS-007399.

Crow, Kenneth. 2001. Product development metrics list. *npd-solutions.* [Online] DRM_Associates, 2001 http://www.npd-solutions.com/metrics.html.

Damm, Lars-Ola and Lundberg, Lars. 2007. Company-wide Implementation of Metrics for Early Software Fault Detection. [Online] 2007. http://paginaspersonales.deusto.es/cortazar/doctorado/articulos/2007/1.MetricsForFaultDetectionICSE07.pdf.

Dávila-Nicanor, Leticia and Mejía-Álvarez, Pedro. 2004. Reliability Improvement of Web-Based Software Applications. *DEPARTAMENTO DE COMPUTACIÓN CINESTAV.* [Online] 2004. http://delta.cs.cinvestav.mx/~pmejia/qsic2004.pdf. **Bibliografía 82**

BIBLIOGRAFÍA

Documento interno GSIG. 2008. *Especificación de Requisitos del SIG-UCI v1.2(Documento interno).* 2008.

—. 2008. *Lista de Riesgos SIG-UCI v1.0 (Documento interno).* 2008.

Dynamic Coupling Measurement for Object-Oriented Software. **Arisholm, Eric, Briand, Lionel C. and Foyer, Audun. 2004.** 8, Lysaker, Noruega : IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 2004, Vol. 30. 0098-5589.

Ebert, Christof. 2002. Metrics for Identifying Critical Components in Software Projects. [Online] 2002. http://paginaspersonales.deusto.es/cortazar/doctorado/articulos/ebert-handbook.pdf.

Emam, Khaled. 2000. *A methodology for validating software product metrics.* s.l. : National Research Council Canada, 2000. ERB-1076.

Empirical Analysis of Object-Oriented Design Metrics for Predicting High and Low Severity Faults. **Zhou, Yuming and Leung, Hareton. 10/2006.** 10, s.l. : IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 10/2006, Vol. 32.

English, Michael, Buckley, Jim and Cahill, Tony. 2007. Fine-Grained Software Metrics in Practice. [Online] 2007. http://paginaspersonales.deusto.es/cortazar/doctorado/articulos/2007/5.ESEM07-MetricsforFriendConstruct.pdf. 0-7695-2886-4.

Febles, Eilyn, Álvarez, Sofía and Fernández, Humberto. 2000. Calidad de software y la empresa, enseñanza de un tema imprescindible para el Ingeniero Informático. *Sociedad Mexicana de Computación en Educación - SOMECE.* [Online] 2000. http://www.somece.org.mx/memorias/2000/docs/123.DOC.

Febles, Eilyn, Piñeiro, Pedro Y. and Fernández, Yamilis. 2003. Diseño de un segundo perfil de calidad de software para graduados de Ingeniería Informática. *Informática 2003-X Convención y Feria Internacional.* [Online] 2003. http://www.informaticahabana.co.cu/evento_virtual/files/CAL048.doc.

Fenton, Norman. 1994. Software Measurement: A Necessary Scientific Basis. *IEEE.* [Online] 03 1994. http://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=32.

Fenton, Norman, Krause, Paul and Neil, Martin. 2002. Software Measurement: Uncertainty and Causal Modelling. [Online] 2002http://paginaspersonales.deusto.es/cortazar/doctorado/articulos/FentonKrauseNeil_IEESoftware.pdf. **Bibliografía 83**

BIBLIOGRAFÍA

Fernández Aedo, Raúl. 1998. *Los métodos de evaluación de expertos para valorar resultados de las investigaciones.* La Habana, Cuba : s.n., 1998.

Fernández, Luis. 1998. Teoría de la medición. *Campus Gipuzkoa.* [Online] 11 30, 1998. http://www.sc.ehu.es/jiwdocoj/remis/docs/teoriamedicion.html.

Fuente, Aquilino Juan. 1996. *Necesidad de sistemas formales de métricas para proyectos OO.* Oviedo : Universidad, 1996.

García, Annie and Almaguer, Adriana. 2008. Propuesta de métricas para la evaluación de Requisitos No Funcionales. *Biblioteca UCI.* [Online] 06 2008. http://bibliodoc.uci.cu/TD/TD_1071_08.pdf. MIS-007120.

García, F., et al. 2004. *Una ontología de la medición del software.* Castilla : Universidad de Castilla-La Mancha, 2004.

García, Lourdes, Álvarez, Sofía and Quintero, Fernando. 2000. Necesidad de la medición en la gestión de la calidad de los proyectos de software para el Ministerio del Azúcar. *CentroCiencia-Biblioteca de la Red de Ciencia en Villa Clara.* [Online] 04 2000. http://biblioteca.idict.villaclara.cu/UserFiles/File/ciencia/71.pdf.

Gil Morell, Melchor. 2003. Carta del Rector de la Universidad de Ciencias Informáticas (UCI). *Portal de la Universidad de Ciencias Informáticas.* [Online] 2003. [Cited: 05 22, 2009.] http://www.uci.cu/?q=node/47.

Ginige, Athula and Murugesan, San. 2002. Web Engineering: An Introduction. *Alpen-Adria Universitat Klagenfurt.* [Online] 2002http://www-itec.uni-klu.ac.at/~harald/proseminar/web1.pdf.

Grupo de Calidad UCI. 2008. Lineamientos Mínimos de Calidad ver5.3. *Dirección de Calidad de Software UCI.* [Online] 2008. http://calidadsoft.prod.uci.cu/index.php?option=com_content&view=section&id=5&Itemid=24.

Guevara, Bedsy. 2007. Procedimiento Propuesto para medir la Calidad en. *Biblioteca UCI.* [Online] 07 2007. http://bibliodoc.uci.cu/TD/TD_0765_07.pdf. MIS-006086.

Humphrey, Watts S. 2005. *A Self-Improvement Process for Software Engineers.* Mexico City : Addison Wesley, 2005. 0-321-30549-3.

IBM-Rational Software Corporation. 2003. *Rational Unified Process (RUP) help.* [framework] s.l. : IBM, 2003. **Bibliografía 84**

BIBLIOGRAFÍA

IEEE Computer Society. 2004. *Guide to the Software Engineering Body of Knowledge.* Los Alamitos, California : The Institute of Electrical and Electronics Engineers, Inc., 2004. 0-7695-2330-7.

—. 1990. *IEEE Std 610.12: 1990 Standard Glossary of Software Engineering Terminology.* USA : Standards Coordinating Committee of the IEEE Computer Society, 1990.

International Organization for Standardization. 2005. INGENIERÍA DE SOFTWARE—CALIDAD DEL PRODUCTO—(ISO/IEC 9126-1:2001, IDT). *Dirección de Calidad del Software UCI.* [Online] 04 2005. http://calidadsoft.prod.uci.cu/tmp/documentos/normas/iso/NC-ISO-IEC%209126-1.pdf. 35.080.

Janiszewski, Steve. 2001. *Introduction to PSP & TSP.*

Jacobson, Ivar, Booch, Grady and Rumbaugh, James. 2000. *El proceso unificado de desarrollo de software.* Madrid : Addison Wesley, 2000. 84-7829-036-2.

Jayaswal, Bijay and Patton, Peter. 2007. Software Quality Metrics. *Developer.com.* [Online] 2007. http://www.developer.com/mgmt/article.php/3644656. 0131872508.

Jones, Cheryl. 2003. Making Measurement Work. [Online] 2003. http://paginaspersonales.deusto.es/cortazar/doctorado/articulos/2003/ctk03jones.pdf.

Kamel, Amr. 2008. Software metrics. *npd-solutions.* [Online] 2008. http://www.cs.ualberta.ca/~sorenson/cmput401/lectures/SoftwareMetrics/tsld001.htm.

Kan, Stephen H. 2002. *Metrics and Models in Software Quality Engineering, Second Edition.* s.l. : Addison Wesley, 2002. 0-201-72915-6.

Kaner, Cem and Pond, Walter P. 2004. *Software Engineering Metrics: What Do They Measure and How Do We Know?* s.l. : IEEE Computer Society, 2004.

Kanti, Mrinal. 1996. Product quality assurance for GIS life-cycle. *GISdevelopment.net .* [Online] 1996http://www.gisdevelopment.net/technology/gis/techgi0042pf.htm.

Keynote Systems, Inc. 2008. Measuring and Monitoring Web 2.0 Applications. *Keynote Systems.* [Online] 2008. http://www.keynote.com/docs/whitepapers/Web2.0_Applications_Solution_Brief.pdf.

Khosravi, Khashayar and Guéhéneuc, Yan-Gaël. 2005. On Issues with Software Quality Models. [Online] 2005. http://paginaspersonales.deusto.es/cortazar/doctorado/articulos/2005/qaoose05SQModels.pdf. **Bibliografía 85**

BIBLIOGRAFÍA

Kitchenham, Barbara, et al. 2006. Lessons Learnt from the Analysis of Large-scale Corporate Databases. [Online] 05 28, 2006. http://paginaspersonales.deusto.es/cortazar/doctorado/articulos/2006/p439-kitchenhamICse06.pdf. 1-59593-085-X.

Kuan Tan, Hee Beng, Zhao, Yuan and Zhang, Hongyu. 2006. Estimating LOC for Information Systems from their Conceptual Data Models. [Online] 05 28, 2006. http://paginaspersonales.deusto.es/cortazar/doctorado/articulos/2006/p321-tanICSE06.pdf. 1-59593-085-X.

la Torre, Ludisley and Cepero, Mariela. 2007. Propuesta de Métricas para Perfeccionar la Gestión de la Calidad en los Procesos de Desarrollo de Software. *Biblioteca UCI.* [Online] 06 2007. http://bibliodoc.uci.cu/TD/TD_0499_07.pdf. MIS-005820.

Lafuente, Guillermo Javier. 2000. Automatizando métricas en la Web. *Universidad Nacional de Luján, Buenos Aires, Argentina.* [Online] 07 2000. http://mdk.ing.unlpam.edu.ar/~labweb/downloads/pdfs/tesiswebsitema.pdf.

Lau, Yung-Tun. 2007. Reference Metrics for Service-Oriented Architectures. [Online] 12 2007. http://paginaspersonales.deusto.es/cortazar/doctorado/articulos/2007/2.0712Lau_CrosstalkDec07_MetricsSOA.pdf.

McDonald, Andrew and Welland, Ray. 2001. Web Engineering in Practice. *University of Glasgow.* [Online] 2001. ttp://www.dcs.gla.ac.uk/~andrew/webe2001.pdf.

Mendes, Emilia. 2007. A Comparison of Techniques for Web Effort Estimation. [Online] 2007. http://paginaspersonales.deusto.es/cortazar/doctorado/articulos/2007/2.ESEM07_WebEffort.pdf.

Meyer, Bertrand. 1998. The role of object-oriented metrics. *Eiffel Software.* [Online] Eiffel Software, 1998. http://archive.eiffel.com/doc/manuals/technology/bmarticles/computer/metrics/page.html.

Meyers, Tymothy M. and Binkley, David. 2007. An Empirical Study of Slice-Based Cohesion and Coupling Metrics. [Online] 2007. http://paginaspersonales.deusto.es/cortazar/doctorado/articulos/2007/3.ACM_TSEMa2-meyers_CohesionSlicingCoupling_Evolution.pdf.

Mills, Everald E. 1988. Software metrics. [Online] 12 1988. http://www.sei.cmu.edu/pub/education/cm12.pdf. **Bibliografía 86**

MINREX. 2005. La informatización en Cuba. *CubaMinRex.* [Online] Ministerio de Relaciones Exteriores de Cuba, 2005. http://www.cubaminrex.cu/Sociedad_Informacion/Cuba_SI/informatizacion.htm.

Morasca, Sandro. 2002. Software Measurement. *Página Personal de Rebbeca Cortazar.* [Online] 2002. http://paginaspersonales.deusto.es/cortazar/doctorado/articulos/morasca-handbook.pdf.

Nagappan, Nachiappan, Ball, Thomas and Zeller, Andreas. 2006. Mining Metrics to Predict Component Failures. [Online] 05 28, 2006. http://paginaspersonales.deusto.es/cortazar/doctorado/articulos/2006/p452-nagappanICSE06_interesante.pdf. 1-59593-085-X.

Naggapan, Nachiappan, Ball, Tomas and Murphy, Brendan. 2007. Using Historical In-Process and Product Metrics for Early Estimation. *Microsoft Research.* [Online] 2007. http://research.microsoft.com/users/nachin/papers/NagappanN-Historical.pdf.

NCGIA. 1990. Core Curriculum in GIS. *gabrielortiz.com.* [Online] 1990. http://www.gabrielortiz.com/.

New York States Archives. 1996. Geographic Information System Development Guides. *New York States Archives.* [Online] 06 1996. http://www.archives.nysed.gov/a/records/mr_pubGIS03.shtml.

Oliva, Daismary and Cruz, Olga Esther. 2008. Métricas para la evaluación del proceso de desarrollo de Software Educativo. *Biblioteca UCI.* [Online] 06 2008. http://bibliodoc.uci.cu/TD/TD_1144_08.pdf . MIS-007195.

Orme, Anthony M., Yao, Haining and Etzkorn, Letha H. 2006. Coupling Metrics for Ontology-Based Systems. [Online] 2006. [Cited: 04 18, 2009.] http://paginaspersonales.deusto.es/cortazar/doctorado/articulos/2007/2.IEEESoft_Apr06_OntologyCoupling.pdf. 0 7 4 0 - 7 4 5 9.

Peña, Yudisleidys and Hernández, Yuniersy. 2007. SIMETSE – SIstema de METricas para evaluar el. *Biblioteca UCI.* [Online] 07 06, 2007.http://bibliodoc.uci.cu/TD/TD_0803_07.pdf. MIS-006124.

Peter In, Hoh, et al. 12/2006. *A Quality-Based Cost Estimation Model for the Product Line Life Cycle.* 12, s.l. : COMMUNICATIONS OF THE ACM, 12/2006, Vol. 49.

PLM_World. 2006. Establishing effective metrics for new. *PLM_World.* [Online] PLM_World, 2006. http://newsletter.plmworld.org/archive/Vol5No2/br_metrics_best_pract.pdf.

Podgorelec, Vili and Hericko, Marjan. 03/2007. *Estimating Software Complexity from UML Models.* 2, Maribor, Slovenia : ACM SIGSOFT, 03/2007, Vol. 32. **Bibliografía 87**

BIBLIOGRAFÍA

Pressman, Roger S. 2005. *Ingeniería del Software. Un enfoque prático.* Ciudad de La Habana : Félix Varela, 2005.

Ramírez, Risell. 2007. Modelo de evaluación del proceso de desarrollo del Software Educativo. *Biblioteca UCI.* [Online] 07 2007. http://bibliodoc.uci.cu/TD/TD_0781_07.pdf. MIS-006102.

Rau, Andreas. 2001. A whitepaper on metrics. *University of applied scineces-Hochschule Esslingen.* [Online] 08 06, 2001. http://www.it.fht-esslingen.de/~rau/forschung/metrics.htm.

Reibing, Ralf. 2001. Towards a Model for Object-Oriented Design Measurement. *Institute of Computer Science, University of Stuttgart.* [Online] 2001. http://paginaspersonales.deusto.es/cortazar/doctorado/articulos/QAOOSE2001.pdf.

Reifer, Donald J. 2002. Estimating Web Development Costs:There Are Differences. [Online] 2002. http://www.stsc.hill.af.mil/crosstalk/2002/06/reifer.pdf.

Rivera, Mailen. 2007. Propuesta de métrica de productividad para el proceso de producción de software en la Facultad 4. *Biblioteca UCI.* [Online] 07 2007. [Cited: 02 16, 2009.] http://bibliodoc.uci.cu/TD/TD_0494_07.pdf. MIS-005815.

Rodríguez, Nuria and Martínez, William. 2007. Planificación y evaluación de proyectos informáticos. *google.books.com.cu.* [Online] 2007. [Cited: 04 18, 2009.] http://books.google.es/books?id=UK5Ys_kBlwYC&printsec=frontcover#PPA138,M1. 9977649898, 9789977649894.

Romero, Jenny and Salcedo, Yordanky. 2007. Propuesta de Modelo de Calidad para Portales Web. *Biblioteca UCI.* [Online] 06 2007. http://bibliodoc.uci.cu/TD/TD_0894_07.pdf. MIS-006216.

Rosenberg, Linda, Hammer, Ted and Shaw, Jack. 1998. *Software metrics and reliability.* Alemania : s.n., 1998.

Rubalcaba, María de los Angeles and Zambrana, Yayneris. 2008. Medición de la calidad de Software durante el Proceso de Pruebas en el Proyecto Modernización del CICPC. *Biblioteca UCI.* [Online] 06 2008. http://bibliodoc.uci.cu/TD/TD_1123_08.pdf. MIS-007174.

Tong, Yi, Fangjun, Wu and Chenghzi, Gan. 2004. *A Comparison of Metrics for UML Class Diagrams.* 5, Nanjing, China : ACM SIGSOFT, 2004, Vol. 29. **Bibliografía 88**

Scotto, Marco, et al. 2004. A relational approach to software metrics. [Online] 2004. http://paginaspersonales.deusto.es/cortazar/articulos.html.

Seront, Gregory, et al. 2005. On the Relationship between Cyclomatic Complexity and the Degree of Object Orientation. [Online] 2005. http://paginaspersonales.deusto.es/cortazar/doctorado/articulos/2005/qaoose05CyclomaticOO.pdf.

Snehal, Thakkar, Knoblock, Craig A. and Ambite, José Luis. 2007. Quality-Driven Geospatial Data Integration. *Information Science Institute.* [Online] 2007. http://www.isi.edu/~ambite/tkaacmgis2007.pdf.

Symons, Charles. 2007. Advancing Functional Size Measurement – which size should we measure? [Online] 2007. http://paginaspersonales.deusto.es/cortazar/doctorado/articulos/2007/2.SMEF07_Symmons_WhatSizeToMeasure.pdf.

Tapanes, Maylín. 2008. Propuesta de métricas para la estimación de proyectos. *Biblioteca UCI.* [Online] 07 2008. http://bibliodoc.uci.cu/TD/TD_1070_08.pdf. MIS-007119.

The Impact of Size and Volatility on IT project performance. **Sauer, Chris, Gemino, Andrew and Horner Reich, Blaize. 11/2007.** 11, s.l. : COMMUNICATIONS OF THE ACM, 11/2007, Vol. 50.

The Interpretation and Utility of Three Cohesion Metrics for Object-Oriented Design. **Counsell, Steve, Swift, Stephen and Crampton, Jason. 04/2006.** 2, Londres, Inglaterra : ACM Transactions on Software Engineering and Methodology, 04/2006, Vol. 15.

van den Berg, Klaas. 1995. Software Mesurement and Functional Programming (PhD Thesis). [Online] 06 23, 1995. http://www.cs.colostate.edu/~bieman/Pubs/philJSS90preprint.pdf.

Vivanco, Rodrigo A. and Jin, Dean. 2007. Selecting Object-Oriented Source Code Metrics to Improve Predictive Models Using a Parallel Genetic Algorithm. [Online] 10 25, 2007. http://paginaspersonales.deusto.es/cortazar/doctorado/articulos/2007/1.OOPSLA07vivancoOOSourceCodePredictiveGeneticAlg.pdf.

Washizaki, Hironori, Yamamoto, Hirokazu and Fukazawa, Yoshiaki. 2003. A Metrics Suite for Measuring Reusability of Software Components. *IEEE Computer Society.* [Online] 2003. http://www2.computer.org/portal/web/csdl/doi/10.1109/METRIC.2003.1232469. 1530-1435/03. **Bibliografía 89**

BIBLIOGRAFÍA

Web-Based Development and Functional Size Measurement. **Cleary, David. 2000.** Melbourne, Australia : CHARISMATEK Software Metrics, 2000.

Westfall, Linda. 2005. 12 Steps to Useful Software Metrics. *The Westfall Team.* [Online] 2005. http://www.westfallteam.com/Papers/12_steps_paper.pdf.

—. 2003. Are We Doing Well, Or Are We Doing Poorly? *The Westfall Team.* [Online] 2003. http://www.westfallteam.com/Papers/Are_We_Doing_Well.pdf.

Williams, Laurie. 2001. An introduction to Object-Oriented Metrics. *Open Seminar in Software Engineering.* [Online] 2001. http://agile.csc.ncsu.edu/SEMaterials/OOMetrics.htm.

Yut, Arlenys and Santo, Laritza. 2008. Propuesta de métricas web para medir la calidad de los portales de la Facultad 10. *Biblioteca UCI.* [Online] 07 2008. http://bibliodoc.uci.cu/TD/TD_1328_08.pdf . MIS-007386. **Acosta, Adieren and Betancourt, Daisel. 2008.** Propuesta de métricas para evaluar el flujo de trabajo Análisis y Diseño. *Biblioteca UCI.* [Online] 06 2008. http://bibliodoc.uci.cu/TD/TD_1149_08.pdf . MIS-007200.

Baresi, Luciano and Morasca, Sandro. 2007. Three Empirical Studies on Estimating the Design Effort of Web Applications. [Online] 2007http://paginaspersonales.deusto.es/cortazar/doctorado/articulos/2007/4.ACM_TSEM_a15-baresi_WebEstimation.pdf.

Barker, Richard and Tempero, Ewan. 2007. A Large-Scale Empirical Comparison of Object-Oriented Cohesion Metrics. [Online] 2007. http://paginaspersonales.deusto.es/cortazar/doctorado/articulos/2007/1.APSEC07_OOComparisonCohesionMetrics.pdf.

Berenbach, Brian and Borotto, Gail. 2006. Metrics for Model Driven Requirements Development. [Online] 05 28, 2006. http://paginaspersonales.deusto.es/cortazar/doctorado/articulos/2006/p445-berenbachICSE06.pdf

Trabajos citados

***Carrasco. 1995*.** Propuesta de Métricas para Perfeccionar la Gestión de la Calidad en los Procesos de Desarrollo de Software. *Biblioteca UCI.* [Online] 06 2007. http://bibliodoc.uci.cu/TD/TD_0499_07.pdf. MIS-005820.

Sicilia. 2009. Product development metrics list. *npd-solutions.* [Online] DRM_Associates, 2001 http://www.npd-solutions.com/metrics.html.

Contrera. 2005. Calidad de video llevada al más alto nivel publicado. http://www.chw.net/2010/10/amd-vpp-el-proximo-motor-de-procesamiento-de-video-de-amd-2/

Asencio. 2002. Fundamentals of Digital Image Processing. Prentice Hall, 1989. ISBN 978-0133361650.

Laura. 2007. La norma ISO/IEC 25000 (SQuaRE: Software Product Quality Requeriments and Evaluation)

calisoft. 2009. Curso calidad. *Introducción a la Pruebas de software.* C. Habana : s.n., 2009.

McCall.J. 1977. Factors in software Quality:General Electric. *Factors in software Quality:General Electric.* 1977.

Mc-Graw-Hill. 1095. *Pressman, R.Ingenieria del software: un enfoque práctico.* 1995.

McGraw-Hill. 1998. *R.S. Pressman.Ingeniería de Software. Un enfoque Práctico.* s.l. : 4ta Edición, 1998.

Mc-Graw-Hill. 2002. *Pressman, R.Ingenieria del software: un enfoque práctico.* 2002.

McGraw-Hill. 2005. *R.S. Pressman.Ingeniería de Software. Un enfoque Práctico.* s.l. : 4ta Edición, 2005.

OLSINA, Luis Antonio. *Metodología Cuantitativa para la Evaluación y Comparación de la Calidad de Sitios Web.*