Publicaciones en Ciencias y Tecnología
Vol 6, año 2012, N^0 1, pp.5–19, ISSN:1856-8890, Depósito Legal: pp200702LA2730

HACIA LA EXTENSIÓN DEL MÉTODO GRAY WATCH BASADO EN EL ESTÁNDAR DE CALIDAD ISO/IEC 25010

[*] [**]Jorge Luis Pérez-Medina [***]Iliana Cristina Sánchez

Recibido: 02/04/2012 **Aprobado:** 12/06/2012

Resumen

Hablar de calidad de software implica la necesidad de contar con parámetros que permitan establecer los niveles mínimos que un producto de este tipo debe alcanzar para que se considere de calidad. El presente artículo tiene como finalidad proponer una extensión del método GRAY WATCH, específicamente en los procesos técnicos de análisis y diseño acoplando los productos obtenidos al proceso de implementación. La orientación de nuestra propuesta consiste en utilizar el estándar de calidad del producto ISO/IEC 25010, que establece criterios para la especificación de requisitos de calidad de productos de software, sus métricas y su evaluación, e incluye un modelo de calidad compuesto por características y subcaracterísticas. El resultado de esta propuesta, agrega un valor importante al método extendido, permitiendo que analistas de sistemas y otros profesionales de computación puedan precisar las actividades específicas a realizar para obtener los requisitos de calidad. Para realizar esta labor hemos apoyado nuestros esfuerzos en el proceso para la Ingeniería de Dominio basado en Calidad de Software denominado InDoCaS como metodología para la definición de actividades y productos en los procesos de análisis, diseño e implementación de la aplicación.
Palabras clave: Método GRAY WATCH, métodos de desarrollo de software, calidad de software, ISO/IEC 25010, proceso InDoCaS.

TOWARDS THE EXTENSION OF THE GRAY WATCH METHOD BASED ON THE QUALITY STANDARD ISO/IEC 25010

Abstract

Talk about software quality implies the need to rely on parameters that should allow to establish the minimal levels that a product of this type must reach in order to be considered of quality. This paper aims to propose an extension of the method GRAY WATCH, specifically in the technical processes of Analysis and Design connecting the products obtained to the process of Implementation. The orientation of our proposal consists of using the standard of product quality ISO/IEC 25010, which establishes criteria for the specification of quality requirements of software products, their metrics and evaluation, and includes a quality model composed by characteristics and subcharacteristics. The result of this proposal, adds significant value to the extended method, Allowing to system analysts and Computer professionals to specify the precise activities to be performed to obtain quality requirements. To make this work we have supported our efforts in the Domain Engineering process based in Software Quality named InDoCaS as methodology for the definition of activities and products in the processes of Analysis, Design and Implementation of the Application.
Keywords: GRAY WATCH Method, sofware development methods, software quality, ISO/IEC 25010, InDoCaS process.

Introducción

La ingeniería de métodos (IM) es una rama de la ingeniería de software (IS) que tiene por objetivo ayudar a la construcción de nuevos métodos y técnicas de ingeniería de sistemas de información. Los tres

[*] *Decanato de Ciencias y Tecnología, Universidad Centroccidental Lisandro Alvarado, Barquisimeto, Venezuela.*
[**] *Centro de Análisis, Modelado y Tratamiento de Datos, CAMYTD, Facultad de Ciencias Tecnología, Universidad de Carabobo. jorgeperez@ucla.edu.ve*
[***] *Instituto Luis Beltrán Prieto Figueroa. Universidad Pedagógica Experimental Libertador, Barquisimeto, Venezuela. ilianac.san@gmail.com*

principios fundamentales de la IM son la optimización, la reutilización y la adaptación [1]. Brinkkemper [2] define la IM como "una disciplina de conceptualización, de construcción y de adaptación de métodos, técnicas y herramientas para el desarrollo de sistemas de información". Esta trata la definición de nuevos métodos de ingeniería de los sistemas de información (SI). Otras definiciones resaltan la noción de la IM a la construcción de nuevos métodos, por ejemplo, Punter [3] define la IM como "un enfoque de construcción de métodos combinando diferentes (partes de) métodos para desarrollar una solución optima del problema dado". Kumar [4], al contrario, propone una definición más general que no impone el uso de métodos existentes como punto de partida de la IM al definir esta última como "una para el diseño y el desarrollo de una metametodología destinada al diseño de métodos de desarrollo de los sistemas de información". Nosotros definimos la IM como la disciplina que apunta sus esfuerzos en aportar soluciones eficaces al diseño, el mejoramiento y la evolución de los métodos de desarrollo de los Sistemas de Información (SI).

En el dominio de los SI, los referenciales, normas y estándares representan generalmente el estado del arte y los saberes-hacer de un contexto dado. Al lado de las normas de producto, las normas de gestión aparecen más presentes; estas introducen un nivel organizacional a los aspectos técnicos naturalmente tomados en cuenta por los servicios informáticos. La ISO 9001[1] especifica las exigencias requeridas por un sistema de manejo de la calidad aplicable a todas las actividades y negocios. [5]. No obstante, hablar de calidad del software resultante implica la necesidad de contar con parámetros que permitan establecer los niveles mínimos que un producto de este tipo debe alcanzar para que se considere de calidad [6]. En la práctica, los parámetros de calidad de los productos de software son frecuentemente descritos en términos generales [7], siendo difícil para el grupo de proyecto validar la calidad del producto de software contra los requisitos.

Varios autores han creado diversos modelos que permitan validar la calidad del mismo, entre los que se destacan: FURPS [8], McCall [9], Boehm [10], ISO 9126 [11], IEEE 1061 [12], Dromey [13] e ISO/IEC [14]. Estos modelos son presentados sobre la forma de factores de calidad y criterios [9], factores y subfactores [12], colecciones de características y subcaracteristicas dependientes las cuales son conectadas a indicadores y métricas [11][14] permitiendo evaluar el producto de software mediante las métricas establecidas.

Por su parte la ISO/IEC 25000, que proporciona una guía para el uso de las nuevas series de estándares internacionales llamados Requisitos y Evaluación de Calidad de Productos de Software (SQuaRE) [14], establece criterios para la especificación de requisitos de calidad de productos de software, sus metas y su evaluación. SQuaRE está formada por la división del modelo de calidad mediante la Norma 25010 que recomienda el uso de un modelo de calidad mediante ocho características de calidad tales como: Funcionalidad, Rendimiento, Seguridad, Compatibilidad, Usabilidad, portabilidad, fiabilidad, y Mantenibilidad.

En este contexto hacemos referencia al método WATCH [15] en su más reciente versión GRAY WATCH [16]. Este método se encuentra enmarcado en el desarrollo de software empresarial bajo el paradigma de reutilización de componentes. GRAY WATCH considera en sus primeras etapas dentro del proceso de desarrollo, específicamente en la fase de Ingeniería de Requisitos, la identificación de requisitos funcionales y no funcionales para evaluar la calidad de la arquitectura, sin precisar las actividades especificas para obtener los requisitos de calidad. Un requisito de calidad [17] corresponde a propiedades que caracterizan una solución arquitectónica. En este orden de ideas Canelón [18] considera que los requisitos no funcionales expresados en términos de calidad son fundamentales para el diseño arquitectural, en consecuencia, para lograr asegurar la calidad en el producto final se debe tener un proceso que considere la especificación de requisitos de calidad, el diseño de modelos de calidad y arquitecturas basadas en atributos de calidad, que permitan implementar el software utilizando artefactos de calidad.

El presente artículo introduce nuestra propuesta de extensión del método GRAY WATCH. La extensión consiste en definir e incorporar las actividades y los productos que permitan expandir los procesos de Análisis y Diseño aplicando el estándar de calidad ISO/IEC 25010 y acoplar el proceso de implementación del método GRAY WATCH con los artefactos generados en los procesos de Análisis y Diseño. Para

[1]La norma ISO 9001 es parte de la serie de normas ISO 9000, relativas a los sistemas de gestión de la calidad.

realizar esta labor hemos apoyado nuestros esfuerzos en el proceso para la Ingeniería de Dominio basado en Calidad de Software denominado InDoCaS [19] como metodología para la definición de actividades y productos en los procesos de Análisis, Diseño e Implementación de la Aplicación. Ambos procesos se basan en el estándar de calidad ISO/IEC 25010. Por razones de espacio de este documento, este artículo se centra en explicar, el trabajo efectuado en el proceso técnico de Análisis del método GRAY WATCH, específicamente en la fase de especificación de los requisitos. Futuros artículos trataran los trabajos efectuados a nivel del diseño y de la implementación. El artículo se organiza de la siguiente manera. La sección 2 presenta la descripción del caso de estudio que se utilizó para la aplicación de la propuesta. La sección 3 presenta un marco conceptual de los principios y conceptos fundamentales abordados por la investigación, entre ellos mencionamos: la Ingeniería de Métodos, el método GRAY WATCH, la norma de calidad ISO/IEC 25010, el modelo de clasificación de requisitos RECLAMO, el proceso InDoCaS basado en calidad de software y el proceso InDoCaSE. La sección 4 presenta un breve estado del arte respecto al uso de la calidad del producto en tempranas fases del desarrollo de los SI. La sección 5 presenta nuestra propuesta de extensión del método GRAY WATCH en su proceso de Análisis. Finalmente la última sección las conclusiones y perspectivas.

Caso de estudio

Para explicar nuestra propuesta nos centramos en un caso de estudio denominado **"Programa de Investigación UPEL-IPB"**. Las razones por las cuales hemos elegido este contexto se deben a que conocemos los procesos y estos forman parte de nuestras labores cotidianas. El objetivo es demostrar como nuestra propuesta de extensión puede ser aplicada en el desarrollo de una aplicación. Este programa que tiene como objetivo gestionar el desarrollo de proyectos, actividades y eventos de investigación de alto nivel científico que posibiliten la excelencia académica. Por tanto, requiere de una aplicación que de soporte a los procesos del negocio mediante una interfaz web que facilite el intercambio de datos e información a través de una red Intranet, Extranet o Internet. Aunado a ello, se deben ofrecer servicios relacionados con la labor investigativa a las diferentes unidades internas de la institución y a los organismos del Estado tomar la información requerida en cuanto a los indicadores de productividad de los investigadores y de sus actividades investigativas.

De igual manera, el sistema también debe ofrecer una gama de servicios de información de utilidad exclusiva de determinados usuarios en base a su perfil ejecutivos, investigadores, visitantes y otros, respecto de programas, proyectos, eventos y productos resultantes de las actividades de investigación desarrolladas en la institución.

Marco conceptual

Esta sección presenta los principios y conceptos que soportan nuestra propuesta de extensión. En primer lugar abordaremos el concepto de la Ingeniería de Métodos. Seguidamente, presentaremos el método GRAY WATCH, la norma de calidad ISO/IEC 25010, el modelo de clasificación de requisitos RECLAMO, el proceso InDoCaS basado en calidad de software.

A. La ingeniería de métodos

El termino método viene del greco "methodos" que significa "medio de investigación". Harsem [20] determina que son los medios de investigación: una colección de procedimientos, técnicas, descripciones de productos y herramientas para el soporte efectivo, eficaz y consistente del proceso de ingeniería de un sistema de información.

Otras definiciones han sido propuestas en: [21][22][23][24][25][2]. Estas definiciones establecen una clara separación entre el producto que el método elabora y el proceso que elabora el producto. Esta aceptación es sintetizada por G. Booch [26] que define un método como un proceso riguroso permitiendo generar un

conjunto de modelos que describen diversos aspectos de un producto de software en construcción. En otros términos, un método trata los dos aspectos de la ingeniería, el producto y el proceso, y está basado en dos elementos: uno o varios modelos de producto y uno o varios modelos de procesos [27], [21]. Este enfoque es usado por el método GRAY WATCH que metodológicamente se basa en un modelo de procesos, un modelo de productos y un modelo de actores. A continuación se describe este método de desarrollo.

B. El método GRAY WATCH

WATCH es un método [15] enfocado al desarrollo de software empresarial. A lo largo de los años el método WATCH ha sido refinado por varios autores. Su primera extensión fue realizada por Hamar [28] en el año 2003 para desarrollar el ciclo de vida de un componente reutilizable, desde su especificación hasta su liberación. Esta sección presenta una visión general de la versión denominada GRAY WATCH [16], en particular, nos concentraremos en su proceso de Análisis donde se definen y especifican el conjunto de requisitos funcionales y no funcionales que la aplicación empresarial debe satisfacer.

GRAY WATCH es un marco metodológico que describe los aspectos técnicos, gerenciales y de soporte que deben emplear los equipos de trabajo que tendrán a su cargo el desarrollo de aplicaciones de software empresarial. Un marco metodológico es un patrón que debe ser *instanciado*, es decir adaptado cada vez que se use [16]. Cada equipo de trabajo deberá usar el método como un patrón o plantilla metodológica, a partir de la cual dicho equipo debe elaborar el proceso específico de desarrollo de la aplicación que se desea producir. GRAY WATCH cubre todo el ciclo de vida de las aplicaciones; desde el modelado del dominio de la aplicación, pasando por la definición de los requisitos de los usuarios, hasta la puesta en operación de la aplicación.

GRAY WATCH está compuesto por tres (3) modelos fundamentales: el modelo de productos, el modelo de actores y el modelo de procesos [16]. Este último establece los procesos necesarios para gestionar proyectos de desarrollo de aplicaciones empresariales y llevar a cabo las actividades técnicas y de soporte que requieren estos proyectos. Los procesos técnicos del método se dividen en tres grupos: Procesos de Análisis, Diseño e Implementación. La figura 1 ilustra esta clasificación.

Figura 1: Procesos del Método GRAY WATCH [16]

La figura anterior resalta tres (3) grandes grupos de procesos. Los procesos de análisis de la Aplicación cubren los procesos de Modelado de Negocios y la Ingeniería de Requisitos. La fase de diseño consiste de los procesos de Diseño Arquitectónico y Diseño de Componentes, mientras que los procesos de Implementación agrupan los procesos de Programación e Integración, Pruebas y Entregas de la Aplicación.

Procesos de análisis

Estos procesos tienen como objetivos (1) entender y modelar el Sistema de Negocios que constituye el dominio de la aplicación empresarial; y (2) definir y especificar el conjunto de requisitos que la aplicación empresarial debe satisfacer.

En la figura 2 se aprecia que el proceso de Ingeniería de Requisitos requiere de la ejecución de cinco subprocesos complementarios: el Descubrimiento de Requisitos, Análisis de Requisitos, Especificación de Requisitos Validación de Requisitos y Gestión de Requisitos.

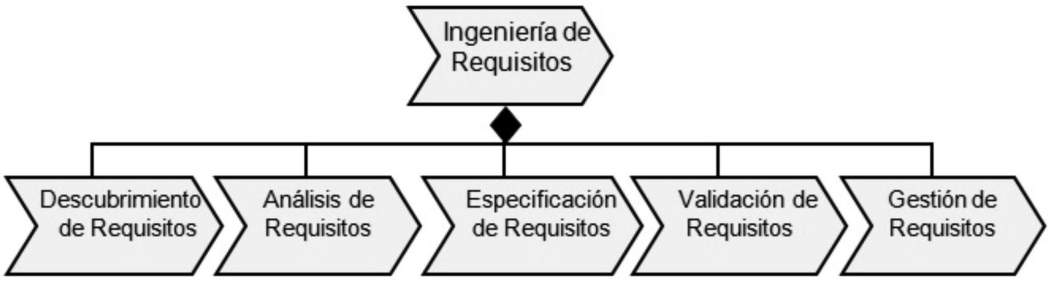

Figura 2: Subprocesos del Proceso de Ingenieria de requisitos [16]

El subproceso de Análisis de Requisitos consiste en determinar y resolver posibles conflictos entre los requisitos y establecer la interacción de la aplicación empresarial con su dominio o ambiente. La figura 3 muestra el diagrama de actividades de este subproceso.

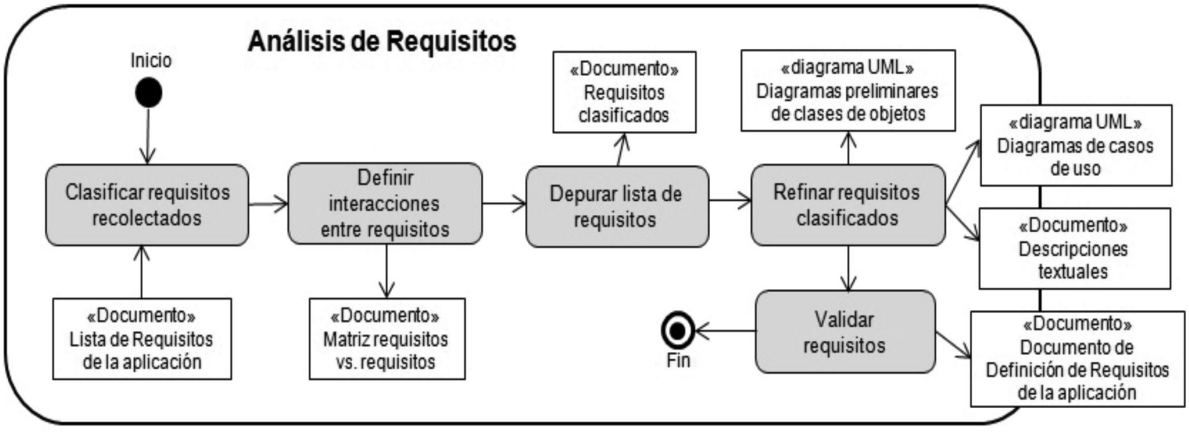

Figura 3: Diagrama de Actividades subproceso de Análisis de Requisitos para el método GRAY WATCH [16]

C. La calidad del software

El término calidad se utiliza con frecuencia en diversos ámbitos de la sociedad. Pressman [6] indica que la calidad del sistema es la concordancia que debe existir entre los requisitos, especificaciones y el

diseño del sistema. Por su parte, la calidad de diseño se refiere a las características que especifican los ingenieros de software para un elemento. El grado de materiales, tolerancias y las especificaciones del rendimiento contribuyen a la calidad del diseño. La calidad de concordancia es el grado de cumplimiento de las especificaciones de diseño durante su realización. Una vez más, cuanto mayor sea el grado de cumplimento, más alto será el nivel de calidad de concordancia.

La calidad de concordancia es un aspecto centrado principalmente en la implementación. Si la implementación sigue el diseño, y el sistema resultante cumple los objetivos de requisitos y de rendimiento, la calidad de concordancia es alta. Dicho de otra manera, la calidad del software se debe diferenciar entre la calidad del producto y la calidad del proceso de desarrollo de éste (calidad de diseño y fabricación). No obstante, las metas que se establezcan para la calidad del producto van a determinar los objetivos del proceso de desarrollo. Dentro de este marco de referencia tenemos las normas ISO que definen la calidad de software, específicamente el estándar 25000 que se refiere a la calidad del producto de software.

La norma de calidad ISO/IEC 25010

La ISO[2] (International Organization for Standarization) y la IEC[3] (International Electrotechnical Commission) forman el sistema especializado para la estandarización a nivel mundial. En el campo de tecnologías de información, ISO e IEC han establecido un comité técnico conjunto denominado ISO/IEC JTC 1, cuya tarea principal es preparar estándares internacionales. El estándar ISO/IEC 25010 [14] fué preparado por el comité técnico conjunto ISO/IEC JTC 1, Tecnología de Información, Subcomité SC 7, Ingeniería de Software y Sistemas, y forma parte de la serie de estándares internacionales SQuaRE (Software product Quality Requirements and Evaluation), la cual contiene división del modelo de calidad ISO/IEC 2501n [14].

La división del modelo de calidad está formada por estándares internacionales que presentan modelos de calidad detallados para sistemas de computadoras y productos de software, incluyendo características de funcionalidad interna y externa, y calidad en el uso. También proporciona orientación sobre el uso de los modelos de calidad. En la figura 4 se puede observar que el estándar ISO/IEC 25010 describe un modelo bipartito para la calidad del producto de software, siendo sus modelos principales: 1) Calidad interna; 2) Calidad externa y 3) Calidad en el uso.

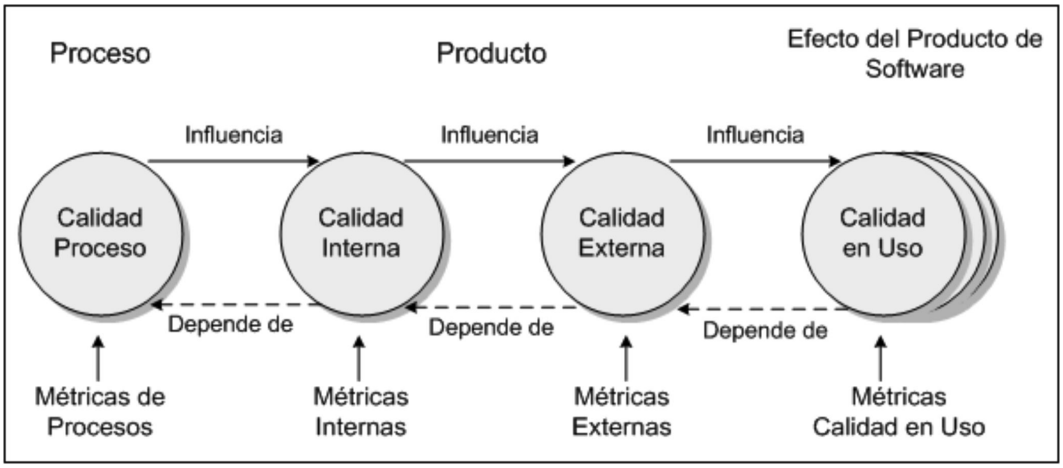

Figura 4: Enfoque de calidad ISO/IEC [14]

La Calidad Interna, proporciona una visión de la "caja blanca" del software y trata las características

[2]ISO: Organización Internacional para la Estandarización - http://www.iso.org/
[3]IEC: Comisión Internacional de Electrónica - http://www.iec.ch/

del producto de software que están disponibles durante el desarrollo. Está relacionada con las características estáticas del software y tiene un impacto en la calidad externa del software, que tiene a su vez un impacto en la calidad funcional. Así mismo, la Calidad externa, proporciona una visión de la "caja negra" del software y trata las características relacionadas con la ejecución del software. La calidad de uso es una medida de la calidad del sistema en su ambiente operacional para usuarios específicos que realizan tareas específicas. La calidad funcional del software es la capacidad de permitir la misma en su ambiente operacional para ejecutar tareas específicas que realizan los usuarios.

La primera parte del modelo especifica ocho características para la calidad interna y externa: funcionalidad, rendimiento, compatibilidad, usabilidad, fiabilidad, seguridad, mantenibilidad y portabilidad. Como se aprecia en la figura 5, las características se subdividen en sub-características, que se manifiestan externamente cuando el software se utiliza como parte de un sistema informático, y es resultado de las cualidades internas del software. Este estándar internacional no elabora el modelo para la calidad interna y externa debajo del nivel de sub-características.

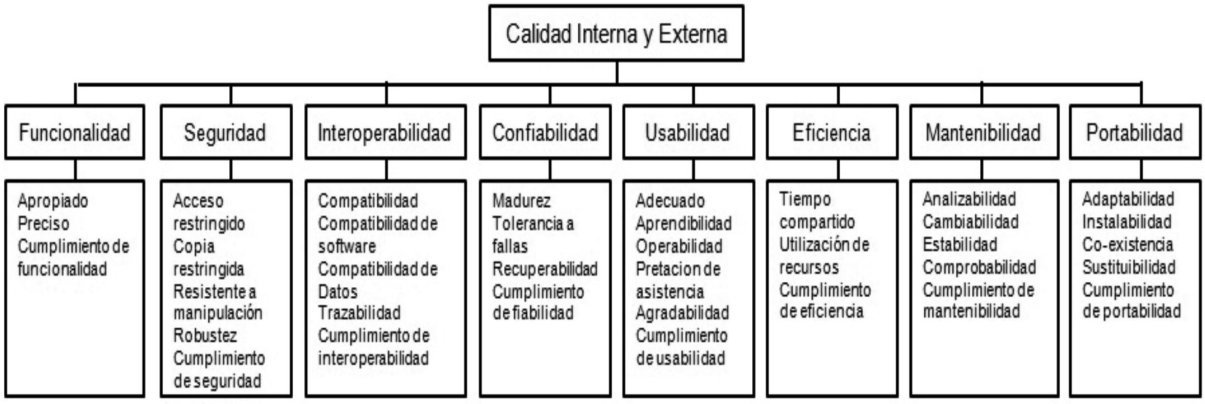

Figura 5: Modelo de calidad interna y externa ISO/IEC 25010 [14]

La segunda parte del modelo ISO/IEC 25010 especifica cinco características de la calidad en el uso. La figura 6 muestra esta clasificación que corresponde al efecto combinado para el usuario de las ocho características de la calidad del producto de software. Las características definidas son aplicables a todo tipo de software. Las características y subcaracterísticas proveen terminología consistente para la calidad de producto de software. También proveen una estructura para requisitos de calidad específicos para software, y obtener compensaciones entre las capacidades del producto de software.

El modelo RECLAMO

Requirements Classification Model (RECLAMO) [29] es un modelo de clasificación de requisitos basado en una perspectiva de calidad. RECLAMO apoya el proceso de especificación de los requisitos. Su objetivo es facilitar la identificación de diferentes clases de requisitos implicados en la definición de un sistema de software, en particular, los requisitos no funcionales son relacionados con los requisitos de calidad. RECLAMO permite agrupar los requisitos de calidad en tres vistas: 1) Interna; 2) Externa; y en uso o sistemas. Estas vistas están basadas en las vistas de calidad del estándar ISO/9126-1. De esta manera los requisitos de calidad se definen como un conjunto de características de calidad y sus relaciones.

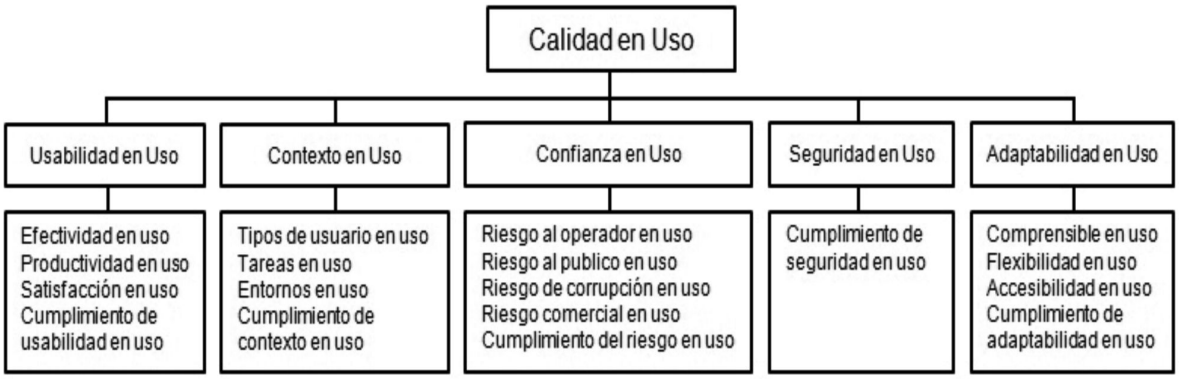

Figura 6: Modelo de calidad en uso ISO/IEC 25010 [14]

D. El proceso para la Ingenieria de Dominio basado en Calidad de Software - InDoCas

InDoCaS [19] es un proceso para la ingeniería del dominio que puede ser utilizado en el enfoque de desarrollo de líneas de producto de software, el cual puede ser instanciado para un dominio especifico y los activos de software producidos pueden ser reutilizados para generar un producto de una familia particular del dominio. El dominio es considerado como una familia de productos que tienen características comunes. La estructura del proceso InDoCaS está basada en: RECLAMO [29], ISO/IEC 25010 [14], un proceso de análisis del dominio para construir el modelo de calidad propuesto por [30], el modelo de variabilidad de FODA [31] [32], los métodos de diseño dirigido por atributos, usados para la formulación de escenarios de calidad de ADD [33] y la evaluación arquitectónica de ATAM [34].

La fase de análisis del dominio del proceso InDoCaS es fundamental para el desarrollo de las líneas de producto de software, en ella se especifican los aspectos comunes a todas las familias del dominio y los particulares de cada una de ellas. Este subproceso permite la caracterización del dominio, identifica las propiedades de calidad que deben ser garantizadas y define el modelo de calidad asociado al dominio que brinda soporte al resto de las fases.

En la figura 7 se puede apreciar que el análisis del dominio del proceso InDoCaS contempla seis actividades: Identificación de Requisitos Obtener Modelo de Similitudes y Variabilidad, Identificación de Propiedades de Calidad Asociadas al Conjunto Minimal de Requisitos Funcionales y no Funcionales, Obtener el Modelo de Calidad Asociado al Dominio, Creación de Escenarios de Calidad del Dominio e Identificar los Estilos Arquitecturales para el Dominio. Asimismo, la imagen muestra las entradas y las salidas de cada una de las actividades y secuencia de ejecución, del mismo modo se indican que técnicas son aplicadas.

El uso de la calidad del producto en tempranas fases del desarrollo

La calidad del producto de software en etapas tempranas de desarrollo se utiliza para diseñar software que cumplan con los requisitos y especificaciones de los clientes, y a su vez se establezcan características y subcaracterísticas que permitan realizar evaluación de la calidad del software. Chung et al. [35], proponen un proceso que Integra Requisitos Funcionales (RF) y No Funcionales (RNF) en el modelo de casos de uso y llega a una arquitectura inicial utilizando el enfoque dirigido por objetivos de Yu y Mylopoulos [36]. Un proceso de análisis de dominio propuesto por [37] utiliza un modelo de calidad de acuerdo al estándar ISO/IEC 9126-1 para la especificación temprana de los (RNF). Para Losavio et al. [17] la calidad del

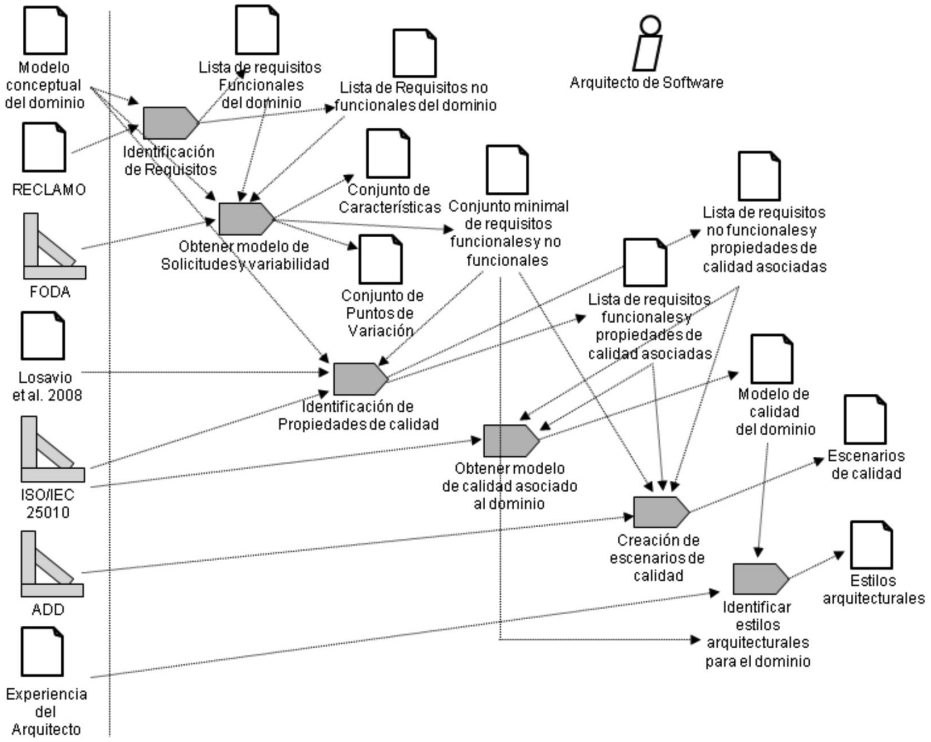

Figura 7: Diagrama de actividades para el análisis del dominio InDoCaS [19]

producto se basa en satisfacer las características deseadas del producto, y esto se puede validar a través del diseño de la arquitectura del sistema, que debe responder a los requisitos exigidos por el sistema: funcionales y no funcionales.

En este sentido, cabe mencionar el trabajo de Canelón et al. [18] que construyó un modelo de calidad para el dominio de las aplicaciones móviles sensibles al contexto incorporando calidad del producto en la especificación de requisitos, utilizando la taxonomía del modelo RECLAMO para la clasificación de requisitos funcionales y no funcionales y aplicando el estándar de calidad ISO/IEC 25010 para identificar los requisitos de calidad en base a características y subcaracterísticas de calidad interna, externa o en uso, que finalmente permiten obtener el modelo de calidad del dominio. Por otra parte, Castillo y otros [38] diseñaron un modelo Conceptual denominado Requirements, Aspects Software Quality (REASQ). Para lograr este propósito, consideraron la metodología de desarrollo de software orientada a aspectos: Aspects Oriented Software Development (AOSD) [39] que propone la especificación de requisitos no funcionales relacionados al comportamiento de la funcionalidad del sistema o contexto en etapas más tempranas, el Modelo RECLAMO y el estándar de calidad ISO/IEC 25000 para el diseño del modelo.

Por su parte en [40] definen un Proceso Extendido del proceso propuesto por Chung para integrar RF y RNF en el modelo de casos de uso y bajo un enfoque orientado a objetivo ó "goal oriented approach". La extensión considera estándares de calidad para la especificación de requisitos no funcionales y la identificación temprana de incumbencias transversales, conocido como el enfoque "early-aspects". El

proceso Extendido de Chung se inspira en [37] que especifica los requisitos de calidad asociados a RF y RNF de acuerdo al estándar ISO/IEC 25010, actualizando el trabajo de [17] en el cual se usó el estándar ISO/IEC 9126-1 versión previa al ISO/IEC 25010.

La extensión del GRAY WATCH en su proceso de análisis

La propuesta consistió en aplicar calidad en los procesos de análisis y diseño del método GRAY WATCH, para ello se incorporaron las actividades del proceso InDoCaS [19], que permitieron extender y acoplar los subprocesos resultantes. Para la representación de los diagramas de actividad que muestran la extensión de las actividades y artefactos en los subprocesos del método GRAY WATCH se utilizó la diagramación mediante SPEM [41]. La primera parte del proceso de extensión se concentró en el subproceso de **Análisis de Requisitos** contenido en el proceso de **Ingeniería de Requisitos**como se aprecia en la figura 3. Este subproceso se encarga de definir y especificar el conjunto de requisitos funcionales y no funcionales que la aplicación debe satisfacer. En el cuadro 1 se puede observar la especificación para la actividad Identificar Requisitos de la propuesta. Asimismo, en los cuadros 2 y 3 se pueden apreciar los artefactos para los requisitos funcionales y no funcionales propuestos. Estos artefactos proporcionan al Analista de Requisitos la guía para la clasificación.

ACTIVIDAD	DESCRIPCION
Nombre	Identificar requisitos
Actor	Analista de requisitos
Objetivo	Clasificar requisitos
Técnica utilizada	RECLAMO
Artefactos de entrada	Lista de requisitos de la aplicación
Artefactos de salida	Requisitos funcionales y no funcionales

Cuadro 1: Actividad identificar Requisitos del Subproceso Análisis de Requisitos del método GRAY WATCH extendido

ARTEFACTO	DESCRIPCION
Nombre	Requisitos Funcionales de la aplicación
Constructor	Analista de requisitos
Formato asociado	Adaptado de InDoCaS.

Nro. Identificación	Requisitos Funcionales

Cuadro 2: Artefacto Requisitos funcionales del Subproceso Análisis de Requisitos del método GRAY WATCH extendido

La extensión de este subproceso consistió en incorporar la actividad **"Identificación de requisitos"** de InDoCaS y fusionarla con la actividad **"clasificar requisitos recolectados"** de GRAY WATCH que permite identificar los requisitos funcionales y no funcionales. Esta fusión permitió aplicar el modelo de clasificación de requisitos RECLAMO [29], que guía la clasificación de los requisitos funcionales que se derivan de los requerimientos de los usuarios y representan la funcionalidad del sistema, mientras que los requisitos no funcionales se derivan del dominio del problema, del ambiente de ejecución del sistema, de los requisitos de definición de datos y de las reglas del negocio. La clasificación de los requisitos funcionales y no funcionales proporciona una guía en la fase de Diseño Arquitectónico para obtener las propiedades de

ARTEFACTO	DESCRIPCION
Nombre	Requisitos No Funcionales de la aplicación
Constructor	Analista de requisitos
Formato asociado	Adaptado de InDoCaS

Nro.	Reglas del negocio asociadas al dominio	Requisitos no Funcionales derivados de las reglas del negocio
	Políticas	
	Procesamiento	
	Implementación	

Cuadro 3: Artefacto Requisitos no funcionales del Subproceso Análisis de Requisitos del método GRAY WATCH extendido

calidad derivadas de los requisitos funcionales y no funcionales y asociadas al estándar ISO/IEC 25010. El resto de las actividades permanecen con su misma funcionalidad tomando como artefacto de entrada la lista de los requisitos clasificados. La figura 8 ilustra el subproceso resultante para la fase de Análisis de los Requisitos.

En relación al caso de estudio, la ejecución de este subproceso permite obtener la lista de los requisitos funcionales y no funcionales que se resumen en los siguientes cuadros. Para obtener estos resultados se aplicó el modelo RECLAMO permitiendo clasificar la lista de los requisitos en requisitos en funcionales (ver Cuadro 2) y requisitos no funcionales considerando la especificación de los clientes y las reglas del negocio (ver Cuadro 3). Estos resultados son a su vez los artefactos de entrada al subproceso de diseño para identificar los requisitos de calidad que será detallado en una siguiente entrega.

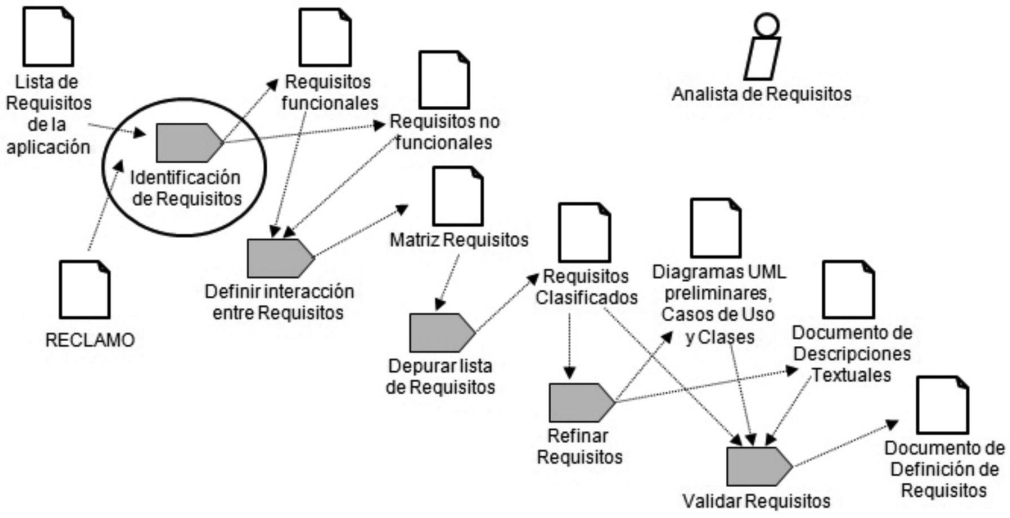

Figura 8: Diagrama de Actividades propuesto del subproceso de Análisis de Requisitos para el método GRAY WATCH

Nro.	Requisitos Funcionales
1	Ofrecer una gama de servicios de información de utilidad exclusiva a determinados usuarios en base a su perfil directivos, investigadores, administrativos y otros
2	Gestionar el registro y seguimiento de proyectos de investigación al personal de investigación, y emitir reportes de los proyectos.
3	Emitir un conjunto de indicadores de gestión y de estadísticas sobre las actividades de investigación.
4	Proporcionar servicios de información de estadísticas de investigadores y de las actividades de investigación a otros entes internos de la Universidad.
5	Suministrar información de los investigadores activos y de los proyectos de investigación a los organismos del Estado como MPPES, OPSU y MCT.

Cuadro 4: Artefacto Requisitos funcionales Caso de estudio: Programa de Investigación UPEL-IPB

Nro.	**Reglas del negocio asociadas al dominio**	**Requisitos no funcionales derivados de las reglas de negocio**
	Políticas	
1	Seguridad en el acceso al sistema	La aplicación debe ofrecer mecanismos de seguridad, que permita solo a los usuarios registrados acceder a las funcionalidades del sistema, que le corresponde según su rol.
2	El sistema debe estar disponible en la red de la Institución	Garantizar el servicio en la red, considerar ancho de banda y protocolos de comunicación.
	Procesamiento	
3	La solución debe ofrecer adecuados niveles de servicio donde la disponibilidad y recuperación de fallos sea garantizada.	El sistema debe tener tolerancia a fallas (el sistema debe mantenerse operativo y recuperarse de posibles fallas)
	Implementación	
4	El sistema debe permitir cambios sin afectar el rendimiento del mismo	El sistema debe permitir la incorporación de nuevos componentes sin que causen un impacto en la ejecución de la aplicación.
5	Capacidad de interacción con otros sistemas de la Institución y externos.	La aplicación debe soportar la capacidad de comunicarse con sistemas externos a nivel de datos y procesos.

Cuadro 5: Artefacto Requisitos no funcionales Caso de estudio: Programa de Investigación UPEL-IPB

Conclusiones

El desarrollo de software es un proceso complejo, requiere la aplicación de principios, métodos, modelos y técnicas de Ingeniería de Software y Gerencia de Proyectos. Nuestras investigaciones consideran el método GRAY WATCH diseñado por Montilva et al [16] como un método de desarrollo de software, cuyo marco metodológico describe los procesos técnicos, gerenciales y de soporte que deben emplear los equipos de trabajo, para desarrollar los componentes arquitectónicos de una aplicación e integrarla al sistema de negocios para el cual ella es desarrollada. Este artículo describe un trabajo de investigación consistente en la extensión de un método existente. El resultado obtenido agrega un valor importante, al método extendido, al tomar en consideración estándares y técnicas conocidas de Calidad del Software como lo es el estándar ISO/IEC 25010. El artículo hace una revisión bibliográfica extensa relacionada con los conceptos, métodos y técnicas empleadas en la elaboración del resultado. El resultado obtenido es útil y de aplicación inmediata. Puede ser utilizado por analistas de sistemas y otros profesionales de la Computación durante las fases de Ingeniería de Requisitos. La propuesta presentada en este artículo

incorpora al método GRAY WATCH aspectos de calidad del producto, a través del uso del estándar ISO/IEC 25010, específicamente en las primeras fases de desarrollo del software como lo es la Ingeniería de Requisitos. La extensión permite conocer con detalle las actividades y los artefactos necesarios para obtener requisitos de calidad que a su vez permiten evaluar la arquitectura inicial. De igual forma, pudimos conocer los pasos a seguir para obtener el modelo de calidad del software. Para ello, se modificaron y extendieron subprocesos del proceso de Análisis de la aplicación del método GRAY WATCH, los procesos y los productos generados se adaptaron al resto de los procesos de la Cadena del método GRAY WATCH. Asimismo, se aplicó la extensión del método GRAY WATCH al caso de estudio Programa de Investigación de la UPEL-IPB, que ejemplifica la necesidad de construir una aplicación Web para la gestión de sus procesos de negocio. Al utilizar el método GRAY WATCH propuesto se obtuvo una clasificación de los requisitos funcionales y no funcionales que servirán para realizar el modelo de calidad y una arquitectura que satisfaga los requisitos arquitectónicos basados en el estándar ISO/IEC 25010.

Futuros trabajos enfocarán los esfuerzos en revisar otros procesos técnicos del método GRAY WATCH en los cuales se pudiera incorporar aspectos de calidad aplicando otra serie del estándar de la división de calidad del producto ISO/IEC 25000, por ejemplo: consideramos que es posible expandir los procesos de Prueba de la Aplicación del método GRAY WATCH al incorporar procesos de evaluación del producto que puedan realizarse aplicando el estándar de evaluación de calidad ISO/IEC 25040 dado que proporciona una guía para la evaluación de la calidad del software. Igualmente, consideramos examinar la posibilidad de incorporar métricas de calidad que permitan evaluar la calidad en uso del Sistema desarrollado, igualmente consideraremos aplicar el estándar de mediciones de calidad ISO/IEC 25020, que presenta aplicaciones de métricas para la calidad de software desde la perspectiva interna, externa y en uso. Finalmente, se invita a la comunidad de Ingeniería de Software a considerar otros aspectos o elementos que pudieran evolucionar la extensión del método GRAY WATCH y crear nuevas experiencias en cuanto al desarrollo de software con normas de calidad.

Referencias

[1] Rolland C. (2005). Lingénierie des méthodes: une visite guidée. e-TI - la revue électronique des technologies d'information. Disponible en : http://www.revue-eti.netdocument.php?id=726.

[2] Brinkkemper S. (1996). Method Engineering. Engineering of information systems development methods and tools, Information and Software Technology, Vol. 38, No.4. Año 1996. Pp 275-280.

[3] Punter H., Lemmen K. (1996). The MEMA model: Towards a new approach for Method Engineering. Information and Software Technology, Vol. 38, No.4. Año 1996. Pp 295-305.

[4] Kumar K., Welke R-J. (1992). Methodology Engineering: a proposal for situation-specific methodology construction, dans Cotterman, W.W. et Senn J.A. (Eds.), Challenges and Strategies for Research in Systems Development, John Wiley & Sons Ltd. Pp 257-269.

[5] Rivet A. (2007). Normes de qualité et systmes dinformation. Les Journées Réseaux - JRES, Strasbourg.

[6] Pressman, R. (2002). Ingeniería del Software, Un enfoque Practico. 5da Edición. McGraw Hill. Madrid, 640 paginas.

[7] Veenendaaal, E., Hendriks, R. y Vonderen, R. (2002). Measuring software product quality. Software Quality Professional. Vol. 5. nro. 1. American Society for Quality. Milwaukee.

[8] Grady R. (1992). Practical software metrics for project management and process improvement, Prentice Hall. 282 paginas.

[9] McCall J., Richards P. y Walters G. (1977). Factors in Software Quality, Nat'l Tech.Information Service, no. Vol. 1, 2 and 3, AD/A-049-015/055. Springfield.

[10] Boehm B., Brown J., Kaspar H., Lipow M., McLeod G. y Merritt M. (1978). Characteristics of Software Quality, New York. Elsevier North Holland Publishing Company, Inc, 1978.

[11] ISO/IEC. (1991). International Organization for Standardization: ISO Standard 9126: Information Technology Software product evaluation Quality characteristics and guidelines for their use.

[12] IEEE std 1061. (1992). Standard for a Software Quality Metrics Methodology.

[13] Dromey R. (1995). A model for software product quality, IEEE Transactions on Software Engineering, no. 2. Pp 146-163.

[14] ISO/IEC JTCI/SC7N4522. (2009). FCD 25010 Software Engineering- Software Product Quality Requirements and Evaluation (SQuaRE) Quality model, FCD ballot.

[15] Montilva, J. Hazam, K., y Gharawi, M. (2000). The Watch Model for Developing Business Software in Small and Midsize Organizations. IV World Multiconference on Systemics, Cybernetics and Informatics - SCI2000. Orlando, Florida. Vol. XII. Pp 263-268.

[16] Montilva, J., Barrios, J. y Rivero M. (2008). GRAY WATCH Método de desarrollo de software para aplicaciones empresariales. Version 1. Proyecto METHODIUS, FONACIT 2005000165. Mérida.

[17] Losavio F., Matteo A. y Pacilli I. (2009). Proceso dirigido por objetivos para análisis de dominio bajo estándares de calidad. Enl@ce Revista Venezolana de Informacion, Tecnologia y Conocimiento, 6(3), Año 2009. Pp 11-28.

[18] Canelón, R., Losavio, F., Matteo, A. y Chirinos, L. (2009). Modelo Conceptual para Modelación de Aplicaciones Móviles sensibles al contexto. Rev. Fac. Ing. UCV. Vol 24(2), Pp 93-103.

[19] Canelon R. (2011). Un proceso para la Ingenieria del dominio basado en calidad de software. Una aplicación al dominio del Aprendizaje móvil sensible al contexto. InDoCas. Tesis de doctorado en ciencias de la computación. Facultad de ciencias. Universidad central de Venezuela, Febrero 2011.

[20] Harmsen F. (1997). Situational Method Engineering. Doctoral Disertation. Utrecht: Moret Ernst & Young Management Consultants, 1997.

[21] Seligmann P., Wijers G. y Sol H. (1989). Analyzing the structure of I.S. methodologies, an alternative approach. In R, Maes (ed) Proceedings of the First Dutch Conference on Information Systems, Amersfoort, The Netherlands, Año 1989.

[22] Brinkkemper S. (1990). Formalisation of information systems modelling, Ph. D. Thesis, University of Nijmegen, Thesis Publishers, Amsterdam.

[23] Wynekoop J., Russo N. (1993). System development methodologies: unanswered questions and the research practice gap. In Proceedings of the 14th Intl. Conf. Inf. Syst., New York, ACM Pub. Pp 181- 190.

[24] Harmsen A., Brinkkemper J. y Oei J. (1994). Situational Method Engineering for Information System Projects. In Olle T. W. and A. A. Verrijn Stuart (Eds.), Methods and Associated Tools for the Information Systems Life Cycle. Proceedings of the IFIP WG8.1 Working Conference (CRIS'94). North-Holland Amsterdam. Pp. 169-194.

[25] Prakash N. (1994). A Process View of Methodologies, 6th Int. Conf. on Advanced Information Systems Engineering, CAISE'94, Springer Verlag.

[26] Booch G. (1991). Object Oriented Analysis and Design with Applications, Benjamin Cummings Publishing Company, Redwoord, Año 1991.

[27] Prakash N. (1999). On Method Statics and Dynamics. Information Systems. Vol.34, No.8, Año 1999. Pp. 613-637.

[28] Hamar, V. (2004). Aspectos metodológicos del desarrollo y reutilización de componentes de software. Trabajo de Maestria. Universidad de los Andes. Facultad de Ingenieria. Postgrado en Computacion.

[29] Losavio F. Chirinos L. y Matteo A. 2004. Identifying Quality-Based Requirements. Information systems Management (ISYM). Editorial: Auerbach Publications, Vol. 21, nro 1, January 2004. Pp 15-21.

[30] Losavio, F. Matteo, A. y Rahamut, R. (2008). Unifying Quality Standards to Capture Architectural Knowledge for web Services Domain. In Proceedings of the 14th International Conference on Distributed Multimedia Systems, Boston, Massachussetts, USA, Año 2008. Pp 65-70.

[31] Kang K.C., Cohen S.G., Hess J.A., Novak W.E., y Peterson A.S. (1998). Feature-Oriented Domain Analysis (FODA). Feasibility Study. Technical Report CMU/SEI-90-TR21 (ESD-90-TR-222), Software Engineering Institute, Carnegie-Mellon University, Pittsburgh, Pennsylvania 15213, Año 1998.

[32] Hofmeister C., Kruchten P., Nord R., Obbink H., Ran A. y America P. (2007). A general model of software architecture design derived from five industrial approaches. The journal of systems and software. Elsevier inc. Año 2007. Pp 106-126.

[33] Bass L., Clements, P. y Kazman, R. (2003). Software Architecture in Practice. SEI Series in Software Engineering. Third Edition. Boston: Addison-Wesley. Pp 25-37.

[34] Clements, P. Northrop, L.M. (2001). Software Product Lines: Practices and Patterns, Addison Wesley, August, 2001, paginas 608.

[35] Chung L., Supakkul S. (2004). Integrating FRs and NFRs: A Use Case and Goal Driven Approach. In Proceedings of the 2nd Intl. Conference of Software Engineering Research, Management & Applications (SERA04), Los Angeles, CA. May 5-7, 2004. Pp 30-37.

[36] Yu E., Mylopoulos J.: Using goals, rules and methods to support reasoning in Business Process Reengineering, 27th Hawaii International Conference on System Sciences, Maui, Hawaii, (1994).

[37] Losavio F., Matteo A. y Rahamut R. (2008). Web Services Domain Analysis Based on Quality Standars 2nd ECSA, Cyprus, Sept. 29- Oct. 1. R Morrison, D. Balasubramaniam, and K. Falkner (Eds.): ECSA 2008, Springer-Verlag Berlin Heidelberg. LNCS Vol. 5292. Pp 354-358.

[38] Castillo, I., Losavio, F., Matteo, A. y Boegh, J. (2010). REquirements, Aspects and Software Quality: The REASQ Model. Journal of Object Technology (JOT). ETH Publications, Vol. 9, nro. 4, Julio 2010. Pp 69-91.

[39] Kiczales, G., Lamping J., Mendhekar A., Maeda C., Videira-Lopes C., Loingtier J.-M. y Irwin J. (1997). Aspect-Oriented Programming. In proceedings of the 11th European Conference on Object-Oriented Programming (ECOOP 1997), Jyvskyl, Finland, Lecture Notes in Computer Science 1241, Springer-Verlag. Pp 220-242.

[40] Losavio F., Matteo A. y Pacilli I. (2012). Proceso Extendido de Chung como Análisis del Dominio, Identificación de Aspectos y Estándares de Calidad. Memorias del II Simposio Científico y Tecnológico en Computación (SCTC 2012). O. Rodríguez, E. Coto (Eds.) Caracas, Venezuela. 7-9 de mayo 2012. Pp 148-155.

[41] OMG. (2008). Software & Systems Process Engineering Meta-Model Specification, Version 2.0. Formal/2008-04-01, Object Management Group.