Ingeniare. Revista Chilena de Ingeniería
ISSN: 0718-3291
facing@uta.cl
Universidad de Tarapacá
Chile

Erazo Martínez, Jennifer; Florez Gómez, Andrés; Pino, Francisco J.
Generando productos software mantenibles desde el proceso de desarrollo: El modelo de referencia MANTuS
Ingeniare. Revista Chilena de Ingeniería, vol. 24, núm. 3, julio, 2016, pp. 420-434
Universidad de Tarapacá
Arica, Chile

Generando productos software mantenibles desde el proceso de desarrollo: El modelo de referencia MANTuS

Creating maintainable software products from the development process: The reference model MANTuS

Jennifer Erazo Martínez[1] Andrés Florez Gómez[1] Francisco J. Pino[1*]

Recibido 4 de mayo de 2015, aceptado 20 de noviembre de 2015
Received: May 4, 2015 Accepted: November 20, 2015

RESUMEN

El mantenimiento de software es una actividad muy importante y crítica para las empresas que conforman la industria del software. Se puede considerar que el problema de costos durante la etapa de mantenimiento reside en que no se tienen en cuenta aspectos de mantenibilidad durante el desarrollo del producto software. Es por esto que este artículo propone el modelo de referencia MANTuS el cual ofrece buenas prácticas para el proceso de desarrollo de software, las que pretenden que los artefactos software generados, durante las diferentes etapas del ciclo de vida de desarrollo, incluyan las subcaracterísticas de mantenibilidad establecidas por la norma ISO/IEC 25010. Este modelo de referencia contiene una serie de prácticas base, definidas para los procesos de análisis de requisitos de software, diseño arquitectural de software, diseño detallado de software, construcción de software, pruebas de evaluación de software y gestión de la documentación del software. Incluir las prácticas en el proceso de desarrollo podría ayudar a potenciar cada una de las subcaracterísticas de mantenibilidad del producto software descritas en ISO/IEC 25010. El modelo de referencia MANTuS ha sido evaluado por medio de un estudio de caso donde se evidenció que llevar a cabo las prácticas propuestas por el modelo de referencia para el proceso de construcción de software permitió potenciar la mantenibilidad del producto obtenido.

Palabras clave: Modelo de referencia, mantenibilidad de software, subcaracterísticas de mantenibilidad, proceso de desarrollo de software, producto software.

ABSTRACT

Software maintenance is a very important and critical activity for software companies. Problem of costs during the maintenance stage lies on the fact that maintainability aspects are not taken into account during the development of the software product. For this reason, this paper proposes the MANTuS reference model, which provides a set of practices for software development process in hopes that the software artifacts generated during the different development life cycle stages include the maintainability sub-characteristics (established by the standard ISO/IEC 25010). This reference model contains a number of base practices defined for the processes: software requirements analysis, software architectural design, software detailed design, software construction, software qualification testing, and software documentation management. To include practices in the development process could help to maximize each software product maintainability sub-characteristic described in ISO/IEC 25010. Outcomes of apply the MANTuS reference model through a case study shown that using practices for software construction process has allowed to improve the maintainability of the obtained product.

Keywords: Reference model, software maintainability, maintainability sub-characteristics, software development process, software product.

[1] Universidad del Cauca. Grupo IDIS. Facultad de Ingeniería de Electrónica y Telecomunicaciones. Calle 5 Nº 4-70. Popayán, Cauca, Colombia.
E-mail: jderazo@unicauca.edu.co; asflorez@unicauca.edu.co; fjpino@unicauca.edu.co
* Autor de correspondencia.

INTRODUCCIÓN

La mantenibilidad es una de las características de calidad esenciales del producto, debido a que las tareas de mantenimiento consumen una gran proporción del esfuerzo total gastado en el ciclo de vida del software [1]. El costo de esta etapa consume entre el 50% y el 80% de los recursos del proyecto [2] y el 66% de los costos del ciclo de vida del software son invertidos en el mantenimiento del producto [3]. Además, el 61% del tiempo que dedican los programadores al desarrollo es invertido en la etapa de mantenimiento, y solo el 39% es empleado en nuevos desarrollos [4]. Lo anterior refleja que la etapa de mantenimiento: (1) requiere el mayor porcentaje de los costos del ciclo de vida del software, (2) incrementa el esfuerzo realizado, (3) impide que una gran parte del tiempo sea utilizado para nuevos desarrollos.

Para facilitar la ejecución de la etapa de mantenimiento es conveniente tener en cuenta la característica de mantenibilidad del producto software. La norma ISO/IEC 25010 [5] define la mantenibilidad como el grado de efectividad o eficiencia con la que un producto o sistema puede ser modificado y la divide en cinco subcaracterísticas: modularidad (modularity), reusabilidad (reusability), capacidad para ser analizado (analysability), capacidad para ser modificado (modifiability) y capacidad para ser probado (testability).

Incluir durante el proceso de desarrollo estas subcaracterísticas al producto software podría ayudar a incrementar la mantenibilidad, reduciendo así el esfuerzo de mantenimiento, lo que permitiría usar estos mismos recursos para realizar más cambios o lograr los mismos cambios con menos recursos [6]. El tener un modelo de referencia que cuente con un conjunto de procesos que: (i) agrupan buenas prácticas para el desarrollo de software, (ii) permitan incluir en los productos software generados por los atributos que influyen sobre la mantenibilidad y (iii) permitan identificar los atributos que afectan a cada subcaracterística, puede facilitar la inclusión de la mantenibilidad al producto software logrando así conseguir un producto altamente mantenible.

En este sentido, en este artículo se presenta el modelo de referencia denominado MANTuS, creado con el objetivo de apoyar la inclusión de subcaracterísticas de mantenibilidad al producto software durante el proceso de desarrollo. Este modelo consta de seis procesos donde se definen prácticas base con el fin de potenciar cada una de las subcaracterísticas de mantenibilidad (descritas en la ISO/IEC 25010), mediante la inclusión de atributos software que afectan estas subcaracterísticas. El modelo de referencia ha sido definido siguiendo los lineamientos que para tal fin establece la norma ISO/IEC 33004 [7]. Los resultados de aplicar el modelo de referencia MANTuS en un estudio de caso que evidencia que este puede ser útil para potenciar la mantenibilidad del producto software.

Además de la presente introducción, este artículo muestra en la sección 2 los trabajos relacionados y la estrategia de investigación utilizada para la construcción del modelo de referencia MANTuS, en la sección 3 la construcción del modelo de referencia MANTuS, en la sección 4 la evaluación del estudio de caso para evaluar el modelo de referencia y, finalmente, en la sección 5 conclusiones y trabajo futuro.

ANTECEDENTES

Trabajos relacionados

Al realizar una búsqueda en la literatura acerca de la inclusión de características de mantenibilidad al producto software durante el proceso de desarrollo y temas relacionados, se han encontrado diferentes estudios relacionados con metodologías de mantenimiento:

- April, Huffman, Abran y Dumke [8] proponen un modelo de madurez de mantenimiento de software (SMmm) que permite evaluar y mejorar esta etapa. Este modelo describe actividades de mantenimiento que ayudan a identificar posibles mejoras en esta etapa y está diseñado como un complemento para el modelo Capability Maturity Model integration (CMMi). En este modelo se propone reagrupar los procesos de mantenimiento de software en tres grupos: Procesos primarios, que son procesos operacionales de mantenimiento de software, procesos de soporte que apoyan a los procesos primarios y, por último, los procesos organizacionales.
- Polo, Piattini, Ruiz y Calero [9] presentan la metodología MANTEMA para el proceso del

ciclo de vida del mantenimiento, que es una adaptación del estándar ISO/IEC 12207[10]. ISO 12207 es un framework de referencia que cubre todos los aspectos del ciclo de vida del software; sin embargo, no detalla cómo implementar las tareas y actividades incluidas en el proceso. MANTEMA incorpora una serie de actividades bien organizadas de acuerdo al tipo de mantenimiento, apoyando el control de documentación, se definen cinco tipos de mantenimiento: correctivo urgente, correctivo no urgente, perfectivo, preventivo y adaptativo. Además de esto, se considera la participación de tres tipos de organizaciones en el proceso de mantenimiento: el cliente que es el propietario del software y requiere del servicio de mantenimiento; el mantenedor es la organización que cumple con el servicio de mantenimiento; el usuario que utiliza el software mantenido. De igual forma MANTEMA también proporciona plantillas estándar para cada uno de los documentos generados.

- Pino, Ruiz, García y Piattini [4] presentan Agile MANTEMA como una propuesta metodológica para el mantenimiento de software que se centra en las pequeñas organizaciones. Especifica una estrategia de mantenimiento que define qué se necesita hacer, cuándo, cómo y por quién. También establece una serie de elementos como los tipos de mantenimiento, niveles de servicio y niveles de capacidad, con el fin de manejar la complejidad que es inherente al proceso de mantenimiento. Esta metodología permite a las pequeñas compañías definir su propio proceso de mantenimiento, tomando en cuenta sus características y necesidades particulares. El objetivo de Agile MANTEMA es hacer los elementos de proceso descritos en MANTEMA más livianos e incorporarlos en la gestión de proyectos ágiles usando el método Scrum. Esta metodología describe un proceso, niveles de servicio, niveles de desempeño de proceso y niveles de capacidad de proceso.
- Una propuesta que trata el mantenimiento de software en detalle es el modelo de madurez para el mantenimiento correctivo CM3, el que es tratado en [11], donde se sugiere que se deben crear modelos de procesos especializados para cada uno de los tipos de mantenimiento. CM3 es un modelo de procesos especificamente para la categoría de mantenimiento correctivo, que contiene un conjunto de procesos definidos que colaboran entre ellos con el fin de mejorar este tipo de mantenimiento.

Los estudios [4, 8-9, 11-12] tienen como tema principal la etapa de mantenimiento de software, proponen modelos y metodologías de mantenimiento de software, con el fin de controlar y mejorar este proceso, ya que esta etapa es la más costosa y conflictiva del ciclo de vida del software. Los estudios anteriores se centran únicamente en la etapa de mantenimiento del software al final del ciclo de vida del mismo, planteando una serie de actividades organizadas que permiten mejorar esta etapa con el fin de reducir el problema de altos costos que se presenta durante la misma. Sin embargo, estas propuestas no tienen en cuenta la característica de mantenibilidad de software ni plantean soluciones para incorporarla al producto durante el proceso de desarrollo, siendo este el principal objetivo del modelo de referencia MANTuS presentado en este trabajo. Este modelo de referencia de procesos ha sido definido teniendo en cuenta los requerimientos para modelos de referencia de procesos indicados en la norma ISO/IEC 33004, los procesos definidos en este siguen la misma estructura de los presentados en la norma ISO/IEC 15504-5 [13].

Estrategia de investigación

Para la construcción del modelo de referencia MANTuS se utilizó una estrategia de investigación que se apoya en el método Investigación-Acción multiciclo con bifurcación [14-15]. La estrategia de investigación consta de tres ciclos:

- el ciclo de investigación conceptual donde se analizaron los trabajos relacionados con inclusión de características de mantenibilidad al producto software durante el proceso de desarrollo;
- el ciclo de investigación metodológico ejecutado para la definición del modelo de referencia, este ciclo está dividido en tres fases: identificación de los atributos software, clasificación de los atributos software y realización del modelo de referencia. Los resultados obtenidos en las dos primeras fases se encuentran documentados en [16]. En la fase de realización del modelo de referencia se definen las prácticas base (actividades) necesarias para incorporar al producto software cada uno de los elementos identificados con el fin de potenciar las sub-

características de mantenibilidad relacionadas. Se define el modelo de referencia que permita incorporar las características de mantenibilidad de software al producto durante el proceso de desarrollo;
- el ciclo de evaluación ejecutado con el fin de evaluar la idoneidad del modelo de referencia propuesto. Durante este ciclo se realiza la evaluación del modelo de referencia definido mediante un estudio de caso, el que siguió los lineamientos propuestos por Yin [17] y Brereton [18].

MODELO DE REFERENCIA DE PROCESOS MANTuS

Para definir el modelo de referencia MANTuS se realizaron las siguientes actividades: (i) Identificar los atributos de software que influyen en la mantenibilidad del producto, (ii) Agrupar conceptos, (iii) Filtrar atributos, (iv) Clasificar los atributos identificados, (v) Analizar como es posible potenciar cada subcaracterística, (vi) Estructurar el modelo de referencia general.

Como resultado de la ejecución de las cuatro primeras actividades, se obtuvo la clasificacion de 18 atributos software teniendo en cuenta: (i) las subcaracterísticas de mantenbilidad de la ISO/IEC 25010 sobre las que influye, y (ii) el flujo de trabajo de desarrollo de software de RUP [19] en los que se presentan. La intención fue generar una clara relación entre los atributos que influyen en la mantenibilidad, las subcaratersiticas de mantenibilidad que estos apoyan y las etapas del proceso de desarrollo donde estos atributos pueden ser incluidos. La ejecución detallada de estas actividades se encuentra en [20]. Para dar una mejor estructura al modelo de referencia se clasificaron los atributos de acuerdo a los procesos para el desarrollo de software, definidos en el estándar internacional ISO/IEC 15504-5, teniendo en cuenta la clasificación realizada previamente por flujos de trabajo. Los procesos para el desarrollo de software que han tenido en cuenta son: análisis de requisitos de software, diseño arquitectural de software, diseño detallado de software, construcción de software, pruebas de evaluación de software y gestión de la documentación. Fue necesario realizar un análisis semántico de las definiciones de los flujos de trabajo del RUP, los procesos definidos en la ISO/IEC 15504, y los atributos que influyen en la mantenibilidad del producto con el fin de obtener la relación entre estos elementos. El resultado de establecer esta relación y clasificación se muestra en la Tabla 1.

Analizar cómo es posible potenciar cada subcaracterística

El modelo de referencia propuesto sigue el paradigma orientado a objetos ya que los estudios revisados en la literatura proponen atributos relacionados con este paradigma. Para cada una de las cinco subcaracterísticas de mantenibilidad se analizó cómo es posible potenciarla mediante la realización de prácticas que permitan incluir los diferentes atributos que influyen sobre la misma, dichas prácticas son separadas de acuerdo al proceso en el cual deben realizarse, todo esto teniendo en cuenta la clasificación desarrollada (la cual se presenta en la Tabla 1).

A manera de ejemplo, a continuación, se presenta el análisis realizado para incorporar el atributo acoplamiento con el fin de potenciar la subcaracterística de reusabilidad, es importante que el software cuente con bajo acoplamiento. Para lograr esto, durante el proceso de diseño se debe descomponer el sistema en pequeñas partes de funcionalidades (módulos) que tengan la menor interrelación posible. Para que estos módulos (paquetes, clases, métodos) sean fáciles de reutilizar es oportuno abstraer la funcionalidad de dichos módulos, así los módulos podrían ser reutilizados en diferentes contextos fuera del sistema original. Además de esto se debe reducir el número de respuestas por clases. Cada uno de los análisis realizados para relacionar cada atributo identificado con las correspondientes subcaracterísticas se encuentran descritos detalladamente en [20].

Descripción general del modelo

A continuación se presenta una descripción del modelo de referencia MANTuS. Esta sección contiene la declaración del objetivo del modelo de referencia, el contexto previsto de uso y las acciones que fueron tomadas para alcanzar el consenso.

Objetivo

El objetivo del modelo de referencia MANTuS es especificar, en términos de sus propósitos y sus resultados, un conjunto de procesos que permitan incluir atributos de mantenibilidad al producto

Tabla 1. Clasificación de los atributos por subcaracterísticas de mantenibilidad y procesos.

Subcaracterísticas / Atributos	CA	M	CM	CP	R	Procesos
Acoplamiento	X	X	X	X	X	Diseño arquitectural, Diseño detallado
Anidación	X		X	X		Construcción de software
Capacidad de expansión			X			Operación de software
Cohesión	X	X	X	X	X	Diseño detallado
Comentarios	X		X		X	Construcción de software
Complejidad	X	X	X	X	X	Diseño arquitectural, Diseño detallado, Construcción de software
Consistencia	X		X			Análisis de requisitos, Diseño arquitectural, Diseño detallado, Construcción de software, Pruebas de evaluación de software, Operación de software
Documentación	X		X	X	X	Análisis de requisitos, Diseño arquitectural, Diseño detallado, Construcción de software, Pruebas de evaluación de software, Operación de software
Duplicación	X		X	X		Construcción de software
Encapsulamiento	X	X	X	X		Diseño detallado
Estandarización	X		X			Construcción de software
Facilidad de lectura	X		X	X	X	Construcción de software
Facilidad de entendimiento	X		X	X	X	Diseño arquitectural, Diseño detallado, Construcción de software
Herencia	X		X	X	X	Diseño detallado
Polimorfismo	X		X		X	Diseño detallado
Simplicidad	X	X	X	X	X	Análisis de requisitos, Diseño arquitectural, Diseño detallado, Construcción de software
Tamaño	X		X	X	X	Construcción
Trazabilidad	X		X			Análisis de requisitos, Diseño arquitectural, Diseño detallado, Construcción de software, Pruebas de evaluación de software

CA = Capacidad para ser analizado, M = Modularidad, CM = Capacidad para ser modificado, CP = Capacidad para ser probado, y R = Reusabilidad.

software durante el proceso de desarrollo con el fin de potenciar esta característica y que sean aplicables en el contexto de empresas desarrolladoras de software. Al igual que otros modelos de referencia el alcance de este es la descripción concreta de los procesos y no la especificación detallada de los elementos particulares de la implementación, es decir, la descripción de los procesos establece lo que se debe lograr mas no determina cómo debe lograrse. Es por esto que los procesos pueden ser implementados de diferentes formas obteniendo el mismo resultado.

Procesos del modelo

El modelo de referencia MANTuS asume que la mantenibilidad de software es un aspecto fundamental del producto que debe ser considerado desde el incio del ciclo de vida del mismo. Es por esto que se establecen los siguentes seis procesos: Análisis de requisitos de software desde la perspectiva de mantenibilidad, Diseño arquitectural de software desde la perspectiva de mantenibilidad, Diseño detallado de software desde la perspectiva de mantenibilidad, Construcción de software desde la perspectiva de mantenibilidad, Pruebas de evaluación de software desde la perspectiva de mantenibilidad, Gestión de la documentación desde la perspectiva de mantenibilidad. En la Tabla 2 se presenta para cada proceso del modelo de referencia su nombre, identificador y propósito. Además, las descripciones de los procesos del modelo de referencia MANTuS se hacen en términos de sus propósitos y resultados. Los enunciados de los propósitos y resultados de los procesos son elementos obligatorios según el estandar ISO/IEC 33004. La descripción de los procesos del modelo de referencia sigue la estructura de los procesos del estándar internacional ISO/IEC 15504-5. Estas descripciones incorporan el enunciado del

Tabla 2. Procesos que componen el modelo de referencia.

Proceso	Identificador	Propósito
Análisis de requisitos de software desde la perspectiva de mantenibilidad.	DEV-MANT.1	Establecer los requerimientos de los componentes del software teniendo en cuenta atributos que ayuden a potenciar la característica de mantenibilidad.
Diseño arquitectural de software desde la perspectiva de mantenibilidad	DEV-MANT.2	Proveer un diseño para el software, que permita incluir atributos de mantenibilidad al producto.
Diseño detallado de software desde la perspectiva de mantenibilidad.	DEV-MANT.3	Proporcionar un diseño para el software que sea lo suficientemente detallado que permita incluir atributos de software relacionados con la mantenibilidad del producto.
Construcción de software desde la perspectiva de mantenibilidad.	DEV-MANT.4	Producir unidades de software ejecutables que incluyan atributos de software para potenciar la mantenibilidad del producto.
Pruebas de evaluación de software desde la perspectiva de mantenibilidad.	DEV-MANT.5	Que la unidad de prueba incluya atributos que favorezcan a la mantenibilidad del producto.
Gestión de la documentación de software desde la perspectiva de mantenibilidad.	SUP-MANT.1	Desarrollar y mantener un registro de la información del software teniendo en cuenta atributos que permitan potenciar la mantenibilidad del producto.

propósito del proceso y el conjunto de resultados del proceso. El enunciado del propósito describe a un alto nivel el objetivo general de llevar a cabo el proceso. El conjunto de resultados del proceso describe los elementos con los que se demuestra el logro exitoso de su propósito, como la producción de un artefacto, un cambio significativo de estado o el cumplimiento de restricciones especificadas.

Relaciones entre los procesos del modelo

El modelo de referencia que apoya la inclusión de subcaracterísticas de mantenibilidad al producto software durante el proceso de desarrollo está compuesto por dos vistas: la vista específica para cada una de las cinco subcaracterísticas y la vista general para la característica de mantenibilidad, la cual agrupa las vistas específicas. En la Figura 1 se observa la vista general del modelo de referencia con el número de prácticas base definidas para cada proceso de las subcaracterísticas de mantenibilidad. El modelo de referencia obtenido se basa en la información y análisis realizado en cada una de las actividades anteriores.

Los seis procesos que conforman el modelo de referencia MANTuS deben ser entendidos desde una vista general. Estos procesos no presentan relación entre sí, son independientes, es decir, para alcanzar los resultados de un proceso no es necesario haber logrado los resultados de otro u otros procesos, lo que da flexibilidad al modelo de referencia. En la Figura 1 se observa que estos procesos son independientes y que el proceso de Gestión de la documentación, desde la perspectiva de mantenibilidad es transversal a los otros, ya que este es un proceso de soporte que puede ser ejecutado durante los otros cinco procesos. La vista general del modelo de referencia combina y sintetiza las prácticas base definidas en los procesos de todas las subcaracterísticas de mantenibilidad.

Comunidad de interés y Contexto previsto de uso

El modelo de referencia MANTuS fue construido para ser aplicado en empresas desarrolladoras de software que deseen garantizar la mantenibilidad de sus productos durante el proceso de desarrollo. Además, empresas que estén interesadas en obtener la certificación de mantenibilidad del Sistema de Gestión de la Calidad del Producto Software ISO 25000 de AENOR [21-22].

Los atributos identificados, sobre los cuales se ha basado el modelo de referencia, están relacionados directamente con las propiedades a evaluar en un producto software por el esquema de evaluación de mantenibilidad ISO 25000 realizada por AQC Lab [23]. Dichas propiedades están relacionadas con el incumplimiento de reglas, la documentación del código, la complejidad, la estructuración, el tamaño de los métodos, el código duplicado, el nivel de acoplamiento, el balance inestabilidad-abstracción, los ciclos y la cohesión. En el trabajo de clasificación de atributos, ejecutado realizado en esta investigación, se realizó una comparación

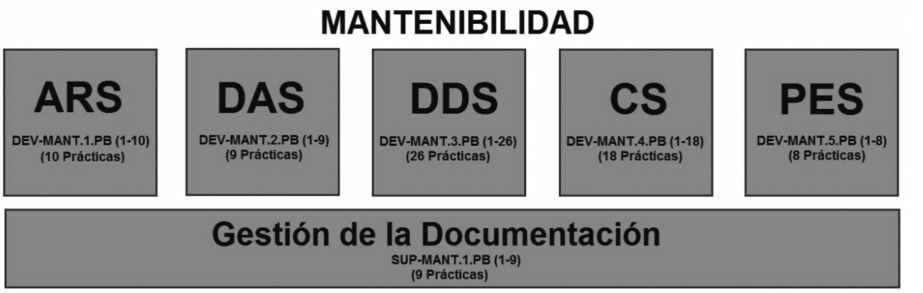

Figura 1. Vista general del modelo de referencia (ARS= Análisis de requisitos de software, DAS= Diseño arquitectural de software, DDS= Diseño detallado de software, CS= Construcción de software, PES= Pruebas de evaluación de software).

de los atributos de mantenibilidad identificados versus las propiedades de mantenibilidad descritas por el esquema de evaluación. Del análisis de esta comparación se puede establecer de manera general que los atributos facilidad de entendimiento, complejidad y simplicidad guardan relación con todas las propiedades de mantenibilidad, mientras que los atributos comentarios, encapsulamiento, estandarización, herencia y polimorfismo tienen relación con un menor número de propiedades. También se observó que la propiedad complejidad tiene relación con todos los atributos identificados; sin embargo, la propiedad balance inestabilidad-abstracción presenta menor relación con estos. Por otra parte, la comparación realizada permitió comprobar que todos los atributos identificado; están relacionados con a lo menos dos de las propiedades de mantenibilidad, y además que cada una de estas propiedades se ve influenciada por al menos seis de los atributos definidos.

El modelo de referencia MANTuS fue construido para ser usado en contextos de desarrollo de software donde sea importante incluir al producto software atributos de mantenibilidad para potenciar esta característica. Asimismo, la especificación de los resultados permite determinar aspectos a incluir o mejorar en la implementación de los procesos existentes.

Descripción detallada de los procesos

Debido a restricciones de espacio, en esta sección solo se presenta el detalle del proceso *construcción de software desde la perspectiva de mantenibilidad* (ver Tabla 3), los demás procesos y el modelo de referencia MANTuS completo se puede encontrar en [20]. Es importante resaltar que en la sección resultados del proceso cada resultado está asociado a las subcaracterísticas de mantenibilidad en las que influye, siendo CA= capacidad para ser analizado; M=modularidad; CM=capacidad para ser modificado; CP=capacidad para ser probado; R=reusabilidad. En la sección prácticas base se proporciona la definición de las actividades y las tareas necesarias para alcanzar el propósito del proceso y cumplir con los resultados del mismo; cada práctica base está asociada a uno o más resultados. Por último, cada proceso describe los productos de trabajo de salida (PTS) correspondientes, los que están relacionados con los diferentes resultados de proceso.

EVALUACIÓN DEL MODELO DE REFERENCIA MANTuS

La evaluación del modelo de referencia MANTuS se realizó mediante un estudio de caso que permitió la aplicación del proceso de Construcción de software desde la perspectiva de mantenibilidad. Este estudio de caso fue realizado con ocho desarrolladores de software que ejecutaron cambios a dos versiones del mismo programa: una versión que evidencia la utilización de algunas prácticas base del proceso de construcción del modelo de referencia MANTuS (código B) y otra que no las utiliza (código A). Para la ejecución de este estudio de caso se tomó como referencia la plantilla protocolo para planeación de estudios de caso planteado en [18]. A continuación, se describe cada uno de los pasos realizados en el estudio de caso, antecedentes, diseño, sujetos de investigación y unidad de análisis, procedimiento de campo, preparación de recolección de datos, ejecución de caso y análisis.

Tabla 3. Proceso de construcción de software que compone el modelo de referencia.

Identificador del proceso	DEV-MANT.4

Nombre del proceso

Construcción de software desde la perspectiva de mantenibilidad.

Propósito del proceso

El propósito del proceso de construcción de software desde la perspectiva de mantenibilidad es producir unidades de software ejecutables que incluyan atributos de software para potenciar la mantenibilidad del producto.

Resultados del proceso

Como resultado de la implementación exitosa del proceso de construcción de software desde la perspectiva de mantenibilidad:

		CA	**M**	**CM**	**CP**	**R**
a)	Las unidades de software cuentan con buenos comentarios.	X		X		X
b)	Las unidades de software tienen baja complejidad.	X	X	X	X	X
c)	Las unidades de software son fáciles de entender.	X		X	X	X
d)	Las unidades de software cuentan con alta facilidad de lectura.	X		X	X	X
e)	Las unidades de software son simples.	X	X	X	X	X
f)	Las unidades de software tienen el tamaño adecuado.	X		X	X	X
g)	Las unidades de software cuentan con bajo nivel de anidación.	X		X	X	
h)	Las unidades de software tienen un estándar definido.	X		X		
i)	Las unidades de software evitan la duplicación.	X		X	X	
j)	Las unidades de software son consistentes.	X		X		
k)	Las unidades de software cuentan con trazabilidad.	X		X		

Prácticas base

DEV-MANT.4.PB1: Utilizar comentarios. Hacer uso de comentarios adecuados en el código fuente. [Resultados: a, c, d]
DEV-MANT.4.PB2: Describir propósitos. Explicar el propósito de las funciones, subrutinas, variables y constantes. [Resultados: a, c, d]
DEV-MANT.4.PB3: Verificar los flujos de control. Incluir solamente los flujos de control necesarios. [Resultados: b, c, e, f]
DEV-MANT 4.PB4: Organizar código. El código fuente debe ser indentado. [Resultado: c, d]
DEV-MANT.4.PB5: Asignar nombres claros. Dar nombres descriptivos a las variables. [Resultado: c, d]
DEV-MANT.4.PB6: Controlar abreviaciones. Hacer poco uso de abreviaciones. [Resultado: c, d]
DEV-MANT.4.PB7: Controlar sentencias. Evitar sentencias largas en el código fuente, es mejor mantener las líneas de código cortas, aunque esto implica separar las sentencias en múltiples líneas. [Resultado: c, d]
DEV-MANT.4.PB8: Controlar paréntesis. Hacer correcto uso de los paréntesis para mejorar la facilidad de lectura de las expresiones aritméticas y lógicas. [Resultado: c, d]
DEV-MANT.4.PB9: Determinar funcionalidades. Implementar solamente las funcionalidades necesarias. [Resultados: b, c, e, f]
DEV-MANT.4.PB10: Dividir métodos. Mantener métodos simples, es decir, dividir los métodos que tengan muchas condiciones. [Resultados: b, c, e, f]
DEV-MANT.4.PB11: Controlar tamaño. Evitar que el tamaño del código crezca innecesariamente. [Resultados: c, f]
DEV-MANT.4.PB12: Controlar nivel de anidación. Evitar las estructuras de control anidadas innecesarias, ya que estas son más complejas que las estructuras de control secuenciales. [Resultados: b, c, e, f, g]
DEV-MANT.4.PB13: Evitar duplicación. Detectar código duplicado, extrayéndolo en un nuevo procedimiento y reemplazando todas las instancias del código duplicado por llamadas al nuevo procedimiento. [Resultados: b, e, f, i]
DEV-MANT.4.PB14: Definir estándares. Tener un conjunto de estándares de programación en la escritura de código para evitar la individualidad entre los programadores. [Resultados: d, h]
DEV-MANT.4.PB15: Utilizar convenciones. Hacer uso de convenciones estándar para el nombrado de paquetes, clases, métodos y variables. [Resultados: d, h]
DEV-MANT.4.PB16: Controlar tamaño de unidad. Las piezas de funcionalidad de más bajo nivel deben mantenerse pequeñas para que sean centradas y fáciles de entender. [Resultados: b, c, e, f]
DEV-MANT.4.PB17: Verificar trazabilidad. Verificar periódicamente que los artefactos de una etapa sean consistentes con artefactos de la etapa anterior. [Resultados: c, j, k]
DEV-MANT.4.PB18: Actualizar artefactos. Actualizar los artefactos cuando se presenten cambios. [Resultados: c, j, k]

Producto de trabajo de salida

PTS-06 Unidad de software que considera la mantenibilidad [Resultados: a, b, c, d, e, f, g, h, i, j, k]

PTS-04 Registro de trazabilidad [Resultados: j, k]

Antecedentes

Aunque existen diversos enfoques que abordan la mantenibilidad se debe resaltar que el área de aplicación de esta investigación es la inclusión de prácticas a las diferentes etapas del proceso de desarrollo que permitan potenciar la mantenibilidad del producto. Teniendo en cuenta lo anterior, la pregunta principal de investigación que se pretende responder con este estudio de caso es:

PP: ¿La utilización de prácticas base del proceso de construcción del modelo de referencia MANTuS permite potenciar la mantenibilidad del producto?

Otras preguntas adicionales que se pueden plantear son:

PA1: ¿Aumenta la tasa de éxito para realizar cambios correctamente con la utilización de prácticas base del proceso de construcción del modelo de referencia MANTuS?

PA2: ¿La utilización de prácticas base del proceso de construcción del modelo de referencia MANTuS permite que el tiempo invertido al realizar cambios al producto software genere los resultados esperados?

PA3: ¿Utilizar las prácticas base del proceso de construcción del modelo de referencia MANTuS permite aumentar la tasa de éxito en el encuentro de la causa de fallos?

PA4: ¿Disminuye el tiempo de análisis de fallos con la utilización de prácticas base del proceso de construcción del modelo de referencia MANTuS?

Diseño

Según el enfoque propuesto en [17], el tipo de diseño del estudio de caso para esta investigación es simple-embebido. Simple porque se tiene un único contexto para el mismo estudio de caso y embebido porque son dos unidades de análisis (dos versiones del mismo código fuente, una versión que incorpora aspectos de aplicar las prácticas del proceso y la otra que no los incorpora). Este estudio de caso compara los resultados obtenidos al abordar las dos unidades de análisis. Por otro lado, las medidas usadas para indagar sobre las preguntas de investigación definidas son: (i) número de funcionalidades modificadas correctamente por todos los participantes a un código fuente creado sin utilizar las prácticas y a otro código fuente que evidencia la utilización de las prácticas descritas en el proceso de *Construcción de software desde la perspectiva de mantenibilidad* del modelo de referencia, (ii) el tiempo requerido para realizar cambios a un código sin prácticas y a un código que evidencia el uso de las prácticas definidas en el modelo de referencia, relacionado con la subcaracterística capacidad para ser modificado, (iii) tasa de éxito en el encuentro de la causa de los fallos a un código sin prácticas y a un código que evidencia el uso de las prácticas definidas en el modelo de referencia, relacionado con la subcaracterística capacidad para ser analizado, (iv) tiempo de análisis de fallos de un código sin prácticas y de un código que evidencia el uso de las prácticas definidas en el modelo de referencia, relacionado con la sub-característica capacidad para ser analizado.

Sujetos de investigación, Unidad de análisis, Procedimiento de campo y Preparación para la recolección de datos

Los sujetos de investigación participantes en este estudio de caso fueron ocho desarrolladores de software (estudiantes de últimos semestres del programa Ingeniería de Sistemas de la Universidad del Cauca). Para este estudio de caso se tuvo dos unidades de análisis las que son dos versiones del mismo programa: una versión que evidencia la utilización de prácticas base del proceso de construcción del modelo de referencia MANTuS (código B) y otra que no las utiliza (código A). El programa está implementado en C++ y simula algunas de las funcionalidades que se pueden realizar en un banco, para esto permite crear una cuenta bancaria que puede ser de dos tipos: corriente o de ahorros, cada una de las que realiza diferentes transacciones.

Las prácticas evidenciadas por el código B son las siguientes:

- DEV-MANT.4.PB1: Utilizar comentarios
- DEV-MANT.4.PB2: Describir propósitos
- DEV-MANT.4.PB3: Verificar los flujos de control
- DEV-MANT.4.PB5: Asignar nombres claros
- DEV-MANT.4.PB6: Controlar abreviaciones
- DEV-MANT.4.PB10: Dividir métodos
- DEV-MANT.4.PB11: Controlar tamaño
- DEV-MANT.4.PB12: Controlar nivel de anidación

- DEV-MANT.4.PB13: Evitar duplicación
- DEV-MANT.4.PB14: Definir estándares
- DEV-MANT.4.PB15: Utilizar convenciones
- DEV-MANT.4.PB16: Controlar tamaño de unidad

Para llevar a cabo el estudio de caso, se estableció el siguiente protocolo: contextualización a los participantes, entrega de instrumentos para la realización del estudio de caso, ejecución de código a modificar, inicio de aplicación, entrega de resultados y diligenciamiento de encuesta.

Para la recolección de datos se utilizó una guía donde se indican las diferentes modificaciones a realizar y se presentan unas plantillas para ingresar valores de tiempo invertido ejecutar realizar cada tarea de mantenimiento y describir opiniones respecto a las modificaciones realizadas. La guía consta de 8 ítemes, de los cuales, el ítem 6 plantea cambio correctivo y los ítemes 1, 2, 3, 4, 7, 8 cambios perfectivos, además el ítem 5 se relaciona con el encuentro de causas de fallos. Esta guía permitió evaluar el tiempo invertido en cada cambio por cada participante, el tiempo de análisis del código y si los cambios fueron efectivos o no.

También se utilizó una encuesta en Google Drive que fue diligenciada por los participantes al finalizar todas las modificaciones propuestas. La realización de la encuesta tiene el objetivo de conocer las opiniones de los participantes con respecto a características de mantenibilidad de cada uno de los códigos (códigos A y B), como capacidad para ser analizado y modificado. A manera de ejemplo, para evaluar la capacidad para ser analizado del producto, algunas de las preguntas que se hicieron son las siguientes:

a) ¿Le pareció fácil encontrar las causas de fallos? ¿Por qué?
b) ¿Le resultó fácil entender el código? ¿Por qué?
c) ¿Las variables tenían nombres claros y entendibles?

Las respuestas de los participantes a esta encuesta soportaron los resultados cuantitativos obtenidos a partir de los tiempos registrados en las guías. De manera similar se realizaron preguntas para evaluar la subcaracterística capacidad para ser modificado.

Ejecución del caso

La aplicación del estudio de caso siguió el procedimiento de campo mencionado anteriormente:

- Contextualización a los participantes, donde se explicó el objetivo del estudio de caso y la idea general del programa a modificar. Se indicó cómo se iba a desarrollar el ejercicio aplicativo y cuáles eran los artefactos a entregar, es decir, el código con las modificaciones, la guía y encuesta diligenciadas.
- Entrega de instrumentos para la realización del estudio de caso, los participantes se dividen en dos grupos: Grupo A que trabajan con el código A el que no incluye prácticas de mantenibilidad y Grupo B que trabajan con el código B el que sí incluye algunas prácticas de mantenibilidad definidas para el proceso de construcción. Posteriormente se le entrega a cada participante la guía del trabajo a realizar y el código al que le debe realizar las modificaciones. Por último se le envía al correo el enlace de la guía a diligenciar.
- Ejecución de código a modificar: cada participante tiene cinco minutos para ejecutar el código entregado, con el fin de que verifique todas las funcionalidades del programa.
- Inicio de aplicación: los participantes desarrollan todos los ítemes de la guía entregada, para cada ítem llevan el control del tiempo invertido llenando la tabla que se encuentra después de la descripción del ítem. Los participantes indican mediante comentarios las líneas de código en las que modifican. Los investigadores resuelven las dudas que tengan los participantes al diligenciara y realizar los ítemes de la guía.
- Entrega de resultados: cuando el participante termine todos los ítemes propuestos en la guía, entrega el código con las modificaciones realizadas y la guía totalmente diligenciada.
- Diligenciamiento de encuesta: una vez el participante entregue el código y la guía, este diligencia la encuesta que se le envió al correo anteriormente, respondiendo a todas las preguntas que se encuentran en ella.

Con los tiempos registrados por los participantes al realizar las modificaciones solicitadas, se elaboraron dos tablas resúmenes con los tiempos invertidos (Duración) por cada uno de ellos y ¿Correcto? que comprende los valores sí o no, los cuales indican que la modificación fue realizada con éxito o no, respectivamente. En la Tabla 4 se muestran los tiempos invertidos por los participantes al realizar las modificaciones de los ítemes 1, 2, 3, 4, 6, 7,

Tabla 4. Tiempo modificación.

Participante		ítem 1	ítem 2	ítem 3	ítem 4	ítem 6	ítem 7	ítem 8	Tiempo total
PA1	Duración	00:27:22	00:46:18	00:02:24	00:03:47	00:16:57	00:05:29	00:01:26	1:43:43
	¿Correcto?	Sí	No	No	Sí	No	No	No	
PA2	Duración	00:55:55	00:27:40	00:12:05	00:02:00	00:08:07	00:05:10	00:01:05	1:52:02
	¿Correcto?	No	No	No	Sí	No	No	No	
PA3	Duración	00:53:40	00:31:50	00:08:40	00:05:44	00:08:08	00:12:00	00:00:42	2:00:44
	¿Correcto?	Sí	No	No	Sí	No	No	No	
PA4	Duración	00:38:42	00:20:38	00:18:52	00:02:21	00:05:00	00:06:10	00:06:20	1:38:03
	¿Correcto?	Sí	Sí	Sí	Sí	No	No	No	
Grupo A	T. total	02:55:39	02:06:26	00:42:01	00:13:52	00:38:12	00:28:49	00:09:33	
	T. promedio	00:43:55	00:31:36	00:10:30	00:03:28	00:09:33	00:07:12	00:02:23	
PB1	Duración	00:36:00	00:17:05	00:16:48	00:03:12	01:01:51	00:10:12	00:14:59	2:40:07
	¿Correcto?	Sí	Sí	Sí	Sí	Sí	Sí	Sí	
PB2	Duración	00:23:00	00:21:40	00:15:56	00:05:56	00:27:30	00:15:00	00:08:00	1:57:02
	¿Correcto?	Sí	Sí	Sí	Sí	Sí	Sí	Sí	
PB3	Duración	00:19:25	00:30:33	00:20:11	00:01:51	00:47:10	00:02:07	00:04:42	2:05:59
	¿Correcto?	Sí	Sí	Sí	Sí	Sí	Sí	Sí	
PB4	Duración	00:10:10	00:21:40	00:21:45	00:02:12	00:56:39	00:04:04	00:16:22	2:12:52
	¿Correcto?	Sí	Sí	Sí	Sí	Sí	Sí	Sí	
Grupo B	T. total	01:28:35	01:30:58	01:14:40	00:13:11	03:13:10	00:31:23	00:44:03	
	T. promedio	00:22:09	00:22:44	00:18:40	00:03:18	00:48:17	00:07:51	00:11:01	

8, y el tiempo total al realizar todos los cambios. Además la tabla incluye el cálculo del tiempo total y promedio invertido por todos los participantes para cada uno de los ítemes.

En la Tabla 5 se presenta para cada participante los tiempos invertidos al realizar el análisis del código para los ítems 4, 5, 7, 8 y el tiempo total.

Análisis de resultados

Para dar respuesta a las preguntas planteadas en este estudio de caso se realizó el análisis presentado a continuación.

PA1: ¿Aumenta la tasa de éxito para realizar cambios correctamente con la utilización de prácticas base del proceso de construcción del modelo de referencia MANTuS?

Para cada grupo se halló la tasa de éxito de modificación total teniendo en cuenta los siete ítems de la guía relacionados con la realización de cambios. Este cálculo fue encontrado teniendo en cuenta el total de cambios a realizar por todos los participantes, el total de cambios realizados con éxito y sin éxito. Partiendo de este análisis se obtuvo como resultado general que la tasa de éxito de modificación es mucho mayor para el código B (100%) que para el A (42,9%), lo que evidencia que el incluir las prácticas del modelo de referencia definidas para el proceso de construcción aumenta la tasa de éxito de realizar los cambios correctamente.

PA2: ¿La utilización de prácticas base del proceso de construcción del modelo de referencia MANTuS permite que el tiempo invertido al realizar cambios al producto software genere los resultados esperados?

El tiempo total invertido para realizar todos los cambios al producto fue hallado para cada participante sumando los tiempos finales empleados en el desarrollo de cada ítem de la guía tanto en modificación como en análisis. Los datos hallados indican que el participante del grupo A que realizó los cambios en menor tiempo lo hizo en 02:05:53 (hh/mm/ss) mientras que para el grupo B el participante con menor tiempo utilizó 02:06:05 (hh/mm/ss). A pesar de que estos dos tiempos difieren por poco (00:00:12), la tasa de éxito del participante del grupo A (57,1%) es considerablemente menor que la del

Tabla 5. Tiempo de análisis.

Participante		Ítem 4	Ítem 5.1	Ítem 5.2	Ítem 5.3	Ítem 7	Ítem 8	Tiempo total
PA1	Duración	00:00:50	00:18:39	00:09:43	00:02:10	00:00:51	00:00:30	0:32:43
	¿Correcto?	Sí	Sí	Sí	Sí	No	No	
PA2	Duración	00:01:10	00:16:55	00:01:15	00:02:10	00:00:50	00:00:55	0:23:15
	¿Correcto?	Sí	No	No	No	No	No	
PA3	Duración	00:01:40	00:07:59	00:04:04	00:03:00	00:02:41	00:01:07	0:20:31
	¿Correcto?	Sí	No	No	No	No	No	
PA4	Duración	00:00:50	00:05:27	00:09:53	00:08:10	00:01:40	00:01:50	0:27:50
	¿Correcto?	Sí	No	No	No	No	No	
PB1	Duración	00:00:30	00:48:46	00:01:08	00:00:51	00:00:10	00:05:20	0:56:45
	¿Correcto?	Sí	Sí	Sí	Sí	Sí	Sí	
PB2	Duración	00:00:48	00:05:00	00:01:40	00:00:30	00:00:05	00:01:00	0:09:03
	¿Correcto?	Sí	Sí	Sí	Sí	Sí	Sí	
PB3	Duración	00:00:47	00:13:50	00:01:15	00:01:11	00:01:19	00:00:52	0:19:14
	¿Correcto?	Sí	Sí	Sí	Sí	Sí	Sí	
PB4	Duración	00:00:37	00:05:45	00:00:00	00:01:28	00:01:40	00:00:43	0:10:13
	¿Correcto?	Sí	Sí	Sí	Sí	Sí	Sí	

participante del grupo B (100%), lo que indica que el tiempo invertido por el participante del grupo A no se ve reflejado en los resultados obtenidos al realizar las modificaciones. Al realizar este mismo análisis con el participante que obtuvo el segundo menor tiempo en cada uno de los grupos, se puede notar que el participante del grupo B invirtió mayor tiempo (02:23:03) con respecto al participante del grupo A (02:15:17), sin embargo, logró realizar correctamente todos los cambios solicitados, lo que se ve reflejado en su tasa de éxito (100%) que es mucho mayor a la del participante PA2 (14,3%).

Haciendo el mismo análisis para los participantes restantes de los dos grupos, se observa el mismo comportamiento, lo cual indica que la inclusión de las prácticas base del modelo de referencia al proceso de construcción permite que el tiempo invertido al realizar cambios al producto software genere los resultados esperados.

PA3: ¿Utilizar las prácticas base del proceso de construcción del modelo de referencia MANTuS permite aumentar la tasa de éxito en el encuentro de la causa de fallos?

La tasa de éxito en el encuentro de causa de fallos (TEF) fue hallada para cada uno de los participantes teniendo en cuenta los resultados del ítem relacionado con el análisis de fallos planteado en la guía. Para esto se tuvo en cuenta la métrica tasa de éxito en el encuentro de fallos planteada en la norma ISO/IEC 9126 con la fórmula (1):

$$TEF = 1 - (A/B) \quad (1)$$

$0 \leq TEF \leq 1$, entre más cercano a 1 mejor.

Donde:

A = número de fallos cuyas causas no se han encontrado
B = número de fallos registrados.

Los valores obtenidos para TEF indican que la tasa de éxito en el encuentro de causa de fallos para el grupo A es del 25% y para el grupo B es 100%, lo que permite verificar que incluir las prácticas base del modelo de referencia al proceso de construcción permite aumentar la tasa de éxito en el encuentro de la causa de fallos.

PA4: ¿Disminuye el tiempo de análisis de fallos con la utilización de prácticas base del proceso de construcción del modelo de referencia MANTuS?

El tiempo de análisis de fallos se halló haciendo uso de la métrica tiempo de análisis de fallo establecida en la norma ISO/IEC 9126 que se calcula mediante la fórmula (2):

$$TAF - Sum(T)/N \quad (2)$$

$0 \leq TAF$, entre más pequeño sea TAF mejor.

Para hallar el tiempo (T) se hace uso de la fórmula (3):

$$T = Tout - Tin \quad (3)$$

Donde:

Tout = tiempo en el que las causas del fallo son encontradas
Tin = tiempo en el que el reporte del fallo es recibido
N: número de fallos registrados.

Debido a que el número de fallos cuyas causas se deben hallar para cada ítem planteado en la guía es uno, al reemplazar N la métrica queda como se muestra en la fórmula (4):

$$TAF = Tour - Tin \quad (4)$$

Los valores obtenidos para TAF indican que el tiempo de análisis de fallos es menor en cada ítem de la guía para todos los participantes del grupo B respecto a los del grupo A, lo que indica que el tiempo de análisis de fallos disminuye con la inclusión de las prácticas base del modelo de referencia al proceso de construcción.

En general, se puede observar que el incluir al código prácticas como documentación, nombrado coherente, controlar abreviaciones, determinar y separar funcionalidades, ayudaron a los participantes del grupo B a realizar correctamente todas las modificaciones planteadas en la guía, pues como ellos mismos lo resaltaron este tipo de prácticas les ayudó a entender, leer, analizar mejor el código para encontrar la parte a modificar y asimismo realizar dichos cambios. Por otra parte se analiza que el código A al no estar programado con este tipo de prácticas generó mayor dificultad en la modificación del código, ya que como los participantes de este grupo lo indicaron, no pudieron entender ni leer bien el código, lo que hizo que los cambios no se hicieran de forma correcta. Es por esto que se puede decir que la utilización de prácticas base del proceso de construcción del modelo de referencia MANTuS permite potenciar la mantenibilidad del producto.

Plan de validez

Para disminuir las amenazas a la validez del estudio de caso se consideraron los siguientes aspectos:

Validez de conclusión: para evitar el riesgo de variación en los resultados debido a diferencias individuales de los sujetos de investigación los participantes elegidos fueron desarrolladores de software (estudiantes de últimos semestres de Ingeniería de Sistemas de la Universidad del Cauca) los que tenían conocimientos adecuados de programación orientada a objetos y manejo del lenguaje C++. Lo anterior permitió reducir la heterogeneidad en el grupo de estudio ya que ellos tenían conocimientos y antecedentes similares.

Validez de constructo: en la recolección de datos se tuvieron en cuenta tanto medidas de tiempo como las observaciones individuales de los participantes para cada modificación, lo que permitió hacer un análisis más completo comparando estos dos aspectos.

Validez interna: la guía planteaba modificaciones sencillas para evitar que los participantes se aburrieran o cansaran rápidamente, evitando así que se afectaran los resultados del estudio. De igual forma, las plantillas de recolección de datos fueron diseñadas en forma sencilla y clara para evitar confusión.

Limitaciones del estudio

La selección de los participantes podría reducir la validez externa del estudio de caso ya que al ser estudiantes, los sujetos no son seleccionados de una población general, lo cual podría limitar el poder de generalización de los resultados obtenidos. El número de participantes en el estudio de caso podría ser insuficiente para generalizar los resultados obtenidos. A pesar de que todos los participantes en el estudio de caso contaban con conocimientos previos tanto en orientación a objetos y manejo del lenguaje C++, se observó inconformidad respecto al lenguaje seleccionado para el estudio, ya que muchos comentaron que hace mucho tiempo no trabajan con él.

Discusión

Los resultados obtenidos a partir del análisis realizado se ven soportados por la opinión dada por los participantes a las preguntas realizadas en la encuesta, todos los participantes que modificaron el código A manifestaron que este no tiene alta mantenibilidad por diferentes aspectos como: la falta de documentación, que dificulta la modificación del código; la gran cantidad de condicionales, que impide realizar cambios mayores de forma sencilla; la ausencia de métodos y funciones. Además

comentaron que el código puede ser simplificado para evitar la redundancia y los menús pueden ser incluidos en funciones para facilitar su modificación.

Por el contrario, todos los participantes que trabajaron con el código B concluyeron que este sí tiene alta mantenibilidad debido, a que las funcionalidades estaban separadas correctamente y el código estaba documentado e indentado. Uno de los participantes afirmó que las clases tenían un bajo acoplamiento y una alta cohesión, lo cual permitía que un cambio, solo afectara a la parte directamente relacionada con él y no a todo el programa. Otro participante concluyó que la estructura del código puede ser extendida fácilmente para agregar nuevas funcionalidades.

CONCLUSIONES Y TRABAJO FUTURO

En este trabajo se ha propuesto un modelo de referencia para la inclusión de subcaracterísticas de mantenibilidad al producto software durante el proceso de desarrollo, este fue elaborado a partir de: (i) la identificación de los atributos de software que influyen en la mantenibilidad del producto, (ii) la clasificasión de los atributos identificados de acuerdo a las subcaracterísticas de mantenibilidad y a los procesos en los que se presentan, (iii) el analisis de como potenciar cada subcaracterística, (iv) la estructuración del modelo de referencia y (v) la evaluación del modelo de referencia realizada por medio de un caso de estudio. Se observa que la mantenibilidad de software es una característica de calidad que se debe considerar desde etapas tempranas del ciclo de vida del producto y durante todo el proceso de desarrollo de software con el fin de disminuir los costos de desarrollo y obtener un producto altamente mantenible, es por esto que las prácticas base definidas para los procesos del modelo de referencia son sencillas, esto con el fin de no aumentar el esfuerzo durante el desarrollo; sin embargo, es importante resaltar que estas deben considerarse durante el proceso de desarrollo y no en etapas posteriores del ciclo de vida del producto. De la aplicación del estudio de caso se puede concluir que la inclusión de las prácticas base del modelo de referencia al proceso de construcción permite potenciar la mantenibilidad del producto software, ya que la capacidad para ser analizado y modificado del producto aumentan considerablemente. Los resultados obtenidos en el estudio de caso aplicado demuestran que la utilización de prácticas base del proceso de construcción del modelo de referencia MANTuS permite potenciar la mantenibilidad del producto software.

Como trabajo futuro se ha iniciado un estudio de caso con el fin de evaluar todo el modelo de referencia propuesto. Este se está llevando a cabo en la asignatura Proyecto II del Programa de Ingeniería de Sistemas de la Universidad del Cauca, incorporando las prácticas del modelo de referencia a tres procesos de desarrollo diferentes (Scrum-XP, Tutelcan, UP-VSE). De igual forma se propene, realizar la validación del modelo de referencia proyectado en un entorno empresarial mediante un estudio de caso y aplicar el modelo de referencia estimado en una organización desarrolladora de software que desee obtener la certificación de mantenibilidad realizada por AQC, paravalidar si inclusión de las prácticas definidas permiten alcanzar este fin.

Con la aplicación del modelo de referencia en un entorno empresarial se puede obtener información que permita definir cuál es el costo de incluir MANTuS en el proceso de desarrollo de la organización objetivo, esto con el fin de analizar el costo-beneficio e impacto que tiene la aplicación de las prácticas base sobre las actividades de desarrollo. Sin embargo, a partir de los resultados iniciales obtenidos del caso de estudio realizado en esta investigación, se puede observar que utilizar MANTuS puede traer beneficios a una organización ya que las actividades de mantenimiento se pueden realizar de manera más efectiva, lo que conlleva a reducir costos operacionales.

Aunque la validación de la propuesta aun es limitada, como se ha mencionado anteriormente, se está trabajando en aplicar el modelo de referencia MANTuS en la industria con el fin de obtener realimentación y validación de su aplicación en contextos industriales.

AGRADECIMIENTOS

Este trabajo ha sido apoyado por el proyecto “Marco de trabajo para la elicitación de requisitos no funcionales basado en la gestión del conocimiento” (Vicerrectoría de Investigaciones de Unicauca - VRI ID 4354). Además Francisco J. Pino agradece a la Universidad del Cauca donde trabaja como profesor titular.

REFERENCIAS

[1] J.M. Conejero, E. Figueiredo, A. Garcia, J. Hernández and E. Jurado. "On the relationship of concern metrics and requirements maintainability". Information and Software Technology. Vol. 54, pp. 212-238. 2012.

[2] C.E.H. Chua, S. Purao and V.C. Storey. "Developing maintainable software: The Readable approach". Decision Support Systems. Vol. 42, pp. 469-491. 2006.

[3] J.-C. Chen and S.-J. Huang. "An empirical analysis of the impact of software development problem factors on software maintainability". Journal of Systems and Software. Vol. 82, pp. 981-992. 2009.

[4] F.J. Pino, F. Ruiz, F. García and M. Piattini. "A software maintenance methodology for small organizations: Agile MANTEMA". Journal Of Software Maintenance And Evolution: Research And Practice. Vol. 24, pp. 851-876. 2011.

[5] ISO. "ISO/IEC 25010: Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - System and Software Quality Models". 2011.

[6] E.B. Swanson. "IS maintainability: should it reduce the maintenance effort?". ACM SIGMIS Database. Vol. 30, pp. 65-76. 1999.

[7] ISO. "ISO/IEC 33004: Information technology - Process assessment - Requirements for Process Referents, Process Assessment and Organizational Maturity Models". 2012.

[8] A. April, J. Huffman Hayes, A. Abran and R. Dumke. "Software Maintenance Maturity Model (SMmm): the software maintenance process model". Journal of Software Maintenance and Evolution: Research and Practice. Vol. 17, pp. 197-223. 2005.

[9] M. Polo, M. Piattini, F. Ruiz and C. Calero. "MANTEMA: A complete rigorous methodology for supporting maintenance based on the ISO/IEC 12207 Standard". In Software Maintenance and Reengineering. Proceedings of the Third European Conference on, pp. 178-181. 1999.

[10] ISO. "ISO/IEC 12207: Systems and software engineering - Software life cycle processes". 2008.

[11] M. Kajko-Mattsson. "Corrective Maintenance Maturity Model: Problem Management". In Software Maintenance. Proceedings. International Conference on, 2002. pp. 486-490.

[12] M. Polo, M. Piattini and F. Ruiz. "Improving the Quality of the Maintenance Process". In The Second World Congress for Software Quality, Pacifico Yokohama Conference Center (Tokyo Bay Arena), pp. 325-330. 2000.

[13] ISO. "ISO/IEC 15504-5: An exemplar software life cycle process assessment model". 2011.

[14] F.J. Pino, M. Piattini and G. Travassos. "Managing and Developing Distributed Research Projects by Means of Action-Research". Rev. Fac. Ing. Univ. Antioquia. Vol. 68, pp. 61-74. 2010.

[15] J. McNiff. "Action research: Principles and practice". 3rd ed. Routledge. New York, USA. 2013.

[16] J. Erazo, A. Florez and FJ. Pino. "Análisis y clasificación de atributos de mantenibilidad-una revisión desde la literatura". Entre ciencia e ingeniería, Universidad Católica de Pereira (en revisión). 2015.

[17] R.K. Yin. "Case Study Research: Design and Methods". Third Editon ed. Vol. 5. SAGE Publications. 2003.

[18] P. Brereton, B. Kitchenham, D. Budgen and Z. Li, "Using a protocol template for case study planning". In Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering. University of Bari. Italy. 2008.

[19] P. Kroll and P. Kruchten. "The rational unified process made easy: a practitioner's guide to the RUP". Addison-Wesley Professional. 2003.

[20] J. Erazo and A. Florez. "Modelo de referencia para la inclusión de subcaracterísticas de mantenibilidad al producto software durante el proceso de desarrollo", Tesis de Pregrado, Ingenieria de Sistemas. Universidad del Cauca. Popayán, Colombia. 2014.

[21] AENOR. "Certificación de la Calidad del Producto Software ISO 25000". URL: http://www.aenor.es/aenor/certificacion/calidad/calidad_software_25000.asp#.VT-1LyGqqkp

[22] C.M. Fernández, M. Rodríguez and M. Piattini. "ISO 25000: Calidad del producto software". Revista de la Normalización y la Certificación (Revista AENOR), pp. 30-35. 2013.

[23] AQCLab. "Evaluación de la mantenibilidad del producto software (Reporte Técnico)". Alarcos Quality Center. 2013.