

Mejora de historias de usuario y casos de prueba de metodologías ágiles con base en TDD.

Improving User Stories and Agile Test Cases Based on TDD.

Katerine Villamizar Suaza*
John Jairo Tabares García**
Carlos Mario Zapata Jaramillo***



Tipo de artículo: Resultado de Investigación.

Recibido: 7 de Julio, 2015
Aceptado: 24 de Septiembre, 2015

Resumen

Las historias de usuario se utilizan en los métodos ágiles para especificar los requisitos de una aplicación de software. El desarrollo dirigido por pruebas (TDD—Test Driven Development) es una técnica usada en los métodos ágiles que consiste en generar pruebas unitarias automáticas basadas en las historias de usuario. Esta técnica presenta dificultades en la confiabilidad de las pruebas funcionales integrales y en la especificación de las historias de usuario. Diversos investigadores proponen nuevas técnicas para mejorar las pruebas funcionales de software y las historias de usuario, pero no solucionan directamente la problemática de la técnica TDD. Por ello, en este artículo se propone una mejora a las historias de usuario con base en un formato único y completo para las historias de usuario que integre elementos de los casos de prueba. En las historias de usuario se propone la inclusión de esquemas preconceptuales que parten del dominio del sistema, lo que implica una mejora en la historia de usuario y una reducción en tiempo de desarrollo del producto. Esta propuesta se ejemplifica con un caso de estudio.

Palabras clave: Historias de usuario, desarrollo dirigido por pruebas, pruebas ágiles.

Abstract

User stories are used in agile methods to specify the requirements of a software application. Test Driven Development (TDD) is a user-story-based technique used in agile methods to generate automatic unit tests. This technique exhibits difficulties in the reliability of integrated functional testing and user story specification. Several researchers have proposed new techniques to improve functional testing of software and user stories, but problems related to TDD are not addressed. Consequently, in this paper we propose an improvement of user stories based on a single, comprehensive user story format including test case elements. Pre-conceptual schemas are proposed to be included in the user stories. Such schemas are based on the system domain, so user stories are improved and product development time is reduced. This proposal is exemplified by a case study.

Keywords: Agile testing, user stories, Test Driven Development.

* Magíster en Ingeniería de Sistemas. Docente Ocasional, Tecnológico de Antioquia – Institución Universitaria. kvillamizar@tdea.edu.co

**Magíster en Ingeniería de Sistemas. Universidad Nacional de Colombia. jjtabares84@gmail.com

*** Ph.D. en Ingeniería. Profesor Asociado, Universidad Nacional Colombia. cmzapata@unal.edu.co

Introducción

Los métodos ágiles de software se basan, fundamentalmente, en la colaboración con los usuarios durante todo el proceso de desarrollo, en la facilidad para adaptar el producto a cambios en requisitos y en la entrega incremental del producto (Brailovsky, 2009). Entre sus artefactos para el inicio de la construcción de software se encuentran las historias de usuario, que se emplean en la especificación de los requisitos de una aplicación de software.

Dentro del desarrollo ágil de software se encuentran las pruebas ágiles, que abarcan el proceso completo de desarrollo. Las historias de usuario son productos de trabajo que permiten modificar y refactorizar el código existente. En XP (Extreme Programming) se desarrolló una técnica de desarrollo dirigido por pruebas (TDD) donde las pruebas son las que permiten guiar el desarrollo del producto de software, partiendo de historias de usuario para la creación de pruebas unitarias automáticas (Laboratorio Nacional de Calidad del Software, 2009).

La técnica TDD presenta problemas en cuanto a la confiabilidad de las pruebas funcionales integrales (Brailovsky, 2009):

- Los desarrolladores, quienes realizan las pruebas, no tienen en cuenta aspectos generales del negocio.
- Los interesados no especifican completamente las historias de usuario.
- La técnica se basa en pruebas unitarias, sin tener en cuenta las pruebas integrales del sistema.

Los proyectos que buscan mejorar la técnica TDD tienen dos tendencias:

- Nuevas técnicas para la mejora de las pruebas funcionales integrales mediante la adición de pruebas de aceptación (Gutiérrez, 2005; Blé, 2010; Yagüe & Garbajosa, 2009; Letelier & Penadés, 2002).
- Gestión de las historias de usuario mediante su versionamiento con el lenguaje XML (Sánchez

et al., 2010; Laboratorio Nacional de Calidad del Software, 2009).

Sin embargo, estos proyectos no permiten solucionar directamente la problemática, ya que para las historias de usuario no se propone una estructura que permita reducir la ambigüedad en las mismas. Por ello, en este artículo se propone una mejora a las historias de usuario, que incluye una redefinición de su estructura, la adición de un esquema preconceptual ejecutable y la integración con elementos de los casos de prueba. Esta mejora se ejemplifica con un caso de estudio.

El análisis condujo al uso de los métodos ágiles para especificar los requisitos de una aplicación de software que se enfoca en dar solución a los problemas de una historia de usuario. La historia de usuario es el resultado de las conversaciones entre los interesados del proyecto y los desarrolladores de software, que son los encargados de realizar las pruebas. Con este estudio se ejemplifica el uso de los métodos ágiles que permiten generar automáticamente los casos de prueba a partir de esquemas preconceptuales.

Este artículo posee la siguiente estructura: inicialmente, se presenta un marco teórico de los conceptos necesarios para entender la problemática presentada; luego, se presenta la definición de la problemática de las pruebas ágiles con TDD; seguidamente, se analizan antecedentes de los trabajos desarrollados para solucionar dicha problemática; luego, se propone la mejora de las historias de usuario que se integran con casos de prueba y se muestra un caso de estudio completo de la solución planteada; finalmente, se discuten las conclusiones y el trabajo futuro que se pueden derivar de esta propuesta.

Marco teórico

Métodos ágiles

Los métodos de desarrollo ágil se basan, fundamentalmente, en la colaboración con los usuarios de software durante todo el proceso de

desarrollo, la facilidad para adaptar el producto a cambios en requisitos y la entrega incremental del producto. Estos métodos se diseñan para ofrecer la aplicación de software que el usuario necesita, en el momento en que la necesita (Calderón & Valverde, 2007).

Entre los métodos ágiles existentes se encuentra XP. Este método ágil se basa en la retroalimentación continua entre el cliente y el equipo de desarrollo y la comunicación fluida entre todos los participantes. XP se adecúa a proyectos con requisitos imprecisos y muy cambiantes y donde existe un alto riesgo técnico (Calderón & Valverde, 2007).

Historias de usuario

Una historia de usuario es el resultado de conversaciones entre los interesados del proyecto, los analistas de negocios, los encargados de pruebas y los desarrolladores. Una historia de usuario es una representación de un requisito de software escrito en una o dos frases utilizando el lenguaje común del interesado. Las historias de usuario se utilizan en las metodologías de desarrollo ágil para la especificación de requisitos (acompañadas de las discusiones con los usuarios y las pruebas de validación). Cada historia de usuario debe ser limitada, pues se debería poder escribir sobre una nota adhesiva pequeña. Dentro de la metodología XP, los clientes deberían escribir las historias de usuario.

Las historias de usuario son formas rápidas de administrar los requisitos de los usuarios sin tener que elaborar gran cantidad de documentos formales y sin requerir mucho tiempo para administrarlos. Las historias de usuario permiten responder rápidamente a los requisitos cambiantes.

Las historias de usuario se emplean en XP para especificar los requisitos del software. Se trata de tarjetas de papel en las cuales el cliente describe, brevemente, las características que el sistema debe poseer, sean requisitos funcionales o no

funcionales. El tratamiento de las historias de usuario es muy dinámico y flexible. En cualquier momento, las historias de usuario se pueden romper, reemplazar por otras más específicas o generales, añadir nuevas o modificar. Cada historia de usuario es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en algunas semanas (Álvarez et al., 2012).

Pruebas ágiles

El concepto de pruebas en entornos ágiles no se limita al diseño y ejecución del software, sino que abarca el proceso completo de desarrollo desde una perspectiva de aporte de calidad desde el inicio. Las pruebas ágiles permiten al equipo realizar un proyecto completo mediante una adaptación continua y un cambio en los patrones de pruebas desde el punto de vista de los resultados para el cliente.

Las pruebas ágiles involucran a todo el equipo de desarrollo desde el inicio, probando y analizando el código desde que está disponible y suficientemente estable. Incorporan, además, toda una serie de herramientas orientadas al diseño basado en comportamiento, automatización de las pruebas, integración continua, propiedad colectiva del código y métricas de calidad del software (Beck, 2000).

TDD (Desarrollo Dirigido por Pruebas, por sus siglas en inglés)

Las pruebas ágiles son productos de trabajo que permiten modificar y refactorizar el código existente. XP incluye el enfoque para el Desarrollo Dirigido por Pruebas (TDD) donde las pruebas sirven para guiar el desarrollo del producto software, partiendo de historias de usuario para la creación de pruebas unitarias. TDD se aplica, esencialmente, en el ámbito de la implementación, particularmente siguiendo el planteamiento “no escribir código hasta no disponer de las pruebas que debe satisfacer dicho código” (Beck, 2000).

Problemática

La técnica TDD presenta problemas en cuanto a las pruebas que se ejecutan. En primer lugar, la técnica TDD se basa en las historias de usuario como punto de partida para generar los casos de pruebas que se ejecutan sobre el código que se está construyendo, pero son los interesados los que crean las historias de usuario y ellos, usualmente, no saben lo que necesitan del sistema de software (Cooper & Aywan, 1998). Además, estas historias no son bien detalladas. En segundo lugar, esta técnica no es fácil de usar para las pruebas de software integrales funcionales, ya que sólo se basan en pruebas unitarias, dejando a un lado el

sistema como un todo (Sánchez et al., 2010); esto se debe a que las pruebas unitarias se centran en la menor unidad de diseño del software, el módulo (un método o clase) y, por ende, no se toma todo el sistema (Tuya et al., 2007). Finalmente, la ejecución de las pruebas no es confiable, debido a que los desarrolladores que las ejecutan no tienen en cuenta los aspectos de negocio del cliente (Yagüe & Garbajosa, 2009).

En la Figura 1 se presenta la problemática descrita en un diagrama causa-efecto.

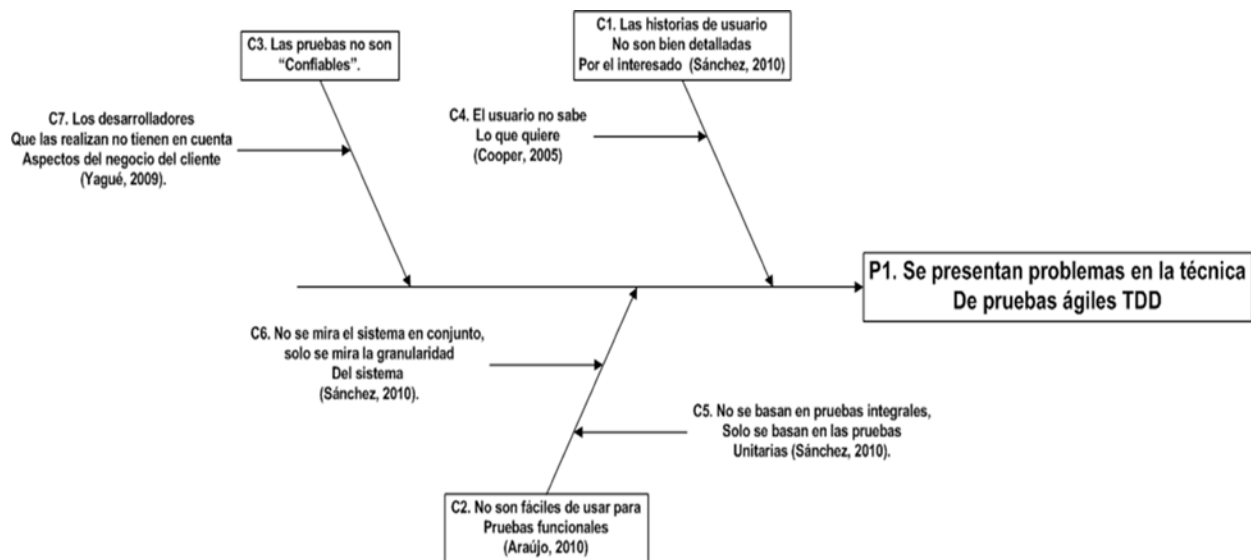


Figura 1. Causa-efecto, problemas de la técnica TDD.

Fuente: Elaboración propia (2014).

Antecedentes

Desarrollo Dirigido por Pruebas de Aceptación (ATDD)

Es una técnica que realiza una conexión entre SCRUM y XP, y se basa en la realización de una lista de ejemplos de cada historia de usuario, que son similares a casos de uso (Gutiérrez, 2005). En esta lista se representa el proceso de desarrollo (Blé, 2010). La técnica sobre el diseño ágil con ATDD, establece las pruebas como una

herramienta del diseño del código, sugiriendo escribirlas antes que el código. Los requisitos son pruebas de aceptación de las historias de usuario “lo suficientemente claros para que los interesados los aprueben” (Zapata et al., 2010). Esta técnica centra su aplicación en la validación y prueba de software con base en pruebas de aceptación (Yagüe & Garbajosa, 2009). En la Figura 2 se presentan ejemplos de historias de usuario.

A partir de las historias de usuario se generan unas pruebas de aceptación (preguntas restringidas para validar la información con el interesado), que son afirmaciones en lenguaje natural. Luego, el equipo de desarrollo hace el esfuerzo de conectar esas frases con los puntos de entrada y salida del código. Básicamente, lo que proponen es escribir las frases en un formato determinado, como por

ejemplo HTML, usando etiquetas de una manera específica para delimitar qué partes de la frase son variables de entrada para el código y cuáles son datos para validación del resultado de la ejecución (Letelier et al., 2002). En las Figuras 3 y 4 se puede apreciar un ejemplo de las preguntas y el paso a las etiquetas para especificar las variables de entrada del código.

- *Formulario de inscripción*
- *Login en el sistema*
- *Reservar una habitación*
- *Añadir un libro al carrito de la compra*
- *Pago con tarjeta de crédito*
- *Anotar un día festivo en el calendario*
- *Informe de los artículos más vendidos*
- *Darse de baja en el foro*
- *Buscar casas de alquiler en Tenerife*

Figura 2. Ejemplos de historias de usuario.

Fuente: Yagüe & Garbajosa (2009).

- *¿Qué hace el sistema si el libro que se quiere añadir al carrito ya está dentro de él?*
- *¿Qué sucede si se ha agotado el libro en el almacén?*
- *¿Se le indica al usuario que el libro ha sido añadido al carrito de la compra?*

Figura 3. Preguntas de una historia de usuario.

Fuente: Yagüe & Garbajosa (2009).

- *Añadir libro X en stock produce: 'El libro X ha sido añadido al carrito'*
- *Libro X está contenido en el carrito*
- *Libro X ya no está en catálogo de libros*

Figura 4. Definición de etiquetas.

Fuente: Yagüe & Garbajosa (2009)

A pesar de que esta técnica permite analizar la problemática de las pruebas integrales en el desarrollo de software, deja a un lado las pruebas individuales. Además, el empleo de las historias de usuario no presenta ninguna mejora en cuanto a las utilizadas en la técnica TDD.

Mejorando la gestión de historias de usuario en XP (Sánchez et al., 2010)

Debido a la necesidad de gestionar las historias de usuario, se propone una plantilla esencial representada en XML y basada en una plantilla

de DTD (Definición de Tipo de Documento). Esta plantilla en XML se genera directamente desde una historia de usuario y se representa en forma de escenarios para facilitar el control de versiones. De esta manera se puede disponer de

un formato sencillo que se pueda procesar en las herramientas informáticas.

En la Figura 5 se presenta un ejemplo de un código XML generado a partir de una historia de usuario.

```
<?xml version="1.0"?>
<!ELEMENT user_story (story_id,name,creation_date,customer,
priority,dependence*,estimation, risk,
release,iteration,upgrade*,base?,
description)>
<!ELEMENT story_id (#PCDATA)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT creation_date (#PCDATA)>
<!ELEMENT customer (#PCDATA)>
<!ELEMENT priority (#PCDATA)>
<!ELEMENT dependence (#PCDATA)>
<!ELEMENT estimation (#PCDATA)>
<!ELEMENT risk (#PCDATA)>
<!ELEMENT release (#PCDATA)>
<!ELEMENT iteration (#PCDATA)>
<!ELEMENT upgrade (#PCDATA)>
<!ELEMENT base (#PCDATA)>
<!ELEMENT description (#PCDATA)>
```

Figura 5. Ejemplo XML generado a partir de HU.

Fuente: Sánchez et al., (2010).

Adicionalmente, en este proceso se dan algunas pautas para abordar las alternativas de evolución que puede sufrir la historia de usuario y se presenta el prototipo de una herramienta para gestionar historias de usuario que apoya dichas pautas (Laboratorio Nacional de Calidad del Software, 2009).

Sin embargo, esta herramienta se orienta hacia el versionamiento, debido a que las historias de usuario son “volátiles” y, por lo tanto, la gestión de las mismas se puede complicar. Si estas especificaciones de historias de usuario en archivos XML se orientaran a una codificación automática inicial, se podría intentar solucionar la problemática presentada en cuanto a las historias de usuario. Además, este proceso no presenta una estructura restringida y limitada de las historias de usuario para traducirlas directamente a código, sino que se copia de la estructura general de las historias de usuario en lenguaje XML (Sánchez et al., 2010).

Metodología

Como solución a la problemática presentada anteriormente, se plantea un mejoramiento de la técnica TDD por medio de la estructuración de una historia de usuario que integre los casos de prueba. Además, se presenta como opción la generación de una pantalla automática por medio de la creación de esquemas preconceptuales ejecutables. A continuación, se presenta una breve explicación de los esquemas preconceptuales y los esquemas preconceptuales ejecutables.

Esquemas preconceptuales

Representación intermedia entre las especificaciones textuales en lenguaje natural (“modelos verbales”) y los diferentes esquemas conceptuales que permiten el modelado de una aplicación de software perteneciente a un dominio específico (Zapata, 2007).

En la Figura 6 se presentan los elementos básicos de los esquemas preconceptuales.

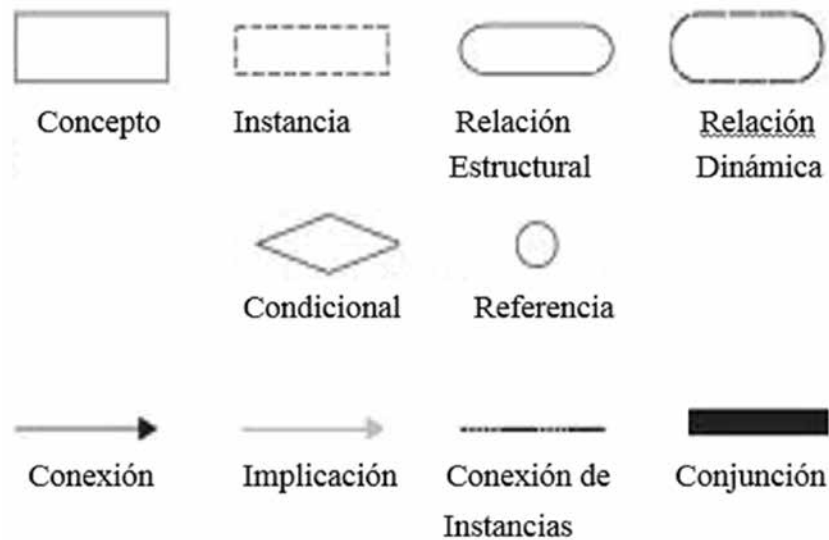


Figura 6. Simbología esquema preconceptual.

Fuente: Zapata (2007).

Esquemas preconceptuales ejecutables

Los esquemas preconceptuales ejecutables son herramientas que se emplean para representar gráfica y computacionalmente el dominio de una situación o problema. Una modificación en la terminología del esquema preconceptual no es suficiente para hacerlos ejecutables, pues pueden surgir todavía problemas de entendimiento, claridad y falta de información sobre las actividades que se realizan para clarificar la representación de un dominio. Así, se proponen varias consideraciones para la implementación de relaciones dinámicas dentro de los esquemas preconceptuales, con el fin de posibilitar la visualización de instancias del sistema. Con ello, se busca identificar posibles fallas que se pueden presentar a la hora de la entrega del producto, mediante la validación con el interesado de esas instancias (Zapata et al., 2011).

Formato de historia de usuario

En la Tabla 1 se detalla el formato de historia de usuario propuesto que integra los casos de pruebas y los esquemas preconceptuales ejecutables.

En la historia de usuario se plantean dos partes, dependiendo del actor que llene la historia de usuario.

- Interesado: llena los campos nombre, actor, objetivo y condición de satisfacción.
- Analista de requisitos: luego de tener una entrevista con el interesado, llena los campos: HU relacionada (Historia de usuario relacionada con la historia de usuario actual, lo que permite tener historias de usuarios integradas con todo el sistema), módulo (colección de historias de usuario que pertenecen a un mismo subdominio), pantalla (detalle de los campos y esquema preconceptual ejecutable).
- Analista de pruebas: después de que el software se construya, llena los campos “Evaluación”, para evaluar los casos planteados directamente desde la historia de usuario y la pantalla construida.

Tabla1. Propuesta formato historia de usuario.

Historia de Usuario / Caso de Prueba						
Código						
Nombre						
Actor						
Objetivo	Actor + Acción + Nombre					
HU Relacionada	Código:		Nombre:			
Módulo						
Condición de Satisfacción	Resultado		Acción		Mensaje	EVALUACIÓN
Pantalla	Esquema Pre conceptual					
	Campos					
	Nombre					
	Código					
	Tipo Campo	Campo editable -Icono -Botón -Chek box -Lista -Radio botton				
	Caracteres Especiales					
	Restricción					
	Valor por defecto					
	Ordenamiento inicial					
	Permite modificar					
	Permite Ordenamiento					
	Links					
	Obligatoriedad					
	Dependencia					
EVALUACIÓN						

Fuente: Elaboración propia (2014).

Con esta propuesta se logra tener un avance en cuanto a la problemática planteada, de la siguiente manera:

1. Detallado de las historias de usuario de una manera estructurada, construida de forma mancomunada entre interesados, analistas de requisitos y analistas de pruebas.
2. Integralidad en el sistema, al tener las relaciones entre las diferentes historias de usuario levantadas para un mismo dominio.
3. Confiabilidad en las pruebas en cuanto a que se tienen en cuenta los diferentes aspectos del negocio, al realizar las pruebas directamente sobre las historias de usuario.

Resultados

A continuación, se presenta un caso práctico de una aplicación naviera en donde el cliente ingresa nominaciones (solicitudes de carga) y comentarios; se tiene que una de las historias de usuario para la aplicación sería la de administrar los comentarios que ingresan los usuarios a la aplicación con el fin de poderles atender en sus observaciones.

1. En el método tradicional (Beck, 2000), se tendría la Tabla 1.

Tabla 2. Historia de usuario Administrar los comentarios.

Historia de usuario Administrar los comentarios
Como administrador quiero administrar los comentarios que ingresan los usuarios a mi aplicación para atenderlos en sus observaciones.

Fuente: elaboración propia (2014).

La historia de usuario presentada en la Tabla 2 no especifica claramente los requisitos funcionales o criterios de aceptación que tendría la historia de usuario para iniciar su desarrollo y posteriormente las pruebas.

Si se lleva dicha historia de usuario a una definición más elaborada (Álvarez, 2012), tendríamos la Tabla 3.

Tabla 3. Historia de usuario más elaborada.

Nombre	Administrar los comentarios
Descripción	Como administrador quiero administrar los comentarios que ingresan los usuarios a mi aplicación para atenderlos en sus observaciones.
Criterios de aceptación	<ul style="list-style-type: none"> • Se deben presentar todos los comentarios que ingresan los clientes. • Se debe permitir el ingreso de un nuevo comentario. • Se debe permitir deshabilitar los comentarios.

Fuente: elaboración propia (2014).

Realizando la revisión de la historia de usuario anterior, se observa que sigue siendo incompleta para poder realizar un desarrollo y, por ende, una prueba a partir de la misma.

De acuerdo con el método propuesto, se presenta en la Tabla 4 la historia de usuario para administrar los comentarios que ingresan los usuarios a la aplicación. A continuación, en las figuras 7 y 8 los esquemas preconceptual y preconceptual ejecutable de la historia de usuario presentada.

Realizando una comparación entre las historias de usuario inicialmente presentadas y la propuesta, se puede observar que la propuesta es completa, de manera que permite tener una relación de “Módulo”, para este caso Nominación, lo que indica que la historia de usuario no sólo se probará de manera individual (unitaria) si no de manera integral con las historias de usuario que pertenezcan al mismo módulo.

Tabla 4. Historia de usuario “Comentarios”.

Historia de Usuario / Caso de Prueba						
Código	001					
Nombre	"Comentarios"					
Actor	Administrador					
Objetivo	Administrador Ingresa "Comentarios"					
HU Relacionada	Código:	NA	Nombre:	NA		
Módulo	Nominación					
Condición de Satisfacción	Acción	Resultado		Mensaje	EVALUACIÓN	
	Ingresar a la pantalla	muestran los comentarios de la nominación que se esté trabajando.			P	
	Ingresar a la pantalla	Los comentarios listados se encuentren ordenados según los criterios de ordenamiento.			F	
	Dar clic en el botón "Agregar comentario"	Que se presente el comentario guardado.				
	Dar clic al ícono "Deshabilitar"	se muestra un mensaje para confirmar que un comentario se pasará por alto.		será pasado por alto, desea continuar"		
Pantalla	Esquema Pre conceptual					
	<u>Esquema Pre conceptual</u>					
	Campos					
	Nombre	<u>Usuario</u>	<u>Fecha</u>	<u>Descripción</u>	<u>Deshabilitar</u>	<u>Agregar comentario</u>
	Código	C001	C002	C003	C004	C006
	Tipo Campo	Campo No editable	Campo No editable	Campo No editable	Botón	Botón
	Caracteres Especiales	NA	NA	NA	NA	NA
	Restricción	NA	El ordenamiento es en forma ascendente.	NA	Cambia el estado al Comentario para que sea pasado por alto.	NA
	Valor por defecto	NA	NA	NA	NA	NA
	Ordenamiento inicial	NA	NA	NA	NA	NA
	Permite modificar	NA	NA	NA	NA	NA
	Permite Ordenamiento	Si	Si	Si	NA	NA
	Links	NA	NA	NA	NA	NA
	Obligatoriedad	Si	Si	Si	NA	NA
Dependencia	NA	NA	NA	NA	NA	
EVALUACIÓN	F	P				

Fuente: elaboración propia (2014).

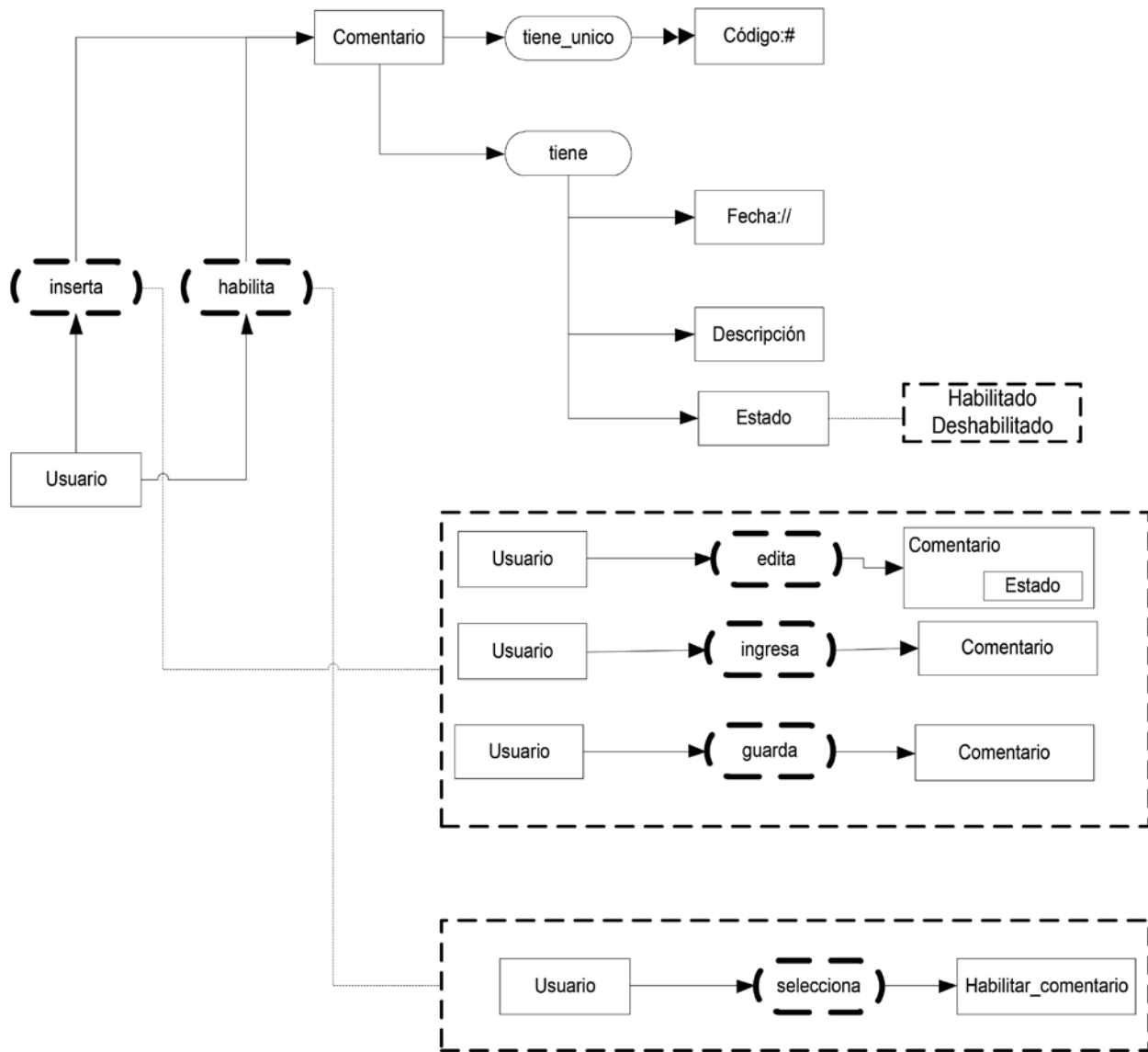


Figura 7. Esquema preconceptual de la historia de usuario “Comentarios”.
 Fuente: elaboración propia (2014).

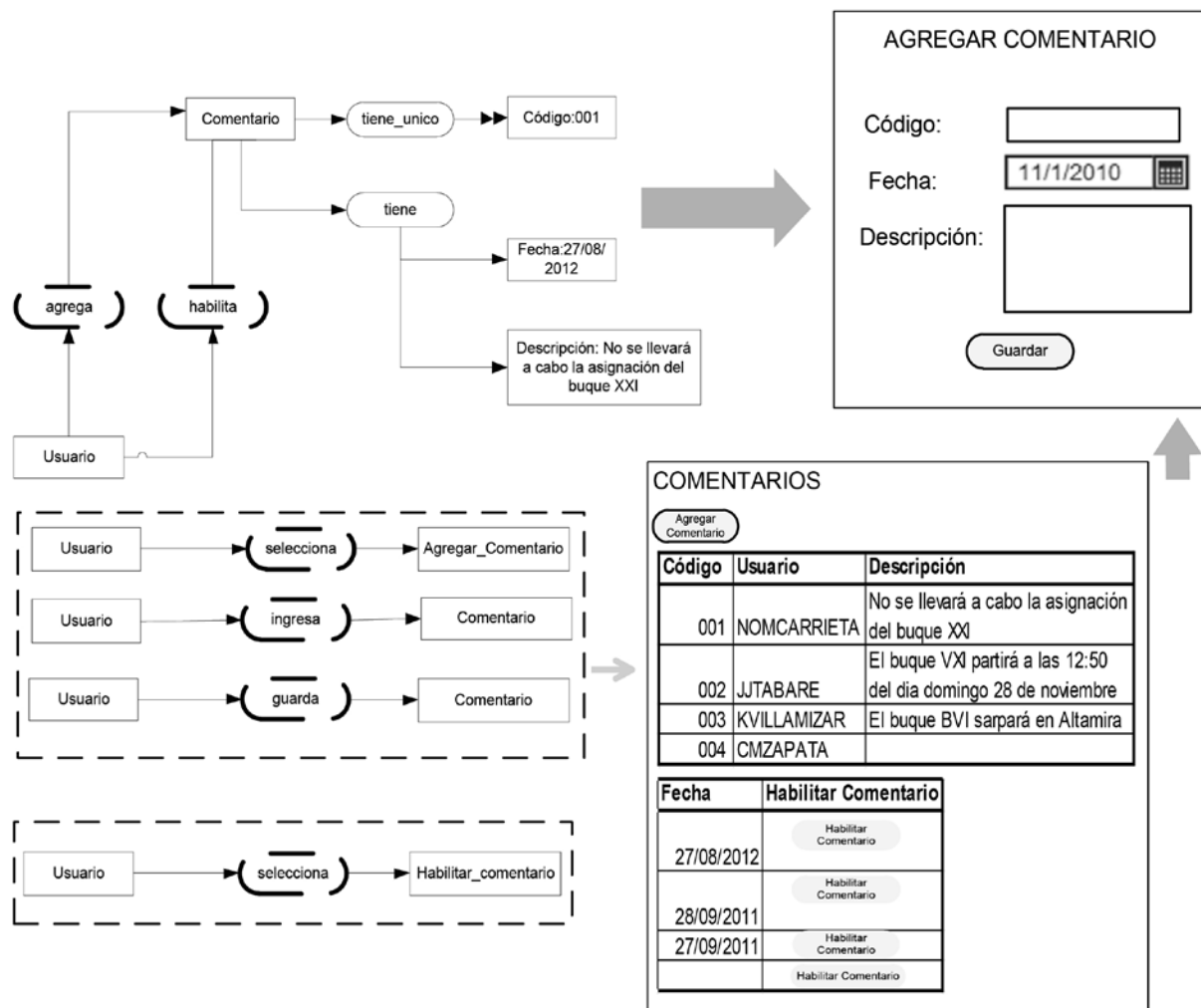


Figura 8. Esquema preconceptual de la historia de usuario “comentarios”
Fuente: elaboración propia (2014)

Conclusiones

En este artículo se propuso una estructura más completa de las historias de usuario, detallándolas mediante esquemas preconceptuales, en los que se especifica claramente la necesidad del interesado y se pueden realizar validaciones con el interesado en un lenguaje no ambiguo.

Se pueden usar las historias de usuario fácilmente para las pruebas funcionales, debido a que en las mismas se presenta la relación que tiene una historia de usuario con otra y el módulo al que pertenece.

Las pruebas son confiables en la medida en que las historias de usuario se pueden traducir a pruebas automáticas, pues se tienen en cuenta los aspectos del negocio que requiere el interesado.

Como trabajo futuro, se propone la generación automática de los casos de prueba, tomando como base los elementos adicionales que se incluyeron en la estructura de la historia de usuario.

Referencias

- Álvarez, A., De las Heras del Dedo, R. & Lasa, C. (2012). Métodos Ágiles y Scrum. Madrid: Ediciones Anaya Multimedia.
- Beck, K. (2000). Extreme Programming Explained. US: Addison Wesley.
- Blé, C. (2010). Diseño Ágil con TDD. Creative Commons.
- Brailovsky, N. (2009). Introducción a las Metodologías Ágiles. Recuperado de https://monoinfinito.files.wordpress.com/2008/10/metodologias_agiles.pdf
- Calderón, S. & Valverde, J. (2007). Metodologías ágiles. Perú: Universidad Nacional de Trujillo.
- Cooper, R. & Aywan, S. (1998). La Inteligencia Emocional aplicada al liderazgo y a las organizaciones. Colombia: Editorial Norma.
- Gutiérrez, J. (2005). Generación de pruebas de sistema a partir de la especificación funcional. Sevilla: Universidad de Sevilla.
- Laboratorio nacional de calidad del software. (2009). Ingeniería del software: Metodologías y ciclos de vida. Colombia: INTECO.
- Letelier, P. & Penadés, C. (2002). Metodologías ágiles para el desarrollo de software: eXtreme Programming (XP). Buenos Aires: Cyta.
- Sánchez P., Emilio A. & Caños L. (2010) Mejorando la gestión de historias de usuario en eXtreme Programming. Valencia: Departamento de Sistemas Informáticos y Computación. Universidad Politécnica de Valencia, Camino de Vera.
- Tuya J., Ramos, I. & Dolado, J. (2007). Técnicas cuantitativas para la gestión en la ingeniería del software. España: Netbiblo, S. L.
- Yagüe, A. & Garbajosa, J. (2009). Comparativa práctica de las pruebas en entornos tradicionales y ágiles. Redalyc, (5), 19.
- Zapata, C. (2007). Definición de un esquema preconceptual para la obtención automática de esquemas conceptuales de UML. Tesis de Doctorado. Medellín: Universidad Nacional de Colombia.
- Zapata, C. Arboleda, C. & Castrillón, C. (2010) Una representación gráfica del testing ágil. Avances en Sistemas e Informática, (7), 18-26.
- Zapata, C. M., Giraldo G. & Londoño, S. (2011). Esquemas preconceptuales ejecutables. Revista Avances en Sistemas e Informática, (8), 15-24.



“La mente que se abre a una nueva idea, jamás volverá a su tamaño original”.

Albert Einstein.