Evaluación de metodologías y técnicas para la generación de pruebas aplicadas un caso real a partir de sus especificaciones funcionales

Silvia Sanchez Zuaín, Lilia Palomo y Beatriz Fernández Reuter
Facultad de Matemática Aplicada - Universidad Católica de Santiago del Estero
silvia.sanchez@ucse.edu.ar - lilia.palom@ucse.edu.ar - bfreuter@gmail.com

RESUMEN

Las pruebas son una parte importante del proceso de verificar si el software satisface las expectativas del usuario y deben presentarse a lo largo de todo el ciclo de vida del desarrollo, como un aspecto decisivo en el control de calidad del producto final. Como parte de las mismas están las pruebas funcionales, para las cuales se requiere una buena planificación que consiste en definir los aspectos a chequear y la forma de comprobar su correcto funcionamiento, punto en el cual adquieren sentido los casos de prueba. También es necesario contar con una metodología y herramientas adecuadas que permitan validar los requisitos funcionales, corregir los errores y mejorar su calidad.

Este trabajo tiene como objetivo presentar una visión global de distintas metodologías para la generación de pruebas a partir de la definición de requisitos funcionales, aplicadas a un caso real, a fin de determinar su efectividad en la detección de errores de sistemas de forma temprana.

Palabras clave: Pruebas del sistema, metodologías de pruebas tempranas, generación de pruebas, casos de prueba, requisitos funcionales, casos de uso.

CONTEXTO

La Facultad de Matemática Aplicada, impulsa el desarrollo de un conjunto de proyectos que apuntan a incentivar la investigación desde las cátedras promoviendo la interacción vertical y horizontal. Esto se centra en posibilitar la participación de los docentes en investigación y obtener resultados transferibles a las actividad de las cátedras.

Es por esto, que desde las cátedras de Programación I, Sistemas de Información II e Ingeniería del Software, en el año 2011 surge el proyecto “Generación de pruebas de sistemas a partir de las especificaciones funcionales aplicadas a un caso real” con el objetivo de promover la investigación aplicada, formación de recursos humanos e innovación de los contenidos de las cátedras y en las prácticas profesionales.

El presente artículo es una línea de investigación que surge del proyecto mencionado anteriormente, ante la necesidad de determinar cual o cuales metodologías utilizar para la generación de pruebas en un caso real, compararlas y así determinar su efectividad en la detección de errores de sistemas en etapas tempranas.

INTRODUCCIÓN

El desarrollo de software implica una serie de actividades de producción donde las posibilidades de que aparezcan fallos humanos son grandes. Los errores pueden aparecer desde el primer momento del

proceso, a causa de objetivos especificados de forma errónea, o bien en pasos posteriores del diseño y desarrollo. Por lo tanto, el desarrollo de software debe ir acompañado de una actividad que garantice su calidad. Un aspecto crucial en el control de calidad del desarrollo de software son las pruebas, cuyo objetivo, consiste en asegurar que el sistema hace lo que el cliente quiere que haga, es decir, verificar que el software cumple con sus requisitos.

Es importante comprender que los errores pueden insertarse en un requerimiento, en el diseño, en un componente del código, en la documentación, o en cualquier momento durante el desarrollo o el mantenimiento.

Como corregir los errores en un sistema tiene un costo elevado a medida que el proceso de desarrollo de software avanza en sus etapas, es fundamental detectarlos en la fase más temprana posible. Una forma de hacerlo es generando casos de prueba en paralelo con el desarrollo del software y donde estos casos de prueba definidos, simulen interacciones del actor con el sistema para verificar que el sistema hace lo que se espera de él [2].

Otra manera de detectar de forma temprana los errores del sistema es a partir de sus especificaciones funcionales. Algunas investigaciones proponen diversas metodologías que permiten la generación de casos de prueba a partir de la definición de casos de uso, tal y como lo plantea Correa y Giandini [1], que genera casos de prueba, a partir de cada caso de uso; y casos de pruebas de datos, a partir de los datos necesarios para la ejecución de un caso de uso. Además, se vale de diagramas de actividades de UML, derivados del modelo de casos de uso, para especificar el comportamiento de los casos de prueba; y de una transformación modelo a texto para obtener como resultado final de este proceso, las pruebas. Otro caso es el de Heumann et al [5] que desarrolla un método que parte de los casos de uso para generar los casos de prueba, identificando en cada uno los posibles escenarios, o caminos de ejecución, y definiendo los valores a probar con cada caso de prueba. Por último, obtiene una lista de casos de prueba, con los valores que se deben probar y los resultados esperados para cada caso.

En los últimos años se ve una tendencia hacia la automatización del proceso de generación de casos de prueba a través de diferentes métodos y herramientas que parten de casos de uso escritos en lenguaje no formal y aplican diferentes métodos para obtener objetivos de prueba [3], o bien, de casos de usos bien definidos que se toman de entrada para ejecutar algoritmos que generan diagramas de actividad de los que se derivan los casos de prueba [4] [7] [8] [9] [10].

Estas metodologías que se mencionan precedentemente indican lo que una prueba debe hacer, al ejecutar un escenario posible de un caso de uso, qué información hay que suministrarle y qué información va a devolver. Por lo tanto, lo primero que es necesario saber para poder implementar las pruebas es tener en cuenta los puntos comunes de las propuestas de generación de pruebas del sistema [6], de los que se se consideran entre otros: a- El objetivo de estas propuestas es obtener un conjunto completo de pruebas del sistema que permitan garantizar que el sistema software cumple con la especificación funcional dada, lo cual permite asegurar su calidad. b- Todas parten de los requisitos funcionales del sistema y la mayoría permiten comenzar a desarrollar los casos de prueba del sistema en cuanto se

comience a disponer de los requisitos funcionales.

Del análisis de lo expuesto surge el planteo de la conveniencia sobre la aplicación de algunas de estas metodologías para determinar su alcance, es decir, que tan efectivas son para detectar errores, que tan fácil de realizar son, si es conveniente utilizar una técnica manual o automatizada, que nivel de cobertura tienen, entre otros aspectos.

LINEAS DE INVESTIGACIÓN Y DESARROLLO

Las líneas de investigación del presente trabajo tienen como eje central las metodologías y herramientas para las pruebas tempranas del software, y en particular la generación de casos de pruebas. Estas líneas de investigación en pruebas del software se resumen a continuación:

- Investigación de las actuales formas usadas para derivar casos de prueba funcional.
- Análisis y comparación de metodologías y herramientas existentes que soportan los procesos de pruebas.
- Estudio del proceso de aplicación de las metodologías de generación de casos de prueba para un caso real.
- Evaluación de la cobertura y efectividad de las pruebas y diseño de pruebas.

El equipo de investigadoras, se propone desarrollar proyectos compatibles con esta línea de investigación y diseñar una estrategia para su desarrollo. Con ello, se quiere dar impulso, por ejemplo, a pequeños proyectos o actividades que puedan ser desarrollados como trabajos de grado y/o monografías de especialización, que servirán para confirmar o rechazar hipótesis, y con el fin de alimentar futuros proyectos.

OBJETIVOS Y RESULTADOS ESPERADOS

El objetivo general de este trabajo es analizar y comparar las diferentes metodologías para la generación de pruebas tempranas a partir de la especificación de requisitos funcionales, para determinar cuál o cuáles de ellas aplicar.

Para llevar a cabo este objetivo primero se deberá seleccionar un caso real al cuál se aplicarán dichas metodologías. Luego, determinar las metodologías que se utilizarán y los criterios con los cuales se realizará la comparación de las mismas. Por último, generar casos de prueba para comprobar si el sistema definido en el caso real, implementa el comportamiento especificado en sus requisitos funcionales.

De esta investigación se espera que sus resultados se incorporen en los contenidos de las cátedras relacionadas y al espacio curricular correspondiente.

Como resultados indirectos se espera la consolidación del grupo de investigación, la formación de nuevos investigadores y la motivación y entrenamiento en investigación de los estudiantes de grado.

FORMACIÓN DE RECURSOS HUMANOS

El equipo de trabajo está integrado por 3 docentes de la carrera de Ingeniería en Informática, 2 adjuntos y 1 Jefe de trabajos prácticos, todos con dedicación semi-exclusiva.

El grupo hace difusión y formación de recursos humanos desde las cátedras: Programación I, Sistemas de Información II e Ingeniería de Software.

Se considera de gran interés la incorporación de becarios para motivar a los alumnos de la

carrera de Ingeniería en Informática a realizar su trabajo final de grado en el área de este proyecto.

Además, se prevé que una de las docentes que integran el equipo, desarrolle su tesis de Maestría en el marco de este trabajo.

REFERENCIAS

[1] Natalia Correa, Roxana Giandini (2011). Generación Automática de Casos de Prueba a partir de
Casos de Uso: Una Propuesta Basada en MDD/MDT. 40JAIIO - ASSE 2011 - ISSN: 1850-2792 - Página 168

[2] Javier J. Gutiérrez, María J. Escalona, Manuel Mejías y Jesús Torres (2006). Hacia una propuesta de Pruebas Tempranas del Sistema. XV Jornadas de Ingeniería del Software y Bases de Datos JISBD 2006 José Riquelme - Pere Botella (Eds) .CIMNE, Barcelona, 2006.

[3] Javier J. Gutiérrez, María J. Escalona, Manuel Mejías, Jesús Torres, Arturo Torres-Zenteno. Generación automática de objetivos de prueba a partir de casos de uso mediante partición de categorías y variables operacionales.

[4] J. J. Gutiérrez, M. J. Escalona, M. Mejías, J. Torres. Derivation of test objectives automatically.

[5] J. Heumann, “Generating Test Cases From Use Cases,” The rational edge, http://download.boulder.ibm.com/ibmdl/pub/software/dw/rationaledge/jun01/GeneratingTestCasesFromUseCasesJune01.pdf, 2001.

[6] J. Gutiérrez, M. J. Escalona, M. Mejías, "Analysis of Proposals to Generate System Test Cases From System Requirements," in CAiSE'05 Forum, Porto, Portugal, 2005.

[7] Natalia Correa, Roxana Giandini (2012). Casos de Prueba del Sistema Generados en el Contexto MDD/MDT. 41 JAIIO - ASSE 2012 - ISSN: 1850-2792 - Pág. 91-105

[8] Marc-Florian Wendland, Ina Schieferdecker and Alain Vouffo-Feudjio. 2011. Requirements-driven testing with behavior trees. 2011 Fourth International Conference on Software Testing, Verification and Validation Workshops.

[9] Yasmine Ibrahim Salem y Riham Hassan. 2011. Requirement-Based Test Case Generation and Prioritization. 978-1-61284-185-4/111 - 2011 IEEE

[10] Bill Hasling, Helmut Goetz, Klaus Beetz. 2008. Model Based Testing of System Requirements using UML Use Case Models. 2008 International Conference on Software Testing, Verification, and Validation