Planificación de las Pruebas del Software

Gladys Kaplan[1, 2], Walter Panessi[1], Claudia Ortiz[1], Eugenia Cespedes[1]

[1] Departamento de Ciencias Básicas, Universidad Nacional de Luján (UNLu)
[2] Departamento de Ingeniería e Innovación Tecnológica, Universidad Nacional de La Matanza (UNLaM)
(gkaplan@unlam.edu.ar, wpanessi@unlu.edu.ar, cortiz@unlu.edu.ar, eugeniacespedes@outlook.com)

RESUMEN

La complejidad actual en la construcción de los sistemas de software, ha impuesto la necesidad de mejorar los procesos de construcción con el menor costo y esfuerzo posible. Las pruebas del software, a diferencia de otras etapas del proceso de construcción, no tienen la misma visibilidad para el usuario, lo que determina que ante presiones de tiempo o costos, sea esta etapa la que se redefina según lo planeado, a pesar de que esta decisión atenta directamente sobre la calidad del software. Para asegurar la correcta construcción del software, en relación a las pruebas, se hace indispensable pensarlas integradas y lo más automatizadas posible. En el proyecto de investigación, se ha trabajado sobre la derivación semiautomática de Casos de Prueba a partir de escenarios futuros y en el presente trabajo se ha avanzado sobre la planificación de esos Casos de Prueba obtenidos, para asegurar que la funcionalidad propuesta en los escenarios contemple las dependencias de recursos y funcionales existentes entre las tareas. De esta manera se utiliza la integración de los escenarios para identificar las dependencias y ordenar la ejecución de los Casos de Prueba en las denominadas suites de prueba, que permiten analizar la funcionalidad parcial y total del futuro sistema de software.

Palabras clave: ingeniería de requisitos, casos de prueba funcionales, suite de pruebas.

CONTEXTO

Este trabajo se lleva a cabo en el marco del Proyecto de Investigación "Generación semi automática de casos de prueba a partir de escenarios", radicado en la Universidad Nacional de Luján, en su Departamento de Ciencias Básicas. [DISPOSICION CDD-CB:027-15].

Como parte de las actividades se ha estudiado un mecanismo para generar tempranamente los casos de prueba partiendo del conocimiento obtenido en la etapa de Ingeniería de Requisitos (IR) [1], particularmente en el proceso de requisitos basado en escenarios [2].

Se profundizó luego la relación escenarios futuros – casos de prueba (EF-CP) y se describió un mecanismo de derivación semiautomática [3].

1. INTRODUCCIÓN

Desde hace mucho tiempo se plantea la necesidad de escribir los test de aceptación conjuntamente con los requisitos. Esto permite, por un lado, asegurase que los requisitos sean verificables y por otro, que el futuro desarrollo y su respectivo desarrollador, cuente con ejemplos que le permitan comprender mejor el requisito.

Este trabajo es una evolución de otros anteriores, siempre basándonos en el proceso de requisitos [2], que crea un glosario denominado LEL [4], [5] del dominio de aplicación. El proceso continúa con la construcción de un conjunto de Escenarios Futuros (EF) a partir de los Escenarios Actuales (EA) [6] y su consecuente decisión de incluirlos en el nuevo sistema.

Trabajos previos de diferentes autores han planteado la necesidad de generar los casos de prueba a partir de varios artefactos que permiten documentar requisitos, como la generación de Casos de Prueba a partir de Casos de Uso [7], [8], [9], [10], [11], a partir de algoritmos metaheurísticos [12], utilizando el lenguaje de transformación QVT, otros a partir de diagramas de secuencia extendidos [13] o a partir de diagramas de actividad [14], [15], [16], [17], a partir de diagramas de secuencia [18] o de diagramas de estado [19]. Algunos lo han planteado en general para el lenguaje UML [20], [21], [22], [23] también a partir de especificación de Requisitos [24].

En el presente trabajo continuamos la idea de derivar en forma semiautomática los Casos de Prueba a partir de los Escenarios Futuros [3] pero se incorpora la necesidad de planificar las pruebas agrupando funcionalidad en diferentes niveles de pruebas.

2. LÍNEAS DE INVESTIGACIÓN Y DESARROLLO

Como se ha mencionado anteriormente, los CP obtenidos desde los escenarios futuros, son estrictamente funcionales y con un alto grado de contextualización. Esto permite darle al caso de prueba información muy valiosa para comprender la mejor manera de analizar la funcionalidad propuesta. Es de destacar que los EF no son independientes entre sí, sino que se rigen por dependencias operativas, de recursos y de contextos. Este ordenamiento funcional se representa en el proceso de requisitos al modelar los escenarios integradores [25]. Cuando los EF son integrados se construyen las jerarquías de escenarios que luego se ordenan en secuencias para su ejecución, conformando los escenarios integradores. El mismo proceso de jerarquización y secuencia se aplicó a los casos de prueba para planificar las pruebas del software.

Las mismas pruebas del software se repiten cuando hay cambios en los requisitos. Existen requisitos que no se modifican, lo que hace que parte de la funcionalidad se mantenga inalterable durante parte o todo el proceso de construcción. Debido al costo de volver a realizar todas las pruebas, es que se

permite seleccionar qué parte de la funcionalidad se desea volver a probar. Esto requiere de la intervención del ingeniero de software y de una aplicación que controle y alerte de las dependencias con otros requisitos. Como puede verse en la Tabla 2 la primera columna es para identificar la funcionalidad con el nombre del escenario futuro, para ello se incorpora el escenario raíz de cada jerarquía obtenida en la integración. La columna de Ultima Prueba cumple la función de informar si después del último cambio realizado a ese escenario la prueba fue Aprobada o si se modificó el escenario y no se realizó o fue rechazada. Obviamente las pruebas rechazadas o que no se hicieron deben rehacerse obligatoriamente, pero cuando existen varios escenarios que no fueron modificados es decisión del ingeniero de software volver a probar todo o no hacerlo. Esto atiende a pruebas de gran volumen con parte de los requisitos estables.

Escenario	¿Se debe probar solo?	Última Prueba
EF**a**	No	Aprobado
EF**b**	SI	
EF**c**	SI	
EF**d**	SI	
Ef**e**	SI	
EF**f**	Si	

Tabla 2 – Tabla de alcance de la prueba

En un primer momento del proceso se determinó cuáles eran los CP para cada EF. En la Fig.1 se da un ejemplo aclaratorio del proceso:

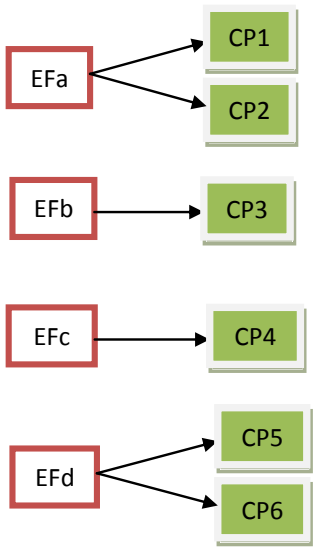

Fig. 1 – Derivación de los CP

Con los CP identificados para cada EF, se debe ir a la integración para generar las suites de prueba. Tomar cada integrador y ordenar los CP correspondientes.

Escenarios Integradores	Escenarios Futuros y sus CP	
I1	EFe	CP7
	EFa	CP1 y CP2
	EFc	CP4
I2	EFb	CP3
	EFd	CP5 y CP6

Tabla 3 – Escenarios integradores con sus EF y cada EF con sus CP derivados

Con esta información se está en condiciones de crear las suites de pruebas. Cada escenario integrador corresponde a una suite. Por lo tanto, según el ejemplo, tenemos 2 Suite de Pruebas con la siguiente secuencia:

Suite 1 {CP7, CP1, CP2, CP4}

Suite 2 {CP3, CP5, CP6}

En resumen, por un lado se cuenta con las pruebas individuales que corresponde a como se debe probar cada EF. Por el otro lado, tenemos los conjuntos de escenarios que se agrupan en integradores que a su vez determinan las suites de pruebas.

3. RESULTADOS OBTENIDOS/ESPERADOS

Los resultados han sido alentadores debido a su vinculación directa con el proceso de requisitos basado en escenarios que ha sido ampliamente probado en más de 100 casos. De todas maneras se planifica probar todo el mecanismo propuesto en nuevos casos reales para mejorar su completitud y refinar el proceso. También se espera analizar cómo se incorporan los cambios de los requisitos en las iteraciones de la generación de los CP.

Como se aclaró en la introducción, el presente trabajo se ha realizado con base en el proceso de requisitos basado en escenarios que modela la solución con escenarios futuros, pero se considera que el proceso es viable para Casos de Uso con algunas pocas modificaciones.

4. FORMACIÓN DE RECURSOS HUMANOS

Se planifica la finalización de la carrera de grado Licenciatura en Sistemas de Información de la alumna Eugenia Cespedes (31 materias aprobadas) y del alumno Julían Massolo (34 materias aprobadas) en UNLu. También la presentación de la tesis de Maestría en Ingeniería de Software de Claudia Ortiz y Walter Panessi en UNLP y la finalización de la tesis doctoral de Gladys Kaplan y de David Petrocelli en UNLP.

5. BIBLIOGRAFÍA

[1] Kaplan Gladys, Doorn Jorge, Panessi Walter, Ortiz Claudia, Cespedes Eugenia, Massolo Julian, Petrocelli David, “Generación semi automática de casos de prueba a partir de escenarios”, WICC 2015.

[2] Leite, J.C.S.P., Doorn, J.H., “Perspectives on Software Requirements: An introduction” en el libro “Perspectives on Software Requirements”, Kluwer Academic Publishers, EEUU, ISBN: 1-4020-7625-8, Capítulo 1, 2004.

[3] Kaplan Gladys, Doorn Jorge, Panessi Walter, Ortiz Claudia, Cespedes Eugenia, “Derivación de casos de prueba a partir de escenarios”, WICC 2017

[4] Leite J.C.S.P., Franco, A.P.M., (1990) “O Uso de Hipertexto na Elicitaçao de Linguagens da Aplicaçao”, Anais de IV Simpósio Brasilero de Engenharia de Software, SBC.

[5] Hadad G.D.S., Doorn J.H., Kaplan G.N. Creating Software System Context Glossaries, In: Mehdi Khosrow-Pour (ed) Encyclopedia of Information Science and Technology. IGI Global, Information Science Reference, Hershey, PA, USA, ISBN: 978-1-60566-026-4, 2nd edn, Vol. II. 2008.

[6] Leite, J.C.S.P., Rossi, G., Balaguer, F., Maiorana, V., Kaplan, G., Hadad, G., Oliveros, A., “Enhancing a Requirements Baseline with Scenarios”, Requirements Engineering Journal, Vol.2, N° 4, 1997.

[7] M. Riebisch, I. Philippow, and M. Götze, “UML-Based Statistical Test Case Generation,” in Revised Papers from the International Conference NetObjectDays on Objects, Components, Architectures, Services, and Applications for a Networked World, 2003, pp. 394-411.

[8] Palacio Liliana González, “Método para generar casos de prueba funcional en el desarrollo de software”, Revista Ingenierías Universidad de Medellín, vol. 8, No. 15 especial, pp. 29-36 - ISSN 1692-3324,150 p. Medellín, Colombia, 2009

[9] Jordán Enriquez, Odalys; Vázquez Ruiz, Orelvis, "Generacion de Casos de Prueba a partir de Casos de Uso en las pruebas del software", Ingeniería Industrial, vol. XXVII, núm. 1, 2006, pp. 7-10 Instituto Superior Politécnico José Antonio Echeverría La Habana, Cuba ISSN: 0258-5960, 2006.

[10] Natalia Correa, Roxana Giandini, "Casos de Prueba del Sistema Generados en el Contexto MDD/MDT", LIFIA- Laboratorio de Investigación y Formación en Informática Avanzada, Universidad Nacional de La Plata, 40 JAIIO, ASSE 2011.

[11] Javier J. Gutiérrez, María J. Escalona, Manuel Mejías y Jesús Torres, Hacia una propuesta de pruebas tempranas del sistema, XV Jornadas de Ingeniería del Software y Bases de Datos. JISBD 2006.

[12] Raquel Blanco, Eugenia Díaz, Javier Tuya," Generación automática de casos de prueba mediante búsqueda dispersa", Revista Española de Innovación, Calidad e Ingeniería del Software, Vol.2, No. 1, 2006.

[13] Beatriz Pérez Lamancha, Macario Polo, "Generación automática de casos de prueba para líneas de producto de software", Revista Española de Innovación, Calidad e Ingeniería del Software, Vol.5, No. 2, 2009.

[14] Chen Mingsong, Qiu Xiaokang, Li Xuandong, "Automatic Test Case Generation for UML Activity Diagrams", Journal of Object Technology. Vol. 8, No. 3, May/June 2009.

[15] Wang Linzhang, Yuan Jiesong ; Yu Xiaofeng ; Hu Jun ; Li Xuandong; Zheng Guoliang, "Generating test cases from UML activity diagram based on Gray-box method", Software Engineering Conference, 2004. 11th Asia-Pacific

[16] P. N. Boghdady, Nagwa L. Badr, M. A. Hashim, Mohamed F. Tolba, "Test Cases Automatic Generator (TCAG): A Prototype", First International Conference, AMLTA 2012, Cairo, Egypt, December 8-10, 2012.

[17] Boghdady, P.N. Badr, N.L.; Hashim, M.A.; Tolba, M.F., "An enhanced test case generation technique based on activity diagrams", Computer Engineering & Systems (ICCES), 2011

[18] Javed, A.Z. "Automated Generation of Test Cases Using Model-Driven Architecture", Automation of Software Test, 2007.

[19] Mourad Badri, Linda Badri and Maxime Bourque-Fortin, "Automated State-Based Unit Testing for Aspect Oriented Programs A Supporting Framework", Journal of Object Technology. Vol. 8, No. 3, May/June 2009.

[20] Abbors, F., Backlund, A.; Truscan, D., "MATERA - An Integrated Framework for Model-Based Testing", Engineering of Computer Based Systems (ECBS), 17th IEEE International Conference 2010

[21] 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

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

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

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

[25] Gladys Kaplan, Graciela Hadad, Jorge Doorn, "Apunte de Ingeniería de Requisitos", UNLaM