Generación semi automática de casos de prueba a partir de escenarios

Gladys Kaplan[1,2], Jorge Doorn[2,3], Walter Panessi[1], Claudia Ortiz[1], Eugenia Cespedes[1], Julian Massolo[1], David Petrocelli[1]

[1] Departamento de Ciencias Básicas, Universidad Nacional de Luján (UNLu)
[2] Departamento de Ingeniería e Innovación Tecnológica, Universidad Nacional de La Matanza (UNLaM)
[3] Escuela de Informática, Universidad Nacional del Oeste (UNO)

(gkaplan@ing.unlam.edu.ar, jdoorn@exa.unicen.edu.ar, wpanessi@unlu.edu.ar, cortiz@unlu.edu.ar, eugeniacespedes@outlook.com, massolo.j@gmail.com, dmpetrocelli@gmail.com)

Resumen

En el presente proyecto se estudiará un mecanismo para generar tempranamente los casos de prueba partiendo del conocimiento obtenido en la etapa de Ingeniería de Requisitos (IR), particularmente en el proceso de requisitos basado en escenarios [Leite 97] [Leite 04]. Se espera definir la planificación de los casos de prueba y la definición cada uno de ellos utilizando el conocimiento existente en los escenarios futuros y en la Especificación de Requisitos de Software (ERS). Para esto se generará una plantilla de registro y seguimiento de los casos de prueba y se analizarán algunos aspectos de rastreabilidad. Es un proyecto inter-etapas pero se enmarca en la de IR debido a que su principal objetivo es validar los requisitos del software, sin embargo se espera que en algunos casos sea necesario adentrarse en el diseño para ubicar el caso de prueba en su entorno real de ejecución. Finalmente como se generará información para la etapa de testing se deben tener en cuenta aspectos propios de sus actividades. Existe una brecha temporal significativa entre la generación de los casos de prueba y su utilización; esto que a primera vista aparece como un inconveniente puede ser utilizado en beneficio del proyecto ya que se obtendría una suerte de control de integridad del proyecto.

Palabras clave: ingeniería de requisitos, casos de prueba, validación de la ERS.

Contexto

Este proyecto es una continuación de otros realizados en diferentes Universidades. Desde el año 1995 a 2000 en proyectos desarrollados en la UB, luego desde 2001 a 2004 en UTN-FRBA y finalmente desde 2005 a la fecha en UNLaM. En paralelo con UNLaM se comienza a trabajar en la UNO y en UNLu, como es el caso del presente proyecto. También existen trabajos en

otras Universidades como ser en la UNLP, UCA, UNICEN, PUC-Rio, UNPA entre otras. En estos proyectos se ha estudiado el proceso de requisitos basado en escenarios definiendo una estrategia, los procesos de construcción de los modelos que lo conforman, las actividades de verificación y validación, la de especificación, priorización y rastreabilidad de los requisitos.

Introducción

La estrategia de requisitos propuesto por Leite et al. [Leite 97][Leite 04], en la cual se ancla el presente proyecto, consiste en, primero, la creación de un glosario denominando léxico extendido del lenguaje (LEL) [Leite 90][Hadad 08] y de un conjunto de escenarios actuales, que representan situaciones observables para la comprensión del universo de discurso, segundo, la construcción de un conjunto de escenarios futuros [Doorn 02] que representan situaciones con el nuevo sistema del software a construir, y finalmente, la especificación de los requisitos del sistema de software basándose en el conocimiento adquirido y registrado en las etapas previas.

La etapa de pruebas del software se encarga de controlar el buen funcionamiento del software y que el mismo cumpla correcta y completamente con la ERS [Meyrs 11]. Para lograr estos objetivos, se deben generan casos de prueba que reflejan como se prevé que el sistema debe ser utilizado y que respuesta es la adecuada. Las pruebas de comportamiento se utilizan para determinar errores en el código donde se evalúa un atributo o capacidad de un programa para determinar si cumple con los resultados esperados [Hetzel 88] [Kuhn 04]. A pesar de que cada tipo de prueba tiene un objetivo propio, la ejecución de una no excluye a la otra, o sea que mientras se ejecutan las pruebas de cubrimiento de la ERS se comprueban errores y viceversa.

Los casos de prueba son la definición de datos de entradas al software el que debe comportarse de una determinada manera y responder según lo esperado. Para cubrir un requisito puede ser necesario realizar varios casos de prueba. Por lo tanto, los casos de prueba tienen dependencias entre ellos.

Existen algunos estudios que han generado casos de prueba a partir de casos de usos. Otros se centran en la generación de casos de prueba específicamente para el código (jPET para programas Java, etc) o aplican los casos de prueba a transacciones (GXTest Generator) [Jacobson 92] [Heumann 01] [Jacobson 03] [Khun 04] [Gutiérrez 07] [Correa 11].

Para visualizar claramente las dificultades que se deben enfrentar al intentar vincular los requisitos de software con los casos de prueba a ser utilizados, exige comprender el hueco tecnológico entre el contenido semántico de los requisitos y sus descripciones contextuales y el contenido semántico de los casos de prueba propiamente dichos. Debe quedar claro que la tarea de definir los casos de prueba

no puede basarse en una estrategia que "salte" ese hueco conceptual, sino en una estrategia que lo "rellene" con conocimiento.

Hay que tener claro entonces que para poder definir los casos de prueba hay que transitar por todas las actividades de diseño, reconociendo todas las decisiones tomadas. Posiblemente uno de los elementos de configuración más importante para lograr la vinculación entre los requisitos y las pruebas es el documento de requirement allocation.

La afirmación anterior permite hacer visible la primera metáfora relacionada con la problemática de crear casos de prueba relacionados con los requisitos. Habría que pensar esta vinculación como una suerte de "test allocation" donde sea posible visualizar dos cosas: i) ¿De qué forma se prueba este requisito? y ii) ¿Con que requisito está vinculado esta prueba? Si bien esta metáfora no es perfecta, puede ayudar a pensar el objeto de estudio de este proyecto en forma más sencilla y clara.

Retornando a la idea de "rellenar" el hueco tecnológico entre los requisitos y los casos de prueba, puede verse que esto puede hacerse en forma temprana o en forma tardía. Hacerlo en forma tardía significa definir un conjunto de casos de prueba en términos de los servicios esperados por el cliente o usuario y luego materializarlos en casos de prueba efectivos atendiendo a todas las decisiones tomadas durante el desarrollo del sistema de software. Hacerlo en forma tardía significa seguir la pista de cada requisito y en los documentos de requirement allocation, postergando la definición de las pruebas mismas.

A lo largo del proyecto se espera probar la hipótesis de trabajo subyacente que considera la calidad de los casos de prueba originados en la etapa de requisitos es superior a la calidad de los casos de prueba creados con posterioridad.

Líneas de Investigación, Desarrollo e Innovación

Se cuenta con aproximadamente 150 casos, algunos académicos y otros profesionales, en los cuales se ha aplicado el proceso de requisitos basado en escenarios los cuales serán utilizados para detectar aspectos funcionales y no funcionales de los escenarios futuros o en la propia ERS que permitan generar casos de prueba.

Luego, se espera realizar la comprobación de los casos de prueba en casos nuevos cuyo proceso de construcción del software se encuentre avanzado. En estos casos se pasará a una nueva fase de comprobación real y efectiva de las pruebas propuestas. Esto permitirá validar el correcto funcionamiento del mecanismo propuesto y su cubrimiento.

Resultados y Objetivos

Recordando que el documento de especificación de requisitos (ERS) es el

resultado más importante de todo proceso de requisitos y que los escenarios futuros dan contexto al mismo; es que se pretende centrar los casos de prueba en aquel documento, logrando el contexto necesario con los escenarios futuros y los documentos de diseño que corresponda.

Se espera además poder ofrecer un control de calidad del tratamiento de los cambios en los requisitos a lo largo de todo el proceso de desarrollo de software ya que a la hora de materializar las pruebas, cada caso de prueba debe ser semánticamente consistente con el o los requisitos relacionados.

En un plano más general se espera identificar características comunes en los modelos de requisitos y en su relación con algunos modelos de diseño que permitan definir los casos de prueba. Se buscarán patrones de funcionalidad que permitan reutilizar esos casos de prueba en distintos proyectos. Paralelamente, se generará una ficha de registro para documentar los casos de prueba y los resultados obtenidos. El análisis de estos registros con sus rastros hacia sus orígenes permitirá definir una heurística para generar los casos de prueba funcionales de manera automática.

La difusión del avance del proyecto se realizará en UNLu en la asignatura de grado “Sistemas de Información IV” y en la UNLaM en los cursos de grado “Proceso de Software”, “Ingeniería de Requerimientos” y en el curso de posgrado “Tópicos de Ingeniería de Requisitos”. Esta transferencia de conocimiento permitirá capacitar a jóvenes profesionales en el ámbito informático, los cuales harán uso directo o indirectamente de los procesos formales que se desarrollarán en el devenir del proyecto de investigación.

Formación de Recursos Humanos

Se planifica la finalización de la carrera de grado Licenciatura en Sistemas de Información de la alumna Eugenia Cespedes (23 materias aprobadas) y del alumno Julían Massolo (25 materias aprobadas) en UNLu.

Se planifica la finalización de la tesis de Doctorado en Informática de Gladys Kaplan en UNLP.

Referencias

[Correa 11] Correa Natalia, Giandini Roxana, 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.

[Doorn 02] Doorn J., Hadad G., Kaplan G. (2002) Comprendiendo el Universo de Discurso Futuro, WER’02 - Workshop on Requirements Engineering, Valencia, Spain.

[Gutiérrez 07] Gutiérrez Javier, Escalona María, Mejías Arturo Torres Manuel, Torres Jesús, Generación e implementación de pruebas del sistema a partir de casos de uso. Revista

Española de Innovación, Calidad e Ingeniería del Software, Vol.3, No. 3, 2007.

[Hetzel 88] Hetzel, William C., The Complete Guide to Software Testing, 2nd ed. Publication info: Wellesley, Mass.: QED Information Sciences, 1988.

[Hadad 08] Hadad G.D.S., Doorn J.H., Kaplan G.N. (2008) 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.

[Heumann 01] Heumann, Jim, Generating Test Cases From Use Cases, TheRationalEdge, June, 2001.

[Jacobson 92] Jacobson I. ; Christerson M. Jonsson P., Övergaard G, Object-Oriented Software Engineering: A Use Case DrivenApproach, Addison-Wesley, 1992.

[Jacobson 03] Jacobson I, Booch G, Rumbeagugh J., Use Cases: Yesterday, Today, and Tomorrow, TheRationalEdge, March, 2003.

[Khun 04] Kuhn, D. R., Wallace, D. R., and Gallo, A. M. Software fault interactions and implications for software testing. IEEE Transactions on Software Engineering, 2004.

[Leite 97] 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.

[Leite 90] 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.

[Leite 04] 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.

[Meyrs 11] Glenford J. Myers, Corey Sandler and Tom Badgett, The Art of Software Testing, 2011