Incorporando Alloy en Desarrollos basados en Metodologías Agiles

María Belén Bonino[1], Ana Garis[2], Daniel Riesco[2]

[1]Universidad Tecnológica Nacional - Facultad Regional San Francisco
Av. de la Universidad 501, San Francisco, Córdoba, Argentina
bonino.mb@frsfco.utn.edu.ar

[2] Universidad Nacional de San Luis
Ejército de los Andes 950, San Luis, Argentina
{agaris,driesco}@unsl.edu.ar

Resumen

Las *Metodologías Ágiles* (MA) son ampliamente elegidas en proyectos de software actuales. No obstante, frecuentemente deben afrontar problemas inherentes a su falta de formalidad; tales como la dificultad de recoger, entender y mantener los requerimientos del sistema.

Alloy es un lenguaje formal, soportado por una amigable herramienta de verificación y validación. Estas caracterísiticas pueden ser aprovechadas en favor de las MA.

La línea de investigación se enfoca puntualmente en la adopción de Alloy dentro de las MA mas populares. Esto incluye analizar no solo los beneficios, sino también la forma en que dicho lenguaje puede ser incorporado en el proceso de desarrollo de software.

Palabras clave: Métodos Formales, Alloy, Metodologías Ágiles.

Contexto

La línea de Investigación se enmarca en el Proyecto de Investigación "Ingeniería de Software: Aspectos de Alta Sensibilidad en el Ejercicio de la Profesión del Ingeniero de Software". Facultad de Ciencias Físico Matemáticas y Naturales, Universidad Nacional de San Luis.

Introducción

Los Métodos Formales (MF) y las MA, tradicionalmente planteados como opuestos, proponen diferentes enfoques para el desarrollo de software. Mientras que los primeros se basan en el uso de métodos matemáticos rigurosos, los segundos sugieren un desarrollo más informal.

Las MA se basan en el manifiesto ágil [1] que estipula un conjunto de valores básicos, priorizando

- al individuo y el equipo de desarrollo sobre el proceso y las herramientas,
- al desarrollo de un software que funcione, más que conseguir una buena documentación,
- la colaboración activa y continua del cliente durante todo el desarrollo,
- la habilidad de respuesta ante los cambios.

Estos valores determinan principios caracterizados por un desarrollo informal, procesos menos controlados, más

flexibles, quitando énfasis en la arquitectura del software y procurando la generación rápida de productos funcionales.

Por otro lado, los MF se basan en la representación del software con notaciones formales: un lenguaje con sintaxis y semántica precisa basados en nociones matemáticas. Ello supone ventajas en el desarrollo de software ya que permite realizar especificaciones no ambiguas, identificar y demostrar cuáles son las propiedades deseables del sistema, entre otras.

De esta manera, los MF y las MA se plantean tradicionalmente como conceptos destinados a definir y proponer enfoques opuestos para el desarrollo de software, confrontando desarrollos matemáticos rigurosos con otros más informales.

Las MA son ampliamente elegidas en proyectos de software de diferentes características [2, 3]. Uno de los desafíos abiertos que afrontan estos enfoques es la necesidad de recoger, entender y mantener los requerimientos del sistema de una manera eficiente y efectiva, y mejorar la comunicación informal del equipo de desarrollo, provocada principalmente por la carencia de un diseño inicial y la inversión en el modelado dentro del ciclo de vida. Otro aspecto crucial se refiere a la automatización del proceso de prueba o testing, factor clave en este tipo de metodologías, y su trazabilidad en relación a los requerimientos del sistema.

A pesar de que los MF tienen una amplia aceptación en el desarrollo de sistemas críticos, tienen poca penetración en la Industria de Software convencional, en parte debido a que los ingenieros de software, en general, los consideran difíciles de aprender y usar.

Algunos trabajos han mostrado que es posible combinar MF y MA en proceso de desarrollo de software [4, 5, 6]. Wolff propone un mecanismo para utilizar MA en el desarrollo de sistemas críticos, aplicando MF en Scrum [4]. Shafiq et al. presentan una solución conceptual para integrar MF en XP (por sus siglas en inglés, de eXtreme Programming) [5]. Mochio et al. establecen un método de desarrollo de software formal ágil escalable, basado en VDM++ [6].

Alloy [7] es un lenguaje de modelado catalogado como *liviano* dentro de la comunidad de los métodos formales. Su lenguaje se basa en la simple noción matemática de relación, el cual es soportado por un poderoso analizador automático SAT.

Aunque Alloy está alcanzando un estado de madurez como lenguaje de modelado, todavía es necesario profundizar en el estudio para extraer su máximo potencial dentro de las metodologías de desarrollo de software actuales.

La incorporación de Alloy dentro de las MA podría favorecer el desarrollo de aplicaciones de alta calidad, exponiendo defectos y excepciones. Por ejemplo, Alloy podría ser explotado para generar casos de prueba automáticamente, clarificar los requerimientos y diseño a alto nivel, articular asumisiones implícitas (generando requerimientos implícitos a partir de reglas semánticas explícitas) e identificar asumisiones no-documentadas o inesperadas.

La línea de investigación propone la adopción de Alloy dentro de las MA actuales mas populares. El trabajo esta dirigido a estudiar técnicas y herramientas con base en Alloy para contribuir a mejorar la calidad de los productos creados a partir de las MA.

Líneas de Investigación, Desarrollo e Innovación

Los ejes de la investigación se basan en los siguientes puntos.

- El análisis de antecedentes en dónde se describa la utilización de Alloy en las MA, en general; y por otro lado, el estado del arte referido al uso de MF en MA.

- El estudio para la adopción de Alloy en las MA en función de los beneficios que ofrece el lenguaje y su herramienta. Entre los atributos a considerar se encuentran la capacidad para facilitar la definición de requerimientos, identificar asumisiones implícitas, defectos y excepciones, y para generar casos de prueba a partir del modelo.

- La evaluación de los puntos antes citados, sobre las MA mas populares. Teniendo en cuenta las particularidades de cada una de las MA evaluadas, establecer cómo Alloy debería ser incluido en el proceso de desarrollo de software.

Resultados y Objetivos

Luego de llevar a cabo un estudio preliminar de antecedentes, se ha observado el potencial de Alloy para MBT (por sus siglas en inglés de Model Based Testing) y su relación con TDD (por sus siglas en inglés de Test-Driven Development) [8, 9, 10]. El presente trabajo plantea experimentar con estas técnicas procurando no solo generar casos de prueba automáticamente, sino también estudiar su trazabilidad con los requerimientos del sistema.

Estos atributos, provistos por Alloy, deben ser considerados en función de las MA mas populares de la actualidad. En este sentido, se propone a Scrum [11] como metodología a analizar. La línea de investigación tiene por objetivo determinar, si es posible aplicar Alloy en Scrum. Para esto es necesario establecer en qué fase del desarrollo de software podría ser incluido y de qué forma.

Como trabajo futuro se pretende estudiar el enfoque conocido como Example Driven Modeling [12] y su aplicación en el contexto de las MA con Alloy. En forma complementaria se espera profundizar en el estudio de la generación de requerimientos implícitos a partir de reglas semánticas explícitas utilizando Alloy, y la generación automática de modelos (o código) a partir de modelos Alloy.

Formación de Recursos Humanos

La línea de investigación forma parte de un trabajo de tesis correspondiente a la Maestría en Ingeniería de Software, Universidad Nacional de San Luis.

Referencias

[1] Manifiesto por el Desarrollo Ágil del Software, disponible en http://www.agilemanifesto.org/

[2] E. Bahit, “Scrum & eXtreme Programming para programadores”, Distribuido bajo Licencia Creative Commons, 2012.

[3] S. R. Kenneth, “Essential Scrum: A Practical Guide to the Most Popular Agile

Process", Addison-Wesley Signature Series (Cohn), 2012.

[4] S. Wolff, "Scrum goes formal: Agile methods for safety-critical systems," Software Engineering: Rigorous and Agile Approaches (FormSERA), 2012 Formal Methods in , vol., no., pp.23,29, 2012.

[5] S. Shafiq y N. M. Minhas, "Integrating Formal Methods in XP—A Conceptual Solution", Journal of Software Engineering and Applications, Vol.7 No.4, 2014.

[6] H. Mochio y K. Araki, "VDM++ as a basis of scalable agile formal software development". Proceedings of the 9th Overture Workshop, Aarhus University, 2011.

[7] D, Jackson, "Software Abstractions: Logic, Language, and Analysis", MIT Press, edición revisada, 2012.

[8] D. Marinov y S. Khurshid, "TestEra: A Novel Framework for Automated Testing of Java Programs". En Proc.16th ASE, pp 22–31, 2001.

[9] Ch. Boyapati, S. Khurshid y D. Marinov, "Korat: Automated Testing Based on Java Predicates". En Proc.ISSTA, 2002.

[10] D. Rayside, A. Milicevic, K. Yessenov, G. Dennis y D. Jackson, "Agile specifications". OOPSLA Companion 2009: 999-1006, 2009.

[11] K. Schwaber y M. Beedle, "Agile Software Development with Scrum". Prentice Hall, Upper Saddle River, 2002.

[12] K. Bak, D. Zayan, K. Czarnecki, M. Antkiewicz, Z. Diskin, A. Wasowski, y D. Rayside. "Example-Driven Modeling: Model; Abstractions; Examples". En Software Engineering (ICSE), 35th International Conference on, pages 1273–1276, 2013.