Raccis, 1(1), 27-29, 2011.

Revista Antioqueña de las
Ciencias Computacionales y la Ingeniería de Software

ISSN: 2248-7441

www.fundacioniai.org/raccis/index.htm
raccis@fundacioniai.org

An Empirical Study to Specify Requirements Using UML and Z

Un Estudio Empírico para Especificar Requisitos Utilizando UML y Z

Ana María Potosí[1], Teresita Cordero[2]
[1] UCA. Panamá, Panamá. a*mapo(AT)ucapanama.org*
[2] UCA. Panamá, Panamá. *tecoro(AT)ucapanama.org*

INFORMACIÓN DEL ARTÍCULO

Tipo de artículo
Investigación

Historia del artículo
Recibido: 17-09-2011
Correcciones: 10-12-2011
Aceptado: 15-12-2011

Categories and Subject Descriptors
D.2.1 [**Software Engineering**]: Requirements/Specifications – *Methodologies*.

General Terms
Software Engineering, Requirements Engineering, Formal Methods.

Keywords
Empirical Study, Requirements Specification, UML, Z Language.

Palabras clave
Estudio Empírico, Especificación de Requisitos, UML, Lenguaje Z.

ABSTRACT
This article presents the results of an empirical study which used some UML diagrams –semi-formal modeling– and Z language –formal modeling– to do a specification of system requirements. The results showed the benefits of formal modeling to improve the quality of specification requirements of an information system. In the experiment also was perceived the need for a clear correspondence between the approaches modeling used. This correspondence was established during the experiment, allowing the correct joint specification.

RESUMEN
En este artículo se presentan los resultados de un estudio empírico en el que se utilizaron algunos diagramas UML –modelado semi-formal– y el lenguaje Z –modelado formal– para realizar una especificación de requisitos de un sistema. Los resultados mostraron los beneficios del modelado formal para mejorar la calidad de la especificación de requisitos de un sistema de información. En el experimento también se percibió la necesidad de una correspondencia clara entre los enfoques de modelado utilizados. Esta correspondencia fue establecida durante el experimento, lo que permitió articular correctamente la especificación.

1. INTRODUCCIÓN

Modelar parte de los aspectos de un sistema software se ha convertido actualmente en una importante actividad del proceso de desarrollo, sobre todo en las primeras fases cuando se presta especial atención a la definición de requisitos. Por lo general, los requisitos se elicitan, modelan, discuten y finalmente se especifican, de tal forma que se origina un documento en el que se especifica detalladamente lo que el software debe hacer. Este documento se utiliza como una guía para las próximas fases del ciclo de vida del producto software.

La mayoría de documentos de especificación contienen especificaciones descritas en lenguaje natural y notaciones cognitivas, que es una especificación semi-formal que utiliza diagramas y una modalidad que utiliza gran parte de la comunidad del desarrollo de software. Sin embargo, la especificación semi-formal con diagramas, que utilizan los Entity-Relationship Diagram ERD y los Data-Flow Diagram DFD y, más recientemente, el Unified Modeling Language UML [1], frecuentemente presenta ambigüedades e imprecisiones que pueden ocasionar implementaciones erróneas del software.

En los últimos 20 años se propusieron varios métodos formales para especificar el software, con el objetivo de ofrecer notaciones y semánticas precisas para la especificación, de tal forma que se evitara tanto la ambigüedad como la imprecisión y, por consiguiente, para disminuir los errores en la implementación del software. Sin embargo, en comparación con los métodos semi-formales, en este mismo período de tiempo los métodos formales han sido poco utilizados por la comunidad del desarrollo de software. Algunas de las razones son:

- Todavía no existen herramientas de software "maduras" que generen código fuente para utilizar en la implementación del software, lo que genera una brecha entre la especificación formal y la implementación del programa.

- Los desarrolladores de software tienen poco contacto con los métodos formales en los procesos de formación académica.

- Las notaciones formales son difíciles de comprender en un corto tiempo, debido a que por lo general se requieren buenos conocimientos en matemáticas y los programas de formación no suplen adecuadamente ese requerimiento.

Esta investigación tiene como objetivo de contribuir a la popularización de los métodos formales para el desarrollo cotidiano de software, especialmente en el segmento de los Sistemas de Información. El experimento aplicado reúne los enfoques de especificación formal y semi-formal, lo que demuestra que es posible y útil utilizar ambos enfoques en un mismo proceso de especificación de requisitos.

2. DESARROLLO DEL ESTUDIO EMPÍRICO

El Sistema de Información seleccionado fue un subconjunto de un sistema de plan de salud. La función de este subconjunto es gestionar la información acerca de la atención domiciliaria de pacientes. Para la organizaci{on, este tipo de tratamiento cuenta con 377 pacientes y 12 profesionales de la salud que atienden 1.700 llamadas al mes. Los actores del sistema son: tres usuarios, un jefe de sección y dos ingenieros de requisitos –uno senior y uno junior.

2.1 *Especificación semi-formal*

La especificación de software comenzó con un modelado semi-formal de la información alrededor del sistema en análisis. La información se obtuvo mediante varias entrevistas con las personas que utilizan el Sistema de Información: los profesionales de la salud, los empleados y el jefe de sección. Con la información obtenida en las entrevistas se diseñó el primer modelo: el diagrama de casos de uso. Después de que los usuarios validaron los requisitos iniciales, representados en el diagrama de casos de uso, se construyó un modelo de datos de la aplicación utilizando el diagrama de clases de UML. Este modelado semi-formal fue comprendido por todos los actores, lo que les permitió validar sin ningún problema los documentos generados.

2.2 *Especificación formal*

El siguiente paso fue construir la especificación formal. Se inició a partir de la información detallada en los modelos de especificación semi-formal, es decir, el diagrama de casos de uso y el diagrama de clases. La notación adoptada para el modelo formal fue el Lenguaje Z [2, 3]. Durante este proceso la estrategia adoptada fue la siguiente:

1. Especificar todas las clases del diagrama de clases como un tipo en Z, utilizando su esquema de notación sin la parte predicativa.

2. Especificar todas las relaciones entre las clases como funciones o relaciones en Z.

3. Especificar todos los casos de uso del diagrama como un esquema en Z, mostrando en el predicado parte del esquema de las limitaciones y de las operaciones, necesarias para el funcionamiento eficaz de los casos de uso.

El documento base para estructutar los tipos Z fue el diagrama de clases. La definición de los tipos en Z fue la primera parte del modelado formal del Sistema de Información. Para cada clase en el diagrama se creó un tipo en Z, en el que cada clase tenía varios atributos que requerían la definición de otros. Debido a que las clases representan estructura de datos, los tipos correspondientes definidos en Z se especificaron utilizando la noción de esquema, lo que facilita la correspondencia entre las clases de UML y los tipos en Z.

El siguiente paso consistió en especificar formalmente las relaciones entre las clases. Para tal propósito y debido a que Z ofrece buenos recursos para los modelos de relaciones se utilizó el concepto de relación entre conjuntos. Para el "Modelo Paciente" se identificaron las siguientes: tres relaciones, dos biyecciones, una función total, una función parcial, una inyección parcial, una inyección total, una sobreyectiva parcial y una sobreyectiva total. Para encontrar la correspondencia correcta entre las relaciones de clases en UML y la relación de tipos en Z, se realizó una actividad que requirió gran esfuerzo debido a que los ingenieros de requisitos no estaban muy familiarizados con el lenguaje Z y a que tales correspondencias no eran tan evidentes. Por lo tanto, la correspondencia entre estos dos enfoques de modelado de relaciones fue muy importante para especificar de forma correcta las relaciones entre los tipos Z, que fueran equivalentes al modelo de clases.

Los casos de uso expresan los requisitos funcionales de un sistema. Fue muy útil utilizar diagramas de caso de uso para el trabajo inicial con los requisitos, debido a que desde el primer momento abstraen los detalles inútiles –durante el proceso de elicitación de requisito– y este diagrama puede ser comprendido fácilmente por los usuarios. Sin embargo, después de elicitar los requisitos funcionales fue necesario analizarlos de forma más profunda, explorando las relaciones y las dependencias entre ellos y analizando los datos relevantes en el contexto de cada requisito, además de sus limitaciones. Con la especificación formal de los casos de uso en el experimento se obtuvieron importantes beneficios, lo que permitió alcanzar el perfeccionamiento deseado de los casos de uso. La precisión del modelado formal hace posible identificar nueva información acerca de los requisitos y permite una mejor visión de los mismos.

3. CONCLUSIONES

La experiencia reportada demuestra que los modelos semi-formales son apropiados para la especificación de requisitos de un sistema, debido a que ayudan a abstraer detalles innecesarios para la primera fase de la Ingeniería de Requisitos, facilitan la comunicación entre los actores del sistema y permiten que el proceso de análisis de requisitos sea fluido y productivo. Sin embargo, para la

última fase de la Ingeniería de Requisitos se requiere una especificación detallada de los mismos, por lo que los modelos semi-formales son insuficientes debido a que no tienen recursos para una especificación precisa y estricta. Este es el punto donde es muy útil la especificación formal, lo que se confirma en el experimento reportado en este trabajo.

Los requisitos de calidad se mejoraron con la especificación Z y, por tanto, esto contribuyó para un mejor desarrollo del software del Sistema de Información analizado. Al final del experimento se constató que, con la especificación formal, se mejoraron los posteriores atributos de calidad de la especificación de requisitos, en lo que tiene que ver con precisión, integridad, exactitud y no ambigüedad.

Al articular la especificación semi-formal –UML– y la especificación formal –Z– se obtuvo un documento de especificación de requisitos mejorado, porque los modelos se complementaron y fue posible construir una documentación sólida de cómo deben implementarse en el software.

Luego de la realización de este trabajo, la propuesta es explorar mejor las correspondencias entre los elementos de modelado del UML y de Z, porque el experimento puso en evidencia la importancia de esa correspondencia para articular estas técnicas de especificación de software.

4. REFERENCIAS

[1] OMG. 2001. Unified Modeling Language Specification.

[2] Moura, A. V. 2001. Especificações em Z: uma Introdução. Editora da Unicamp.

[3] Spivey, J. M. 1998. The Z Notation: A Reference Manual. Prentice Hall International