Extracción de Información Estática de Programas Escritos Usando el Paradigma Orientado a Objetos

Arnaldo Ceballos, Hernán Bernardis, Enrique Miranda, Mario Berón, Daniel Riesco

Departamento de Informática/ Facultad de Ciencias Físico Matemáticas y Naturales/ Universidad Nacional de San Luis

Ejercito de Los Andes 950 - San Luis - Argentina

arnaldoceballos@gmail.com, {hbernardis, eamiranda, mberon, driesco}@unsl.edu.ar

Resumen

La Comprensión de Programas (CP) es una disciplina de la Ingeniería del Software cuyo objetivo es facilitar el entendimiento de los sistemas. Esta disciplina está influenciada en gran medida por el tamaño de los mismos. Es decir, mientras más grande es el código del sistema, más complejo se hace su entendimiento.

Comprender un sistema de manera correcta y rápida disminuye claramente el costo de actividades tales como: mantenimiento, reingeniería, evolución; lo cual representa una de las características más importantes de esta disciplina.

Entre los principales desafíos en la CP se destaca lo siguiente: lograr reconstruir la relación entre el Dominio del Problema y el Dominio del Programa. El primero hace referencia a la salida del sistema y el segundo a las componentes utilizadas para producir dicha salida.

Para el caso particular del Dominio del Programa, uno de los lugares más significativos desde donde se puede extraer información es el código fuente. La información que se obtiene de esta extracción se clasifica en estática, si es obtenida sin ejecutar el programa y dinámica, si es obtenida en tiempo de ejecución.

En este artículo se describe una línea de investigación centrada en la extracción de información estática de los sistemas para facilitar el proceso de comprensión de los mismos.

Palabras claves: *Comprensión de Programas, Dominio del Programa, Extracción de la Información Estática.*

Contexto

La línea de investigación descrita en este artículo se encuentra enmarcada dentro del proyecto: Ingeniería de Software: Aspectos de alta sensibilidad en el ejercicio de la profesión de Ingeniero de Software de la Universidad Nacional de San Luis. Dicho proyecto, es reconocido por el programa de incentivos y es la continuación de diferentes proyectos de investigación de gran éxito a nivel nacional e internacional.

Este proyecto se desarrolla en el marco del Área de Programación y Metodologías de Desarrollo de Software y del Laboratorio de Calidad e Ingeniería de Software, de la Facultad de Ciencias Físico Matemáticas y Naturales de la Universidad Nacional de San Luis (Argentina) y del Grupo de

Procesamiento de Lenguajes del departamento de informática de la Universidade do Minho (Portugal).

Introducción

La Comprensión de Programas [1,2,3] es un área de la Ingeniería del Software destinada a elaborar métodos, técnicas y herramientas, basadas en un proceso cognitivo y un proceso de ingeniería, para lograr un conocimiento profundo de un sistema de software.

El proceso cognitivo [4,5] implica el estudio y análisis de las fases y pasos seguidos por los programadores para comprender un sistema. El proceso de ingeniería [1] tiene la finalidad de representar la información del sistema de manera que tal que enfatice sus principales aspectos. Este proceso implica el estudio de áreas tales como: Visualización de Software, Extracción de la Información, Administración de la Información.

El proceso de comprensión de programas se traduce en la habilidad de entender una pieza de código escrito en un lenguaje de alto nivel. El lector de un programa consigue extraer el significado del mismo cuando comprende de que forma el código cumple con la tarea para la cual fue creado [6].

Por lo expuesto en el párrafo anterior, se puede afirmar que el principal desafío en la CP consiste en lograr relacionar correctamente el Dominio del Problema y el Dominio del Programa [1]. Una forma de alcanzar este objetivo consiste en:

1. Extraer información del Dominio del Problema y del Dominio del Programa.
2. Proveer representaciones para ambos dominios a partir de esta información.
3. Definir una estrategia de vinculación que permita unir ambas representaciones.

Para poder construir la representación de cada dominio, previamente es necesario extraer información (EI) de cada uno de ellos. La Figura 1 visualiza los pasos previamente mencionados.

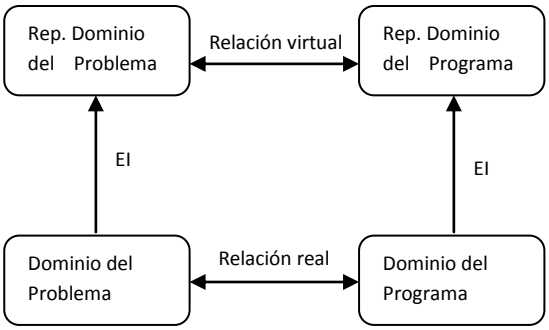

Figura 1. Modelo de Comprensión de Programas.

El proceso de extracción de información desde el Dominio del Problema es un gran desafío en la Ingeniería del Software, debido a su elevada complejidad. Dicha complejidad generalmente se debe a:

- La carencia de conocimiento del programador del sistema provocada, entre otras cosas, por la falta de documentación o cuando aquellos que intentan comprenderlo no son los creadores del mismo.
- El olvido de los aspectos relacionados al mismo por parte de los creadores luego de un tiempo.

Por otro lado, el Dominio del Programa es más palpable y concreto que el Dominio del Problema, porque está relacionado a las componentes que generan el comportamiento del sistema, presentes en el código fuente. Por lo tanto, el análisis sobre el código fuente permite obtener una gran cantidad de información del sistema y, con ella, crear representaciones del Dominio del Programa. La información que puede extraerse desde éste, se puede clasificar en dos tipos:

- Dinámica: es la que se obtiene a través de la ejecución del programa. Un ejemplo de estrategias de Extracción de la Información Dinámica es la Instrumentación de Código [7,8]. Esta estrategia consiste en insertar instrucciones dentro del código fuente del programa de modo que cuando éste se ejecute, proporcione información relacionada al comportamiento del mismo durante su ejecución.
- Estática: es aquella que se adquiere a través del uso de técnicas de compilación tradicionales, como por ejemplo: análisis lexicográfico, análisis sintáctico y análisis semántico del código fuente. Este tipo de información es la que comúnmente manejan los compiladores para: determinar la correctitud de los programas, generar código, optimizar código, entre otras tantas.

En este artículo se describe una línea de investigación que se centra en el estudio y creación de Técnicas de Extracción de la Información Estática desde el Dominio del Programa.

La organización de este artículo se expone a continuación. La sección 3 describe la línea de investigación. La sección 4 presenta los resultados obtenidos y esperados. Finalmente, la sección 5 describe brevemente las tareas realizadas en el contexto de los recursos humanos.

3. Línea de Investigación y Desarrollo

La información estática es obtenida sin ejecutar el programa. Una forma posible de hacerlo es usando técnicas de compilación. Dichas técnicas permiten capturar nombres de variables, tipos de variables, nombres de procedimientos/métodos, variables locales a un procedimiento/método, etc.

Los procesos de extracción de información estática son fundamentales para la Representación del Dominio del Programa. Éstos permiten detectar errores lexicográficos, sintácticos y semánticos, relaciones entre las componentes del sistema, entre otras.

Básicamente esta línea de investigación se centra: i) En el estudio de diferentes herramientas de extracción de la información, tales como: Analizadores Lexicográficos, Analizadores Sintácticos, Depuradores, Profilers, entre otros; ii) En la elaboración e implementación de representaciones del Dominio del Programa basadas en la información estática. A modo de ejemplo se pueden mencionar: El grafo de funciones, el grafo de dependencias de módulos, el grafo de dependencias del sistema, etc. iii) En la definición de métodos para seleccionar la herramienta de extracción más apropiada para un Dominio del Programa específico, iv) En el análisis y creación de técnicas de compilación.

Además de las temáticas mencionadas en el párrafo precedente, el grupo de investigación también estudia diferentes formas de visualizar la información extraída. Esta tarea tiene como finalidad proporcionar un conjunto de técnicas y herramientas que disminuyan sustancialmente la brecha existente entre el conocimiento del ingeniero y el subyacente al sistema que se desea entender.

4. Resultados y Objetivos

Los resultados obtenidos hasta el momento se vinculan estréchamente al lenguaje de programación Java y a un generador de analizadores sintácticos que ha demostrado

ser robusto. Dichos resultados se mencionan a continuación:

- Se analizó una especificación de la gramática del lenguaje Java y se pudo concluir que es posible insertar acciones semánticas, para obtener mediante atributos sintetizados y heredados la información requerida.
- Se estudiaron herramientas de creación automática de Analizadores Lexicográficos y Sintácticos que permiten utilizar los conceptos anteriormente mencionados (atributos sintetizados, heredados y acciones semánticas) en la gramática. Del estudio previamente mencionado se pudo concluir que AnTLR [9] es la herramienta más apropiada para llevar a cabo la inspección estática del código.
- Se usó AnTLR con gramáticas de ejemplo para realizar inspecciones de prueba.

Dentro de los objetivos planteados a corto plazo se encuentran:

- La construcción de una herramienta que genere distintas vistas a partir de la información estática obtenida de programas escritos en Java.
- Ampliar la cantidad de información extraída, y de esta manera poder crear representaciones aún más completas.
- Crear ontologías sobre la información extraída de manera de facilitar la reconstrucción virtual de la relación real Dominio del Problema - Dominio del Programa. Se pretende crear ontologías para describir los conceptos y las relaciones pertenecientes al Dominio del Problema, con el objetivo de sistematizar una representación de este dominio, y facilitar la identificación del mapeo de estos conceptos con secuencias de ejecución.

Todos los trabajos futuros mencionados previamente permiten percibir la importancia de la línea de investigación presentada en este trabajo para la Comprensión de Programas y la Ingeniería Inversa.

5. Formación de Recursos Humanos

Actualmente, los temas abordados por esta línea de investigación están siendo desarrollados como parte de trabajos de Licenciatura en Ciencias de la Computación.

Además se está trabajando en conjunto con integrantes que abordan temas fuértemente relacionados con el de la temática descrita en este artículo. Este grupo de investigadores se encargan de:

1. Extraer información para la construcción de ontologías que permitan representar el Dominio del Programa.
2. Extraer información dinámica para representar el Dominio del Programa.
3. Elaborar estrategias de visualización de software.
4. Elaborar métodos de evaluación para facilitar la toma de decisiones en un amplio rango de herramientas y métodos usados en Comprensión de Programas.
5. Estudiar y definir lenguajes específicos del dominio.
6. Estudiar lenguajes orientados a la web.

El objetivo principal de este trabajo colaborativo es: Proporcionar resultados relevantes para el área de la Comprensión de Programas, y a partir de estos resultados definir tesis de maestría y doctorado. Se pretende que las tesis, antes mencionadas, se puedan llevar a cabo en el ámbito nacional

como así también en el internacional. Fomentando, de esta manera, la cooperación inter-universitaria.

Referencias

[1] Mario Berón; Roberto Uzal; Pedro Rangel Henriques; Maria João Varanda Pereira. Inspección de Código para relacionar los Dominios del Problema y Programa para la Comprensión de Programas. X Workshop de Investigadores en Ciencias de la Computación. 2008.

[2] Mario M. Berón, Daniel Riesco, Germán Montejano, Pedro R. Henriques, Maria J. Pereira. Estrategias para Facilitar la Comprensión de Programas. XII Workshop de Investigadores en Ciencias de la Computación. 2010.

[3] Liberman H.; Fry C. Bridging the Gulf Between Code and Behavior in Programming. ACM Conference on Computers and Human Interface. Denver, Colorado. 1994.

[4] M.-A.D. Storey, F. D. Fracchia, and H. A. Mueller. 1997. Cognitive Design Elements to Support the Construction of a Mental Model during Software Visualization. In Proceedings of the 5th International Workshop on Program Comprehension (WPC '97).IEEE Computer Society, Washington, DC, USA, 17-.

[5] Wang Kechao; Wang Tiantian; Su Xiaohong; Ma Peijun. "Overview of Program Comprehension", Computer Science and Electronics Engineering (ICCSEE), 2012 International Conference on, vol.1, no., pp.601,605,23-25 March 2012. doi: 10.1109/ICCSEE.2012.285.

[6] Walenstein, A., "Theory-based analysis of cognitive support in software comprehension tools", Program Comprehension, 2002. Proceedings. 10th International Workshop on, vol., no., pp.75,84,2002.

[7] Hernán Bernardis. Instrumentación de Programas Escritos en Java para Interconectar los Dominios del Problema y del Programa. 40º Jornadas Argentinas de Informática e Investigación Operativa (JAIIO). 2011.

[8] Berón Mario; Henriques, Pedro; Pereira, Maria João; Uzal, Roberto. Instrumentación de programas escritos en C para interrelacionar las vistas comportamental y operacional de los sistemas de software. In XV CACIC'09-Argentine Congress on Computer Science. S. S. de Jujuy. 2009.

[9] ANTLR, ANother Tool for Language Recognition, http://www.antlr.org/.