Universidade Federal de Pernambuco
Graduação em Ciência da Computação
Centro de Informática

Proposta de Trabalho de Graduação

Tipos para uma Linguagem de Transformação

Aluno: Alexandra Barreto Assad de Barros (abab@cin.ufpe.br)
Orientador: Paulo Henrique Monteiro Borba (phmb@cin.ufpe.br)

Recife, 1 de Junho de 2004.

1 Contexto

Transformação de programas é uma técnica poderosa no suporte a atividades de engenharia de software como: refactoring [7, 10], desenvolvimento formal de software [1, 9], geração de código [6] e tradução de linguagens [6]. De fato, as aplicações de transformações de programas possuem grande importância, mas seu uso em projetos reais, de grande escala, não é possível sem automação. O suporte de ferramentas é vital para a aplicação de transformações de programas, pois aumenta a produtividade e diminui a chance de introduzir erros, o que ocorre muitas vezes quando se realiza uma tarefa tediosa e cansativa.

A fim de implementar transformações de programas, precisamos escolher uma maneira de representar a linguagem objeto em uma meta-linguagem [4]. Por linguagem objeto entende-se a linguagem utilizada para construir os programas onde serão aplicadas as transformações. A meta-linguagem, por sua vez, é usada para implementar essas transformações.

A maioria das meta-linguagens escolhidas não possuem um sistema de tipos, ou seja, a informação sobre tipo em um programa objeto não pode ser refletida no tipo de sua representação em uma meta-linguagem. Sendo assim, apesar da meta-programação ser uma abordagem poderosa para transformação de programas, ela é notoriamente passível a erros, já que a maioria das meta-linguagens não possui um sistema de tipos. Além do mais, elas não oferecem nenhuma assistência para capturar erros de tipos em tempo de compilação [5].

Várias ferramentas para transformação de programas já foram implementadas. Castor e Borba [2] definiram JaTS - uma linguagem para especificação de transformações de programas Java, que foi posteriormente implementada por um sistema de mesmo nome [3]. Esse sistema, específico para a linguagem Java, traz alguns benefícios em relação a outros sistemas de transformações de programas, pelo fato de definir uma linguagem para especificar transformações que é um superconjunto da linguagem Java. Isto diminui o gap semântico entre a linguagem objeto e a meta-linguagem. Outra vantagem é o fato de que JaTS leva a semântica de Java em consideração, o que torna possível especificar transformações muito mais complexas do que se somente a sintaxe fosse levada em consideração. A existência dessa linguagem e de um sistema capaz de dar suporte a ela também é uma vantagem em relação a outros sistemas, pois o programador não fica limitado a um conjunto pré-definido de transformações.

Apesar das vantagens apresentadas pela linguagem JaTS, ela ainda não possui um sistema de tipos bem definido, o que torna possível a implementação de transformações com erros de tipos que só podem ser detec-

tados em tempo de execução. A situação é agravada pelo fato de erros em meta-programas geralmente serem mais difíceis de detectar e corrigir do que erros em programas convencionais. Conseqüentemente, faz-se necessária a definição de um sistema de tipos para JaTS, a fim de permitir que erros de tipos em meta-programas sejam detectados em tempo de compilação.

2 Objetivos

Diante do cenário apresentado, o objetivo do trabalho é definir um sistema de tipos para JaTS. Na implementação atual da linguagem, a noção de tipos já existe, mas foi definida apenas de maneira implícita e informal. É necessário especificar o sistema de tipos de JaTS com precisão e garantir que a implementação da linguagem está em conformidade com o sistema de tipos definido. Para modelar formalmente esse sistema, será usado um cálculo minimal para Java [8].

O produto final deste trabalho será composto por duas partes: um documento descrevendo o sistema de tipos da linguagem e a extensão da implementação de JaTS para suportar esses sistema de tipos e localizar erros em tempo de compilação.

3 Cronograma

	Mês															
Atividade	*Maio*				*Junho*				*Julho*				*Agosto*			
Estudo de um cálculo minimal para os elementos principais do sistema de tipos de Java.	*															
Estudo de transformações de programas tipadas.	*	*	*													
Estudo de prova de propriedades em transformação de programas.			*	*												
Definição do sistema de tipos do JaTS.					*	*	*	*	*	*	*	*				
Manutenção do type checker para Java.	*	*	*	*	*	*	*	*								
Implementação do type checker para o JaTS.					*	*	*	*	*	*	*	*	*	*	*	*
Escrita do relatório final.					*	*	*	*	*	*	*	*	*	*	*	*
Elaboração da apresentação oral.															*	*

Tabela 1: Cronograma

Referências

[1] P. Borba and A. Sampaio. *Basic Laws of ROOL: an object-oriented language*. III Workshop de Métodos Formais, pages 33–44. João Pessoa - PB, Brasil, 2000.

[2] F. Castor and P. Borba. *A language for specifying Java transformations*. V Simpósio Brasileiro de Linguagens de Programação, pages 236–251. Curitiba - PR, Brasil, 2001.

[3] F. Castor, K. Oliveira, A. Souza, G. Santos, and P. Borba. *JaTS: A Java transformation system*. XV Simpósio Brasileiro de Engenharia de Software, pages 374–379. Rio de Janeiro - RJ, Brasil, 2001.

[4] C. Chen and H. Xi. *Implementing typeful program transformations*. Proceedings of the ACM SIGPLAN 2003 Workshop on Partial evaluation and semantics-based program manipulation, pages 20–28. San Diego - California, USA, 2003.

[5] C. Chen and H. Xi. *Meta-programming through typeful code representation*. Proceedings of the Eighth ACM SIGPLAN International Conference on Functional Programming, pages 275–286. Uppsala, Suécia, 2003.

[6] M. F. Felix and E. H. Hausler. *LET: Uma Linguagem para Especificar Transformações*. III Simpósio Brasileiro de Linguagens de Programação, pages 109–121. Porto Alegre - RS, Brasil, 1999.

[7] M. Fowler. *Refactoring: Improving the Design of Existing Code*. Object Technology Series. Addison Wesley, 2000.

[8] A. Igarashi, B. Pierce, and P. Wadler. *Featherweight Java: A minimal core calculus for Java and GJ*. ACM 1999 Symposium on Object-Oriented Programming, Systems, Languages and Applications, pages 132–146. Denver - Colorado, USA, 1999.

[9] C. Morgan. *Programming from specifications*. International Series in Computer Science. Prentice-Hall International, 1990.

[10] W. F. Opdyke. *Refactoring Object-Oriented Frameworks*. PhD thesis, University of Illinois at Urbana-Champaign, Dept. of Computer Science, 1992.

Datas e Assinaturas

Recife, 1 de Junho de 2004

Paulo Henrique Monteiro Borba
(Orientador)

Alexandra Barreto Assad de Barros
(Proponente)