OCONGRA - Ferramenta para geração de Grafos de Controle de Fluxo de Objetos

Paulo Roberto de A. F. Nunes, Ana C. V. de Melo
Universidade de São Paulo
Instituto de Matemática e Estatística
Departamento de Ciência da Computação
prnunes@linux.ime.usp.br, acvm@ime.usp.br

9 de maio de 2004

Resumo

Existem diversas ferramentas que auxiliam a realização de testes em programas orientados a objetos. No entanto exigem a aquisição de licenças de uso. Neste artigo é apresentada uma ferramenta de código aberto que utiliza o critério de análise de fluxo de objetos. para construção de um grafo de controle de fluxo com o objetivo de facilitar a análise e geração de testes.

Abstract

Many testing tools were developed to help us to get good results in testing oriented objects programs. However, we need to get a non-free license of use. This paper presents a open source tool that build an Object Control Flow Graph to support system analyses and generating testing cases.

1 Introdução

O teste de programas é uma fase complexa no desenvolvimento de software. Em sistemas complexos, os quais envolvem diversas atividades de produção, a probabilidade de encontrarmos falhas na programação é muito alta [4]. Para diferentes paradigmas de desenvolvimento com diferentes características, são necessárias técnicas de testes que abordem essas características para que erros sejam descobertos.

O paradigma de Orientação a Objetos (OO) apresenta diferenças em relação ao de programação procedural. Requisitos de teste aplicáveis a sistemas procedurais tornam-se inadequados para os testes de sistemas Orientados a Objetos - os requisitos mudam de acordo com o paradigma. Dessa forma, alguns critérios de teste existentes para o paradigma prodedural foram adaptados para o OO. Existem atualmente diversas técnicas para testes OO, como por exemplo, baseado em estado [2], teste incremental [3], todos os caminhos [1], baseado no fluxo de objetos [5], etc.

A análise de fluxo de objetos segue uma estratégia de validação que se baseia na cobertura total de estados de transformação de cada objeto do sistema. Tais transformações são denotadas pela redefinição de valores para os atributos dos objetos através da passagem de mensagens entre eles. Para monitorar o comportamento de cada um dos objetos são analisados todos os pares de definição e uso, ou seja, os locais onde cada objeto é definido ou usado. Um objeto é definido quando o construtor do objeto é invocado, um atributo é definido (tem seu valor alterado) ou um método que inicializa ou modifica o(s) atributo(s) é invocado. Um objeto é usado quando

um de seus atributos tem seu valor utilizado, um método que utiliza o valor de um de seus atributos é invocado ou o objeto é passado como parâmetro.

Em [5], Chen & Kao definem 5 passos para a construção de um grafo de controle de fluxo de objetos (GCFO) - que consiste basicamente na ligação entre as classes do sistema e seus métodos - e os pares de definição-uso (*du*): criar o conjunto de definição e uso; construir o GCFO inicial (somente nós com classes e métodos); adicionar ao grafo as arestas intra-métodos; adicionar ao grafo as arestas inter-métodos; selecionar os pares *du*.

A cobertura de todos os pares *du* aumenta a probabilidade de se encontrar falhas no programa. Com o grafo criado torna-se muito mais fácil a visualização do sistema e também a identificação dos pares *du*.

A comercialização de ferramentas desenvolvidas para testes muitas vezes inviabiliza a aquisição das mesmas fazendo com que possivelmente haja perda de qualidade no produto final. Diante disso surge a necessidade de uma ferramenta com código aberto que auxilie os testes orientados a objetos. A OConGra (Object Control Graph): *uma ferramenta para geração de grafos de controle de fluxo de objetos.* é um software livre que aplica a técinica sugerida em [5] que gera um grafo e possibilita a elaboração de casos de testes mais precisos.

A Seção 2 apresenta as características principais da ferramenta, onde são abordados os passos executados pela ferramenta para geração do grafo. A Seção 3 detalha um exemplo de funcionamento da ferramenta, onde é lido um sistema e em seguida apresentado o resultado obtido. Finalmente, a Seção 4 conclui apresentando outras utilizações para a ferramenta e propostas de trabalhos futuros.

2 Características da Ferramenta

A OConGra recebe como entrada um sistema (escrito em Java) e tem como saída o grafo de controle de seus objetos. O usuário, através de interface gráfica, designa o local onde estão localizados os arquivos fonte do sistema. Após a análise de cada classe do sistema, a ferramenta produz o grafo e o conjunto de pares *du*.

2.1 Geração do Grafo

A construção do GCFO de um dado sistema obedece aos mesmos passos sugeridos em [5]. Dessa forma, a ferramenta foi desenvolvida com base nesses passos.

A seguir serão descritas as fases executadas pela OConGra para a geração do GCFO e o conjunto de pares de definição e uso.

1. Leitura e Identificação do Projeto: o usuário especifica o local do projeto. O software então lê as classes e organiza internamente as informações;

2. Construção do *OCFG* com as classes-mãe: baseado nos resultados da fase anterior, será construído o grafo contendo a relação entre cada classe do sistema lido na entrada. Assim, essas classes terão suas chamadas de métodos denotadas;

3. Identificação das definições-usos através do *OCFG*: com o grafo definido, as mudanças de estado dos objetos poderão ser analisadas com maior precisão. Analisando o grafo, o software identificará cada definição-uso utilizada pelas classes do sistema;

4. Atualização do *OCFG* baseada nas definições-usos: a identificação das definições e usos (passo anterior) possibilita a implementação de alterações no grafo obtido no passo 2, criando assim um novo grafo;

5. Geração gráfica dos resultados obtidos: o grafo encontrado após as análises do sistema é desenhado e exibido de forma gráfica ao usuário.

2.2 Arquitetura da Ferramenta

O sistema está sendo desenvolvido em Java com a utilização da ferramenta Eclipse [6]. Possui aproximandamente 3500 linhas de código. A arquitetura do sistema é resumida em três partes: Principal, Estrutura Java e Geração de Grafos.

A parte *Principal* é responsável pela interface com o usuário. Ela lê os sistemas (através da parte responsável pela estrutura Java) e exibe como resultado o GCFO (através da parte responsável pela geração de grafos). Na parte *Estrutura Java* é feita a leitura de todos os arquivos do sistema (designado pelo usuário) e organizado de forma a facilitar a geração do GCFO. E finalmente, a *Geração de Grafos* recebe a estrutura criada e constrói o grafo final.

3 Um Exemplo de Uso

Para exemplificar o funcionamento da ferramenta foi criada uma aplicação simples que instancia uma classe para conversão de unidades de temperaturas:
kelvin <-> fahrenheit, celsius <-> kelvin, fahrenheit <-> celsius.

A classe principal contém um objeto `t` *Temperatura* - como pode ser visto na Figura 1 - que é utilizado para conversões e para exibir os valores de unidades específicas.

Figura 1: Um programa Java para conversão de unidades de temperatura.

Inicialmente, após ter informado a localização do projeto, o software exibe a estrutura dos arquivos lidos: nome das classes e suas heranças, seus respectivos métodos e quais números de linhas eles aparecem. A Figura 2 mostra as definições e os usos presentes no método `main` da classe *App*, conforme enunciado anteriormente. Cada linha que possui um objeto é verificada a ocorrência de uma definição ou de um uso.

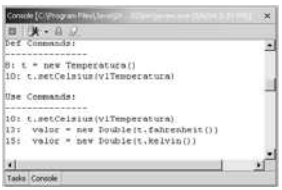

Figura 2: Definições e Usos do método `main` da classe *App*.

Já a Figura 3 mostra o grafo correspondente ao exemplo. Após identificar a estrutura do sistema o programa constrói esta saída gráfica.

Após identificar as definições-uso e o grafo, é necessário exercitar os pares pelo menos uma vez. Para isso são criados os seguintes testes: (0,0) - exercita o par (10, 13) da classe *App* - e (1,0) - exercita o par (10,15) da classe *App*. Esses dois testes exercitam todas as intanciações do objeto `t`.

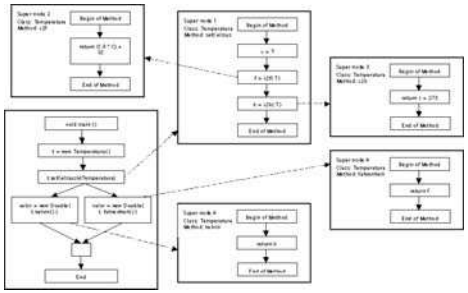

Figura 3: Grafo de controle de fluxo da aplicação.

4 Conclusões

Esse artigo apresentou a ferramenta OConGra utilizada na geração de grafos de controle de fluxo de objetos.

Além da utilização principal, a ferramenta nos auxilia na geração de bons dados para testes, pois garantem a cobertura total dos casos de definição e uso de objetos. Com isso, caso haja um erro, a probabilidade de encontrarmos se torna maior, pois os objetos serão exercitados ao menos uma vez (segundo a técnica [5]).

A OConGra pode ser utilizada no auxílio educacional em matérias de Engenharia de Software na geração gráfica de grafos.

Finalmente, por ser desenvolvida em Java, há a possibilidade de um projeto para adicioná-la ao Eclipse como um plug-in, o que será de grande utilidade para desenvolvedores, pois ao gerar os ".class"poderão também gerar os grafos e também o conjunto de definição e uso.

Agradecimentos: Ao CNPq pelo apoio financeiro e a todos aqueles que participam deste projeto.

Referências

[1] CHAIM, M. L.. *Depuração de Programas Baseada em Informação de Teste* Estrutural. PhD thesis, Faculdade de Engenharia Elétrica e de Computação da Universidade de Campinas, Novembro/2001.

[2] PINTO, I. M. and PRICE, A. M. A.. Um sistema de apoio ao teste de programas orientados a objetos com uma abordagem reflexiva. In *XII Simpósio Brasileiro de Engenharia de Software*, pages 87-102, 1998.

[3] HARROLD, M. J., MCGREGOR, J. D. and FITZPATRICK, K. J.. Incremental testing of object-oriented class structures. In *international Conference on Software Engineering*, pages 68-80. IEEE Computer Society Press, 1992.

[4] PRESSMAN, R. S.. *Software Engineering*. McGraw-Hill, 2001.

[5] CHEN, M. and KAO, H. M. Testing Object-Oriented Programs - An integral aproach. In . *Proccedings of the Tenth International Symposium on Software Reliability Engineering*. IEEE Computer Society Press, 1999.

[6] PROJETO ECLIPSE. Disponível no site Eclipse, URL: http://www.eclipse.org. Consultado em 15/08/2003.