Refatorações para Melhoria da Legibilidade de Código Fortran

Dionatan K. Tietzmann[1], Gustavo Rissetti[1], Andrea S. Charão[1], Eduardo K. Piveta[1]
Adriano Petry[2], Jonas R. de Souza[2]

[1] Universidade Federal de Santa Maria (UFSM)
[2]Instituto Nacional de Pesquisas Espaciais (INPE)
{dionatan,rissetti,andrea,piveta}@inf.ufsm.br
dr.adriano.petry@gmail.com, jonas@dae.inpe.br

Abstract. *Refactoring is a software engineering technique that seeks to improve the design of existing software through source code transformations. This technique has been developed extensively for object-oriented languages, but is still underused in procedural languages like Fortran. In this paper, we present four refactorings implemented as extensions to the Eclipse IDE, aimed at improving the readability of Fortran code. We also present an application of these refactorings in a Fortran code used in the National Institute for Space Research (INPE).*

Resumo. *Refatoração é uma técnica de engenharia de software que efetua transformações em código-fonte a fim de melhorá-lo. Essa técnica se desenvolveu amplamente para linguagens orientadas a objetos, mas é ainda pouco explorada em linguagens procedurais como Fortran. Neste artigo, apresenta-se quatro refatorações implementadas como extensões ao IDE Eclipse, voltadas para melhoria da legibilidade de código Fortran. Apresenta-se também uma aplicação destas refatorações em um código Fortran utilizado no Instituto Nacional de Pesquisas Espaciais (INPE).*

1. Introdução

Refatoração é uma técnica presente em diferentes etapas do ciclo de vida de uma aplicação [Fowler et al. 1999, Opdyke 1992] e vem sendo amplamente utilizada nas linguagens orientadas a objetos mais recentes, como Java, por exemplo. O uso dessa técnica visa melhorar aspectos relacionados à estrutura interna de componentes de uma aplicação sem comprometer seu aspecto funcional, permitindo evoluir os códigos das aplicações.

A evolução é uma propriedade natural em processos de desenvolvimento de software. Durante o ciclo de vida de um sistema, geralmente existe a necessidade de evolução, seja para a adição de um novo requisito, ou para a alteração de funcionalidades existentes. Muitas vezes esse processo de evolução é executado manualmente, sem a existência de uma regra a seguir (como um conjunto de passos bem definidos, por exemplo), e isso acarreta na degradação da qualidade do software.

As refatorações podem ser usadas como auxílio para a evolução de software, permitindo que o programador siga instruções bem definidas para modificar trechos de código. O processo de refatoração, além de poder ser aplicado manualmente, pode ser feito com ferramentas que automatizam as ações de refatoração, facilitando a programação e reduzindo o risco de erros decorrentes das modificações de código.

O conceito de refatoração está fortemente vinculado ao paradigma da orientação a objetos e está presente de forma automatizada em diversas ferramentas alinhadas com este paradigma. Na computação científica, a refatoração de código é uma técnica ainda pouco explorada, principalmente pelo fato de boa parte do código legado de tais aplicações estar escrito em linguagens procedurais [Johnson et al. 2006] como Fortran, que é uma linguagem de programação com mais de cinquenta anos de existência, e que ainda hoje é amplamente utilizada em aplicações científicas.

Este artigo descreve a automação de um conjunto de quatro refatorações para código em Fortran, implementadas como extensões ao *framework* Photran do Eclipse, que é um popular ambiente integrado de desenvolvimento (*Integrated Development Environment* - IDE). Essa automação visa reduzir a lacuna existente entre a grande quantidade de código legado escrito em Fortran e o reduzido número de IDEs voltadas para essa linguagem, distribuídas como software livre.

O artigo está organizado como segue. A seção 2 apresenta os recursos e as características do Eclipse e do *plugin* Photran. A seção 3 descreve as refatorações implementadas e agregadas ao Photran, com uma contextualização e detalhes de implementação de cada uma delas. A seção 4 descreve os resultados da aplicação das refatorações implementadas em um aplicativo de software usado pelo Instituto Nacional de Pesquisas Espaciais (INPE). Por fim, a seção 5 apresenta as considerações finais.

2. Eclipse e Photran

O Eclipse é um IDE com suporte para diversas linguagens de programação, tais como Java, C e C++. Recentemente, por meio do *plugin* Photran, o Eclipse passou também a suportar a linguagem Fortran, em suas versões 77, 90, 95, 2003 e 2008. Arquiteturalmente, pode-se ver o *framework* do Eclipse como sendo uma aplicação que disponibiliza uma interface gráfica padrão (com menus, barras de ferramentas, editores, etc.), na qual as funcionalidades são implementadas por meio de *plugins*.

O Photran [Eclipse.org 2011] é um *plugin* para o Eclipse que estende as funcionalidades do IDE para suportar o desenvolvimento de código Fortran. Atualmente, seu desenvolvimento é mantido como um projeto oficial da Eclipse Foundation e faz parte de um macro projeto denominado PTP (*Parallel Tools Platform*). Um dos principais objetivos do projeto Photran é disponibilizar um *framework* que possibilite o desenvolvimento rápido de ações de refatoração para código Fortran, reutilizando a infra-estrutura fornecida pelo Eclipse [De 2004]. Para isso, são utilizadas reescritas de árvores sintáticas abstratas (ASTs), as quais são manipuladas através da adição, da modificação e da remoção de nós e informações desses nós em suas estruturas.

3. Refatorações Desenvolvidas

Para facilitar o processo de automatização de refatorações, é comum que esteja disponível um analisador sintático específico para a linguagem de programação a ser refatorada. O processo de análise sintática utiliza algoritmos para a decomposição de sentenças de um código-fonte a partir de uma gramática formal e constrói uma árvore gramatical (AST, *Abstract Syntax Tree*) [Aho et al. 2006].

Além da AST, o *plugin* Photran disponibiliza uma estrutura denominada VPG (*Virtual Program Graph*), que agrega às ASTs algumas ligações adicionais entre seus

nós. Essas ligações podem representar, por exemplo, um vínculo entre a utilização de determinada variável (em uma expressão) e sua respectiva declaração, ou ainda um comando e seu escopo (bloco lógico de atuação) [Overbey 2007]. Através de uma VPG é possível obter informações para auxiliar na manipulação dos nós de uma AST.

Neste trabalho, foram automatizadas algumas refatorações aplicadas a características da linguagem Fortran para melhorar a **legibilidade de código**, utilizando-se o *framework* Photran. Em trabalhos anteriores, outras refatorações também foram implementadas para a mesma ferramenta [Boniati et al. 2010], com objetivos de melhorar desempenho das aplicações e melhorar atributos de qualidade do código-fonte das mesmas. As técnicas selecionadas neste trabalho objetivam a evolução da linguagem e organização de código. A seguir, são detalhadas as técnicas de refatoração automatizadas.

3.1. Introduzir Árvore de Chamadas

Chamadas a sub-rotinas são frequentes no código-fonte de programas em Fortran. Quando surge a necessidade de fazer alguma alteração na aplicação, o fato de se ter diversas chamadas de sub-rotinas pode causar dúvidas e complicar o entendimento da lógica do programa. A introdução da árvore de chamadas nos comentários de cada sub-rotina permite saber antecipadamente quais são as invocações efetuadas no seu escopo. A Figura 1 mostra a aplicação dessa refatoração no Photran.

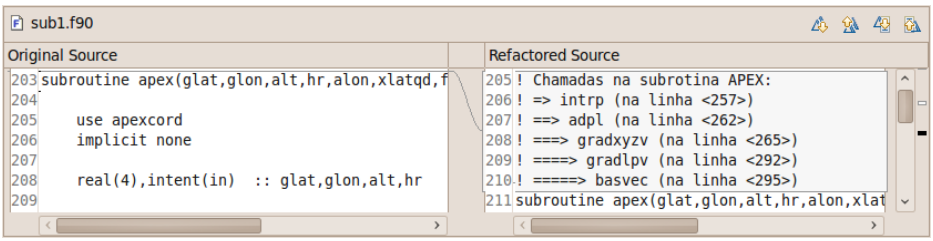

Figura 1. Aplicação da refatoração *Introduce Call Tree* no Photran

A mecânica desta refatoração consiste em percorrer a AST em busca de chamadas a sub-rotinas. Quando uma chamada é encontrada, é verificado o nome da sub-rotina chamada e o número da linha onde a mesma ocorre. Os dados obtidos são inseridos na forma de um comentário Fortran no início de cada escopo do código. As árvores de chamadas são adicionadas em todos os escopos da aplicação onde são usadas sub-rotinas.

3.2. Aviso sobre Variáveis Globais Modificadas

Muitos programas em Fortran utilizam um módulo com variáveis globais que são acessadas por sub-rotinas, sendo uma técnica frequente em versões mais antigas de Fortran. O problema ao usar variáveis globais surge quando um código precisa ser modificado, pois torna-se difícil saber quais variáveis são locais e quais variáveis são globais. Com isso, pode-se cometer enganos ao alterar o valor de uma variável global, introduzindo erros no programa. Com a introdução de avisos no código-fonte sobre alterações em variáveis globais, o usuário visualiza antecipadamente quais e onde estão sendo manipuladas as variáveis globais do código-fonte, facilitando sua manutenção e entendimento. A Figura 2 mostra a aplicação dessa refatoração no Photran.

A mecânica de implementação dessa refatoração consiste em primeiramente percorrer a AST do código em busca de definição de módulos do usuário. O nome do módulo

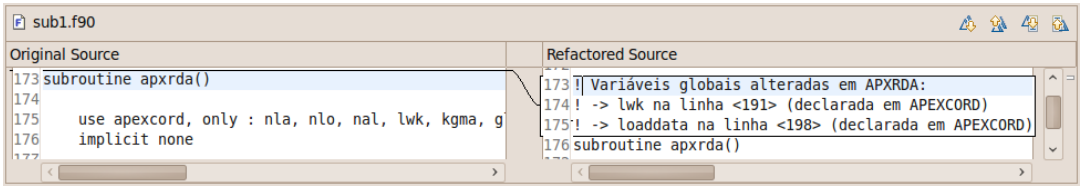

Figura 2. Aplicação da refatoração *Global Variable Changed Warning* no Photran

e todas as definições de variáveis contidas nele são armazenados em uma estrutura do tipo *hash* para consultas posteriores. Após isso, a AST é percorrida em busca de nós que fazem alteração no valor de variáveis. Quando achados, são verificados para constatar se eles são nós que alteram variáveis locais ou globais. Isso é feito consultando cada nome de variável contido em uma lista de variáveis dos módulos utilizados pelo programador. Caso sejam realmente alterações em variáveis globais, um comentário é adicionado no topo do escopo em que a alteração ocorre, informando o nome da variável global alterada, assim como a linha onde a modificação é feita. O programador é informado também sobre a origem da variável (onde ela foi inicialmente declarada).

3.3. Adicionar Cláusula *ONLY* em Comandos *USE*

É comum encontrarmos módulos de código Fortran contendo declarações de variáveis que são usadas dentro de outras sub-rotinas. Normalmente, usam-se módulos para poder ter acesso a variáveis globais, como visto na seção 3.2. Porém, nem sempre que se declara o uso de um módulo (*USE module*) são usadas todas as variáveis nele contidas. Para ficar claro ao programador quais variáveis são realmente usadas, pode-se usar a cláusula *ONLY* seguida de uma lista de variáveis do módulo que são realmente usadas dentro da sub-rotina em questão (*USE module, ONLY : var1, var2, ...*). Com essa refatoração, todos os *USE* são substituídos por *USE ONLY*, explicitando as variáveis usadas e assim facilitando o entendimento do código. A Figura 3 mostra a aplicação dessa refatoração no Photran.

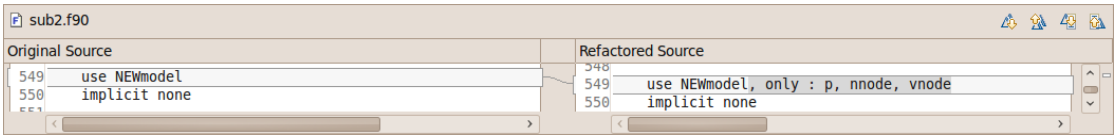

Figura 3. Aplicação da refatoração *Add Use ONLY Statement* no Photran

A mecânica de implementação dessa refatoração consiste em primeiramente percorrer a AST do código em busca de definição de módulos. Os nomes dos módulos encontrados, assim como suas variáveis, são armazenados em uma estrutura *hash* para serem acessados posteriormente. Após isso, a AST é percorrida em busca de nós que representam uma referência a variáveis de cada módulo e em seguida, em busca de nós que representam modificações em cada variável do módulo. Após obtidas todas as variáveis usadas no módulo, é localizado o nó da AST que contém o comando *USE*. Caso a lista de *onlys* deste nó esteja vazia, são adicionadas na lista as variáveis realmente usadas no escopo onde o *USE* é declarado. Caso nenhuma variável do módulo seja usada, e mesmo assim exista uma declaração *USE* para o módulo, o nó que contém tal declaração é removido da AST, pois é desnecessário. No final, obtém-se um código contendo apenas *USE ONLY*, clarificando quais variáveis dos módulos são usadas em cada escopo.

3.4. Substituir Estilos Antigos de Laços *DO*

A linguagem Fortran passou por várias evoluções ao longo do tempo alterando a sintaxe de seus comandos. Dentre as construções que sofreram alterações está o laço de repetição *DO*. A substituição de estilos antigos de laços *DO*, visa modificar tais construções atualizando-as a estilos mais recentes da linguagem, substituindo a construção *DO/CONTINUE*, que utiliza referência a um rótulo[1] para indicar o final laço, pela contrução *DO/END DO*, onde o final do laço é indicado pela instrução *END DO*. A Figura 4 mostra a aplicação dessa refatoração no Photran.

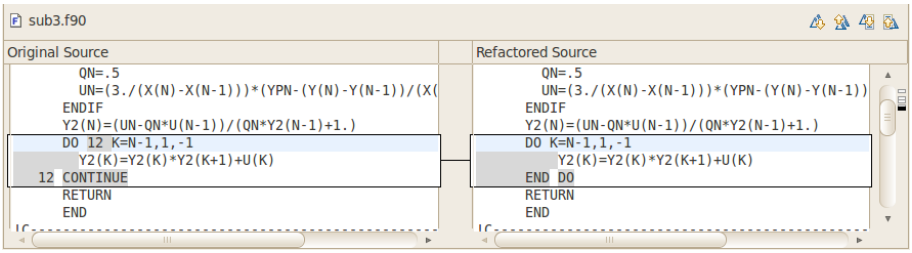

Figura 4. Aplicação da refatoração *Replace Old-Style Do Loops* no Photran

A mecânica desta refatoração consiste em percorrer a árvore sintática em busca de nós que representem laços de repetição no estilo *DO/CONTINUE*, removendo a referência ao rótulo do final do laço e substituindo a instrução *CONTINUE* por *END DO* no final do laço. Ao final, a AST é percorrida novamente verificando se o rótulo é referenciado em outro local. Se não houver referências para o rótulo, ele é removido da AST.

4. Aplicação das Refatorações Implementadas

No Photran, as refatorações implementadas foram nomeadas como *Introduce Call Tree*, *Global Variable Changed Warning*, *Add Use ONLY Statement* e *Replace Old-Style Do Loops*, respectivamente. Para testá-las, elas foram aplicadas no código SU-PIM [Bailey 1978], usado em pesquisas [de Souza 1997] no Instituto Nacional de Pesquisas Espaciais (INPE). O código é uma aplicação escrita em Fortran 90, com construções de Fortran 77, usado em simulações na área de geofísica espacial. A aplicação foi disponibilizada para os testes em um arquivo único, contendo mais de 11 mil linhas de código. Para testar as refatorações implementadas, o código teve de ser particionado em oito arquivos, pois no modo de desenvolvimento do Photran não é suportado um arquivo tão grande, não havendo espaço suficiente de memória para manipulá-lo.

Com a aplicação da refatoração *Introduce Call Tree* foram identificadas 104 chamadas para sub-rotinas em todo o código. Foram adicionadas informações sobre as chamadas em todos os escopos onde as mesmas ocorreram, informando o nome da sub-rotina usada e a linha onde o uso ocorreu. Já, com *Add Use ONLY Statement*, as cláusulas *ONLY* foram adicionadas em 11 regiões do código onde módulos eram usados, clarificando quais as variáveis eram realmente utilizadas em cada escopo.

Com a aplicação de *Global Variable Changed Warning* foram encontradas 51 alterações de variáveis globais. As informações sobre as alterações das mesmas foram adicionadas no início de cada escopo onde as variáveis foram alteradas.

[1] identificador numérico usado para identificar e referenciar linhas específicas do código

Com a utilização da refatoração *Replace Old-Style Do Loops* foram modificados 272 laços de repetição em todo o código. Essas alterações modificaram o estilo de construção dos laços de repetição *DO/CONTINUE* para o estilo *DO/ENDDO*.

5. Considerações Finais

As técnicas de refatoração têm por objetivo amenizar a degradação natural pela qual o código fonte de aplicações sofre com o tempo. O processo de refatoração pode ser executado manualmente, mas o ganho de qualidade em escala se dá quando as técnicas são automatizadas e integradas a ferramentas para desenvolvimento de software (IDEs).

A refatoração de código Fortran é necessária para ajudar a manter os códigos legados existentes. Com o uso de software livre, como o Photran, é possível que pesquisas independentes possam agregar novas funcionalidades às ferramentas. Este trabalho explorou a utilização de refatorações para essa linguagem objetivando melhorar as construções usadas nos códigos legados, através de mudanças sintáticas que adaptam o código-fonte a novos padrões de programação da linguagem. As refatorações propostas foram automatizadas e integradas à ferramenta Photran, para serem disponibilizadas para seus usuários de forma gratuita. Os testes realizados com as refatorações na aplicação SUPIM se mostraram satisfatórios, resultando em um código melhor estruturado e de melhor legibilidade.

Referências

Aho, A. V., Lam, M. S., Sethi, R., and Ullman, J. D. (2006). *Compilers: Principles, Techniques, and Tools*. Addison Wesley, 2 edition.

Bailey, G. (1978). SUPIM - Sheffield University Plasmasphere Ionosphere Model. Disponível em: `http://gbailey.staff.shef.ac.uk/supim.html`. Acesso em: abril de 2011.

Boniati, B. B., Rissetti, G., Charão, A. S., and Piveta, E. K. (2010). Extensões para Refatoração de Código Fortran no Eclipse. In *11º Fórum Internacional de Software Livre – XI Workshop Sobre Software Livre*, pages 74–79, Porto Alegre, Brasil. Consórcio Editorial Livre.

De, V. (2004). *A Foundation for Refactoring Fortran 90 in Eclipse*. Dissertação de mestrado, University of Illinois, Urbana-Champaign, EUA.

de Souza, J. R. (1997). *Modelagem Ionosférica em baixas latitudes no Brasil*. Tese de doutorado, Instituto Nacional de Pesquisas Espaciais.

Eclipse.org (2011). Photran - An Integrated Development Environment for Fortran. Disponível em: `http://www.eclipse.org/photran/`. Acesso em: abril de 2011.

Fowler, M., Beck, K., Brant, J., Opdyke, W., and Roberts, D. (1999). *Refactoring: Improving the Design of Existing Code*. Addison Wesley.

Johnson, R., Foote, B., Overbey, J., and Xanthos, S. (2006). Changing the Face of High-Performance Fortran Code. A White Paper.

Opdyke, W. (1992). *Refactoring Object-Oriented Frameworks*. Tese de doutorado, University of Illinois, Urbana-Champaign, EUA.

Overbey, J. L. (2007). Virtual program graph. Disponível em: `http://jeff.over.bz/software/vpg/doc/`. Acesso em: abril de 2011.