FERUS: Um Ambiente de Desenvolvimento de Especificações CASL

Gleydson Lima, Anamaria Martins Moreira, David Déharbe,
David Pereira, Demostenes Sena, Jorgiano Vidal*

Departamento de Informática e Matemática Aplicada - DIMAp
Universidade Federal do Rio Grande do Norte -UFRN
Campus Universitário de Lagoa Nova
59072-970 Natal, RN, Brasil
email: cofirn@consiste.dimap.ufrn.br

Resumo

Este artigo apresenta FERUS (Apoio Formal à Especificação e Re-Utilização de Software), um ambiente desenvolvido no Departamento de Informática e Matemática Aplicada da UFRN em parceria com o LORIA (França), que permite a criação, manipulação e prototipação de especificações na linguagem CASL. Para otimizar as diferentes possibilidades de trabalho sobre as especificações, a ferramenta trabalha com diferentes formatos de representação: texto, formato de intercomunicação entre ferramentas e formato interno de representação de especificações, adequado para operações de transformação de especificação. O protótipo que apresentamos aqui opera através de uma interface gráfica, mas uma biblioteca de funções correspondentes às funcionalidades da ferramenta também será disponibilizada para facilitar sua integração em outros contextos de operação e sua interoperabilidade com outras ferramentas.

Abstract

This paper presents FERUS (Formal Suppport to the Specification and Reuse of Software), an environment under development at the Departamento de Informática e Matemática Aplicada *(DIMAp-UFRN-Brazil) in cooperation with the LORIA laboratory (Nancy-France). This tools supports the creation, transformation and prototyping of CASL language specifications. To optimize each kind of operation on specifications, the tool deals with different specification representation formats: CASL text, a tools intercommunication format and an internal format. The prototype that we present here operates through a graphical interface, but a library of functions corresponding to the tool's functionalities will also be made available to ease its integration into other operation contexts and its interoperability with other tools.*

1 Introdução

O uso de especificações formais no desenvolvimento de um software pode incrementar sua qualidade e até reduzir custos de algumas etapas do desenvolvimento. No entanto, técnicas que não sejam apoiadas por ferramentas são meramente descartadas, e com aplicação restrita à acadêmia. Nesse sentido, o desenvolvimento de ferramentas de suporte ao desenvolvimento de software e hardware apoiado por métodos formais tem sido foco de pesquisas e de grandes avanços nos últimos anos.

A ferramenta FERUS está sendo desenvolvida para apoiar desenvolvimento de software usando métodos formais, usando a linguagem CASL (*Common Algebraic Specification*

* Projeto financiado pelo CNPq (ProTeM-CC) e INRIA.

Language [2]) para especificar componentes de software. Ela fornece ao usuário um ambiente para o design e prototipação dessas especificações, permitindo ao usuário editar o texto de uma especificação, compilar e executar especificações (no caso de especificações executáveis, um subconjunto das especificações CASL), e manipular especificações através de operações de transformação controladas, como por exemplo a instanciação de uma especificação genérica (parametrizada).

CASL é uma linguagem criada e mantida pelo grupo CoFI (*The Common Framework Initiative*) com o propósito de especificar requisitos e design para softwares convencionais. A linguagem baseia-se no paradigma algébrico que, em outras palavras, refere-se a especificações axiomáticas (orientada a propriedades) de classes de modelos. Sua criação foi baseada em conceitos de linguagens algébricas já existentes, tentando assim uniformizar as notações utilizadas e criar uma linguagem que sirva como padrão.

Para otimizar as diferentes possibilidades de trabalho sobre as especificações, a ferramenta trabalha com diferentes formatos de representação: formato texto CASL, para uso e entendimento pelo usuário; formato de intercomunicação entre ferramentas sob a forma de termos e árvores de sintaxe abstrata (ATerms [7]) e o formato interno FERUS de representação de especificações sob a forma de grafos, adequado para operações de transformação de especificação.

Em seguida apresentamos em mais detalhes as funcionalidades de FERUS (seção 2.1), sua arquitetura (seção 2.2) e dados sobre a sua implementação (seções 2.3 e 2.4). Para concluir, apresentamos um levantamento da situação atual do desenvolvimento e uma previsão de evolução em um futuro próximo.

2 A ferramenta FERUS

FERUS fornece ao usuário um ambiente de criação, manipulação e prototipação de especificações na linguagem CASL. O desenvolvimento de uma especificação em FERUS pode ser feito através de editores de texto integrados ao ambiente, ou de operadores de transformação, incluindo a generalização proposta por A. Moreira em [4], que formam a particularidade do ambiente. Em ambos os casos, o usuário é apoiado por um analisador [5] que verifica a correção sintática e semântica da especificação criada; e por uma ferramenta de prototipagem [3] que permite executar uma especificação caso ela respeite algumas regras de executabilidade (sub-linguagem de CASL).

2.1 Funcionalidades

Edição-compilação-execução

- **Edição:** A ser feita no editor escolhido pelo usuário para ser utilizado no ambiente (atualmente disponíveis um editor do ambiente e emacs) ou em editor chamado externamente ao ambiente.
- **Compilação:** Através da compilação o usuário pode (1) se certificar da correção sintática e semântica de suas especificações e (2) preparar um componente de especificação para sua execução ou manipulação pelos operadores de transformação.
- **Prototipação (execução) de especificações:** A prototipação de especificações permite executar os componentes para observar o resultado de suas operações.

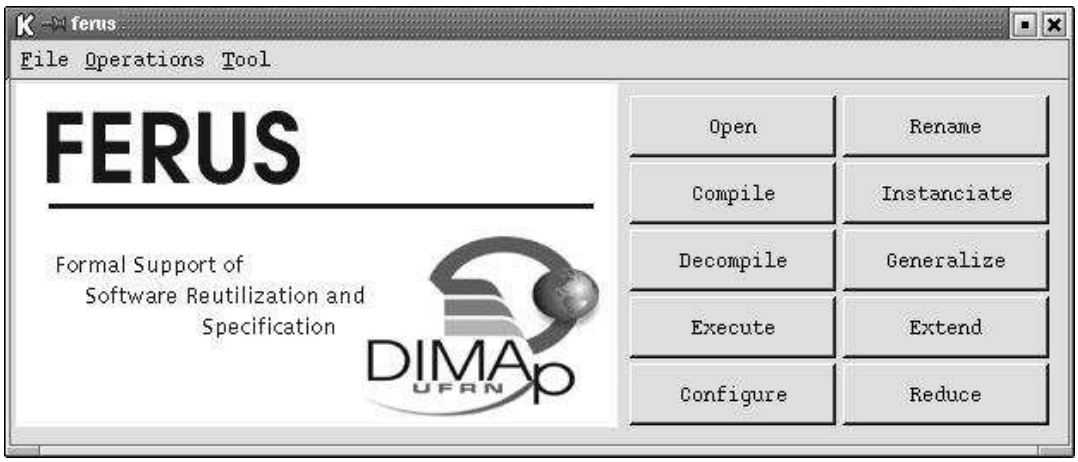

Figura1. Janela principal da ferramenta

Operações de transformação

- **Renomagem:** Permite que símbolos declarados em um componente sejam renomeados, de maneira a adaptarem-se a seus novos contextos de utilização.
- **Generalização:** A generalização de componentes possibilita torná-los mais genéricos ou seja, abstrai níveis de especialização, conservando parcialmente sua semântica. Assim, componentes que descrevem um modelo específico passam a descrever uma classe de modelos, o que resulta em componentes com um maior nível de abstração e maior potencial de reutilização.
- **Instanciação:** Permite que componentes genéricos sejam especializados para aplicações específicas, selecionando um modelo pertencente a classe de modelos especificada pelo componente genérico original.
- **Extensão:** Enriquece modelos através da declaração de novos símbolos (firmando suas propriedades) e/ou especializa a interpretação de símbolos já declarados.
- **Redução:** Esconde ou elimina determinados símbolos em uma especificação.

Adicionalmente, FERUS propõe uma operação de decompilação que realiza a operação inversa da compilação, traduzindo especificações do formato interno ferus para o formato texto. Ela acontece em geral após a execução de uma ou mais operações de transformação (executadas sobre o formato interno).

2.2 Arquitetura

A arquitetura proposta para a ferramenta foi desenvolvida com base nos resultados da fase de análise de requisitos. De posse dessas informações, foi proposta uma descrição arquitetural [6] modular com o objetivo de facilitar a manutenção, podendo ser modificada facilmente e/ou expandida, adequando-se a novas necessidades ou modificações desejadas.

- **Interface:** Módulo de interação entre FERUS e o usuário. Além da comunicação entre o usuário e os outros módulos, esse módulo apresenta funcionalidades de controle do ambiente.

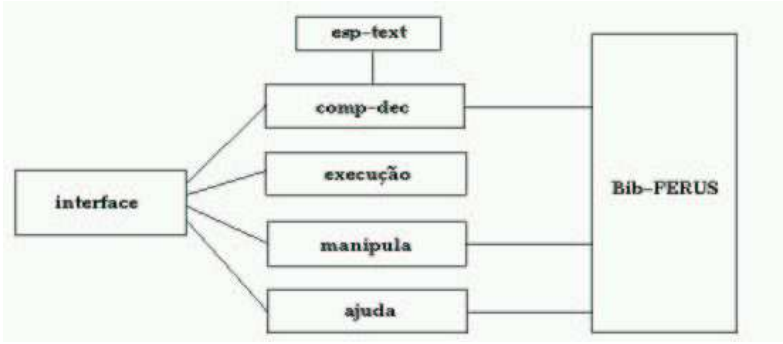

Figura2. Arquitetura da ferramenta

- **Compilação/Decompilação:** Responsável pelas operações de compilação e decompilação. No processo de compilação FERUS utiliza um analisador léxico-sintático e semântico CATS [5] que faz parte das ferramentas de apoio à linguagem CASL, gerando uma árvore sintática no formato intermediário ATerms [7]. O compilador da ferramenta FERUS realiza um casamento de padrão na árvore sintática no formato ATerm e traduz a especificação para o formato interno ferus.
- **Execução:** A execução é feita através de técnicas de reescritura e estratégias disponíveis em ELAN [1]. O sistema ELAN oferece um framework para a combinação de computação e paradigmas de dedução. Ele permite o desenvolvimento de provadores de teoremas, linguagens de programação lógicas, resolvedores de restrições (*constraint solvers*) e procedimentos de decisão. Para a execução de especificações, é o seu aspecto computacional que é utilizado.

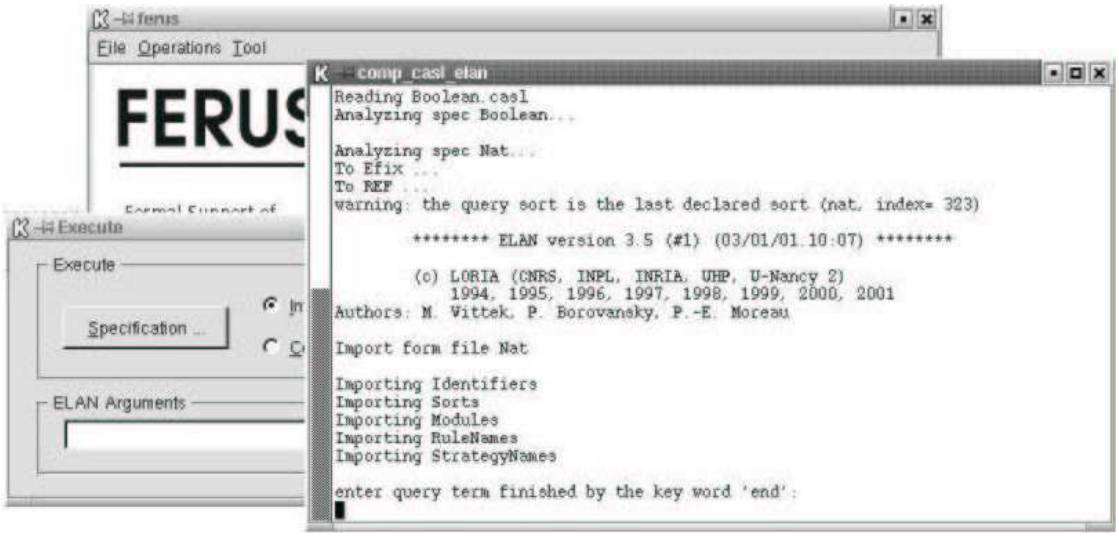

Figura3. Prototipação com o ELAN

- **Manipula:** Módulo que representa as operações de transformação de componentes.
- **Ajuda:** Guia o usuário no uso da operação de generalização, pois nem sempre é obvio para o usuário identificar o nível e as possibilidades de generalização de determinado

componente. Esse módulo deverá interagir com o sistema de reescritura ELAN, que fornecerá apoio sob a forma de provador de teoremas equacionais.
- **Exp-text e Bib-Ferus:** Exp-text representa os formatos de entrada CASL e ATerm que uma especificação pode assumir e Bib-Ferus corresponde à biblioteca de componentes de especificação formal proposta para a ferramenta.

2.3 Formatos Internos

Como ocorre em geral com formatos textuais destinados ao usuário, as especificações CASL em formato texto são de manipulação difícil e ineficiente pelo computador. A ferramenta usa então internamente dois formatos de representação de especificações: ATerms e formato interno ferus.

O formato ATerm é um formato genérico e tem como objetivo servir como meio de comunicação entre as ferramentas conectadas ao ambiente (CATS e ELAN) e FERUS. É através dos ATerms que FERUS recupera o resultado da análise léxico-sintática e semântica efetuada por CATS. O mesmo formato é também utilizado como ponto de partida pelo programa de execução de especificações CASL pelo sistema ELAN [3].

Esse formato é apoiado por uma biblioteca de manipulação centrada em duas operações básicas: o *making* e o *matching*. A operação de making permite a criação de um padrão através da passagem de alguns argumentos. Estes argumentos são a estrutura da árvore de termos e os dados que a compõem. A operação de matching é o inverso, através desta operação é possível realizar o casamento de padrão para obter as informações dos nós da árvore.

O formato interno ferus possui uma estrutura baseada em ponteiros, sendo ideal para a execução eficiente das operações de transformação. Este formato é apoiado pela biblioteca `libcasl` que define os tipos de nós da representação e as operações correspondentes de criação, alteração e supressão de nós.

2.4 Implementação

A implementação da ferramenta é feita em ambiente Linux, usando as linguagens C++ e C[1], a biblioteca para implementação de interfaces gráficas Qt 2.3.0 (desenvolvido pela TrollTech e disponível em http://www.trolltech.com) e a biblioteca ATerms 1.5.5.

O módulo de interface gráfica é implementado em C++, usando a biblioteca Qt. Algumas funcionalidades correspondentes aos demais módulos são implementadas por programas externos: ELAN para a prototipação e de CATS para a análise das especificações CASL. As funcionalidades específicas de FERUS são implementadas na linguagem C sob a forma de bibliotecas de funções: a `libcasl` que gerencia o formato interno ferus e a `libferus` que disponibiliza operações tais como generalizar e decompilar.

3 Conclusões

Este artigo apresentou a ferramenta FERUS, que visa apoiar o desenvolvimento de software através do uso de especificações formais na linguagem CASL. A ferramenta faz parte de um projeto de cooperação internacional também denominado FERUS.

[1] O uso de duas linguagens deve-se às linguagens de implementação das bibliotecas utilizadas. No entanto, a mistura de linguagens não gera conflito pelo fato de C e C++ serem linguagens perfeitamente integráveis.

Atualmente, a ferramenta encontra-se em seu primeiro protótipo, sendo desenvolvida por alunos de graduação e de mestrado. Esse protótipo trata um sub-conjunto da linguagem CASL que deve ser aumentado de maneira incremental até o suporte da totalidade da linguagem.

A ferramenta possui um futuro promissor, pois a linguagem CASL tem como meta de projeto tornar-se um padrão em especificações algébricas. Adicionalmente, novas funcionalidades serão implementadas na ferramenta além do aperfeiçoamento das operações já existentes.

Referências

1. P. Borovanský, C. Kirchner, H. Kirchner, P.-E. Moreau, and Ch. Ringeissen. An Overview of ELAN. In C. Kirchner and H. Kirchner, editors, *Proc. Second Intl. Workshop on Rewriting Logic and its Applications*, Electronic Notes in Theoretical Computer Science, Pont-à-Mousson (France), September 1998. Elsevier.
2. CoFI group. *The CoFI Algebraic Specification Language*, 2001. Available at the CoFI home page: http://www.briks.dk/Projects/CoFI.
3. Hélène Kirchner and Christophe Ringeissen. Executing casl equational specifications with the elan rewrite engine. Technical report, The Common Framework Initiative for algebraic specification and development, electronic archives, 2000. accessible from http://www.brics.dk/Projects/CoFI.
4. A. Martins Moreira. *La Généralisation : un Outil pour la Réutilisation.* PhD thesis, INPG, March 1995.
5. T. Mossakowski. Casl - from semantics to tools. In *Proceedings of TACAS 2000*, number 1785 in LNCS, pages 93–108, Berlin, 2000. Springer Verlag.
6. S. Escobar Peraça and A. Martins Moreira. Proposta de uma ferramenta de apoio formal à especificação e re-utilização de software. In *Anais do III Workshop de Métodos Formais*, João Pessoa, Brasil, 2000.
7. M.G.J. van den Brand, H.A. de Jong, P. Klint, and P.A. Olivier. Efficient annotated terms. Technical Report SEN-R0003, CWI, 2000.