Pontifícia Universidade Católica
do Rio de Janeiro

André Luiz Castro de Almeida Reis

Editor de Diagramas Dirigido por Metamodelos

Dissertação de Mestrado

Dissertação apresentada ao Programa de Pós-graduação em Informática do Departamento de Informática do Centro Técnico Científico da PUC-Rio como requisito parcial para obtenção do título de Mestre em Informática.

Orientador: Prof. Simone Diniz Junqueira Barbosa

Rio de Janeiro
Agosto de 2011

André Luiz Castro de Almeida Reis

Editor de Diagramas Dirigido por Metamodelos

Dissertação apresentada como requisito parcial para a obtenção do grau de Mestre pelo Programa de Pós-graduação em Informática do Departamento de Informática do Centro Técnico e Científico da PUC-Rio. Aprovada pela Comissão Examinadora abaixo assinada.

Profª. Simone Diniz Junqueira Barbosa
Orientadora
Departamento de Informática – PUC-Rio

Prof. Arndt von Staa
Departamento de Informática – PUC-Rio

Prof. Alberto Barbosa Raposo
Departamento de Informática – PUC-Rio

Prof. José Eugenio Leal
Coordenador Setorial do Centro
Técnico Científico – PUC-Rio

Rio de janeiro, 26 de agosto de 2011

André Luiz Castro de Almeida Reis

Graduou-se em Engenharia da Computação pela PUC-Rio em 2008. Participou do Laboratório de Inteligência Computacional (ICA) do departamento de Engenharia Elétrica da PUC-Rio. Já trabalhou em diversas empresas, e desde janeiro de 2008 atua no laboratório de pesquisa Tecgraf, na PUC-Rio. É interessado em aplicações web e desenvolvimento de interfaces para softwares que possam proporcionar melhor qualidade e satisfação aos usuários.

Ficha Catalográfica

Reis, André Luiz Castro de Almeida

Editor de diagramas dirigido por metamodelos / André Luiz Castro de Almeida Reis ; orientador: Simone DJ Barbosa. – Rio de Janeiro: PUC, Departamento de Informática, 2011.

111 f. : il. (color.) ; 30 cm

Dissertação (mestrado) – Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática, 2011.

Inclui referências bibliográficas.

1. Informática – Teses. 2. Metamodelos. 3. Editor gráfico. 4. Modelo baseado em IHC. 5. Interação humano-computador. I. Barbosa, Simone DJ. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de

CDD:004

Este trabalho é dedicado aos meus pais, por me darem
as condições necessárias para chegar até aqui.

Agradecimentos

Em primeiro lugar, agradeço a Deus por me conceder a oportunidade de viver esse momento em que poucos tem a possibilidade de chegar. Obrigado por me conceder a benção de ter uma família maravilhosa repleta de amor e respeito. Obrigado por me cercar de amigos que me ajudaram ao longo da minha jornada.

À minha orientadora Simone Diniz Junqueira Barbosa pelo estímulo e apoio para a realização deste trabalho. Sou muito grato pelos seus conselhos e atenção que sempre teve comigo.

Aos professores que participaram da Comissão examinadora Arndt von Staa e Alberto Raposo.

Ao Tecgraf, pelos auxílios concedidos, sem os quais este trabalho não poderia ter sido realizado.

Ao meu pai Astrogildo Reis, minha mãe Helenira Reis e a minha segunda mãe Antônia Barbosa que sempre apoiaram o meu estudo e me deram condições de crescer.

Aos colegas do SERG, Bruno Silva, Fabiana Simões, Gustavo Miranda, José Motta, Vinícius Costa.

Aos meus amigos Carlos Eduardo Abrão, Cristopher Lee, Edgard Marx, Filip Duarte, Thiago Valente, Marcela Câmara, Marcelo Nascimento, Priscila Nunes e Vitor Pinheiro que de alguma forma me ajudaram a concluir este trabalho.

A todos os amigos e familiares que de uma forma ou de outra me estimularam e ajudaram.

Resumo

Reis, André Luiz Castro de Almeida. Barbosa, Simone Diniz Junqueira. **Editor de Diagramas Dirigido por Metamodelos**. Rio de Janeiro, 2011. 111p. Dissertação de Mestrado – Departamento de Informática. Pontifícia Universidade Católica do Rio de Janeiro.

O uso de editores de diagramas tem se mostrado muito útil para a elaboração de soluções de design na área de Interação Humano-Computador. Eles facilitam o uso da linguagem e o controle sobre os seus elementos, evitando que o usuário utilize itens lexicais que não pertençam ao vocabulário da linguagem. Esses elementos estão definidos em um metamodelo, que basicamente consiste em um conjunto de conceitos dentro de um determinado domínio. Com isso, o usuário ganha agilidade e confiabilidade no processo de criação. Porém, muitos editores não garantem que a solução concebida obedeça à sintaxe da linguagem. Para isso, torna-se necessário um editor que, além de ter controle sobre os símbolos da linguagem, forneça também apoio gramatical para o uso de modelos, de forma não apenas gráfica, mas também fazendo uso das regras de sintaxe de cada metamodelo. Com esse conjunto de regras que define as combinações válidas dos elementos da linguagem, o usuário pode ser alertado sobre possíveis infrações que estejam acontecendo durante a elaboração da solução. As regras descrevem a sintaxe da linguagem através de uma gramática. Analisar sintaticamente diagramas significa tentar encontrar uma sequência de aplicações de regras que derivam de uma gramática ou de alguma representação dela. Levando em consideração essa abordagem, este trabalho apresenta um estudo sobre editores de diagramas dirigidos por metamodelos e uma ferramenta que possibilita ao usuário, a partir da definição de um metamodelo, acoplá-lo a um editor de diagramas genérico para linguagens visuais, em que se possa controlar tanto o vocabulário quanto a gramática dos diagramas criados. Desta forma, o objetivo da atual pesquisa é propor uma ferramenta que englobe estas soluções e que seja focada nas linguagens visuais comuns na área de Interação Humano-Computador como MoLIC, CTT e Statecharts.

Palavras-chave

Metamodelos; Regras; Editor gráfico; Modelo baseado em IHC; Interação Humano-Computador.

Abstract

Reis, André Luiz Castro de Almeida. Barbosa, Simone Diniz Junqueira (Advisor). **Diagram Editor Driven by Metamodels**. Rio de Janeiro, 2011. 111p. MSc Dissertation – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

The use of diagram editors has been very useful for creating design solutions in the area of Human-Computer Interaction. They facilitate the use of modeling languages and provide control over the elements of the solution space, preventing the user from using an invalid lexical item of the chosen language. These elements are defined in a metamodel, which basically consists of a set of concepts with-in a given domain. As a result, users gain speed and reliability in the process of creation. However, many editors do not guarantee that designed solution meets the language syntax. For this, we need an editor that, in addition to having control over the language symbols, also provides support for the use of models, going beyond graphical editing and also making use of the syntax rules defined in each metamodel. With this set of rules that define the form of language, the user may be warned of possible rule violations while building the solution. The rules describe the syntax of the language through its grammar. To parse a diagram means to try to find a sequence of applications of rules that derive from a grammar or some representation of it. Considering this approach, this dissertation presents a study on diagram editors, and a metamodel-driven tool that allows the user to, by defining a metamodel, make use of a generic diagram editor for visual languages that can control the vocabulary and grammar of the created diagrams. Thus, the goal of current research is to propose a tool that encompasses these solutions and is focused on visual languages common in the area of Human-Computer Interaction, such as MoLIC, CTT and Statecharts.

Keywords

Metamodels; Rules; Graphical Editor; Model-based HCI; Human-Computer Interaction.

Sumário

1. Introdução 15

1.1. Motivação 15

1.2. Definição do Problema 16

1.3. Objetivo e organização do Trabalho 16

2. Trabalhos Relacionados 18

2.1. Abordagens Baseadas em Metamodelos 18

2.2. Editores de Diagramas Baseados em Regras 21

2.3. Linguagem de Regras 24

2.4. Ferramentas Existentes 26

2.4.1. Microsoft Office Visio 26

2.4.2. Visual Library 28

2.4.3. JgraphX 29

2.4.4. Dia 30

2.4.5. DiaMeta 31

3. Metamodelo 33

3.1. Elementos 33

3.1.1. Nós 33

3.1.2. Arestas 35

3.2. Regras 39

4. Linguagens de Modelagem de IHC 41

4.1. Concur Task Trees (CTT) 41

4.1.1. Representação da CTT na linguagem de regras 44

4.2. MoLIC 46

4.2.1. Cena 47

4.2.2. Acesso Ubíquo 47

4.2.3. Ponto de Entrada 48

4.2.4. Ponto de Saída 48

4.2.5. Processo do Sistema 49
4.2.6. Fala de Troca de Turno ou Transição 49
4.2.7. Fala de recuperação de ruptura (*breakdown*) 50
4.2.8. Representação da MoLIC na linguagem de regras 50
4.3. Statecharts 55
4.3.1. Representação de Statecharts na linguagem de regras 57
5. O Editor de Diagramas 59
5.1. Organização do metamodelo 59
5.2. Descrição do metamodelo 60
5.3. O aplicativo 61
6. Avaliação do Editor 64
6.1. Perfil dos participantes 65
6.2. Metodologia 68
6.3. Teste-Piloto 70
6.3.1. Questionário Pré-Teste 70
6.3.2. Cenários 71
6.3.3. Questionário Pós-teste 71
6.3.4. Termo de Consentimento 71
6.4. Apreciação dos Resultados 72
6.4.1. Erros capturados pelo editor 72
6.4.2. Apresentação dos erros de sintaxe 73
6.4.3. Apresentação das mensagens de erro 75
6.4.4. Segurança sobre a corretude sintática do modelo elaborado 75
7. Conclusões 77
7.1. Contribuições 77
7.2. Trabalhos Futuros 78
8. Referências bibliográficas 81
Apêndice A 85
Apêndice B 96

Lista de Figuras

Figura 1 - A estrutura do metamodelo do framework e seu mapeamento na estrutura do metamodelo MOF [Gholizadeh & Azgomi, 2010]. 19
Figura 2 - Exemplos de mapeamento dos elementos do formalismo para os elementos do grafo [Gholizadeh & Azgomi, 2010]. 20
Figura 3 - Definição de uma rede Petri baseado na estrutura do metamodelo [Gholizadeh & Azgomi, 2010]. 20
Figura 4 - Antes (a) e depois (b) da aplicação do layout estático [Maier & Minas, 2009]. 22
Figura 5 - Antes (a) e depois (b) da aplicação do layout dinâmico [Maier & Minas, 2009]. 22
Figura 6 - Antes (a) e depois (b) da aplicação da regra de layout [Maier, 2009]. 23
Figura 7 - Conjunto de regras por padrão (a) e condição & ação (b) [Maier, 2009]. 24
Figura 8 - Árvore de análise entre os rótulos [RuleML Tutorial, 2011]. 25
Figura 9 - Programa Microsoft Visio 2010. 26
Figura 10 - Adição de uma regra no Visio 27
Figura 11 - Exemplo de programa que usa a API Visual Library. 29
Figura 12 - Interface do programa Dia. 31
Figura 13 - Interface do editor de diagramas DiaMeta [Minas, 2006]. 32
Figura 14 - Os tipos de seta de início de aresta disponíveis: none, classic, diamond, oval, open, block. 36
Figura 15 - Os diferentes tipos de arestas na sequência: straight, horizontal, vertical, entity, arrow. 37
Figura 16 - Controle sobre as cores das arestas. 37
Figura 17 - Controle de espessura da aresta. 37
Figura 18 - Aresta com linha tracejada. 38

Figura 19 - Os tipos de seta final de aresta disponíveis: none, classic, diamond, oval, open, block. 38
Figura 20 - Diagrama do CTT. 41
Figura 21 - Nós da linguagem CTT. 44
Figura 22 - Arestas da linguagem CTT. 45
Figura 23 - Regras relacionadas a aresta Parent. 46
Figura 24 - Cena com diálogo. 47
Figura 25 - Cena sem diálogo. 47
Figura 26 - Acesso Ubíquo 48
Figura 27 - Ponto de entrada. 48
Figura 28 - Ponto de saída. 48
Figura 29 - Processo do sistema. 49
Figura 30 - Falas de troca de turno do usuário e do preposto designer. 49
Figura 31 - Falas de recuperação de ruptura do usuário e do designer. 50
Figura 32 - Nós da linguagem MoLIC. 51
Figura 33 - Arestas da linguagem MoLIC. 51
Figura 34 - Regra relacionada ao Ponto de entrada. 52
Figura 35 - Regras relacionada ao Ponto de saída. 52
Figura 36 - Regra relacionada a Fala de recuperação de ruptura. 53
Figura 37 - Regra relacionada a Fala de troca de turno ou transição do designer. 53
Figura 38 - Regra relacionada a Fala de troca de turno ou transição do usuário. 54
Figura 39 - Estados simples com XOR meta-state. 56
Figura 40 - Estado concorrente com AND meta-state. 56
Figura 41 - History State. 56
Figura 42 - Nós da linguagem Statecharts. 57
Figura 43 - Aresta da linguagem Statecharts. 57
Figura 44 - Regra relacionada ao nó super-estado. 57
Figura 45 - Regra relacionada ao nó estado padrão. 58
Figura 46 - Regras relacionada ao nó história. 58
Figura 47 - Organização da pasta do metamodelo. 60

Figura 48 - Organização do arquivo do metamodelo. 61
Figura 49 - Arquitetura do editor de diagramas. 62
Figura 50 - Tela do editor de diagramas. 63
Figura 51 - Alerta de erro de sintaxe. 63
Figura 52 - Gradiente com número ímpar de escolhas. 69

Lista de Tabelas

Tabela 1 - Relacionamento de cada participante ao uso de diferentes editores de diagramas. 66
Tabela 2 - Grau da familiaridade com a MoLIC. 67
Tabela 3 - A quantidade de sistemas já projetados (total ou parcialmente) com a MoLIC. 67
Tabela 4 - Frequência que os participantes encontraram erros de sintaxe numa representação de interação já concluída. 68
Tabela 5 - Segurança do participante sobre a corretude sintática dos seus diagramas MoLIC 68
Tabela 6 - A quantidade de erros cometidas pelos participantes capturados pelo editor. 73
Tabela 7 - Opinião sobre a apresentação das mensagens de erro. 73
Tabela 8 - O editor colaborou para detectar se houve erro de sintaxe. 74
Tabela 9 - O editor colaborou para o participante entender o erro. 74
Tabela 10 - Opinião sobre a explicação das mensagens de erro. 75
Tabela 11 - A colaboração do editor para aumentar a segurança sobre a corretude da representação 76