

PONTIFÍCIA UNIVERSIDADE CATÓLICA  
DO RIO DE JANEIRO



**Andréia Miranda de Luna**

## **Geração de Interfaces RIA Dirigida por Ontologias**

**Dissertação de Mestrado**

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

Orientador: Prof. Daniel Schwabe

Rio de Janeiro  
Dezembro de 2009



**Andréia Miranda de Luna**

**Geração de Interfaces RIA Dirigida por  
Ontologias**

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.

Prof. Daniel Schwabe  
Orientador  
Departamento de Informática – PUC-Rio

Prof. Marco Antonio Casanova  
Departamento de Informática – PUC-Rio

Prof<sup>a</sup> Simone Diniz Junqueira Barbosa  
Departamento de Informática – PUC-Rio

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

Rio de Janeiro, 6 de janeiro de 2010

Todos os direitos reservados. É proibida a reprodução total ou parcial do trabalho sem autorização da universidade, da autora e do orientador.

### **Andréia Miranda de Luna**

Graduou-se em Informática e Tecnologia da Informação pela Universidade do Estado do Rio de Janeiro em fevereiro de 2007. Possui interesse nas áreas de Engenharia de Software, Linguagens de Programação, Web Semântica e Segurança da Informação.

#### Ficha Catalográfica

Luna, Andréia Miranda de

Geração de interfaces RIA dirigida por ontologias / Andréia Miranda de Luna; orientador: Daniel Schwabe. – 2009.

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

Dissertação (Mestrado em Informática) – Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, 2009.

Inclui bibliografia

1. Informática – Teses. 2. Web semântica. 3. Desenvolvimento dirigido por modelos. 4. Aplicações de Internet rica. 5. Interfaces Web de usuário. 6. Ontologia. 7. Geração de código. I. Schwabe, Daniel. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. IV. Título.

CDD: 004

Aos meus pais e mestres, pela educação transmitida.

## Agradecimentos

A Jeová Deus, por ter me concedido a realização de mais um objetivo.

Ao prezado professor e orientador Daniel Schwabe, por ter acreditado no meu potencial e por suas inestimáveis contribuições para este trabalho.

A minha família e amigos, por todo o amor e apoio.

A todos os colegas, professores e funcionários do Departamento de Informática da PUC-Rio, pelo companheirismo, aprendizado e auxílio.

## Resumo

Luna, Andréia Miranda de; Schwabe, Daniel. **Geração de Interfaces RIA Dirigida por Ontologias**. Rio de Janeiro, 2009. 179p. Dissertação de Mestrado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Vivemos hoje na era da Web 2.0, onde os navegadores executam interfaces gráficas cada vez mais ricas, permitindo que virtualmente todo tipo de aplicação possa explorar a ubiqüidade dos navegadores Web sem comprometer a experiência do usuário. Os modelos utilizados no desenvolvimento de software, além de mecanismos de abstração e documentação, sob o paradigma do Desenvolvimento Dirigido por Modelos, também são poderosas linguagens de especificação, que, aliadas a técnicas de transformação, podem automatizar a tarefa repetitiva de gerar código de infraestrutura. A proposta deste trabalho consiste em definir uma linguagem de descrição, em alto nível, do funcionamento das interfaces RIA (*Rich Internet Application*), um processador para esta descrição, capaz de gerar o código executável da interface, e a respectiva máquina de runtime para executar as interfaces geradas. Por dar suporte ao desenvolvimento Web baseado no método SHDM (*Semantic Hypermedia Design Method*), o ambiente de prototipação HyperDE foi escolhido como a plataforma-alvo para a geração de interfaces executáveis. O ambiente de modelagem e execução das interfaces RIA também introduz um protocolo assíncrono baseado em fila de mensagens como forma de implementar a comunicação entre as camadas de Modelo e Visão. Se a tecnologia Ajax permite a comunicação assíncrona entre cliente e servidor, de tal forma que diferentes componentes da interface possam ser atualizados de forma independente, as interações entre Visão e Modelo, quando mediadas por um sistema de fila de mensagens, tornam possível a atualização de interfaces com os resultados parciais do processamento de uma requisição.

### Palavras-chave

Web Semântica; Desenvolvimento Dirigido por Modelos; Aplicações de Internet Rica; Interfaces Web de Usuário; Ontologia; Geração de Código.

## Abstract

Luna, Andréia Miranda de; Schwabe, Daniel (Advisor). **Ontology-Driven RIA Interfaces Generation**. Rio de Janeiro, 2009. 179p. MSc. Dissertation - Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

In this Web 2.0 era, the browsers perform ever-richer graphical interfaces. Today, virtually every type of application can benefit from the ubiquity of Web browsers without compromising the user experience. Under the Model-Driven Development paradigm, models represent more than abstraction and documentation tools; they can also perform as powerful specification languages. When transformation rules are applied to these models, this can automate the repetitive task of generating infrastructure code. This work proposes an abstract RIA interface description language and a whole software environment that can make it possible to the application designer to automatically generate an executable interface from an abstract description. Being the Semantic Hypermedia Design Method (SHDM) development environment, the HyperDE framework became the target platform for the RIA interface code generation. Our solution also introduces a message queue-based protocol as a way to implement asynchronous communication between Model and View. It will make it possible to update the interface with the partial results of a request processing and, therefore, improve the user experience, enhancing what Ajax technology has accomplished so far.

### Keywords

Semantic Web; Model Driven Development; Rich Internet Applications; Web User Interfaces; Ontology; Code Generation.

# Sumário

1	Introdução	18
1.1.	Motivação	19
1.2.	Objetivos e Contribuições Esperadas	20
1.3.	Estrutura da Dissertação	22
2	Aplicações Web Ricas	23
2.1.	Plataformas RIA	23
2.2.	Projeto de Interfaces RIA Baseado em Modelos	27
2.3.	Metadados para Descrição de Interfaces RIA	31
2.3.1.	OpenAjax	31
2.3.2.	OpenAjax Hub	32
2.3.3.	OpenAjax Metadata	33
2.3.4.	XForms	34
2.3.5.	IBM Collage Programming Model	38
2.3.6.	Modelagem UML	42
2.3.7.	Jitsu	44
3	Modelagem de Interfaces RIA	48
3.1.	Requisitos para a Modelagem de Interfaces	48
3.2.	Modelagem de Widgets Abstratos e Concretos	50
3.3.	Modelagem de Transições, Decorações e Eventos	52
3.4.	Mapeamento Abstrato-Concreto	56
3.5.	Mapeamento entre Visão e Modelo	58
3.6.	Uma Linguagem Extensível	61
4	Geração Automatizada de Interfaces	63
4.1.	Arquitetura SWUI de Interfaces Ricas	63
4.2.	Geração de Interfaces Ricas	65
4.2.1.	Compilação de <i>Widgets</i> Abstratos em Concretos	65
4.2.2.	Anotação Semântica do código XHTML	67
4.2.3.	Compilação de Descrições Retóricas	69
4.3.	Renderização de Interfaces Ricas	69



4.4. Execução de Interfaces Ricas	70
4.5. Transações	71
5 Especificação e Execução de Interfaces RIA no HyperDE	74
5.1. Modelo de Dados	74
5.2. Ambiente de Modelagem	76
5.2.1. Especificação de Interfaces Abstratas	76
5.2.2. Descrição Retórica	82
5.2.3. Mapeamentos Abstrato-Concreto e entre Visão e Modelo	86
5.2.4. Declaração de Folhas de Estilo	87
5.2.5. Especificação de <i>Widgets</i> Concretos	89
5.2.6. Especificação de Efeitos de Decoração	93
5.2.7. Compilação de Interfaces Abstratas	95
5.3. Ambiente de Execução	96
5.3.1. Renderização de Interfaces	97
5.3.2. Representação dos Objetos do Modelo na Interface Renderizada	101
5.3.3. Interfaces Abstratas Dinâmicas	107
5.3.4. Execução das Interfaces	111
5.3.5. Retórica na Interface	112
5.3.6. Comunicação de Eventos entre Visão e Modelo	113
6 Exemplos	117
6.1.1. Interface ProfessorView	118
6.1.2. Componente PersonalData	119
6.1.3. Componente AcademicData	119
6.1.4. Componente ContactInfo	120
6.1.5. Componente AdviseesList	121
6.1.6. Integração com <i>Cascading Style Sheets</i>	123
6.1.7. Interface Concreta	125
6.1.8. Navegação Intra e Intercontextual	127
6.1.9. Pesquisa Twitter	129
6.1.10. A Interface de Transição	131
6.1.11. Retórica em Interfaces	133
7 Conclusão	136
7.1. Avaliação	136
7.2. Contribuições	140

7.3. Críticas e Trabalhos Futuros	141
8 Referências Bibliográficas	144
Apêndice I – Ontologia de Descrição de Interfaces Ricas	148
Classes da Ontologia	148
Propriedades da Ontologia	154
Apêndice II – Gramática da Linguagem de Descrição de Widgets: SWUI.dtd	172
Apêndice III – Classes CSS do Módulo de Interfaces Ricas	174
Apêndice IV – Interfaces <i>Default</i> do Módulo de Interfaces Ricas	176
DefaultAbstractInterface	176
DefaultContextInterface	176
DefaultIndexInterface	176
DefaultLandmarks	177
DefaultLandmarksForIndexes	177
DefaultNavBar	178
DefaultContextIndex	178
DefaultJSONDisplay	179

## Lista de figuras

Figura 1 – Método RUX-Model	29
Figura 2 – Componentes confiáveis e não-confiáveis em um <i>mashup</i>	33
Figura 3 – Exemplo de formulário em uma aplicação de comércio eletrônico	35
Figura 4 – Modelo de dados XForm correspondente ao formulário	35
Figura 5 – Dados coletados pelo formulário em formato XML	35
Figura 6 – Exemplo de Modelo de dados XForm	37
Figura 7 – <i>Binding</i> usando o atributo <i>ref</i>	37
Figura 8 – Binding usando o atributo <i>bind</i>	37
Figura 9 – Exemplo de declaração de uma ação XForm.	38
Figura 10 – Modelo de Execução Collage	40
Figura 11 – Unidade de interação abstrata do modelo Collage	41
Figura 12 – Exemplo da declaração de um campo de entrada de dados no modelo Collage	42
Figura 13 – Perfil UML para modelagem de <i>Widgets</i> .	43
Figura 14 – Arquitetura associada ao perfil UML	44
Figura 15 – Geração e execução das aplicações Jitsu	46
Figura 16 – <i>Databinding</i> no <i>framework</i> Jitsu	47
Figura 17 – Classes <i>AbstractInterface</i> e <i>AbstractInterfaceElement</i> .	50
Figura 18 – Classes <i>ConcreteInterfaceElement</i> , <i>DHTMLInterfaceElement</i> , <i>HTMLTag</i> e <i>DHTMLRichcontrol</i> .	51
Figura 19 – Interface <i>ProfessorView</i>	52
Figura 20 – Especificação abstrata da interface <i>ProfessorView</i>	52
Figura 21 – Classe <i>Decoration</i>	54
Figura 22 – Classe <i>Event</i>	55
Figura 23 – Classes <i>Transition</i> , <i>RhetStructSeq</i> e <i>RhetStructSet</i>	55
Figura 24 – Classes <i>RhetoricalStructure</i> , <i>DecorationSeq</i> e <i>DecorationSet</i>	56
Figura 25 – Interface <i>ProfessorView</i> formatada com folha de estilo CSS	57
Figura 26 – Especificação abstrata da interface <i>ProfessorView</i> , incluindo os mapeamentos abstrato-concreto	58
Figura 27 – Propriedades de mapeamento visão-modelo	60
Figura 28 – Especificação abstrata da interface <i>ProfessorView</i> , incluindo os mapeamentos visão-modelo	61

Figura 29 – Arquitetura SWUI de Interfaces Ricas	64
Figura 30 – Geração de Interfaces	65
Figura 31 – Propriedades de mapeamento abstrato-concreto	66
Figura 32 – Especificação abstrata da visualização de um objeto, incluindo o mapeamento visão-modelo	68
Figura 33 – Código HTML+RDFa correspondente	68
Figura 34 – Modelo de dados RDF correspondente.	68
Figura 35 – Renderização de Interfaces	70
Figura 36 – Modelo de dados das Transações	71
Figura 37 – Diagrama de Seqüência de uma Transação	72
Figura 38 – Execução de uma operação sob o contexto de uma transação	73
Figura 39 – Modelo de dados do Módulo de Interfaces Ricas do HyperDE	75
Figura 40 – Menu do Módulo de Interfaces Ricas	76
Figura 41 – Tela de edição de interfaces abstratas	78
Figura 42 – Edição de componentes de interface	78
Figura 43 – Especificação abstrata da interface <i>ProfessorView</i> (Representação gráfica)	79
Figura 44 – Especificação abstrata da interface <i>ProfessorView</i> (Notação N3)	79
Figura 45 – Especificação abstrata da componente <i>PersonalData</i> (Representação gráfica)	79
Figura 46 – Especificação abstrata da componente <i>PersonalData</i> (Notação N3)	80
Figura 47 – Especificação abstrata da componente <i>AcademicData</i> (Representação gráfica)	80
Figura 48 – Especificação abstrata da componente <i>AcademicData</i> (Notação N3)	81
Figura 49 – Especificação abstrata da interface <i>ProfessorView</i> (sintaxe XML)	81
Figura 50 – Trecho de código exemplificando o uso da diretiva <code>include</code>	82
Figura 51 – Código correspondente após a resolução da diretiva <code>include</code>	82
Figura 52 – Segunda versão da interface <i>ProfessorView</i>	83
Figura 53 – Especificação da retórica da interface (código DSL)	84
Figura 54 – Especificação de uma transição (Notação N3)	84
Figura 55 – Especificação de uma seqüência de estruturas retóricas (Notação N3)	84
Figura 56 - Especificação de uma seqüência de decorações (Notação N3)	84
Figura 57 – Especificação de decorações (Notação N3)	85
Figura 58 – Especificação da retórica da interface (código DSL)	85
Figura 59 – Especificação de eventos (Notação N3)	86
Figura 60 – Listagem de interfaces abstratas	86

Figura 61 – Tela de mapeamentos da interface	87
Figura 62 – Tela de edição de folhas de estilo CSS	88
Figura 63 – Detalhe da tela de mapeamentos da interface	88
Figura 64 – Editor de instâncias da classe <i>HTMLTag</i>	90
Figura 65 – Editor de instâncias da classe <i>RichControl</i>	91
Figura 66 – Exemplo de controle rico (código HTML)	92
Figura 67 – Exemplo de controle rico (código Javascript)	92
Figura 68 – Editor de instâncias da classe <i>DHTMLEffect</i>	94
Figura 69 – Exemplo de efeito de decoração (código Javascript)	95
Figura 70 – Compilação de interfaces abstratas	96
Figura 71 – Funções <i>readEvents()</i> , <i>readTransitions()</i> e <i>readDecorations()</i> , para a interface <i>ProfessorView</i>	100
Figura 72 – Renderização de Interfaces	101
Figura 73 – Função <i>readJSONData()</i> para uma instância da interface <i>ProfessorView</i>	102
Figura 74 – Terceira versão da interface <i>ProfessorView</i>	104
Figura 75 – Landmarks, e índices de contexto, serializados em um <i>hash</i> JSON	105
Figura 76 – Índice “ReseachAreas”	105
Figura 77 – Índice navegacional serializado em um <i>hash</i> JSON	107
Figura 78 – Exemplo de utilização da chave <i>swui:order</i>	107
Figura 79 – Código Ruby na descrição abstrata da interface	108
Figura 80 – Detalhe de uma instância de <i>DefaultContextInterface</i>	109
Figura 81 – Exemplo de código fonte de uma interface abstrata dinâmica	110
Figura 82 – Exemplo de código compilado de uma interface abstrata dinâmica	111
Figura 83 – Execução de uma operação sob o contexto de uma transação	115
Figura 84 – Exemplo de operação definida no Modelo	115
Figura 85 – Operação reescrita para suportar a execução sob o contexto de uma transação	116
Figura 86 – Esquema navegacional do site do Departamento de Informática (Site DI)	117
Figura 87 – Especificação abstrata da componente <i>PersonalData</i>	119
Figura 88 – Especificação abstrata da componente <i>AcademicData</i>	120
Figura 89 – Especificação abstrata da componente <i>ContactInfo</i>	121
Figura 90 – Especificação abstrata da componente <i>AdviseesList</i>	122
Figura 91 – <i>Widget ProfessorNode</i>	123
Figura 92 – Edição da interface <i>ProfessorView</i>	123

Figura 93 – Interface <i>ProfessorView</i> , sem a aplicação de CSS	124
Figura 94 – Interface <i>ProfessorView</i> , com CSS aplicado	125
Figura 95 – Código concreto da interface <i>ProfessorView</i>	127
Figura 96 – <i>Widget ProfessorNode</i>	128
Figura 97 – Versão final da interface <i>ProfessorView</i>	129
Figura 98 – Código da operação <i>searchTwitter</i>	130
Figura 99 – Detalhe da interface <i>AreasView</i> , junto com a especificação do ativador <i>twitterSearch</i>	131
Figura 100 – Interface de transição <i>TwitterSearch</i>	131
Figura 101 – Código Ruby embutido na especificação da interface <i>TwitterSearch</i>	132
Figura 102 – Especificação abstrata do <i>widget QueryResultsList</i>	132
Figura 103 – Especificação da estrutura retórica <i>HideContextData</i>	133
Figura 104 – Especificação da estrutura retórica <i>DisplayTweet</i>	134
Figura 105 – Especificação do evento <i>TwitterLogoClick</i>	134
Figura 106 – Especificação da transição <i>TwitterSearch</i>	135

## Lista de tabelas

Tabela 1 – Comparação entre os modelos de dados Collage/RDF, Entidade-Relacionamento, UML, Relacional e XML.	38
Tabela 2 – Regras de mapeamento abstrato-concreto	57
Tabela 3 – Regras de tradução para código HTML	67
Tabela 4 – Atributos da classe navegacional <i>Professor</i>	119
Tabela 5 – Índice <i>StudentByProfessor</i>	121

## Lista de abreviaturas

ADO	Abstract Data Object
ADV	Abstract Data View
Ajax	Asynchronous Javascript and XML
CSS	Cascading Style Sheets
DHTML	Dynamic HTML
DOM	Document Object Model
DSL	Domain Specific Language
DTO	Data Transfer Objects
GUI	Graphical User Interfaces
HTML	HyperText Markup Language
JSON	JavaScript Object Notation
MDA	Model Driven Architecture
MVC	Model-View-Controller
OO-H	Object-Oriented Hypermedia
OOHDM	Object-Oriented Hypermedia Design Method
PIM	Platform Independent Model
PSM	Platform Specific Model
RDF	Resource Description Framework
RDFa	Resource Description Framework in attributes
RIA	Rich Internet Application
RUX-Model	Rich User eXperience Model
SHDM	Semantic Hypermedia Design Method
SUL	Simple User Interface Language
SWUI	Semantic Web User Interface
UIML	User Interface Markup Language
UWE	UML-based Web Engineering
UWE	UML based Web Engineering
WebML	Web Modelling Language
WPF	Windows Presentation Foundation
WPF	Windows Presentation Foundation
XAML	eXtensible Application Markup Language
XML	Extensible Markup Language
XSLT	eXtensible Stylesheet Language Transformations
XUL	XML User Interface Language
XUP	eXtensible User Interface Protocol
YUI	Yahoo! User Interface Library



“One machine can do the work of fifty ordinary men. No machine can do the work of one extraordinary man.”<sup>1</sup>

Elbert Hubbard

---

<sup>1</sup> “Uma máquina é capaz de realizar o trabalho de 50 homens comuns. Máquina nenhuma é capaz de realizar a obra de um homem extraordinário.”