

**Christina von Flach Garcia  
Chavez**

**Um Enfoque Baseado em  
Modelos para o Design  
Orientado a Aspectos**

**TESE DE DOUTORADO**

**DEPARTAMENTO DE INFORMÁTICA  
Programa de Pós-graduação em  
Informática**

Rio de Janeiro  
Abril de 2004

PONTIFÍCIA UNIVERSIDADE CATÓLICA  
DO RIO DE JANEIRO



**Christina von Flach Garcia Chavez**

**Um Enfoque Baseado em Modelos para o  
Design Orientado a Aspectos**

**Tese de Doutorado**

Tese apresentada ao Programa de Pós-graduação em Informática do Departamento de Informática da PUC-Rio como parte dos requisitos parciais para obtenção do título de Doutor em Informática

Orientador: Prof. Carlos José Pereira de Lucena

Rio de Janeiro  
Abril de 2004



**Christina von Flach Garcia Chavez**

**Um Enfoque Baseado em Modelos para o  
Design Orientado a Aspectos**

Tese apresentada ao Programa de Pós-graduação em Informática do Departamento de Informática do Departamento de Informática da PUC-Rio como parte dos requisitos parciais para obtenção do título de Doutor em Informática. Aprovada pela Comissão Examinadora abaixo assinada.

**Prof. Carlos José Pereira de Lucena**

Orientador

Departamento de Informática — PUC-Rio

**Prof. Cecília Mary Fischer Rubira**

Instituto de Computação - UNICAMP

**Prof. Paulo Henrique Monteiro Borba**

Centro de Informática - UFPE

**Prof. Arndt von Staa**

Departamento de Informática - PUC-Rio

**Prof. Noemi de La Rocque Rodriguez**

Departamento de Informática - PUC-Rio

**Prof. José Eugenio Leal**

Coordenador Setorial do Departamento de Informática — PUC-Rio

Rio de Janeiro, 02 de Abril de 2004

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

### **Christina von Flach Garcia Chavez**

Graduou-se em Processamento de Dados na Universidade Federal da Bahia em 1987 e obteve o título de Mestre em Ciência da Computação pela Unicamp em 1992. Desde 1990, é Professora da UFBA, vinculada ao curso de Bacharelado em Ciência da Computação, onde leciona disciplinas e orienta trabalhos de Iniciação Científica na área de Linguagens de Programação e Engenharia de Software.

Sua área de pesquisa atual é Engenharia de Software, com especial interesse em Desenvolvimento de Software Orientado a Aspectos.

#### Ficha Catalográfica

Chavez, Christina von Flach Garcia

Um Enfoque Baseado em Modelos para o Design Orientado a Aspectos/ Christina von Flach Garcia Chavez; orientador: Carlos José Pereira de Lucena. — Rio de Janeiro : PUC–Rio, Departamento de Informática, 2004.

298 f: il. ; 30 cm

1. Tese (doutorado) - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática.

Inclui referências bibliográficas.

1. Informática – Teses. 2. Desenvolvimento de software orientado a aspectos. 3. Linguagens orientadas a aspectos. 4. Modelo de aspectos. 5. Metamodelagem. 6. Separação de *concerns*. I. Lucena, Carlos José Pereira de. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. III. Título.

CDD: 004

Para Luis Alberto, Leonardo, Bruno e Alexandre,  
que não sabem o que são aspectos  
(mas é como se soubessem).

## Agradecimentos

A idéia de partir rumo ao Rio de Janeiro para fazer doutorado veio em Setembro de 1997. Trinta minutos após ter enviado uma mensagem bem formal para o professor Sergio Carvalho, consultando sobre a possibilidade de ser orientada por ele, recebi a resposta desejada e suficiente: “Venha!” A ele, onde quer que esteja, manifesto meus agradecimentos pela acolhida imediata e pelas lições várias, entremeadas por rasgos de ironia inteligente e pitadas de bom humor.

Os seis meses seguintes voaram. Concluí as duas disciplinas de graduação que ministrava, redigi relatórios de atividades e afastei-me da chefia de departamento. Aos colegas do Departamento de Computação (DCC) da Universidade Federal da Bahia (UFBA), agradeço pela confiança depositada. À UFBA e à Capes, agradeço pelo suporte financeiro, através de salário e bolsa PICDT/Capes. Ao saudoso ex-reitor Felipe Serpa, agradeço por ter apoiado o Planejamento Estratégico de nosso DCC, que teve como principal meta a viabilização do Mestrado em Computação na UFBA através da qualificação de seus docentes, entre outras ações. A Fabíola, Aline e Claudia, agradeço pela amizade e pelo respeito que nos fizeram trabalhar com muita motivação, unidas em torno de ideais compartilhados, ao longo de uma década.

Partimos de Salvador numa manhã ensolarada de sábado. Na bagagem, brinquedos, livros de histórias e mamadeiras davam uma noção aproximada do desafio da empreitada. E desde o primeiro dia, o Rio de Janeiro nos recebeu de braços abertos. Morar na Gávea é como morar no paraíso. O clima agradável do bairro estendia-se da Escola Parque – escola de referência, onde nossos meninos estiveram felizes e bem cuidados por três anos – até a PUC-Rio e seu ambiente acadêmico de primeiro mundo imerso na Mata Atlântica. Após três anos, estávamos tão adaptados à vida no Rio que, em 2001, ao retornar para Salvador, a família passou por uma espécie de banzo generalizado e às avessas, de pessoas e lugares, de hábitos adquiridos e momentos especiais. Agradeço a todos que nos proporcionaram a alegria da convivência diária nos vários anos em que estivemos fora da Bahia. À cidade do Rio de Janeiro e aos amigos de todos os lugares: aquele abraço!

No final de 1999, fomos acolhidos – cinco alunos de doutorado e um de mestrado – pelo professor Carlos Lucena. Em um momento extremamente delicado para todos nós, ele nos deu a base, o apoio necessários para prosseguirmos, respeitando as linhas de pesquisa e temas previamente combinados com o professor Sergio Carvalho, e nos inserindo no rico ambiente de pesquisa do TecComm/LES. Em 2002, mais acolhimento e respeito: foram do professor Lucena as palavras e os silêncios que me ajudaram a superar outras dificuldades. Apesar das incomparáveis qualidades acadêmicas do professor Lucena, levo como lembrança mais marcante desse período de convivência, a sua excelência em viver, e viver como um homem sensível e de bem. Professor, muito obrigada.

O ingresso no grupo do professor Lucena deu-me a oportunidade de trabalhar com Alessandro, um jovem brilhante, com quem aprendi muito sobre engenharia de software, trabalho em equipe e outras “artesanias” científicas. Espero que possamos trabalhar juntos em outras ocasiões.

Além de Alessandro, tive a oportunidade de trabalhar com diversas pessoas em diferentes contextos durante o doutorado, resultando em trabalhos, artigos científicos e boas recordações. A Noemi Rodriguez e Thais Batista, Sylvia Cruz, Cláudio Sant’Anna, Otávio Silva, Viviane Silva, Anarosa Brandão, Arndt von Staa e Paulo Alencar, e aos diversos professores, colegas e funcionários do Departamento de Informática na PUC-Rio, muito obrigada.

À Banca Examinadora e ao prof. Brian Henderson-Sellers pelas sugestões enriquecedoras para este trabalho, muito obrigada.

◇ ◇ ◇

Amizade e os gestos que fazem o coração transbordar. Sandra, Alessandro, Eduardo, Adriana, India, Lerner, Karin, Miriam, Anarosa, Cláudio, Uirá, Guga, Bruno, Rodrigo Assis, Claudia, Vera, Akeo... E Regina, D. Stella, Luis Guilherme e Denise, Fatima, a querida D. Neveline e muitos nomes que não poderei registrar neste espaço, mas cujas marcas de gentileza estão registradas em meu coração. Obrigada.

Família. Guilherme e Lia, meu pai e minha mãe. Não poderia ter chegado tão longe sem seu apoio, desde os primeiros passos. D. Conchita, uma das pessoas mais admiráveis que conheço. Este trabalho é um pouco dela, pelo seu inesgotável carinho maternal. Obrigada.

Beto, Leo, Bruno e Alexandre, nossas vidas entrelaçadas. As coisas da vida ficam mais belas e simples com vocês, por vocês. Obrigada por me ensinarem o que é o Amor, diariamente.

## Resumo

Chavez, Christina von Flach Garcia; Lucena, Carlos José Pereira de. **Um Enfoque Baseado em Modelos para o Design Orientado a Aspectos**. Rio de Janeiro, 2004. 298p. Tese de Doutorado — Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Programação Orientada a Aspectos (POA) é um paradigma de programação que propõe um novo tipo de abstração – denominado *aspecto* – que permite a descrição modular de propriedades que, em geral, se encontram espalhadas por vários pontos de um sistema. Há distintas abordagens que podem ser classificadas como orientadas a aspectos. Cada abordagem propõe tipos de abstração, mecanismos de composição e terminologia específicos.

Nesta tese, deslocamos o foco de atenção dos mecanismos de implementação que dão suporte à tecnologia de programação orientada a aspectos, para os conceitos e propriedades que a caracterizam como um paradigma emergente para o desenvolvimento de software. Em particular, focalizamos em como estes conceitos e propriedades podem ser explorados na fase de design para construir sistemas que sejam mais fáceis de compreender, evoluir e reutilizar. Esta tese aborda questões atuais relacionadas a design e modelagem orientados a aspectos e propõe: (i) o **modelo de aspectos**, um arcabouço conceitual unificador para POA que fornece terminologia consistente e semântica básica para analisar problemas à luz dos conceitos e propriedades de POA, (ii) **aSideML**, uma linguagem de modelagem para especificação e comunicação de designs orientados a aspectos. A linguagem **aSideML** define uma notação gráfica, semântica e regras que permitem ao projetista construir modelos em que aspectos são tratados explicitamente como cidadãos de primeira classe, e (iii) o metamodelo **aSide**, um modelo lógico que define a semântica de modelos estruturais e comportamentais representados em **aSideML**.

## Palavras-chave

Desenvolvimento de software orientado a aspectos; linguagens orientadas a aspectos; modelo de aspectos; metamodelagem; separação de *concerns*.



## Abstract

Chavez, Christina von Flach Garcia; Lucena, Carlos José Pereira de. **A Model-Driven Approach for Aspect-Oriented Design**. Rio de Janeiro, 2004. 298p. PhD. Thesis — Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Aspect-Oriented Programming (AOP) is a new programming paradigm that promotes advanced separation of concerns by introducing a new modular unit, called aspect, for the modularization of crosscutting concerns. As an emerging programming paradigm, there are many different approaches to AOP, with distinct and varying sets of abstractions and composition mechanisms, specific terminology, properties and language constructs.

In this thesis, we shift the focus from the mechanisms that support the aspect-oriented programming technology to the concepts and properties that characterize it as an emerging paradigm for software development. In particular, we focus on how these concepts and properties can be exploited at an early design stage to build systems that are easier to comprehend, evolve and reuse.

This thesis addresses current issues related to aspect-oriented design and modeling, and proposes: (i) the **aspect model**, an unifying conceptual framework for AOP that provides consistent terminology and basic semantics for thinking about a problem in terms of AOP core concepts and properties, (ii) the **aSideML**, a modeling language for specifying and communicating aspect-oriented designs. The **aSideML** provides notation, semantics and guidelines that enable the designer to build models in which aspects are explicitly treated as first-class citizens, and (iii) the **aSide** metamodel, a logical model that defines the semantics of structural and behavioral models supported by the **aSideML**.

## Keywords

Aspect-oriented software development; aspect-oriented languages; aspect model; metamodeling; separation of concerns.

# Sumário

1	Introdução	17
1.1	Contexto	17
1.2	Declaração do Problema	19
1.3	Solução Proposta	23
1.4	Contribuições	26
1.5	Organização da Tese	27
2	Programação Orientada a Aspectos	31
2.1	Visão Geral	31
2.2	Separação de Concerns	32
2.3	Separação de Concerns Avançada	39
2.4	Desenvolvimento de Software Orientado a Aspectos	67
2.5	Considerações Finais	68
3	Uma Teoria de Aspectos	72
3.1	O Modelo de Aspectos	73
3.2	Utilização do Modelo de Aspectos	87
3.3	Trabalhos Relacionados	93
3.4	Considerações Finais	94
4	Modelagem Orientada a Aspectos	95
4.1	Avanços na Modelagem de Software	96
4.2	Questões da Modelagem Orientada a Aspectos	105
4.3	Requisitos para Linguagens de Modelagem Orientadas a Aspectos	110
4.4	A Linguagem UML	111
4.5	Modelagem Orientada a Aspectos e UML	116
4.6	Abordagens para Modelagem Orientada a Aspectos	118
4.7	Considerações Finais	132
5	A Linguagem de Modelagem aSideML	135
5.1	Visão Geral da Linguagem aSideML	136
5.2	Modelagem Estrutural	141
5.3	Modelagem Comportamental	160
5.4	Modelagem Composicional	171
5.5	Usando aSideML	177
5.6	Trabalhos Relacionados	179
5.7	Considerações Finais	181
6	O Metamodelo aSide	185
6.1	Visão Geral do Modelo aSide	185
6.2	Aspect Core	191
6.3	Elementos Comportamentais	197
6.4	Considerações Finais	200

7	Estudo de Caso: Portalware	<b>201</b>
7.1	Concerns de Agência	202
7.2	Projeto Orientado a Aspectos de Portalware	208
7.3	Revisitando Portalware com aSideML	215
7.4	Trabalhos Relacionados	232
8	Princípios e Diretrizes para o Design Orientado a Aspectos	<b>235</b>
8.1	Princípios	235
8.2	Diretrizes	240
8.3	Trabalhos Relacionados	246
8.4	Considerações Finais	246
9	Conclusões	<b>247</b>
9.1	Contribuições	247
9.2	Trabalhos Futuros	249
	Bibliografia	<b>251</b>
A	Transformações	<b>267</b>
A.1	De aSideML para o metamodelo aSide	267
A.2	De aSideML para AspectJ	269
A.3	De aSideML para Hyper/J	270
B	Especificação do Metamodelo aSide	<b>272</b>
B.1	O Pacote <i>Aspect Core</i>	273
B.2	O Pacote <i>Data Types</i>	287
B.3	O Pacote <i>Behavioral Elements</i>	288
B.4	O Modelo de Componentes	292
B.5	O Pacote <i>Join Points</i>	293
B.6	O Pacote <i>Weaving</i>	296

## Lista de Figuras

1.1	Roteiro da Tese.	28
2.1	Espalhamento e entrelaçamento em programas estruturados.	34
2.2	Decomposição orientada a objetos.	36
2.3	Espalhamento e entrelaçamento em programas orientados a objetos.	38
2.4	Decomposição orientada a Aspectos	39
2.5	Implementação fechada X implementação aberta [108].	42
2.6	Visões subjetivas de uma árvore orientada a objetos.	43
2.7	O método adaptativo <code>sumSalaries</code> [84].	46
2.8	Filtros múltiplos [5].	48
2.9	Um objeto no modelo de Filtros de Composição [14].	48
2.10	O combinador de aspectos [73].	52
2.11	Diagrama de classes do exemplo das figuras.	54
2.12	Grafo de chamadas para <code>Line.moveBy(2,2)</code> [74].	55
2.13	Um Hyperspace.	61
2.14	Hyperslice.	62
2.15	O pacote <code>DisplayUpdate</code> .	65
2.16	O modelo de Filtros de Composição	69
2.17	O modelo de AspectJ	70
2.18	O modelo de Hyperspaces	71
3.1	O <i>modelo de aspectos</i> .	74
3.2	O modelo de componentes.	75
3.3	O modelo de pontos de combinação.	76
3.4	Aspecto.	79
4.1	Princípios de abstração usados na modelagem orientada a objetos [130].	99
4.2	Crosscutting na modelagem orientada a aspectos.	102
4.3	Exemplo de aspecto sistêmico [10].	103
4.4	Exemplo de aspecto subjetivo.	104
4.5	Exemplo de aspecto evolutivo [10].	105
4.6	A arquitetura de quatro camadas de UML.	113
4.7	Mecanismos de extensão [17].	116
4.8	Diagrama de classes em UML com estereótipos e ocorrências de padrão [62].	119
4.9	<code>SubjectObserverProtocol</code> em AspectJ [10].	121
4.10	<code>SubjectObserverProtocolImpl</code> em AspectJ [10].	122
4.11	AODM: Notação [127].	123
4.12	AODM: Notação para Advice [127].	124
4.13	AODM: Colaboração para After Advice [127].	125
4.14	AODM: processo de combinação de advice [127].	126
4.15	Um sujeito no nível de design [32].	127

4.16	Exemplo de integração com Merge [32].	128
4.17	Padrões de composição: o padrão de projeto Observer Pattern [29].	130
4.18	Composição de Observer com Library [29].	131
4.19	Saída de composição: Library com Observer [29].	131
4.20	Metamodelo de padrões de composição[32].	132
5.1	Declaração de aspecto completa.	142
5.2	Declaração de aspecto condensada.	143
5.3	O padrão de projeto Observer modelado como um aspecto.	144
5.4	Observation com Interfaces.	145
5.5	Interface transversal com adições, refinamentos, redefinições e usos.	146
5.6	Interface transversal com compartimentos omitidos.	146
5.7	Interface transversal : detalhes no nível da análise.	149
5.8	Interface transversal com adornos textuais.	150
5.9	Crosscutting.	153
5.10	Crosscutting no estilo um-para-vários.	153
5.11	O relacionamento precedence.	155
5.12	Relacionamento requirement.	156
5.13	Diagrama de aspecto e o aspecto Timing.	157
5.14	Diagrama de classes estendido.	158
5.15	Perspectiva centrada em base.	159
5.16	Perspectiva de ponto de combinação: <i>crosscutting</i> omitido na fonte.	160
5.17	Perspectiva de ponto de combinação: <i>crosscutting</i> omitido no destino.	160
5.18	Diagrama de sequência para stateChange.	161
5.19	Instância de aspecto.	162
5.20	Colaboração aspectual: parte estática.	163
5.21	Colaboração aspectual: Refine-Before em Receiver.	164
5.22	Colaboração aspectual para stateChange..	165
5.23	Interação adornada com o símbolo de ponto de combinação.	167
5.24	Interação aspectual.	168
5.25	Padrão de interação Refine-Before-Execution.	169
5.26	Padrão de interação Refine-After-Execution.	169
5.27	Padrão de interação Refine-Around-Execution.	170
5.28	Padrão de interação Redefine-Execution.	170
5.29	Notação.	172
5.30	Notação.	173
5.31	Estilo não-hierárquico [43].	173
5.32	Classe combinada no estilo não-hierárquico.	174
5.33	Classes combinadas: Button and ColorLabel.	174
5.34	Colaboração base.	175
5.35	Colaboração combinada: click e stateChange..	176
5.36	Colaboração combinada: click, stateChange_ e baseOp..	177
5.37	Ferramentas para aSideML.	178
5.38	Herança e aspectos.	182

6.1	O metamodelo aSide.	188
6.2	Resumo da abordagem.	190
6.3	Pacotes de alto nível.	191
6.4	O pacote Aspects.	192
6.5	Aspect Core: <i>Backbone</i> .	193
6.6	Relacionamento de <i>crosscutting</i> .	194
6.7	Aspect Core: Aspectos	195
6.8	Instâncias de aspecto.	197
6.9	Colaborações aspectuais.	198
6.10	Interações aspectuais.	199
7.1	Concerns de agência.	203
7.2	Relacionamento entre propriedades de agência [53].	208
7.3	O método orientado a aspectos para o desenvolvimento de SMAs [53].	209
7.4	Agentes no SMA Portalware [53].	210
7.5	Visão estática preliminar de agência para Portalware [47].	211
7.6	Visão estática preliminar para agentes de informação [47].	212
7.7	Visão dinâmica preliminar para Portalware [47].	213
7.8	Estado e comportamento de Agent (Conhecimento).	215
7.9	Tipos de agente.	216
7.10	Visão estática do aspecto Interaction.	217
7.11	Diagrama de sequência para <code>outgoingMsg_()</code> .	217
7.12	Interação aspectual para <code>outgoingMsg_()</code> .	218
7.13	Interação aspectual para <code>incomingMsg_()</code> .	218
7.14	Aspecto Interaction afeta Agent e Sensor.	219
7.15	A visão estática do aspecto Autonomy.	220
7.16	Interação aspectual para <code>makesDecision_()</code> .	220
7.17	Autonomy afeta Agent e requer Interaction.	221
7.18	A visão estática do aspecto Adaptation.	221
7.19	Interação aspectual para <code>adaptPlan_()</code> .	222
7.20	Adaptation afeta Agent e requer Interaction.	223
7.21	Agência revisitada.	224
7.22	A visão estática do aspecto Caller.	225
7.23	Refinamento de <code>search()</code> .	225
7.24	Visão estática do aspecto Answerer.	226
7.25	Caller e Answerer afetam InformationAgent.	226
7.26	Visão estática do aspecto Collaboration.	227
7.27	Collaboration afeta InformationAgent.	227
7.28	Visão estática de InformationAgent revisitada.	228
7.29	Visão dinâmica de Portalware revisitada.	229
7.30	Visão dinâmica de Portalware revisitada.	230
7.31	Visão dinâmica de Portalware revisitada.	231
7.32	Visão estática do padrão Interaction [54].	232
7.33	Visão dinâmica do padrão Interaction: enviando e recebendo mensagens [54].	233
7.34	Arquitetura de agentes orientada a aspectos [77].	234
8.1	Dependências no design orientado a aspectos.	237

8.2	Refinamento e adição de operação usando uma interface transversal.	241
8.3	Refinamento e introdução de operação (duas interfaces transversais).	242
8.4	Aspecto para Colaboração.	243
8.5	O aspecto Timing.	244
8.6	Aspecto para evolução.	245
9.1	Conformidade com MOF.	250
B.1	Aspect Core: Backbone	274
B.2	Aspect Core: Aspectos	275
B.3	Aspect Core: Dependências	276
B.4	Crosscutting.	278
B.5	Data Types.	287
B.6	Instâncias de aspecto.	289
B.7	Colaborações aspectuais.	290
B.8	Interações aspectuais.	291
B.9	Modelo de pontos de combinação.	294
B.10	Modelo de Weaving.	297

## Lista de Tabelas

1.1	Resumo da definição do problema.	23
2.1	Pesquisa sobre POA: Perspectiva Histórica.	40
3.1	Características transversais.	80
3.2	Conceitos e propriedades definidos no <i>modelo de aspectos</i> .	87
3.3	Interpretações usando o <i>Modelo de Aspectos</i> .	90
4.1	Crosscutting e Herança.	100
4.2	Conceitos e propriedades definidos no modelo de aspectos.	106
4.3	Comparação entre abordagens.	133
5.1	Dimensões da modelagem orientada a aspectos com aSideML.	138
5.2	Elementos estruturais e pontos de combinação estáticos.	151
5.3	Substituição para parâmetros de template do aspecto.	152
5.4	Comparação.	180
6.1	Arquitetura de metamodelagem de quatro camadas.	189
6.2	Elementos padrão.	196
7.1	Visão geral de propriedades de agência.	204
8.1	Princípios Gerais.	236
8.2	Princípios relacionados à orientação a objetos.	236
A.1	Mapeamento entre aSideML e AspectJ	269
B.1	Elementos Padrão	286
B.2	Sombras Estáticas.	296



Teremos coisas bonitas pra contar.  
E até lá, vamos viver  
temos muito ainda por fazer  
não olhe pra trás  
apenas começamos, o mundo começa agora.  
Apenas começamos.

**Renato Russo**, *Metal contra as nuvens*.