



**Antônio Tadeu Azevedo Gomes**

**LindaX**

**Uma Linguagem de Descrição de  
Sistemas de Comunicação Adaptáveis**

**Tese de Doutorado**

Tese apresentada como requisito parcial para  
obtenção do título de Doutor pelo Programa de Pós-  
Graduação em Informática da PUC-Rio.

Orientador: Luiz Fernando Gomes Soares

Rio de Janeiro, março de 2005



**Antônio Tadeu Azevedo Gomes**

**LindaX**

**Uma Linguagem de Descrição de  
Sistemas de Comunicação Adaptáveis**

Tese apresentada como requisito parcial para obtenção do título de Doutor pelo Programa de Pós-Graduação em Informática da PUC-Rio. Aprovada pela Comissão Examinadora abaixo assinada.

**Luiz Fernando Gomes Soares**

Orientador

Departamento de Informática – PUC-Rio

**Sérgio Colcher**

Departamento de Informática – PUC-Rio

**Renato Fontoura de Gusmão Cerqueira**

Departamento de Informática – PUC-Rio

**Paulo Roberto Freire Cunha**

Centro de Informática – UFPE

**Fábio Moreira Costa**

Instituto de Informática – UFG

**Bruno Schulze**

Coordenação de Ciência da Computação – LNCC/MCT

**José Eugenio Leal**

Coordenador(a) Setorial do Centro Técnico Científico – PUC-Rio

Rio de Janeiro, 21 de março de 2005

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

### **Antônio Tadeu Azevedo Gomes**

Formou-se Bacharel em Informática pela UFRJ. Obteve o título de Mestre em Ciências em Informática pela PUC-Rio. É tecnologista do LNCC/MCT. Também é professor do Curso de Especialização em Redes de Computadores da PUC-Rio e do Curso de Bacharelado em Informática da Universidade Estácio de Sá.

#### Ficha Catalográfica

Gomes, Antônio Tadeu Azevedo

LindaX: uma linguagem de descrição de sistemas de comunicação adaptáveis / Antônio Tadeu Azevedo Gomes; orientador: Luiz Fernando Gomes Soares. – Rio de Janeiro: PUC, Departamento de Informática, 2005.

194 f. ; 30 cm

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

Inclui referências bibliográficas.

1. Informática – Teses. 2. Engenharia de serviços de telecomunicações. 3. Qualidade de serviço. 4. Linguagens de descrição de arquitetura. 5. Linguagens de domínio específico. 6. Redes programáveis. 7. Componentes de software. I. Soares, Luiz Fernando Gomes. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. III. Título.

CDD: 004

*À minha mulher Luciana,  
grande amor da minha vida.*

*À minha filha Ana Carolina,  
que me mostra todos os dias  
que é preciso saber viver.*

*A meus pais amados Sebastião e Maria Antonieta.*

*A meus sogros queridos Antônio Augusto e Solange.*

## Agradecimentos

São João Batista de La Salle, precursor do ensino popular generalizado, repetia: “A ciência infla, só a caridade edifica”. Talvez São João teria repensado sua famosa frase se tivesse conhecido meu orientador, Professor Luiz Fernando Gomes Soares. Com sua refinadíssima didática, seu profundo conhecimento em Ciências e, acima de tudo, sua terna amizade e simplicidade, o “Dotô LF” cativa todos, sejam os seus orientandos ou os menores carentes do CIDS de Vargem Grande, que ele ajudou a fundar. “Dotô”, seus ensinamentos, tanto no plano profissional quanto pessoal, são heranças que espero carregar comigo durante toda minha vida. A você um grandíssimo “Obrigado!”.

Ao Grêmio Científico e Recreativo Escola de Redes, Hipermídia e Samba Acadêmicos do TeleMídia, um grande abraço. Foram oito anos de convivência com pessoas que fizeram e fazem do TeleMídia um lugar especial. Almoço com os chefes, café sem açúcar da Dona Carmen, momentos que estão guardados no fundo do meu coração. Corro o risco de ser injusto com outros TeleMidianos pelos quais nutro grande amizade, mas não posso deixar de exprimir minha admiração e carinho especiais por Sérgio Colcher, Rogério Ferreira Rodrigues e Débora Cristina Muchaluat-Saade. Agradecimentos também a Marcelo Ferreira Moreno e Carlos de Salles Soares Neto, cuja ajuda foi fundamental no fechamento deste trabalho. Valeu por tudo gente!

A todo o Departamento de Informática da PUC-Rio, seus professores e funcionários, agradeço à formação de altíssimo nível que proporcionam a seus alunos.

Agradeço também à Diretoria e às Coordenações de Sistemas e Redes e de Ciência da Computação do LNCC pelo incentivo necessário para que eu pudesse terminar em tempo esta tese.

Por fim, agradeço ao CNPq/MCT, CAPES/MEC, FAPERJ e PUC-Rio pelo suporte financeiro dado em diferentes fases deste trabalho.

## Resumo

Gomes, Antônio Tadeu Azevedo. **LindaX**. Rio de Janeiro, 2005. 194p. Tese de Doutorado - Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

No cenário atual do setor de telecomunicações, percebe-se uma tendência crescente em direção ao uso de sistemas de comunicação que permitam a criação rápida e de baixo custo de serviços. Na busca por arquiteturas de rede que respondam a essa tendência, vários grupos têm centrado seus esforços em pesquisas na área de ‘redes programáveis’. O surgimento da tecnologia de ‘processamento de rede’ no mercado de equipamentos de telecomunicações abriu ainda maior espaço para pesquisas nessa área. Nesse contexto, é imprescindível que os processos de criação de serviços sejam bem estruturados e, o quanto possível, sistemáticos. Esta tese, inserida em um projeto desenvolvido no Laboratório TeleMídia da PUC-Rio, adota uma abordagem de criação de serviços em que técnicas de Arquitetura de Software e de Desenvolvimento Baseado em Componentes são aplicadas consistentemente e de modo ubíquo, desde especificações de alto nível de serviços até a implementação de software básico em unidades programáveis de processamento de rede. Os objetivos principais são expressar a adaptabilidade de múltiplos aspectos nesses serviços e, simultaneamente, reduzir a sobrecarga cognitiva em projetistas e programadores, decorrente dessa multiplicidade de aspectos. Para isso, foi desenvolvida uma linguagem de especificação baseada em XML, chamada ‘LindaX’, que permite descrever arquiteturalmente diversos aspectos de sistemas de comunicação – por meio de um arcabouço sintático único para DSLs – e restrições de adaptação em cada aspecto particular – por meio de estruturas de estilos arquiteturais. Complementando o trabalho, um conjunto de ferramentas de manipulação de descrições arquiteturais em LindaX é definido. Essas ferramentas permitem o refinamento para diferentes linguagens formais ou a síntese de configurações e mecanismos de controle de adaptações para diversas plataformas.

## Palavras-chave

Engenharia de Serviços de Telecomunicações; Qualidade de Serviço; Linguagens de Descrição de Arquitetura; Linguagens de Domínio Específico; Redes Programáveis; Componentes de Software.

## **Abstract**

Gomes, Antônio Tadeu Azevedo. **LindaX**. Rio de Janeiro, 2005. 194p. D.Sc. Thesis - Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

It is increasingly noticeable, in the current telecommunications market, a trend towards using communication systems that allow rapid and cheap deployment of new services. In pursuit of network architectures that keep up with such trend, significant research has been carried out on ‘programmable networks’. This field is set to gain further impetus from developments in ‘network processor’-based equipment. In this context, it is crucial that service creation processes be well structured and, as far as possible, systematic. This thesis, which is part of an ongoing project at the TeleMídia Laboratory, PUC-Rio, adopts a service creation approach in which techniques from Software Architecture and Component-Based Software Development are uniformly and ubiquitously applied at all levels of a communication system, ranging from high-level service specifications to low-level software implementation running in network processors. The main aim is to express adaptability in cross-cutting service aspects and, meanwhile, cut down on the cognitive overhead usually imposed upon designers and programmers due to such multiplicity of concerns. For the sake of the aforementioned aim, an XML-based specification language, called ‘LindaX’, has been developed. Such language allows various system aspects to be architecturally described – by means of a single syntactic framework for DSLs – as well as adaptable in a constrained way – through style structures. Complementing the work, a toolset for handling LindaX architecture descriptions has been defined, which allows their refinement to different formal languages or their synthesis onto system configurations and adaptation controlling mechanisms for diverse platforms.

## **Keywords**

Telecommunications Services Engineering; Quality of Service; Architecture Description Languages; Domain-Specific Languages; Programmable Networks; Software Components.

# Sumário

1	Introdução	17
1.1.	Posicionamento da tese	20
1.1.1.	Técnicas de descrição formal	20
1.1.2.	Linguagens de descrição de arquitetura	21
1.1.3.	<i>Frameworks</i> , <i>patterns</i> e linguagens de domínio específico	23
1.1.4.	Comentários gerais	24
1.2.	Objetivos da tese	25
1.2.1.	Estrutura e uso de LindaX	28
1.3.	Organização da tese	31
2	Ambientes de Criação de Serviços	32
2.1.	Ciclo de vida de serviços	33
2.2.	Classificação dos mecanismos de adaptação	34
2.3.	Requisitos para ambientes de criação de serviços	35
2.4.	Modelo de composição de serviços	37
2.4.1.	Metasserviços	43
2.4.2.	<i>Frameworks</i> de serviços	45
2.5.	Sumário	47
3	A Linguagem LindaX	48
3.1.	Estrutura de propriedades	50
3.2.	Sistema de tipos	51
3.3.	Descrições de arquitetura	56
3.4.	Visão de recursos	60
3.5.	Estilos	62
3.5.1.	Lógica de predicados	64
3.6.	Sumário	68
4	Especializações de LindaX	69



4.1. A DSL LindaRouter	70
4.1.1. Arquitetura geral de um roteador	70
4.1.2. Estilos de LindaRouter	73
4.1.3. Semântica de LindaRouter	77
4.2. A DSL LindaQoS	81
4.2.1. Arquiteturas de provisão de QoS	81
4.2.2. Estilos de LindaQoS	83
4.2.3. Semântica de LindaQoS	87
4.3. Sumário	91
5 Refinamento e Síntese de Especificações no Ambiente LindaStudio	92
5.1. Visão geral	92
5.2. Arquitetura de software do ambiente LindaStudio	95
5.3. A ferramenta Translator	98
5.3.1. Estudo de caso	99
5.3.1.1. A linguagem Wright	100
5.3.1.2. Refinamento de DSLs LindaX para Wright	103
5.4. <i>Framework</i> para gerência de adaptações	106
5.4.1. O controlador genérico de adaptações	107
5.4.2. O controlador transacional	108
5.4.3. O configurador	109
5.5. A ferramenta Generator	110
5.5.1. Estudo de caso	110
5.5.1.1. O <i>toolkit</i> NetKit e a plataforma OpenCOM	111
5.5.1.2. <i>Frameworks</i> de componentes no NetKit	114
5.5.1.3. Aplicação do <i>framework</i> para gerência de adaptações no NetKit	115
5.5.1.4. Gerência de adaptações	116
5.5.1.5. Adaptações transacionais	121
5.5.1.6. Configurações programadas	123
5.5.1.7. Implementação em NPUs	125
5.6. Sumário	128
6 Trabalhos Relacionados	130
6.1. Aster	130

6.2. XelHa	134
6.3. Fractal	139
6.4. FORMAware	142
6.5. MOTEL	143
6.6. Arquitetura TOSCA	144
6.7. NetScript	145
6.8. Análise comparativa	147
7 Conclusões	150
7.1. Sumário de contribuições	152
7.2. Resultados decorrentes	152
7.3. Trabalhos futuros	153
8 Referências	157
9 Apêndice A	165
9.1. Implementação do LindaStudio	165
9.1.1. Interface de Generator	167
9.1.2. Interface de Translator	167
9.1.3. Interface de Expander	167
9.2. Instanciação do framework para gerência de adaptações no OpenCOM	168
9.2.1. Interfaces do GAC	169
9.2.2. Interface do TC	169
9.2.3. Interface do CG	169
10 Apêndice B	170
10.1. Estilo LowestNQoS	170
10.2. Estilo CentralizedNQoS	171
10.3. Estilo DistributedNQoS	173
10.4. Estilo HierarchyNQoS	176
11 Apêndice C	177
11.1. Esquema lindaxprop	177
11.2. Esquema lindaxtyp	179

11.3. Esquema lindaxcnf	182
11.4. Esquema lindaxres	184
11.5. Esquema lindaxsty	185
11.6. Esquema lindaxfolpredicate	187

# Índice de Figuras

Figura 1.1. Processo de criação e uso de DSLs em LindaX.	30
Figura 2.1. Abstrações básicas do SCM.	37
Figura 2.2. Composições arquiteturais no SCM.	39
Figura 2.3. Abstração de pipe no SCM.	39
Figura 2.4. Pipes compostos no SCM.	40
Figura 2.5. Árvore de recursos virtuais no SCM. As folhas da árvore identificam parcelas de recurso sendo utilizadas por determinados consumidores.	41
Figura 2.6. Tarefas no SCM.	42
Figura 2.7. Relacionamento entre abstrações do modelo SCM.	43
Figura 2.8. Metasserviços no SCM.	44
Figura 2.9. Torre de metasserviços no SCM.	45
Figura 2.10. Provisão de QoS fim-a-fim (extraída de (Aurrecoechea et al., 1998)).	46
Figura 3.1. Exemplo de declaração em XML de uma propriedade LindaX.	50
Figura 3.2. Gramática da notação sem tags do sistema de tipos de LindaX.	53
Figura 3.3. Exemplo de descrição em XML e na notação sem tags em LindaX.	54
Figura 3.4. Definição de tipos de entidades de um módulo IP em LindaX.	54
Figura 3.5. Diagrama SCM do módulo IP de um roteador.	55
Figura 3.6. Gramática da notação sem tags para descrições de arquitetura em LindaX.	58
Figura 3.7. Descrição de arquitetura baseada em estilo.	58
Figura 3.8. Gramática da notação sem tags para descrições de recursos em LindaX.	61
Figura 3.9. Descrição de atividades de computação e comunicação em LindaX.	62

Figura 3.10. Gramática da notação sem tags para descrições de estilos em LindaX.	63
Figura 3.11. Descrição de estilo em LindaX.	63
Figura 3.12. Descrição de sub-estilo em LindaX.	64
Figura 3.13. Gramática da notação sem tags para a lógica FOL.	67
Figura 3.14. Identificação dos elementos da lógica FOL.	68
Figura 4.1. Arquitetura geral de um roteador.	71
Figura 4.2. Estrutura de herança de estilos de roteador em LindaRouter.	73
Figura 4.3. Combinações de interfaces push e pull.	74
Figura 4.4. Estilo GenericForwarder.	75
Figura 4.5. Estilo PCBasedForwarder.	76
Figura 4.6. Estilo NPUBasedForwarder.	77
Figura 4.7. Configuração de encaminhamento IP em um roteador.	78
Figura 4.8. Descrição de arquitetura em LindaRouter de um roteador baseado em NPU.	80
Figura 4.9. Notação diagramática dos estilos de negociação em LindaQoS.	83
Figura 4.10. Estilo CentralizedNQoS.	84
Figura 4.11. Estilo DistributedNQoS.	86
Figura 4.12. Configuração de uma arquitetura de provisão de QoS distribuída.	87
Figura 4.13. Descrição de templates para sistemas finais e roteadores em LindaQoS.	89
Figura 4.14. Descrição de arquitetura em LindaQoS de um sistema de sinalização RSVP.	90
Figura 5.1. Visão geral do ambiente LindaStudio.	95
Figura 5.2. Arquitetura de software do ambiente LindaStudio.	97
Figura 5.3. Estrutura da ferramenta Translator.	99
Figura 5.4. Especificação arquitetural estática em Wright.	101
Figura 5.5. Especificação de componente composto em Wright.	102
Figura 5.6. Refinamento de descrição na DSL LindaRouter para Wright.	105
Figura 5.7. Planificação de configurações LindaX.	106
Figura 5.8. Framework para gerência de adaptações.	107

Figura 5.9. Estrutura da ferramenta Generator.	110
Figura 5.10. O modelo de componentes OpenCOM.	112
Figura 5.11. Associações “não-locais” no OpenCOM. A especificação do protocolo de comunicação utilizado entre X e Y não é englobada pelo modelo.	114
Figura 5.12. Exemplo de script de verificação em Lua.	118
Figura 5.13. O componente GAC no OpenCOM.	121
Figura 5.14. Serviço de adaptação transacional no OpenCOM.	122
Figura 5.15. Implementação do CG no OpenCOM.	123
Figura 5.16. Descrição de configuração em Lua	124
Figura 5.17. Implementação do framework para gerência de adaptações no IXP1200.	127
Figura 6.1. Especificação arquitetural de aplicações em Aster.	131
Figura 6.2. Especificação do barramento AsterBus.	132
Figura 6.3. Árvore de propriedades no barramento AsterBus. Os predicados ilustrados são simplificações daqueles apresentados por Issarny e Bidan (1996).	133
Figura 6.4. Exemplo de especificação de sistema em XelHa.	135
Figura 6.5. Exemplo de especificação de tarefas em XelHa.	137
Figura 6.6. Exemplo de especificação de serviço em XelHa.	138
Figura 6.7. Processo de análise de especificações em XelHa.	139
Figura 6.8. Configuração de componentes Fractal.	140
Figura 6.9. Exemplo de especificação na ADL Fractal. Os atributos signature e content_class são preenchidos de acordo com a linguagem de programação em uso.	141
Figura 6.10. A ferramenta WrapperGenerator de FORMAware.	143
Figura 6.11. Framework de monitorização de propriedades da ferramenta MOTEL.	144
Figura 6.12. Composição de protocolos em NetScript.	147
Figura 9.1. Drivers das ferramentas Translator e Generator.	166
Figura 9.2. Captura de tela do ambiente LindaStudio.	166

# Índice de Tabelas

Tabela 2.1. Ciclo de vida de serviços tradicional.	34
Tabela 2.2. Classificação das estratégias de adaptação	35
Tabela 3.1. Tipos de valores para propriedades em LindaX	50
Tabela 5.1. Convenção de refinamento de DSLs LindaX para descrições Wright.	103
Tabela 5.2. Mapeamento das estruturas de estilos LindaX em trechos de código em Lua.	120
Tabela 5.3. Footprint de memória dos componentes no OpenCOM.	128
Tabela 9.1. Número de linhas de código de cada componente do framework.	168

## Lista de Acrônimos

ADL – Architecture Description Language  
API – Application Programming Interface  
ASIC – Application Specific Integrated Circuit  
BL – Base Level  
CBSD – Component Based Software Development  
CF – Component Framework  
CG – ConfiGurator  
COPS – Common Open Policy Service  
CORBA – Common Object Request Broker Architecture  
DSL – Domain Specific Language  
EBNF – Extended Backus Naur Form  
FDT – Formal Description Language  
FOL – First Order Logic  
GAC – Generic Adaptation Controller  
GUID – General Universal Identifier  
IDL – Interface Definition Language  
IP – Internet Protocol  
LWP – LightWeight Process  
MIL – Module Interconnection Language  
ML – Meta Level  
NPU – Network Processing Unit  
ODP – Open Distributed Processing  
OOSD – Object Oriented Software Development  
OSI – Open Systems Interconnection  
OSPF – Open Shortest Path First  
PCI – Peripheral Component Interconnect  
QoS – Quality of Service  
RISC – Reduced Instruction Set Computer  
RSVP – Resource reSerVation Protocol  
SCM – Service Composition Model  
SS7 – Signalling System no. 7  
TC – Transaction Controller  
UML – Unified Modelling Language  
WWW – World Wide Web  
XML – eXtensible Markup Language