DEISYMAR BOTEGA TAVARES

PROCEDIMENTO DE ANÁLISE PARA VALIDAÇÃO DE DIAGRAMA DE CLASSES DE DOMÍNIO BASEADO EM ANÁLISE ONTOLÓGICA

Dissertação apresentada à Universidade Federal de Viçosa, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, para obtenção do título de *Magister Scientiae*.

VIÇOSA

MINAS GERAIS - BRASIL

2008

DEISYMAR BOTEGA TAVARES

PROCEDIMENTO DE ANÁLISE PARA VALIDAÇÃO DE DIAGRAMA DE CLASSES DE DOMÍNIO BASEADO EM ANÁLISE ONTOLÓGICA.

Dissertação apresentada à Universidade Federal de Viçosa, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, para obtenção do título de *Magister Scientiae*.

APROVADA: 15 de Agosto de 2008.

Prof. José Luís Braga
(Co-orientador)

Prof. Mauro Nacif Rocha
(Co-orientador)

Prof. Jugurta Lisboa Filho

Prof. Ricardo de Almeida Falbo

Prof. Alcione de Paiva Oliveira
(Orientador)

Dedicatória

Dedico em especial esta conquista a meu pai que teve a sabedoria de entender que um precioso bem a ser construído para seus filhos seria o conhecimento. A você toda minha gratidão e meu amor. Obrigada meu pai!

Agradecimentos

- Agradeço e bendigo a Deus, por ter me dado sabedoria, discernimento e perseverança para desempenhar tantas tarefas nesta etapa da minha vida.

- Agradeço o meu esposo Luiz por ser, simplesmente, meu companheiro presente e paciente.

- Agradeço à minha filha Gabriela, por me alegrar a cada dia, por me aconselhar com suas palavras de sabedoria: "Calma mamãe, você vai consegui, eu também não consegui fazer minha tarefinha, mas depois eu consegui."

- À minha mãe Acione que, junto de meu pai, foram meus primeiros mestres, mestres do amor, da honestidade, responsabilidade e perseverança.

- À minha irmã Taciana, meu irmão Valdir e meu cunhado Cristiano que mesmo distantes fisicamente, sempre se fizeram presentes, me incentivando a cada dia.

- À Dª Ortência por se fazer presente na vida da Gabriela, principalmente nos momentos de minha ausência. Foi muito importante e confortante pra mim.

- Ao meu orientador Alcione, pela presteza, paciência e simplicidade com que conduziu este trabalho. Quantas vezes, com sua experiência e maturidade me indicou o caminho.

- Ao meu grande mestre José Luís Braga, por quem tenho grande admiração, o meu muito obrigado pelos ensinamentos e amparo.

- Agradeço ao Machado, que não mediu esforços para viabilizar os meus estudos. Foi muito importante toda a sua compreensão e apoio neste período. Certamente

você não só desempenhou seu papel de coordenador, você foi além. Obrigada, muito obrigada.

- A meu grande amigo Mauro, pelo carinho, pela proteção, pelo incentivo. E claro, pelas caronas, infinitas caronas.
- Ao Diego, que de quase aluno tornou-se colega de mestrado e amigo. Sem seu apoio certamene o caminho teria sido muito mais árduo.
- Ao Marcelo Balbino, pela ajuda nos testes e pelo interesse na minha pesquisa.
- À minha amiga Renata, pela alegria que me proporcionou em ler a minha dissertação.
- Ao Ed Carlo, à Nunziata e à Odilaine, amigos presentes, sempre.
- À diretoria de ciências exatas do Unileste-MG, que na pessoa da Maria Cândida permitiu e viabilizou a realização do meu mestrado.
- Ao Unileste-MG pelo apoio financeiro, sem o qual não seria possível realizar tal capacitação.
- À todos os professores do DPI que direta ou indiretamente contribuíram para minha capacitação.
- À Universidade Federal de Viçosa.

Biografia

Deisymar Botega Tavares, filha de Antônio Pedroso Tavares e Acione Botega Tavares, brasileira nascida em 22 de julho de 1974 no município de Nepomuceno, no Estado de Minas Gerais.

Ingressou no curso de graduação em Ciência da Computação na Universidade Federal de Viçosa em 1994, obtendo o título de bacharel em informática no ano de 1998. Em 1999 começou a lecionar no Centro Universitário do Leste de Minas Gerais. Atualmente faz parte do corpo docente do curso de Sistemas de Informação-Computação da referida instituição, lecionando também nos cursos de engenharia. De 2000 a 2001 realizou o curso de especialização pela Universidade Federal de Minas Gerais. Iniciou em 2006 o curso de mestrado em Ciência da Computação na Universidade Federal de Viçosa - UFV defendendo a dissertação em agosto de 2008.

Conteúdo

Lista de Tabelas x

Lista de Figuras xi

Resumo xiii

Abstract xv

1 Introdução **1**

1.1 Objetivos do Trabalho 5

1.2 Organização deste Documento 6

2 Revisão Bibliográfica **8**

2.1 Modelagem Conceitual Utilizando Diagramas de Classes UML 8

2.2 Análise Ontológica 11

2.2.1 Ontologia de Fundamentação Unificada - UFO 12

2.2.2 Ontologia Formal de Propriedades 15

2.3 O Uso da Análise Ontológica na Modelagem Conceitual de Sistemas . . 19

2.3.1 VERONTO:Técnica de Validação de Diagramas de Classe por meio de Propriedades Ontológicas 21

2.3.2 OntoUML: perfil UML ontologicamente bem formado para modelos conceituais estruturais 26

3 OntoCon - Técnica de Validação de Diagrama de Classes de Domínio Baseado em Modelagem Ontológica **36**

3.1 Análise comparativa da técnica VERONTO e do Perfil OntoUML para modelos conceituais UML 37

3.2 OntoCon - Proposta de combinação da técnica VERONTO com o Perfil OntoUML para Modelos Conceituais UML 43

4 PrOntoCon - Procedimento de Análise para Validação de Diagrama de Classes de Domínio Baseado em Modelagem Ontológica 52

4.1 Primeira Fase: Identificação de Estereótipos 56

4.2 Segunda Fase: Verificação Hierárquica para Relacionamentos de Generalização/Especialização 64

4.3 Terceira Fase: Aplicação de Padrão de Projeto para Modelagem de Papéis 70

4.4 Quarta Fase: Verificação de Construtores UML 77

5 Estudo de Casos 79

5.1 Domínio de Controle de Atividades de Estagiários em Psicologia 79

5.1.1 Aplicação da Fase 1 do PrOntoCon 81

5.1.2 Aplicação da Fase 2 do PrOntoCon 88

5.1.3 Aplicação das Fases 3 e 4 do PrOntoCon 89

5.2 Domínio de Controle de Contratos Financeiros para Cursos de Pós-Graduação 92

5.2.1 Aplicação da Fase 1 do PrOntoCon 93

5.2.2 Aplicação da Fase 2 do PrOntoCon 101

5.2.3 Aplicação das Fases 3 e 4 do PrOntoCon 108

6 Conclusões 112

6.1 Trabalhos Futuros 115

A O procedimento PrOntoCon - Fase 1 116

B O procedimento PrOntoCon - Fase 2 125

C O procedimento PrOntoCon - Fase 3 135

D O procedimento PrOntoCon - Fase 4 143

E SPEM - Software Process Engineering Metamodel 145

Referências Bibliográficas 121

Lista de Tabelas

2.1 Propriedades formadas a partir da combinação das meta-propriedades. 20

2.2 Associação das meta-propriedades e regras que estas impõem aos relacionamentos hierárquicos aos construtores de modelagem conceitual 22

2.3 Associação das meta-propriedades relacionadas a unidade e identidade aos construtores de modelagem conceitual UML, e as restrições que essas impõem a relacionamentos "parte-todo". 27

2.4 Um perfil UML ontologicamente bem formado para diagramas de classe UML- OntoUML. 29

3.1 Comparação da técnica VERONTO Villela (2004)com o Perfil OntoUML Guizzardi (2005). Características comparadas: meta-propriedades associadas e construtor UML. 39

3.2 Relação entre os estereótipos da OntoCon e as Propriedades definidas por Guarino and Welty (2000a). Mapeamento dos estereótipos a construtores UML, baseado em Villela (2004) e Guizzardi (2005). 45

3.3 Restrições taxonômicas da técnica OntoCon, herdadas da técnica VERONTO (Villela, 2004) e do Perfil OntoUML para Modelos Conceituais UML (Guizzardi, 2005). 47

3.4 Problemas detectados na técnica VERONTO e no Perfil OntoUML e melhorias proporcionadas pela OntoCon. 50

4.1 Possibilidades e restrições quanto ao uso de construtores UML 78

5.1 Análise Ontológica da Classe Cliente 94

5.2 Análise Ontológica da Classe Pessoa Física 95
5.3 Análise Ontológica da Classe Pessoa Jurídica 95
5.4 Análise Ontológica da Classe Contrato Jurídico 96
5.5 Análise Ontológica da Classe Contrato Físico 97
5.6 Análise Ontológica da Classe Aluno 98
5.7 Análise Ontológica da Classe Tipo de Aluno 99
5.8 Análise Ontológica da Classe Curso 99
5.9 Análise Ontológica da Classe Turma 100
5.10 Análise Ontológica da Classe Planos 100
5.11 Análise Ontológica da Classe Planos Físicos 101
5.12 Análise Ontológica da Classe Planos Jurídicos 102
5.13 Classificação das classes «tipo»/«quase-tipo» nas categorias UFO . . . 102

Lista de Figuras

2.1 Parte da Ontologia Fundacional Unificada A (UFO-A). Universais e Indivíduos. Fonte: Guizzardi et al. (2008). 13

2.2 Fragmento da Ontologia Fundacional Unificada B (UFO-B). Objetos e Eventos. Fonte: Guizzardi et al. (2008). 14

2.3 Parte da Ontologia Fundacional Unificada C (UFO-C). Ações, Agentes e Substanciais Inanimados. Fonte: Guizzardi et al. (2008). 15

2.4 Problemas de modelagem de papéis com múltiplos tipos permitidos. Fonte: Guizzardi (2005). 33

2.5 Padrão de Projeto Ontológico para Modelagem de Papéis. Fonte: Guizzardi (2005). 33

2.6 Versão ontologicamente correta do modelo da Figura 2.4 obtido com a aplicação do padrão de projeto. Fonte: Guizzardi (2005). 34

2.7 Aplicação do Padrão de Projeto de Guizzardi (2005), para casos onde existem múltiplos supertipos «kind» para uma classe esterotipada como «role». 35

4.1 Fases do Procedimento de Validação de Diagramas de Classe - PrOntoCon 54

4.2 Fluxograma representativo das fases e subfases do procedimento PrOntoCon, numa visão hierárquica. 55

4.3 Árvore de decisão para identificação de estereótipo. 57

4.4 Questão 2 - Rigidez. Pergunta utilizada no procedimento com o objetivo de descobrir se a classe é ou não uma classe rígida. 60

4.5 Diagrama de Atividades da Fase 1: Identificação de Estereótipos 62

4.6 Diagrama de Atividades da Fase 2: Verificação Hierárquica dos relacionamentos de Generalização/Especialização. 65
4.7 Diagrama de Atividade da SubFase da Fase 2: Checar Conformidade das Restrições Hierárquicas. 67
4.8 Diagrama de Atividade da SubFase da Fase 2: Checar Hierárquicas Existentes. 68
4.9 Diagrama de Atividades - Aplicação Padrão de Projeto - Caso Hierarquia Incorreta. 71
4.10 Transformações ocorridas na aplicação do Padrão de Projeto para o caso de hierarquia incorreta. Fonte: adaptada Guizzardi et al. (2004b). 72
4.11 Um exemplo de aplicação do Padrão de Projeto para o Caso de Hierarquia Incorreta. Fonte: adaptada de Guizzardi et al. (2004b). 73
4.12 Diagrama de Atividade - Aplicação do Padrão de Projeto - Caso Herança Múltipla. 74
4.13 Um exemplo de aplicação do Padrão de Projeto para o Caso de Herança Múltipla. 76
4.14 (a) Resultado final da aplicação do Padrão de Projeto para os dois casos: hierarquia incorreta e herança múltipla de «tipo» (Guizzardi, 2005). (b) Exemplo adaptado de Guizzardi et al. (2004b). 76
4.15 Diagrama de Atividades da Fase 4 - Verificação de Construtores UML. 77

5.1 Diagrama de Classes UML original do domínio que representa o controle das atividades dos estagiários do curso de Psicologia do Unileste-MG. . 80
5.2 Diagrama parcialmente estereotipado após término da fase 1 do PrOntoCon. 87
5.3 Diagrama de Classes do domínio de Controle de Atividades de Estagiários de Psicologia validado pelo procedimento PrOntoCon. 90
5.4 Diagrama de Classes UML original do domínio que representa o controle dos contratos financeiros para cursos de Pós-Graduação do Unileste-MG. 93
5.5 Diagrama de Classes parcialmente estereotipado após execução da fase 1 do PrOntoCon 103

5.6 Aplicação do Padrão de Projeto - Caso Hierarquia Incorreta no domínio de Controle de Contratos Financeiros de Cursos de Pós-Graduação do Unileste-MG. 106
5.7 Diagrama de Classes estereotipado após término da subatividade Checar Hierarquias existentes. 107
5.8 Diagrama de Classes do domínio de Controle de Contratos Financeiros para Cursos de Pós-Graduação validado pelo procedimento PrOntoCon. 109

A.1 Fases do Procedimento de Validação de Diagramas de Classe UML - PrOntoCon. 117
A.2 Diagrama de Atividades da Fase 1: Identificação de Estereótipos 117
A.3 Guia 1.1 - Atividade Percorrer Árvore de Identificação de Estereótipo . 118
A.4 Árvore de caminhamento para identificação dos estereótipos. 118
A.5 Questão 1 - Possui Identidade. Pergunta utilizada no procedimento com o objetivo de descobrir se a classe possui ou não uma identidade. . 119
A.6 Questão 2 - Rigidez. Pergunta utilizada no procedimento com o objetivo de descobrir se a classe é ou não uma classe rígida. 120
A.7 Questão 3 - Dependência. Pergunta utilizada no procedimento com o objetivo de descobrir se a classe é ou não uma classe que possui dependência externa em relação a outra classe do modelo. 121
A.8 Questão 4 - Anti-Rigidez. Pergunta utilizada no procedimento com o objetivo de descobrir se a classe é ou não uma classe anti-rígida. 122
A.9 Guia 1.4 - Instruções para Anotações da Classificação. Guia contendo instruções sobre como o estereótipo diagnosticado para a classe será acrescentado ao diagrama original. 122
A.10 Guia 1.5 - Retirar Classes Classificadas como Atribuição. Guia contendo instruções sobre como retirar uma classse que foi classificada como Atribuição. 123
A.11 Guia 1.2 - Tabela de Histórico de Classificação de Classes e Tabela Com Classificação das Classes «tipo»/«quase-tipo». Apresentação do layout de tais tabelas. 123

A.12 Guia 1.3 - Classificação das Classes «tipo»/«quase-tipo». Orientações de como se proceder para agrupar as classes «tipo»/«quase-tipo». . . . 124

B.1 Diagrama de atividade da Fase 2 do procedimento PrOntoCon - Verificação Hierárquica. 126

B.2 Guia 2.1 - Definição Hierárquica das Classes <tipo»/«quase-tipo». Guia para orientar o modelador na classificação final das classes até então classificadas como «tipo»/«quase-tipo». 127

B.3 Diagrama de Atividade da SubFase da Fase 2: Checar Conformidade das Restrições Hierárquicas. Objetiva conduzir o modelador a (i) checar se aos relacionamentos hierárquicos existentes estão corretos e (ii) a avaliar as hierarquias obrigatórias. 128

B.4 Guia 2.8 - Orientações sobre Atividade Checar Hierarquias Existentes. . 128

B.5 Diagrama de Atividade da SubFase da Fase 2: Checar Hierárquicas Existentes. Tem por objetivo conduzir o modelador na verificação das hierarquias já existentes, detectando possíveis erros. 129

B.6 Guia 2.3 - Supertipos e Subtipos Possíveis para Estereótipos. Utilizado na atividade de Verificar Supertipo e Subtipos permitindo detectar erros hierárquicos. 130

B.7 Continuação: Guia 2.3 - Supertipos e Subtipos Possíveis para Estereótipos. 131

B.8 Guia 2.7 - Condução à Aplicação do Padrão de Projeto. 132

B.9 Guia 2.6 - Condução à Reavaliação dos Estereótipos. 132

B.10 Guia 2.4 - Checar Existência de Hierarquias Obrigatórias. Orientações que conduzem o modelador na análise de hierarquias obrigatórias no diagrama. 133

B.11 Guia 2.5 - Tabela de Controle das Hierarquias Obrigatórias. Apresentação do layout de tal tabela. 134

B.12 Guia 2.2 - Checar Conformidade com Restrições de Relacionamento. Orientações sobre análise de relacionamentos envolvendo classes «papel material», «papel formal» ou «fase». 134

C.1 Diagrama de Atividade-Aplicação do Padrão de Projeto: Caso Hierarquia Incorreta - Fase 3. 136
C.2 Guia 3.1 - Aplicar Padrão de Projeto: Caso Hierárquica Incorreta. Orientações para aplicação do padrão de projeto. 137
C.3 Guia 3.2 - Verificar Relacionamento Obrigatório no Padrão de Projeto. Orientações para verificação de relacionamentos obrigatórios. 138
C.4 Diagrama de atividades-Aplicação do Padrão de Projeto: Caso Herança Múltipla. Fase 3. 139
C.5 Guia 3.3 - Checar Herança Múltipla em Classes «papel material». . . . 140
C.6 Guia 3.4 - Aplicar Padrão de Projeto: Caso Herança Múltipla de classe «papel material» com mais de uma superclasse «tipo». Orientações para a aplicação do padrão de projeto. 141
C.7 Guia 3.5 - Verificar Relacionamento Obrigatório no Padrão de Projeto. 142

D.1 Diagrama de Atividade Fase 4: Verificação de Construtores. 144
D.2 Guia 4.1 - Tabela de Possibilidades de Construtores. Orientações para checar uso correto dos construtores UML no diagrama de classes. . . . 144

E.1 Principais estereótipos com respectivos ícones do SPEM. Fonte: Lahoz (2004). 147
E.2 Pacote representante da *Discipline* Elicidação. Um Processo pode ser composto por várias *Disciplines*. Fonte: Gengivir et al. (2003). 149
E.3 Diagrama de Atividades do WorkDefinition Compreender Domínio da Aplicação. Fonte: Gengivir et al. (2003). 150

Resumo

TAVARES, Deisymar Botega, M.Sc., Universidade Federal de Viçosa, agosto de 2008. **Procedimento de Análise para Validação de Diagrama de Classes de Domínio Baseado em Análise Ontológica**. Orientador: Alcione de Paiva Oliveira. Co-orientadores: José Luís Braga e Mauro Nacif Rocha.

Com o objetivo de representar o domínio dos sistemas a serem desenvolvidos, a Engenharia de Software faz uso da modelagem conceitual. Um dos desafios existentes nesta área é gerar modelos cada vez mais fidedignos ao domínio do problema, ou seja, com uma representação mais próxima da realidade. Técnicas de modelagem ontológica têm sido criadas com o objetivo de validar modelos conceituais expressos por meio de diagramas de classe UML. O presente trabalho faz uma análise cuidadosa da Técnica VERONTO e do Perfil OntoUML. Ambos utilizam a modelagem ontológica como forma de auxílio na modelagem conceitual. Com base nesta análise estruturou-se uma nova técnica composta de características herdadas da VERONTO e do Perfil OntoUML. Tal técnica, denominada de OntoCon, foi organizada com o objetivo de prover informações devidamente estruturadas, principalmente no que diz respeito a regras e restrições utilizadas para validar relacionamentos de generalização e especialização em diagramas de classe UML. Porém, técnicas que utilizam modelagem ontológica são de difícil aplicação uma vez que fazem uso de conceitos e análises filosóficas; dificilmente dominadas por modeladores. Objetivando tornar viável o uso da OntoCon construiu-se um procedimento de uso da mesma. Intitulado de PrOntoCon, o procedimento de uso da técnica OntoCon guia o modelador na validação de diagramas de classes UML já existentes, corrigindo erros de hierarquias ou indicando

a falta delas. Também faz parte do procedimento resolver problemas de modelagem de papéis através do uso de um padrão de projeto herdado do Perfil OntoUML. Após a validação de diversos diagramas de classes UML com o uso do PrOntoCon, obteve-se diagramas mais escaláveis, mais estáveis, com maior redundância de dados e com maior facilidade de integração. Além de todas essas vantagens, o procedimento PrOntoCon agregou facilidade na aplicação dos conceitos filosóficos existentes na técnica de modelagem ontológica OntoCon. Possibilita-se, então, com tais recursos, a obtenção de sistemas finais de mais fácil manutenibilidade, característica imprescindível para os sistemas atuais.

Abstract

TAVARES, Deisymar Botega, M.Sc., Universidade Federal de Viçosa, August, 2008. **Procedure of Analysis for Validation of Domain Class Diagram based in Ontological Modeling**. Advisor: Alcione de Paiva Oliveira. Co-Advisors: José Luís Braga and Mauro Nacif Rocha.

The Software Engineering field makes use of conceptual modeling aiming to build an abstract model of a domain. The main challenge is to build a reliable model of the domain, i.e., a model that reflects the relevant aspects of the reality. Ontological modeling techniques have been developed with the purpose of validating conceptual models expressed through UML class diagrams. This work makes a careful analysis of the VERONTO technique and the OntoUML profile. Both use the ontological modeling as a way to improve the conceptual modeling. Based on this analysis a new technique that inherits characteristics from VERONTO and OntoUML profile was proposed. This technique, called OntoCon, was organized with the objective of yielding properly structured information, particularly by providing rules and restrictions that can be used to validate generalization/specialization relationships in UML class diagrams. However, ontological modeling techniques are difficult to apply because they make use of philosophical concepts that are hard to be mastered by software engineers. Aiming to make viable the use of OntoCon, a procedure, entitled PrOntoCon, was built. The procedure, based upon the OntoCon technique, guides the modeler, by asking simple questions, trough the process of validating a previously created UML class diagrams. It detects hierarchies' misconceptions or indicates the lack of hierarchical elements. It is part of the procedure to solve problems related to the role modeling through the use of a standard design captured from OntoUML

profile. The UML class diagrams validated with the PrOntoCon procedure showed less data redundancy and seemed more scalable. Thus it is possible to obtain systems that are easier to maintain, an essential feature for the current complex systems.

Capítulo 1

Introdução

As fases iniciais de um processo de desenvolvimento de software são as fases de maior importância de todo o processo, pois é quando se obtém o entendimento do domínio do problema. Um entendimento equivocado de parte do domínio fatalmente incorrerá em um produto final que não representa adequadamente o domínio para o qual foi criado, podendo gerar sérios problemas relacionados com a funcionalidade e manutenibilidade dos sistemas. A modelagem conceitual faz parte dessas fases iniciais e tem por objetivo elaborar uma representação abstrata dos elementos relevantes do domínio que está sendo analisado. Um modelo conceitual incorreto e/ou incompleto pode levar a problemas de projeto, implementação, operação e manutenção dos sistemas.

A utilização de modelos é uma prática comumente usada pelas engenharias. Eles são a base para a construção de diversos artefatos. A Engenharia de Software utiliza essa prática para representar várias características do sistema no decorrer das etapas de seu desenvolvimento. Mas, um modelo, para representar adequadamente o produto que será desenvolvido, tem que possuir algumas características importantes: ter acurácia, ser compreensível, consistente e modificável, isto é, suportar melhor a modificações de requisitos. Ter acurácia significa que os modelos têm que descrever de forma precisa e correta o sistema que está representando. Eles devem ser o mais simples possível para o que se propõem e devem ser fáceis de serem compreendidos. A consistência implica que um modelo não pode expressar coisas que sejam conflitantes umas com as outras. Já a característica de ser modificável é necessária em razão da

mutabilidade dos requisitos dos usuários. Por fim, além de todas essas características um modelo deve ser acima de tudo, fácil de ser elaborado. Para isso é preciso que o modelador tenha orientações que o auxiliem no processo de construção e ou verificação de um determinado tipo de modelo, de forma a garantir que tal modelo terá todas as características necessárias.

Essa última característica é a mais raramente encontrada nos processos de modelagem. Ainda existe uma deficiência de suporte metodológico para auxiliar os usuários de linguagens de modelagem na decisão de como modelar os elementos de um dado domínio. Na maioria das vezes, quando, por exemplo, um modelo de diagrama de classes UML-*Unified Modeling Language* é construído, seus elementos são extraídos e relacionados de forma *ad hoc*.

Uma das formas de auxiliar o processo de modelagem conceitual é fazer uso de uma análise mais detalhada das propriedades dos objetos pertencentes a um domínio com o objetivo de entender sua existência. Como essa análise trata do entendimento da natureza dos elementos que existem em um determinado domínio, denomina-se esta etapa de análise ontológica (Villela, 2004). Modeladores conceituais podem fazer uso deste recurso e assim minimizar erros de modelagem, uma vez que partem de uma compreensão mais aprofundada dos elementos do ambiente.

Pesquisas neste sentido têm sido desenvolvidas para criar mecanismos que, utilizando a modelagem ontológica, possam auxiliar na análise dos elementos pertencentes ao domínio da aplicação a ser modelada. Algumas delas baseiam-se nos trabalhos de Guarino and Welty (2000a,d) e outras nos trabalhos de Bunge (1977) *apud* Wand et al. (1999a) e Bunge (1979) *apud* Wand et al. (1999b).

Guarino and Welty (2000a,d) definiram oito tipos de propriedades baseadas nas meta-propriedades ontológicas de rigidez, identidade, unidade e dependência. Essas meta-propriedades impõem uma série de restrições sobre os relacionamentos que podem existir entre os elementos de uma dada ontologia. O uso correto dessas meta-propriedades e restrições ontológicas na classificação e hierarquização dos elementos de um domínio torna possível validar modelos conceituais expressos por diagramas de classe UML. Villela (2004) apresentou um mapeamento dos tipos de proprie-

dades apresentadas por Guarino and Welty (2000a,d) nos elementos do diagrama de classes da UML. Esse mapeamento, denominado técnica de Verificação Ontológica (VERONTO), pode ser utilizado como uma etapa adicional no processo de desenvolvimento de software, de forma a verificar o modelo conceitual a partir de uma análise ontológica dos elementos do domínio. A aplicação da técnica VERONTO mostrou-se promissora, uma vez que modelos validados na pesquisa tiveram alterações significativas para o contexto e ampliou-se a semântica embutida nos modelos conceituais dos sistemas analisados.

Guizzardi et al. (2004a) apresentaram um Perfil UML para modelagem conceitual e representações ontológicas. Este trabalho foi também fortemente influenciado pelas meta-propriedades definidas por Guarino and Welty (2000a,d). Neste mesmo trabalho, Guizzardi et al. (2004a) propuseram também um Padrão de Projeto, baseado no Perfil UML, para resolver problemas relacionados à modelagem de "papéis", que trata-se de um problema recorrente na literatura. Posteriormente este trabalho foi ampliado permitindo tratar não somente questões relativas a relacionamentos hierárquicos de diagramas de classe UML. Tal perfil UML passou a ser denominado OntoUML: perfil UML ontologicamente bem formado para modelos conceituais estruturais (Guizzardi, 2005).

Apesar de já existirem algumas técnicas de auxílio na análise e construção de diagramas de classe UML, como a VERONTO e o Perfil OntoUML, o que se percebe é que o uso dessas técnicas não é algo trivial. Conseguir extrair os elementos de um domínio e analisá-los conforme suas características ontológicas não é uma tarefa fácil. Os conceitos e formalizações ontológicas existentes nessas técnicas exigem um grande estudo de conceitos filosóficos que envolvem um grande formalismo. Por esse motivo, fatalmente a aplicação das técnicas aqui citadas têm como grande barreira a dificuldade de uso, apesar de promissoras. Para tentar solucionar esse problema é necessário a criação de um procedimento que guie implicitamente o modelador no uso das técnicas de modelagem ontológica: (i) na análise dos elementos do domínio que irão representar as classes do diagrama de classes e (ii) na correta identificação dos devidos relacionamentos entre suas classes.

Acredita-se que a aplicação das técnicas de modelagem ontológica com o auxílio de um procedimento de uso das mesmas será capaz de trazer vários resultados considerados importantes a um sistema, a saber:

- Maior Escalabilidade: o uso da modelagem ontológica na modelagem conceitual gera um modelo mais preparado para o crescimento, uma vez que leva o modelador a explicitar os elementos mais estáveis (rígidos) do domínio. Tais elementos são considerados os elementos base do sistema. Desta forma a adição de novos elementos, durante a vida útil do sistema, tenderá a ser feita de forma mais natural, sem degradar o sistema, pois os novos papéis que surgirem serão facilmente associados a esses elementos base.

- Menor redundância de dados: isso acontece porque com a modelagem ontológica há um cuidado grande em distinguir os elementos que são papéis (*role*) dos elementos que representam os objetos que exercem os papéis, denominados por alguns pesquisadores como tipos naturais (Guarino and Welty, 2000d; Steimann, 2000). Com isso os dados inerentes aos papéis ficam na classe que representa o papel e os dados inerentes aos objetos que exercem os papéis ficam nas classes rígidas. Por exemplo, em um sistema acadêmico o correto é termos uma superclasse Pessoa e as subclasses Estudante, Professor e Funcionário, desta forma, dados como nome e CPF estarão somente na classe Pessoa. Mesmo que uma pessoa esteja exercendo mais de um papel ao mesmo tempo, esses dados não serão replicados. Tal fato aconteceria se as classes Professor, Estudante e Funcionários não fossem especializações de uma única classe.

- Melhora da integração das visões: como cada visão do sistema conterá os elementos básicos criados com o uso da análise ontológica, então existirá pontos comuns de contato entre essas visões, facilitando a integração das mesmas.

- Aumento da estabilidade: com os elementos básicos criados com o uso da modelagem ontológica, o fato de se acoplar novos módulos ao sistema não provocará uma grande alteração da base já existente.

Alguns desses resultados puderam ser constatados neste trabalho pelo do uso do procedimento PrOntoCon - Procedimento de uso da técnica OntoCon na validação de diagramas de classes UML, procedimento este que representa a principal contribuição desta dissertação. O PrOntoCon faz uso da análise ontológica como forma de validação da modelagem conceitual e seu uso propiciou a obtenção de diagramas mais estáveis, escaláveis e com menos redundância de dados.

1.1 Objetivos do Trabalho

O objetivo geral do presente trabalho é propor um procedimento de análise de conceitos que permita guiar o modelador na validação das classes e dos relacionamentos expressos em um diagrama de classes do domínio de uma aplicação, através do uso de técnicas de modelagem ontológica.

Especificamente, objetiva-se:

1. Montar uma técnica que faça uso da análise ontológica no apoio à modelagem conceitual utilizando características da técnica VERONTO (Villela, 2004) e do Perfil OntoUML para diagramas de classe UML definido por Guizzardi (2005).

2. Construir um procedimento que auxilie no uso da análise ontológica na fase de modelagem conceitual do desenvolvimento de sistemas baseado na técnica especificada no item 1. de modo a:

   - Definir formas de identificar as meta-propriedades associadas aos conceitos ontológicos do domínio do sistema a ser modelado.
   - Definir esquemas que permitam uma fácil utilização do mapeamento dos tipos associados às meta-propriedades em elementos do diagrama de classes. Mapeamento este definido na técnica especificada no item 1.
   - Definir formas de permitir uma fácil aplicação das regras e restrições de relacionamentos impostas pela técnica especificada no item 1.

3. Formalizar o procedimento construído em uma linguagem formal de definição de processos.

4. Aplicar o procedimento construído conforme item 2 para validar diagramas de classes UML construídos sem o uso de análise ontológica.

5. Analisar os benefícios do uso de análise ontológica na validação de diagramas de classes UML.

1.2 Organização deste Documento

No Capítulo 2 são apresentados os embasamentos teóricos necessários ao desenvolvimento deste trabalho. Descreve-se sobre modelagem conceitual e análise ontológica destacando-se ontologias de fundamentação (Guizzardi et al., 2008) e de propriedades (Guarino and Welty, 2000a,d,b). Relata-se ainda neste capítulo sobre o uso da modelagem ontológica na modelagem conceitual abordando trabalhos correlatos como a VERONTO e o Perfil OntoUML. Por fim, descreve-se sobre o SPEM, uma linguagem de modelagem de processos.

No Capítulo 3 é feita uma análise comparativa da técnica VERONTO e do Perfil OntoUML. Com base nessa análise apresenta-se a técnica OntoCon, construída com a junção de algumas características da VERONTO e do Perfil OntoUML, além do acréscimo de algumas melhorias.

No Capítulo 4 descreve-se todas as fases do procedimento PrOntoCon, criado para facilitar o processo de aplicação da técnica OntoCon. As quatro fases do procedimento são detalhadas.

No Capítulo 5 são descritos dois estudos de casos com o objetivo de validar diagramas de classe UML já existentes, utilizando o procedimento PrOntoCon. Descrevem-se, passo a passo as transformações ocorridas nos diagramas de classe UML originais em cada uma das fases do PrOntoCon, gerando um novo diagrama de classes UML validado pela técnica OntoCon. Faz-se também nesse capítulo a análise e discussão dos resultados. O resultado obtido em cada estudo de caso é analisado, comparando-se o diagrama de classes original com o diagrama validado, descrevendo-se as melhorias.

A conclusão do presente trabalho bem como as principais contribuições do

mesmo são feitas no Capítulo 6. Descrevem-se também neste capítulo as possíveis melhorias, alterações e trabalho futuros.

Nos Anexos A, B, C e D apresenta-se na íntegra o procedimento PrOntoCon, construído como principal contribuição deste trabalho.

Capítulo 2

Revisão Bibliográfica

2.1 Modelagem Conceitual Utilizando Diagramas de Classes UML

Qualquer engenharia utiliza de modelos para que produtos finais possam ser construídos, baseados nesses modelos. Segundo Eriksson et al. (2004), criar modelos requer um alto grau de criatividade. Na realidade não existe uma solução final única, pronta e totalmente correta. Modelar trata-se de uma arte e requer muita experiência do modelador que deve não só ser conhecedor da sua tarefa de modelar, mas também deve passar a conhecer sobre o domínio do problema. Modelos servem não só para representar um sistema, mas também para permitir uma maior compreensão do problema e permitir simulações. Na fase de análise, os requisitos até então levantados, passam por detalhamentos, estruturações e validações que permitem a construção de um modelo conceitual do sistema (Eriksson et al., 2004). Esta modelagem conceitual, realizada ainda na fase de análise, é necessária para que os conceitos relevantes do domínio do problema sejam devidamente modelados e assim usados como base para as fases seguintes do processo de desenvolvimento.

Segundo Elmastri and Navathe (2000), o modelo conceitual consiste em uma descrição concisa, utilizando-se uma linguagem de modelagem de alto nível, dos requisitos de dados coletados dos usuários, incluindo descrições detalhadas de tipos de

dados, relacionamentos e restrições. Elmastri and Navathe (2000) apontam como vantagens da utilização de tais modelos, primeiramente que eles são mais fáceis de serem entendidos, uma vez que os conceitos ali presentes não incluem nenhum detalhe de implementação, podendo ser utilizados para comunicação com usuários não técnicos; o modelo conceitual de alto nível também pode ser utilizado como referência para assegurar que todos os requisitos de dados do usuário foram alcançados e que não incluem quaisquer conflitos.

Segundo Wazlawick (2004), o modelo conceitual é um artefato do domínio do problema e não da solução do problema. Um modelo conceitual representa as informações que o sistema vai gerenciar. Qualquer modelagem equivocada irá provocar erros durante o processo de desenvolvimento do sistema, gerando um produto inadequado para o domínio no qual será utilizado.

Diagramas são utilizados para especificar como o analista quer representar o produto a ser desenvolvido. Para representar os conceitos que são extraídos na modelagem conceitual pode-se utilizar o diagrama de classes da UML. Segundo Villela (2004), o diagrama de classes é uma representação gráfica dos conceitos abstratos de um domínio e exibe uma visão estática de um sistema, mostrando suas classes e os relacionamentos existentes entre elas.

Como primeiro passo na construção de um diagrama de classes tem-se a identificação das classes que serão obtidas através da análise dos requisitos, já anteriormente identificados. Segundo Eriksson et al. (2004), uma forma de identificar se um conceito deve ou não ser considerado uma classe é questionando se tal conceito deve ser armazenado ou analisado pelo sistema. Quando se detecta que uma informação deve ser armazenada, transformada, analisada ou manipulada, tem-se então uma possível classe candidata. Durante o processo de análise, somente classes que fazem parte do domínio do problema devem ser consideradas. Classes técnicas que definem soluções relacionadas ao sistema a ser construído, como por exemplo, classes de interface de usuário, classes de banco de dados e outras, não podem ser consideradas durante a modelagem conceitual, não fazendo ,portanto, parte do diagrama de classes de análise(Eriksson et al., 2004).

O próximo passo na construção do diagrama é o agrupamento das classes correlatas em pacotes lógicos e em seguida a identificação dos relacionamentos existentes entre tais classes. O objetivo da identificação dos relacionamentos é mostrar como cada classe do domínio se coloca em relação às outras. Segundo Eriksson et al. (2004), as classes e os relacionamentos formam o núcleo de informação do sistema a ser gerado quando se usa programação orientada a objetos.

O diagrama de classes que irá representar o modelo conceitual do domínio do problema, denominado de diagrama de classes de domínio, deve conter somente os detalhes necessários para servir de base para o desenho do sistema. Portanto, deve-se evitar a inclusão de detalhes que estejam relacionados ao domínio da implementação (Paula, 2003).

Alguns equívocos são cometidos quando se controem diagramas de classes UML de domínio. Muitas vezes, classes que caracterizam papéis são criadas sem estar relacionadas a uma classe base (rígida). Como exemplo, num domínio acadêmico têm-se muitas vezes a classe Professor e a classe Aluno totalmente desvinculadas, sendo que na realidade pessoas tramitam entre esses papéis. Assim, uma modelagem adequada teria uma superclasse Pessoa, associada às subclasses Aluno e Professor, evitando redundância e tornando o sistema mais escalável. Outro problema de modelagem muito comum é a inversão de classes num relacionamento de generalização/especialização quando envolve papéis. Tal problema é citado por Guizzardi et al. (2004b) que utiliza o seguinte exemplo: uma classe Cliente como sendo superclasse das classes Organização e Pessoa. Uma pessoa que é hoje um cliente de uma determinada instituição não necessariamente o será por toda vida, mas certamente não deixará de ser pessoa. Ou seja, objetos das classes Pessoa e Organização nunca deixarão de ser objetos de tais classes durante toda sua existência, já objetos da classe Cliente não pertencerão necessariamente a esta classe por toda a sua existência. Logo tal relacionamento não está correto.

Percebe-se que a necessidade de se ter mecanismos que possam auxiliar na validação de diagramas de classes UML é primordial para que se tenha diagramas que representem mais adequadamente o domínio, gerando sistemas mais escaláveis,

estáveis, com menor redundância de dados e de fácil integração.

2.2 Análise Ontológica

Ao desenvolver uma modelagem conceitual de um determinado fenômeno de um domínio obtêm-se os objetos relevantes para a compreensão não só isoladas desses objetos, como principalmente o entendimento dos relacionamentos entre os mesmos. Uma análise mais detalhada das propriedades desses objetos é chamada de análise ontológica. Segundo Villela (2004), modelagem ontológica pode ser definida como sendo a criação de uma ontologia específica de domínio. Essa ontologia é construída através da captura de entidades importantes do domínio e da incorporação dessas entidades em um conjunto de categorias que revelam sua natureza.

Ontologia tem sua definição original na filosofia como sendo a ciência que estuda "os tipos de coisas que existem no mundo". No entanto, o termo ontologia tem sido utilizado com diferentes sentidos na área de Inteligência Artificial e Engenharia do Conhecimento, o que tem provocado certa descaracterização do mesmo (Villela, 2004).

Segundo Guarino (1998), uma ontologia define um vocabulário específico usado para descrever uma certa realidade, mais um conjunto de decisões explícitas fixando de forma rigorosa o significado pretendido para o vocabulário. Uma ontologia envolve, então, um vocabulário de representação que captura os conceitos, relações e propriedades em algum domínio e um conjunto de axiomas, que restringem a sua interpretação.

Na computação o termo ontologia é utilizado com sentidos diferentes dependendo da área. Na comunidade de modelagem de dados e áreas correlatas, ontologia assume as definições filosóficas, denominada, como categorias formais de um sistema independente de domínio bem fundamentadas filosoficamente. Desta forma pode ser usada para articular modelos reais de domínios específicos. Essa é uma classificação mais superior denominada de ontologia de nível topo (ou de fundamentação) que abrange termos gerais que formam o fundamento para a representação de conhecimento em todos os domínios (Guizzardi et al., 2008).

Nas comunidades de inteligência artificial, engenharia de software e web semântica, o termo ontologia é, em geral, utilizado: (i) como artefatos concretos destinados a propósitos específicos sem muito cuidado ou atenção a questões fundacionais e (ii) como uma representação de um domínio específico (política, química orgânica, transporte ferroviário). Esse tipo de ontologia, denominada ontologia de domínio, é utilizada como base de conhecimento do domínio dando suporte à tarefa de modelagem conceitual para sistemas do referido domínio (Guizzardi et al., 2008).

Ontologias de topo podem ser utilizadas com objetivo de classificar os conceitos de uma ontologia de domínio permitindo que seus elementos sejam mais bem entendidos e também os relacionamentos entre os mesmos (Villela, 2004). Portanto, é possível utilizar ontologia de nível topo para auxiliar no processo de modelagem conceitual.

Tanto ontologia topo quanto ontologias de domínio são classificações do que é chamando de ontologia formal. O presente trabalho utiliza o termo ontologia como sendo ontologia de nível topo.

2.2.1 Ontologia de Fundamentação Unificada - UFO

Ontologias de caráter filosófico têm sido propostas na Ciência da Computação, especificamente na área de Ontologia Aplicada. Tais ontologias têm sido denominadas Ontologias de Fundamentação.

Guizzardi et al. (2008) propõe uma ontologia de fundamentação denominada UFO - *Unified Foundational Ontology*, baseada em Ontologia Formal, Lógica Filosófica, Filosofia da Linguagem, Linguística e Psicologia Cognitiva.

A UFO é dividida em três categorias:(i) UFO-A: uma ontologia de objetos; (ii) UFO-B: uma ontologia de eventos; (iii) UFO-C: ontologia de entidades sociais. Tais ontologias têm sido usadas para avaliar, remodelar e dar um sentido semântico real para ontologia de domínio da engenharia de software. Faz-se aqui uma breve descrição de algumas categorias das UFO´s.

A Figura 2.1 apresenta uma parte da UFO-A, uma ontologia cuja principal distinção é feita entre as categorias **Indivíduo** (Particular) e **Universal** (Tipo). Segundo Guizzardi et al. (2008) Indivíduos são entidades que possuem uma identidade única.

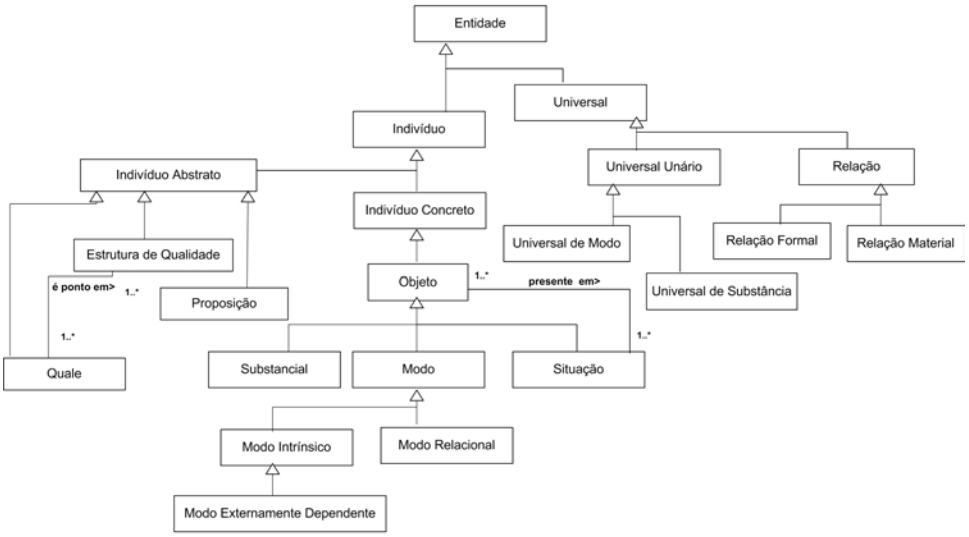


Figura 2.1: Parte da Ontologia Fundacional Unificada A (UFO-A). Universais e Indivíduos. Fonte: Guizzardi et al. (2008).

Já Universais são padrões de características que podem existir em diferentes elementos. **Substanciais** (*Substantial*) são existencialmente independentes. Por exemplo: uma pessoa, um cachorro, uma casa, um carro, Madre Tereza. O termo **Modo** (*mode*) é um indivíduo que só pode existir em outros indivíduos. Modos são todos dependentes de outros indivíduos, ou seja, só podem existir em outros indivíduos. O termo Modo não têm relação com tempo instantâneo. Modos são existencialmente dependentes de outros indivíduos. Através da dependência pode-se fazer a distinção entre **Modos Intrínsecos** e **Modos relacionais**. Modos Intrínsecos (**Intrinsic Modes** são dependentes de um único indivíduo, como por exemplo, cor, carga elétrica, um sintoma. Já Modos Relacionais (**relators** dependem de vários indivíduos, como por exemplo, um emprego, um tratamento médico, um casamento. **Relações** são entidades que estão coladas em outras entidades. Possui duas categorias: **Relações Materiais** e **Relações Formais**. Uma Relação Formal envolve relações do tipo dependência, parte de, subconjunto de , entre outras. Relações Materiais têm estrutura material incluem exemplos como trabalho em, estar matriculado em, e de ser conectado a. Por último destaca-se a entidade **Situação**, que são tipos especiais de objetos. Trata-se de um entidade constituída de muitos possíveis objetos, incluindo outras situações.

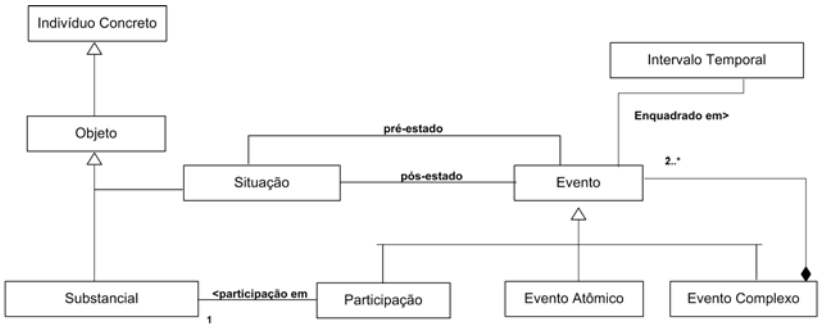


Figura 2.2: Fragmento da Ontologia Fundacional Unificada B (UFO-B). Objetos e Eventos. Fonte: Guizzardi et al. (2008).

Exemplos: 'João está com febre' , 'João está no mesmo local que Paulo enquanto Maria está no mesmo local que Davi'.

A Figura 2.2 apresenta uma parte da UFO-B que é uma ontologia que tem por objetivo fazer a distinção entre objetos e eventos. Eventos são indivíduos compostos de partes temporais e representam possíveis transformações de uma porção de realidade para outra. Exemplos: uma conversação, um jogo de futebol, uma festa de aniversário (Guizzardi et al., 2008). A principal categoria desta ontologia é **Eventos**. Eventos podem ser **Atômicos** ou **Complexos**. Eventos Atômicos não têm partes enquanto Eventos Complexos são compostos de pelo menos dois eventos. Eventos são entidades ontologicamente dependentes, uma vez que sua existência depende de seus participantes (Guizzardi et al., 2008).

Na Figura 2.3 tem-se uma parte da terceira camada da Ontologia de Fundamentação Unificada, denominada UFO-C, ontologia de entidades sociais. Destaca-se nesta ontologia as categorias **Agentes** e **Objetos**. Agentes podem ser agentes físicos como, por exemplo, uma pessoa. Mas podem ser também agentes sociais, como, por exemplo, uma organização ou uma sociedade. O mesmo acontece com a categoria Objetos, que pode ser objetos físicos ou sociais. Como objetos físicos pode-se ter um livro, uma árvore, um carro e como objetos sociais pode-se considerar dinheiro e idioma. Agentes são substanciais que podem possuir tipos especiais de Modos chamados de **Modos Intencionais** (**intentional modes**). Tais modos podem ser do tipo **Crenças)** (*Belief*), **Desejos** (*Desire*) e **Intenções** (*Intention*). **Ações** são

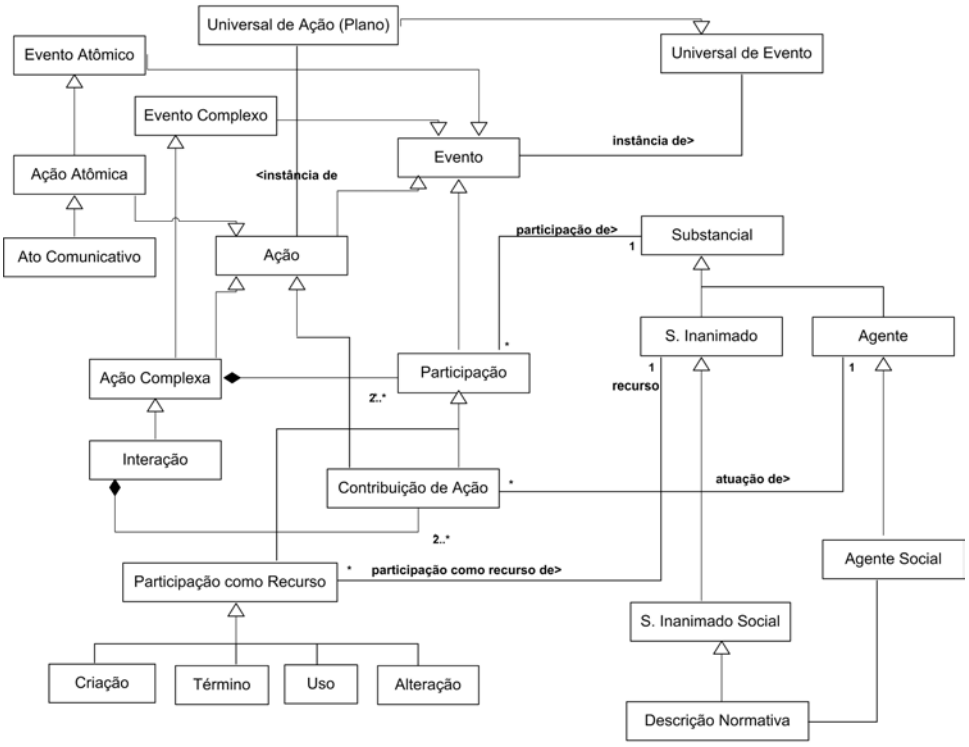


Figura 2.3: Parte da Ontologia Fundacional Unificada C (UFO-C). Ações, Agentes e Substanciais Inanimados. Fonte: Guizzardi et al. (2008).

eventos intencionais, sendo assim instâncias de um **Plano** para satisfazer uma intenção. Exemplos de Ações: um ato de comunicação, escrever um texto, um processo de negociação (Guizzardi et al., 2008).

Detalhes sobre as categorias das ontologias UFO-A, UFO-B e UFO-C estão descritas em Guizzardi et al. (2008).

2.2.2 Ontologia Formal de Propriedades

Guarino and Welty (2000a,d,b) definiram uma ontologia formal baseada em quatro meta propriedades: rigidez, dependência, identidade e unidade. Elas impõem restrições naturais à estrutura taxonômica que facilitam a compreensão ontológica dos conceitos trabalhados, bem como sua comparação e integração. Para fundamentar essas meta-propriedades foram utilizadas noções filosóficas sobre identidade, essência,

unidade e dependência.

A noção de identidade diz respeito à maneira como se reconhece entidades individuais. Unidade está relacionada ao tratamento de problemas de distinção das partes de uma instância do resto do mundo, através de uma relação de unificação que une suas partes. Essência refere-se ao conhecimento de que algumas propriedades mudam no decorrer do tempo e outras não. Por fim, a noção de dependência diz respeito a relações de dependência que podem ser intrínsecas e extrínsecas. As meta-propriedades de rigidez, dependência, identidade e unidade representam o comportamento de uma propriedade em relação às noções filosóficas.

Quando uma propriedade for analisada com base na meta-propriedade Rigidez, ela poderá ser classificada em: Rígida (+R), Não Rígida (-R) e Anti-Rígida (˜R). Uma propriedade é dita rígida (+R) quando é essencial para todas as suas instâncias, ou seja, se um elemento do domínio, que instancia tal propriedade, permanecerá instanciando-a durante toda sua existência. Formalmente tem-se: seja ϕ uma propriedade rígida então: $\forall x \phi(x) \rightarrow \Box \phi(x)$, onde o operador $\Box$ significa "em todos os possíveis mundos". Uma propriedade é dita não rígida (-R) quando não é essencial para alguma de suas instâncias. Formalmente tem-se: $\exists x \phi(x) \wedge \neg \Box \phi(x)$. Propriedade anti-rígida (˜R) é aquela que não é essencial para todas as suas instâncias. Formalmente tem-se: $\forall x \phi(x) \rightarrow \neg \Box \phi(x)$. Uma propriedade é considerada semi-rígida ($\neg$R) quando é não-rígida, mas não anti-rígida (Guarino and Welty, 2000a). A meta-propriedade Rigidez não é herdada ao longo da hierarquia de propriedades.

Objetivando facilitar o entendimento desse conceito, considere as propriedades Pessoa e Professor. Sabe-se que qualquer instância de Professor é também instância de Pessoa. Porém, uma instância da propriedade Professor pode deixar de ser professor em determinada circunstância, mudando de profissão por exemplo, mas jamais deixará de ser instância da propriedade Pessoa. Tal fato leva a concluir que a propriedade Professor é não-rígida (-R), sendo mais precisamente anti-rígida (R), pois todas as instâncias de Professor não serão necessariamente para sempre instâncias de Professor; e a propriedade Pessoa é rígida (+R), pois todas suas instâncias permanecerão como tal por toda a sua existência.

A meta-propriedade dependência externa está relacionada á noção filosófica de dependência, vista anteriormente. Guarino and Welty (2000c) definem esta meta-propriedade da seguinte forma:

'Uma propriedade ϕ é externamente dependente de uma propriedade ψ se, para todas as suas instâncias x, necessariamente alguma instância de ψ deve existir, que não seja uma parte nem constituinte de x: $\forall x \Box(\phi(x) \rightarrow \exists y \psi(\dot{y}) \wedge \neg P(y,x) \wedge \neg C(y,x)$ *'(tradução do autor).*

Para denotarmos uma propriedade externamente dependente, utiliza-se o símbolo +D e -D caso contrário. Maiores detalhes sobre relações de parte e constituinte podem ser verificadas no texto de Guarino and Welty (2000d).

Por exemplo, a propriedade Cliente é externamente dependente da propriedade Empresa, pois uma pessoa somente poderá ser um cliente, se houver uma empresa da qual ela possa adquirir produtos ou serviços prestados. Já a propriedade Pessoa não é externamente dependente de um coração ou corpo, pois uma instância de Pessoa possui um coração como parte e é constituída de um corpo (Villela, 2004).

Para entendimento da meta-propriedade identidade, deve-se definir o conceito de Condição de Identidade (IC). Segundo Guarino and Welty (2000c), uma condição de identidade para uma propriedade , que consiste numa relação , satisfaz a seguinte fórmula:

$$E(x,y) \wedge \phi(x,t) \wedge E(y,t) \wedge \phi(y,t') \wedge x = y \rightarrow \Gamma(x,y,t,t')$$

$$E(x,y) \wedge \phi(x,t) \wedge E(y,t) \wedge \phi(y,t') \wedge \Gamma(x,y,t,t') \rightarrow x = y$$

Diz-se que uma propriedade 'possui uma IC', representada pelo símbolo +I (-I, caso contrário), se existirem, em tempos distintos, instâncias que: se satisfazem a mesma IC, então elas são iguais (IC suficiente). Ou se, em tempos distintos, as instâncias são iguais, então satisfazem a mesma IC (IC necessária) (Guarino and Welty, 2000c).

Diz-se que uma propriedade ϕ 'fornece uma IC', denotada pelo símbolo +O (-O, caso contrário) somente se ela for rígida e possuir uma IC (+I), e esta não for executada por nenhuma propriedade que a subjuga. Isto significa que, se herda IC's diferentes (mas compatíveis) de múltiplas propriedades, ela ainda permanece fornecendo um IC

(Guarino and Welty, 2000c). Quando a hierarquia de propriedades vai se formando, as IC ´s vão sendo herdadas.

Por exemplo, a propriedade Cliente, que é não-rígida, pode apenas executar suas CI's (+I), herdando-as de propriedades rígidas que a subjugam, como a propriedade Pessoa (+O). Isto se dá devido ao fato de uma mesma pessoa poder ser cliente em diferentes tempos de diferentes empresas. Assim, uma CI fornecida por cliente, como, por exemplo, ter o mesmo código, pode ser apenas local; enquanto que uma CI fornecida pela propriedade Pessoa, como ter o mesmo número de Carteira de Identidade ou as mesmas impressões digitais, terá validade global (Villela, 2004).

A meta-propriedade unidade está relacionada com o conceito de Condição de Unidade (UC). Para se compreender o que significa uma certa propriedade possuir uma UC, ou seja, constituir-se um *todo*, deve-se observar as definições de Guarino and Welty (2000d,c): (i) "Seja ω uma relação de equivalência. Em um dado tempo t, um objeto x é um *todo contingente* sob ω se cada parte de x estiver ligada por ω a todas as suas outras partes e a nada mais". (ii) "Seja ω uma relação de equivalência. Um objeto x é um *todo intrínseco* sob ω se, a qualquer tempo em que x exista, ele é um *todo contingente* sob ω."

Então, uma propriedade "possui uma UC", representada pelo símbolo +U, se e somente se existir uma relação de equivalência ω tal que todas as suas instâncias sejam *todos intrínsecos* sob ω.

O símbolo ˜U denota propriedade que possui anti-unidade, isso ocorre quando cada uma de suas instâncias não constitui todos intrínsecos. Uma propriedade "não possui unidade"(-U), se ela não possui uma UC comum a todas as suas instâncias, ou seja se suas instâncias não se constituem todos intrínsecos (˜U) (Guarino and Welty, 2000a,d,c, 2001b).

As meta-propriedades descritas anteriormente geram algumas restrições naturais na estrutura taxonômica da ontologia. Sejam ϕ e ψ duas propriedades quaisquer e seja a notação ϕ^M a indicação de que a propriedade ϕ possui a meta-propriedade M. Tem-se as seguintes restrições:

- $\neg(\psi^{+R} \rightarrow \phi^{\tilde{}R})$: ϕ^{+R} não pode ser subtipo $\psi^{\tilde{}R}$;

- $\neg(\psi^{-I} \rightarrow \phi^{+I})$: ϕ^{-I} não pode ser subtipo ϕ^{+I};
- $\neg(\psi^{-D} \rightarrow \phi^{+D})$: ϕ^{-D} não pode ser subtipo ψ^{+D};
- $\neg(\psi^{-U} \rightarrow \phi^{+U})$: ϕ^{-U} não pode ser subtipo ψ^{+U};
- $\neg(\psi^{+U} \rightarrow \phi^{\tilde{}U})$: ϕ^{+U}; não pode ser subtipo $\psi^{\tilde{}U}$;
- Propriedades com UC's incompatíveis são disjuntas;
- Propriedades com IC's incompatíveis são disjuntas.

Discussões detalhadas sobre as meta-propriedades e sobre os tipos de propriedades são encontradas em Guarino and Welty (2000a,d,b).

Guarino and Welty (2000a) exploraram de forma sistemática as quatro meta-propriedades de forma a detectarem como elas poderiam ser combinadas para formar diferentes tipos de propriedades. Como resultado desta análise foi criado o que é denominado "Ontologia Formal de Propriedades". Veja na Tabela 2.1 os oito tipos de propriedades e cada uma das suas respectivas combinações de meta-propriedades.

As classificações dos tipos de propriedades mostradas na Tabela 2.1 auxiliam a tarefa do modelador de estabelecer o significado de cada elemento do domínio. Uma vez que os tipos de propriedades são baseados nas meta-propriedades e estas possuem restrições hierárquicas em relação aos elementos do domínio, pode-se entender, então, que tais classificações podem ser utilizadas para auxiliar no processo de criação e validação de modelos conceituais (Villela, 2004).

Uma explicação detalhada sobre cada tipo de propriedade pode ser encontrada em Guarino and Welty (2000a).

2.3 O Uso da Análise Ontológica na Modelagem Conceitual de Sistemas

Enquanto a modelagem conceitual concentra-se em relacionar de forma adequada os conceitos extraídos do domínio; a análise ontológica se propõe identificar os elementos

Tabela 2.1: Propriedades formadas a partir da combinação das meta-propriedades.

Meta-propriedades				**Tipo de Propriedades**	
+O	+I	+R	+D	Tipo	SORTAL
			-D		
-O	+I	+R	+D	Quase Tipo	
			-D		
-O	+I	˜R	+D	Papel Material	
-O	+I	˜R	-D	Ordenável com Fase	
-O	+I	¬R	+D	Mixin	
			-D		
-O	-I	+R	+D	Categoria	NÃO SORTAL
			-D		
-O	-I	˜R	+D	Papel Formal	
-O	-I	˜R	-D	Atribuição	
		¬R	+D		
			-D		

Fonte: Adaptada de Guarino and Welty (2000a) e Villela (2004).

do domínio e entender sua natureza, por meio da descrição de suas propriedades (Villela, 2004). Analisando o objetivo dessas duas modelagens pode-se entender que elas podem ser usadas de forma complementar. A base para a modelagem conceitual pode ser extraída de uma modelagem ontológica.

A ontologia formal de propriedades definida por Guarino and Welty (2000a), mostradas na Seção 2.2.2, auxilia no processo de entendimento de cada conceito do domínio bem como na estruturação hierárquica destes, em função das restrições hierárquicas. Uma vez que os conceitos a serem modelados são mais bem entendidos e existem regras que podem ser checadas para validar relacionamentos, principalmente de generalização/especialização, tem-se uma ferramenta de grande valia na construção e validação de modelos conceituais.

Alguns pesquisadores têm desenvolvido técnicas que auxiliam no uso da modelagem ontológica na modelagem conceitual. Nesta seção serão abordadas duas técnicas encontradas na literatura. A primeira é a VERONTO (Villela, 2004) e trata-se de uma proposta de utilização das meta-propriedades e da Ontologia Formal de Propriedades na modelagem conceitual. A segunda trata-se de OntoUML: um perfil UML ontologicamente bem formado para modelos conceituais estruturais (Guizzardi, 2005).

2.3.1 VERONTO:Técnica de Validação de Diagramas de Classe por meio de Propriedades Ontológicas

A VERONTO trata-se de uma técnica que utiliza as meta-propriedades rigidez, dependência externa, identidade e unidade, definidas por Guarino and Welty (2000a,d,b) na validação de modelos conceituais que estejam especificados por meio de diagramas de classe UML. Ao propor essa técnica, Villela (2004) realiza um mapeamento dos tipos de propriedades definidas por Guarino and Welty (2000a) em elementos do diagrama de classes da UML. Através desse mapeamento é possível aplicar as restrições taxonômicas sobre relacionamentos hierárquicos existentes quando se aplica a análise de meta-propriedades nos elementos do diagrama de classes.

Na Tabela 2.2 é apresentada parte da técnica VERONTO: o mapeamento dos

tipos de propriedades propostos na Ontologia Formal de Propriedades nos diagramas de classes da UML e as restrições sobre relacionamentos hierárquicos impostas pelo uso das meta-propriedades. O objetivo principal da VERONTO é que, através do auxílio proporcionado pela análise ontológica na extração adequada de conceitos de um domínio, seja possível construir modelos conceituais mais consistentes e mais fáceis de serem entendidos, compartilhados e integrados Villela (2004).

Tabela 2.2: Associação das meta-propriedades e regras que estas impõem aos relacionamentos hierárquicos aos construtores de modelagem conceitual .

Meta-Propriedades	Classificação da Propriedade	Construtor UML	Restrições Sobre Modelos Hierárquicos
+O +I +R	Tipo	Classe concreta Ex.: Pessoa, Gato	-Superclasse de classes que representem propriedades "sortais", pois fornece uma IC para suas subclasses (sortal é uma classificação das coisas do mundo que possuem princípio de identidade). -Pode ser subclasse de classes correspondentes a Categorias, outros Tipos ou Quase-Tipos (únicos que correspondem a propriedades rígidas).

Meta-Propriedades	Classificação da Propriedade	Construtor UML	Restrições Sobre Modelos Hierárquicos
-O +I +R	Quase-Tipo	Classe concreta Ex.: Animais Invertebrados, Animais Herbívoros	-Deve ser subclasse, no mínimo, de uma classe correspondente a um Tipo, com IC/UC compatível (para herdar a IC). -Pode ser subclasse de classes referentes a Categorias e Mixins (não recomendado). -Pode ser superclasse de classes que representem propriedades "sortais"(pois transmite sua IC para suas subclasses).
-O +I ~R +D	Papel Material	Classe concreta Ex.: Estudante, Casado, Comida	-Deve ser subclasse de uma classe correspondente a, no mínimo, um Tipo, com IC/UC compatível (para herdar a IC). -Pode ser subclasse de qualquer tipo de classe - recomenda-se classes referentes a propriedades rígidas (para fornecer a IC). -Pode ser superclasse de classes correspondentes a papéis materiais (pois é a única que, além de ser anti-rígida, executa uma IC e é externamente dependente). -Corresponde a um papel da superclasse em uma associação.

Meta-Propriedades	Classificação da Propriedade	Construtor UML	Restrições Sobre Modelos Hierárquicos
-O +I ~R -D	Sortal com Fase	Classe concreta Ex.: Lagarta, Borboleta	-Deve ser subclasse de uma classe correspondente a um Tipo, com IC/UC compatível (para herdar IC). -Pode ser subclasse de qualquer classe correspondente a uma propriedade independente (recomenda-se que também seja rígida). -Pode ser superclasse de quaisquer classes não rígidas (recomenda-se apenas classes correspondentes a *sortal* com fases e papéis materiais). -Corresponde a uma fase de sua superclasse, representada por uma classe correspondente a um Tipo ou Quase-Tipo, que subjuga apenas classes correspondentes a *sortal* com fases, que representam as possíveis fases pelas quais passa durante sua existência.

Meta-Propriedades	Classificação da Propriedade	Construtor UML	Restrições Sobre Modelos Hierárquicos
-O -I +R	Categoria	Classe Abstrata Ex.: Entidade Concreta, Entidade Abstrata	-Pode ser superclasse de classes de qualquer espécie. -Normalmente, correspondem a classes de mais alto nível, dentro de uma hierarquia. -Pode ser subclasse somente de classes também correspondentes a Categorias (pois não executa IC e é rígida). -Classe utilizada para fins classificatórios.
-O -I ~R +D	Papel Formal	Classe Abstrata Ex.: Paciente, Instrumento	-Superclasse de classes que representam papéis materiais. -Pode ser uma subclasse de classes correspondentes a Categorias e Papéis Formais (não executa IC). -Corresponde a um papel de uma classe em uma associação. -Classe utilizada para organizar a hierarquia de classes correspondentes a papéis.
-O -I ~R -D -O -I ¬R	Atribuição	Atributo Ex.: Cor, Forma	-Não fornece e não executa IC (-O, -I),também é anti-rígida e não dependente (~R, -D), ou semi-rígida (¬R). -Corresponde a um Atributo de uma classe em UML.

Fonte: Adaptada de Villela et al. (2004)

Quando as meta-propriedades são relacionadas ao conceito de unidade e identidade, algumas restrições aos relacionamentos *parte-todo* são diagnosticadas. A Tabela 2.3 mostra essas restrições e também o mapeamento das meta-propriedades em relação aos construtores de modelagem conceitual UML.

Além dos mapeamentos mostrados nas Tabelas 2.2 e 2.3, faz parte da técnica VERONTO a adição de regras para o uso de relacionamentos, baseadas nas regras propostas por Wand et al. (1999c) mostradas a seguir:

1. Regras para relacionamentos opcionais:

   (a) Associações opcionais (cardinalidade mínima 0) devem ser evitadas. Muitas vezes uma classe é externamente dependente de outra classe, não podendo então existir entre elas um relacionamento de cardinalidade zero.

   (b) A aquisição ou perda de uma associação deve ser modelada como uma mudança de classe.

   (c) A capacidade de instâncias de uma classe participar de uma associação, sem perder propriedades, deve ser modelada como uma subclassificação (especialização).

2. Regras para agregação

   (a) Cada classe componente deve ser associada com a classe composta através de um relacionamento de agregação.

   (b) As propriedades emergentes da classe composta ("todo") devem ser modeladas como atributos e associações.

Implicações práticas de tais regras, bem como algumas vantagens são discutidas por Villela (2004).

2.3.2 OntoUML: perfil UML ontologicamente bem formado para modelos conceituais estruturais

Guizzardi (2005) propõe um perfil UML para auxiliar na construção de modelos

Tabela 2.3: Associação das meta-propriedades relacionadas a unidade e identidade aos construtores de modelagem conceitual UML, e as restrições que essas impõem a relacionamentos "parte-todo".

Meta-Propriedades	Denominação das Propriedades	Construtor UML VERONTO	Restrições sobre Relacionamentos "Parte-Todo"
+I +U	Indivíduo	Classe concreta	-Pode ser uma classe parte de uma classe correspondente a um todo (+U) - Agregação compartilhada -Pode ser uma classe todo em um relacionamento "parte-todo"cujas partes sejam também todos (+U) ou apenas partes (-U).
+I -U	Identificável	Classe concreta	-pode ser apenas parte em um relacionamento "parte-todo- Composição.
-I +U	Inteiro	Classe Abstrata ou Atributo	

Fonte: Adaptada de Villela (2004)

conceituais sobre o ponto de vista ontológico. Tal perfil UML para modelos conceituais é bem fundamentado em conceitos filosóficos e psicológicos; além disso, incorpora dentre várias outras características, um padrão de projeto para resolver um problema recorrente que existe em modelagem de papéis, amplamente discutido em literatura.

Guizzardi et al. (2004a) defendem a idéia de que para os modelos conceituais representarem a realidade, isto é, serem modelos reais, a linguagem de modelagem conceitual dever ser fundamentada em uma ontologia topo.

O perfil UML e o padrão de projeto foram embasados em várias fundamentações filosóficas e psicológicas sobre tipos de classificadores para modelagem conceitual. Tais fundamentações podem ser vistas com detalhe em Guizzardi (2005), citam-se aqui apenas alguns princípios e postulados:

- **Postulado 1:** Todo objeto em um modelo conceitual (MC) de um domínio deve ser uma instância de uma classe do MC representando um *sortal* (sortal pode ser entendido como uma classificação das coisa do mundo e que são providas de princípio de identidade).

- **O princípio da restrição:** se um indivíduo se enquadra em dois *sortals* distintos F e F´, então no curso de sua história há um sortal pelo menos, do qual F e F' são especializações.

- **O princípio de singularidade:** se um indivíduo se enquadra em dois *sortals* distintos F e F´, então no curso de sua história há um *ultimate sortal* do qual F e F' são especializações. Um sortal F é último se não houver nenhum outro sortal F' distinto de F que F especializa.

- **Postulado 2:** Um objeto em um modelo conceitual do domínio não pode instanciar mais que uma classe do MC, que represente uma última *substance sortal* (é o único *ultimate sortal* que fornece princípio de identidade para suas instâncias).

- **Postulado 3:** Uma classe de um MC representando um classificador rígido não pode ser subclasse de uma classe que represente um classificador anti-rígido.

- **Postulado 4:** Uma classe de um MC que representa um *dispersive universal* (abstrações de propriedades comuns de papéis materiais) não pode ser uma subclasse de uma classe que represente um *sortal*.

O Perfil OntoUML proposto por Guizzardi (2005) possui um conjunto de classes estereotipadas que permitem o projeto de modelos conceituais bem fundamentados ontologicamente, de acordo com as fundamentações teóricas já comentadas nesta seção. A Tabela 2.4 extraída de Guizzardi et al. (2004b) e Guizzardi (2005) apresenta o perfil UML para diagramas de classes UML. As classes utilizadas para exemplificar cada um dos estereótipos têm apenas caráter ilustrativo.

Tabela 2.4: Um perfil UML ontologicamente bem formado para diagramas de classe UML- OntoUML.

Estereótipo	Descrição	Restrições
«kind»	Um *kind* representa uma *substance sortal*, isto é, rígida, *universal* externamente independente que prove um princípio de identidade para suas instâncias. Exemplos podem ser instâncias de tipos naturais (Pessoa, Cachorro, Árvore) e artefatos (Cadeira, Carro, Televisão).	Todo objeto em um modelo conceitual usando este perfil deve ser uma instância de um *kind*, direta ou indiretamente (Postulado 1). Além disso, ele não pode ser uma instância de mais de um *ultimate kind* (Postulado 2). Um supertipo de um *kind* não pode ser membro de *«subkind», «phase», «role», «roleMixin»*.

Estereótipo	Descrição	Restrições
«subkind»	Um *subkind* é rígido, carrega o princípio de identidade provida por um *kind*. Um exemplo pode ser o subtipo Pessoa Masculina do tipo Pessoa. Em geral, o estereótipo *subkind* pode ser omitido em um modelo conceitual sem perda de clareza.	Um supertipo de um subkind não pode ser membro de *«phase», «role», «roleMixin»*.
«phase»	Representa um *phased-sortal phase*, ou seja, um *Universal* anti-rígido e externamente independente definido como parte de uma partição de um *kind*. Por exemplo, a partição lagarta, borboleta do tipo Lepdóptero.	Fases são *Universals* anti-rígidos e assim, um «phase» não pode aparecer em um modelo conceitual como um supertipo de um *Universal* rígido (Postulado 3). As fases {P1 ... Pn} que formam a partição de um tipo K são definidas em UML como um conjunto disjunto e completamente generalizado. O tipo K é sempre mostrado com uma classe abstrata.

Estereótipo	Descrição	Restrições
«role»	Representa a *phased-sortal role*, ou seja, um *Universal* anti-rígido e externamente dependente. Por exemplo, o papel estudante exercido por uma instância de um tipo Pessoa.	Papéis são *Universals* anti-rígidos e não podem aparecer em um modelo conceitual como uma superclasse de um *Universal* rígido (Postulado 3). Além disso: seja X uma classe estereotipada como *role* e r seja uma associação representando condições de restrições de X. Então, #X.r$\geq$ 1.
«category»	Representa um *non-sortal* rígido e externamente independente, um *dispersive universal* que agrega propriedades essenciais que são comuns a diferentes tipos. Por exemplo, a categoria EntidadeRacional como sendo uma generalização de Pessoa e AgenteInteligente.	Uma categoria não pode ter instâncias diretas e tem de ser representada como uma classe abstrata. Um supertipo de um categoria não pode ser membro de *«kind», «subkind», «phase», «role», «roleMixin»*.

Estereótipo	Descrição	Restrições
«roleMixin»	Representa um *non-sortal* rígido e externamente dependente, um *dispersive universal* que agrega propriedades que são comuns a diferentes papéis. Inclui papéis formais como *whole* e *part* e *initiatior* e *responder*.	Um role mixin não pode ter instâncias diretas e deve ser representada como uma classe abstrata. Um supertipo de um papel formal não pode ser membro de *«kind», «subkind», «phase», «role»*. Seja X uma classe estereotipada como *roleMixin* e seja r uma associação representando condições de restrições de X. Então, #X.r$\geq$ 1.

Fonte: adaptada de Guizzardi et al. (2004a,b); Guizzardi (2005)

Um dos problemas recorrentes na modelagem conceitual é a modelagem de papéis. A Figura 2.4 mostra que o papel Cliente é um supertipo de Pessoa e Organização. Esta modelagem viola o postulado 3, produzindo um modelo conceitual ontologicamente incorreto. Nem toda pessoa ou organização é cliente o tempo todo e nem todo cliente permanecerá na condição de cliente o tempo todo (Guizzardi, 2005).

Para resolver este problema Guizzardi (2005) propõem um Padrão de Projeto (Design Pattern) baseado no perfil definido na Tabela 2.4, como sendo uma solução ontologicamente correta. A Figura 2.5 apresenta o diagrama que representa este padrão de projeto.

O padrão foi desenvolvido com base na seguinte análise: (i) Uma entidade que seja um tipo dispersivo, não rígido, que possua vários subtipos que obedeçam a princípios de identidade diferentes (ex: Cliente) deverá ser uma generalização de seus subtipos (ex: ClientePrivado e ClienteCorporativo). Tais subtipos são elementos *sortals* que carregam princípios de identidade providos por entidades do tipo *kind* (ex: Pessoa e Organização). A classe A representa um *role mixin* (papel formal) que

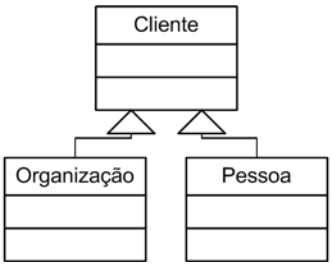


Figura 2.4: Problemas de modelagem de papéis com múltiplos tipos permitidos. Fonte: Guizzardi (2005).

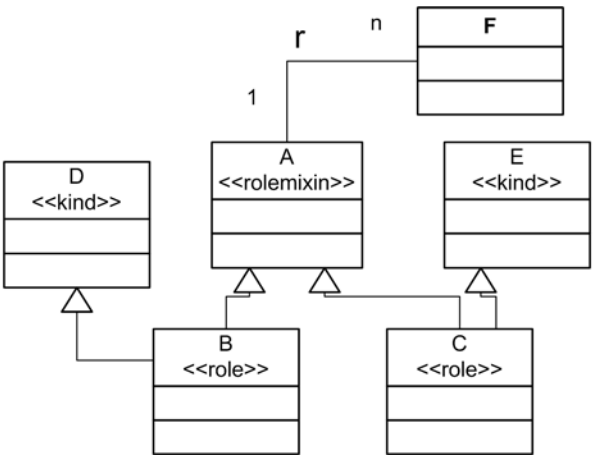


Figura 2.5: Padrão de Projeto Ontológico para Modelagem de Papéis. Fonte: Guizzardi (2005).

abrange diferentes tipos de papéis. As classes B e C são duas classes disjuntas de A que podem ter instâncias diretas, representando um papel *sortal* que carrega um princípio de identidade que comanda os indivíduos que são suas extensões. As classes D e E são *ultimate substance sortal*, ou seja, são tipos *«kind»*, que provê o princípio de identidade herdado por B e C respectivamente. A associação r representa a condição de especialização de B e C, que é representado em A. Por último, a classe F representa o tipo de que a classe A é externamente dependente (Guizzardi, 2005).

A Figura 2.6 mostra a correção do modelo da Figura 2.4 conforme o padrão de projeto proposto.

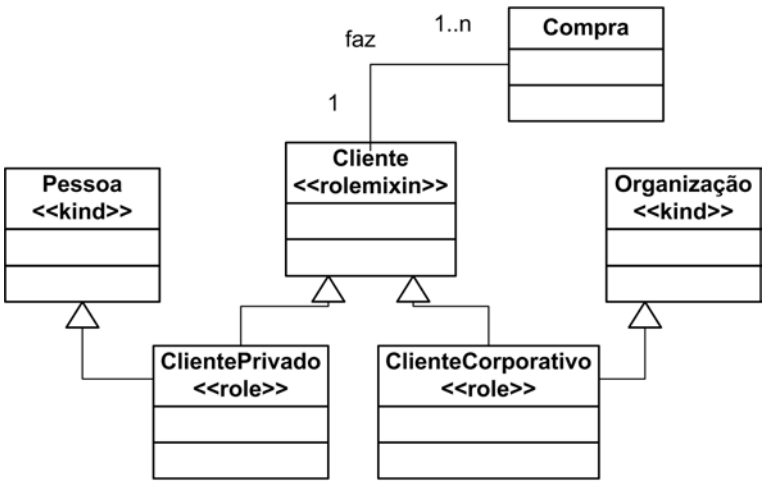


Figura 2.6: Versão ontologicamente correta do modelo da Figura 2.4 obtido com a aplicação do padrão de projeto. Fonte: Guizzardi (2005).

Este padrão de projeto resolve não somente os problemas oriundos de erros hierárquicos como classes estereotipadas como «role» aparecendo como supertipos de classes estereotipadas como «kind». Quando uma classe «role» for subtipo de duas superclasses «kind», caracterizando uma herança múltipla tem-se um problema. Segundo restrições do Perfil OntoUML, sabe-se que um «role», obrigatoriamente, tem que ter um supertipo «kind» do qual irá herdar a identidade. Quando um «role» tem dois supertipos «kind», de qual supertipo irá herdar sua identidade? Se aplicarmos o Padrão de Projeto proposto por Guizzardi (2005) este problema será resolvido

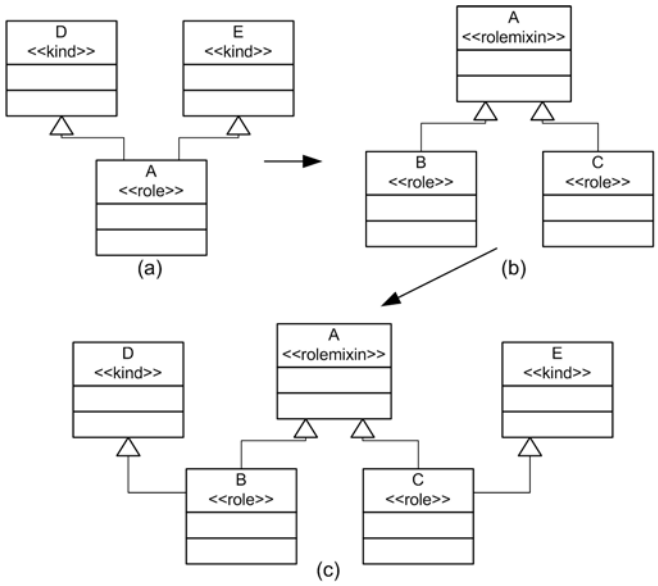


Figura 2.7: Aplicação do Padrão de Projeto de Guizzardi (2005), para casos onde existem múltiplos supertipos «kind» para uma classe esterotipada como «role».

pois a classe estereotipada como «role» será transformada em um «rolemixin», como mostrado com a classe A na Figura 2.7. Serão criadas tantas subclasses «role» quantas forem as superclasses «kind» da classe «role» inicial. Cada classe «role» será subclasse da classe «rolemixin» e será subclasse também da classe «kind» correspondente. Tendo assim apenas uma superclasse «kind» para herdar sua identidade.

O OntoUML não é composto somente da definição dos perfis UML com suas devidas restrições (Tabela 2.4) e do Padrão de Projeto Ontológico para Modelagem de Papéis. Outras contribuições fazem parte do OntoUML e algumas delas se referem a relacionamentos todo-parte. Tais contribuições podem ser analisadas em Guizzardi (2005).

Capítulo 3

OntoCon - Técnica de Validação de Diagrama de Classes de Domínio Baseado em Modelagem Ontológica

O uso da modelagem ontológica na validação de modelos conceituais, mais especificamente Diagramas de Classes UML, tem sido alvo de pesquisas, como pode ser constatado com os estudos feitos nas Seções 2.3.1 e 2.3.2. Tal recurso permite a obtenção de diagramas mais corretos e conseqüentemente mais manuteníveis, o que vem a ser fator de grande relevância para os sistemas atuais, cada vez mais complexos e sempre em expansão. A técnica VERONTO (Villela, 2004) e o perfil OntoUML (Guizzardi, 2005) são duas técnicas fundamentadas nas meta-propriedades de Guarino and Welty (2000a,b,c,d, 2001b, 2002) que permitem verificar principalmente restrições de hierarquias baseadas em generalização/especialização. Além dessas restrições hierárquicas, tais técnicas possuem outros pontos que complementam uma validação de diagramas de classes, como: as regras para o uso de relacionamentos, baseadas nas propostas de Wand et al. (1999c) presente na VERONTO, o padrão de projeto para modelagem de papéis proposto por Guizzardi (2005) e as restrições impostas para relacionamentos todo-parte presente em ambas.

Através do estudo comparativo da VERONTO e do perfil OntoUML, feitos na Seção 3.1 percebeu-se a importância da elaboração de uma técnica que fosse a

composição de ambas. A relevância da elaboração de uma técnica que incorporasse outras duas deve-se ao fato de: (i) poder eliminar redundâncias que ocorrem quando as técnicas são aplicadas em separado; (ii) incorporar melhorias; e (iii) estruturar uma técnica de tal forma que facilite a criação do procedimento para uso da mesma. A importância deste último fato é significativo para esta pesquisa, uma vez que a criação de um procedimento que conduza o modelador na aplicação de uma técnica de validação de um modelo conceitual por meio de modelagem ontológica é o principal objetivo do presente trabalho. Tem-se, então, como ponto inicial a organização da Técnica OntoCon (Uso da Modelagem Ontológica na Modelagem Conceitual) mostrado na Seção 3.2. Na Seção 3.1 faz-se uma análise aprofundada das técnicas VERONTO e do perfil OntoUML.

3.1 Análise comparativa da técnica VERONTO e do Perfil OntoUML para modelos conceituais UML

Tanto a técnica VERONTO quanto o Perfil OntoUML discutidos neste trabalho são fortemente embasados nas meta-propriedades de Guarino and Welty (2000a,d,b). Pode-se perceber que um dos focos dessas duas técnicas é o relacionamento taxonômico, tendo como base as relações de Generalização/Especialização, preocupando-se em verificar se tais relações são ontologicamente corretas. Conforme mostrado nas Tabelas 2.2 e 2.4 existem restrições de relacionamento para cada classificação de propriedade da VERONTO e para cada um dos estereótipos do Perfil OntoUML.

Antes de se fazer uma comparação entre as restrições taxonômicas impostas pelas técnicas, é preciso mostrar a relação entre os conceitos de base que podem ser extraídos de ambas quando o foco são as meta-propriedades. A Tabela 3.1 mostra que a maioria dos tipos de propriedade utilizada na técnica VERONTO tem relação direta com um dos estereótipos definidos por Guizzardi (2005). Isso ocorre porque eles estão embasados nas mesmas meta-propriedades, tendo divergências apenas na meta-propriedade dependência externa. Em Guizzardi (2005) não é feito uma relação explícita das metapropriedades associadas a cada estereótipo do OntoUML. A rela-

ção das meta-propriedades com cada estereótipo do Perfil OntoUML foi estabelecida através de uma análise criteriosa dos postulados e das descrições de cada estereótipo do Perfil OntoUML. Tal relação é descrita no decorrer desta seção.

Analisando a Tabela comparativa 3.1 percebem-se semelhanças, diferenças e características complementares entre a VERONTO e o Perfil OntoUML. Comparando a propriedade *Tipo* com o estereótipo «kind», vê-se que o mapeamento deste último é mais abrangente, uma vez que engloba tanto classe abstrata quanto classe concreta. Tal mapeamento é permitido devido à restrição imposta a estereótipos «kind», onde todo objeto em um modelo conceitual que utilize o Perfil OntoUML deve ser uma instância de um «kind», direta ou indiretamente. O estereótipo «kind» é caracterizado por Guizzardi (2005) como rígido, ou seja, +R. Pode-se concluir que as meta-propriedades +O e +I fazem parte deste estereótipo, uma vez que este provê um princípio de identidade, ou seja, têm um suporte que permite identificar se dois elementos são os mesmos. Para a propriedade *Quase-Tipo* o mapeamento feito na técnica VERONTO remete a um elemento que seja classe concreta. Para o estereótipo «subkind», o autor não explicita tal mapeamento, porém pode-se concluir que poderá representar tanto classe abstrata quanto classe concreta, uma vez que todo estereótipo «subkind», tem que ter como supertipo um estereótipo «kind». Pode-se concluir que as meta-propriedades -O e +I fazem parte do estereótipo «subkind» porque tem o princípio de identidade provido por um «kind».

O mapeamento feito na técnica VERONTO para a propriedade *Papel Material* remete a um elemento classe concreta. Para se analisar a possibilidade do estereótipo «role» poder representar tanto classe abstrata quanto concreta pode-se embasar no princípio de restrição. Segundo Guizzardi (2005), como conseqüência do princípio de restrição, para todo *phased-sortal PS*, como «role» por exemplo, existe um *substance sortal* S, da qual PS é uma especialização. Desta forma «role» são especializações de um «kind», que é um *substance sortal*, podendo ser mapeado como classe concreta ou abstrata. Além disso, sendo especialização de um «kind» ele terá as meta-propriedades -O e +I.

Para a propriedade *Sortal com Fase* o mapeamento feito na técnica VERONTO

Tabela 3.1: Comparação da técnica VERONTO Villela (2004)com o Perfil OntoUML Guizzardi (2005). Características comparadas: meta-propriedades associadas e construtor UML.

VERONTO			**Perfil OntoUML**		
Classificação da Propriedade	Meta-Propriedades	Construtor UML	Estereótipo	Meta-Propriedades	Construtor UML
Tipo	+O +I +R +D ou -D	Classe concreta	«kind»	+O +I +R -D	Classe abstrata ou classe concreta
Quase-Tipo	-O +I +R+D ou -D	Classe concreta	«subkind»	-O +I +R-D	
Papel Material	-O +I ˜R +D	Classe concreta	«role»	-O +I ˜R +D	
Sortal com Fase	-O +I ˜R -D	Classe concreta	«phase»	-O +I ˜R -D	
Categoria	-O -I +R +D ou -D	Classe abstrata	«category»	-O -I +R -D	Classe abstrata
Papel Formal	-O -I ˜R +D	Classe abstrata	«rolemixin»	-O -I ˜R +D	Classe abstrata
Atribuição	-O -I ˜R -D -O -I ¬R	atributo			

Fonte: o autor.

remete a um elemento classe concreta. Para o estereótipo «phase», Guizzardi (2005) não explicita tal mapeamento, porém pode-se concluir que poderá representar tanto classe abstrata quanto classe concreta, uma vez que todo estereótipo «phase» tem que ter um supertipo «kind» em alguma parte da hierarquia de relacionamentos. As meta-propriedades -O e +I fazem parte do estereótipo «phase», pois, sendo especialização de um «kind», possui o princípio de identidade provido por tal estereótipo.

Tanto «role» quanto «phase» são *phased sortal*, ou seja, são anti-rígidos e externamente dependente.

A propriedade Categoria e o estereótipo «category» são muito semelhantes. Uma única diferença encontrada diz respeito ao conceito de dependência externa. Conforme restrições do Perfil OntoUML, nenhum estereótipo que seja +O e ou +I são supertipos de «category». Então, pode-se concluir que ele não possui e nem fornece identidade (-O, -I). O mesmo fato pôde ser concluído para o estereótipo «rolemixin», ele também possui as meta-propriedades -O e -I.

A propriedade Atribuição não foi tratada pelo Perfil OntoUML por se tratar de uma propriedade que é mapeada para atributos das classes dos diagramas de classe UML, não sendo correto um estereótipo para tal. A propriedade Mixin e o estereótipo «mixin» não estão sendo comparados neste trabalho por se tratar de uma classificação pouco usada em modelagem conceitual, sendo seu uso desencorajado por Guarino and Welty (2000a).

Percebe-se que a meta-propriedade Dependência Externa dos tipos de propriedade e dos estereótipos mostrados na Tabela 3.1 não são equivalentes na maioria dos casos. Tanto na VERONTO quanto no Perfil OntoUML a descrição de ser ou não dependente é feita de forma explícita. Porém, as classificações tipo, quase-tipo e categoria diferem dos perfis «kind», «subkind» e «category» no que diz respeito à meta-propriedade dependência externa (+D e - D). Isso se deve ao fato do conceito de dependência externa ser tratado de forma diferente por Guizzardi (2005) e Guarino and Welty (2000a). Para Guarino and Welty (2000a), uma característica é considerada dependente do todo. Por exemplo: buracos em pedaços de queijo são dependentes do queijo. Já para Guizzardi (2005), um buraco em um pedaço de queijo é uma parte

inseparável e não externamente dependente.

Várias conclusões sobre relacionamentos hierárquicos puderam ser feitas, uma vez que as meta-propriedades associadas a cada estereótipo do Perfil OntoUML são as mesmas dos tipos de propriedades definidos por Guarino e Welty, exceto para a meta-propriedade Dependência Externa. Analisando as restrições de relacionamentos do Perfil OntoUML descritas na Tabela 2.4 e comparando-as com as restrições de relacionamento hierárquicos da técnica VERONTO (Tabela 2.2), pode-se destacar:

- Para quase todos os tipos de propriedades usadas na técnica VERONTO, não se menciona explicitamente a necessidade de respeitar a restrição taxonômica relativa à meta-propriedade dependência externa. Seria importante fazer menção sobre a necessidade de se respeitar tal restrição.

- Os supertipos permitidos a elementos classificados como *Tipo* e como estereótipo «kind» tem um ponto de controvérsia. Na VERONTO a propriedade *Quase-Tipo* pode ser supertipo de *Tipo*. No Perfil OntoUML, classes estereotipadas como «subkind» não podem ser supertipo de «kind». Entende-se que existe uma diferença de visão entre as linhas de pesquisa de Guarino and Welty (2000a) adotada na técnica VERONTO e de Guizzardi (2005) adotada no Perfil OntoUML. O termo *Quase-tipo* deve ser atribuído a elementos que sejam quase-tipo, não obrigatoriamente tendo que ser apenas subtipo de outro tipo. Muitas vezes pode-se ter um elemento classificado como *Quase-Tipo* sendo supertipo de um elemento classificado como *Tipo*. Um exemplo seria uma hierarquia onde tem-se a classe Animal classificada como *Tipo*, sendo supertipo das classes Invertebrado e Vertebrado, ambas classificadas como *Quase-Tipo*. Acrescentando a esta hierarquia a classe Pessoa, um *Tipo*, esta será um subtipo da classe Vertebrado, ou seja, uma especialização de um *Quase-Tipo*. Segundo a restrição do Perfil OntoUML tal hierarquia não é permitida, isto porque o termo *Subkind* utilizado no Perfil OntoUML é interpretado como subtipo e não como quase tipo, sendo incoerente então ter «subkind» como generalização de «Kind».

- Os supertipos permitidos a elementos classificados como *Quase-Tipo* e com o es-

tereótipo «subkind» diferem apenas no que diz respeito ao supertipo «subkind» permitido para o estereótipo «subkind» e não mencionado para a propriedade *Quase-Tipo*. Com base nas meta-propriedades e nas restrições taxonômicas (Seção 2.2.2) impostas pelas mesmas, pode-se concluir que *Quase-Tipo* pode ser supertipo de *Quase-Tipo*.

- *Papel Material* pode ter como supertipos propriedades classificadas como: T*ipo, Quase-Tipo, Sortal com Fase, Papel Material, Papel Formal e Categoria*. Para o estereótipo «role» não é feita uma menção explícita dos supertipos possíveis. Uma vez que esses supertipos não violam nenhuma das restrições taxonômicas abordadas na Seção 2.2.2 (Guarino and Welty, 2000a,d,b), pode-se concluir que «role» pode ter os mesmos supertipos do tipo *Papel Material*.

- As considerações feitas no item anterior valem para a classificação *Sortal Com Fase* e o estereótipo «phase».

- Os supertipos permitidos a elementos classificados com *Categoria* e com o estereótipo «category» são equivalentes.

- Os supertipos permitidos a elementos classificados com *Papel Formal* e com o estereótipo «rolemixin» são equivalentes.

- O Perfil OntoUML não relata explicitamente os subtipos possíveis para cada estereótipo, já a técnica VERONTO exibe a relação entre cada propriedade classificatória e seus respectivos subtipos. Uma vez que o Perfil OntoUML está baseado nas mesmas combinações de meta-propriedades que a VERONTO, pode-se concluir que os subtipos utilizados nesta técnica são válidos no Perfil OntoUML desde que observada a restrição quanto à propriedade Dependência Externa.

- Na técnica VERONTO as propriedades *Quase-Tipo, Papel Material e Sortal com Fase* têm a seguinte restrição: devem ser subclasse de uma classe correspondente a um *Tipo*, com IC/UC compatível (para herdar IC). No Perfil OntoUML esta restrição corresponde à restrição colocada no estereótipo «kind» afirmando que

todo objeto tem que ser instância de um *kind* em modelagem conceitual, direta ou indiretamente (postulado 1). Além disso, esta restrição é acrescida de outra: este objeto não pode ser instância de mais de um *ultimate sortal*.

- A técnica VERONTO afirma que um elemento caracterizado como Papel Material corresponde a um papel da superclasse em uma associação. No Perfil OntoUML, através do postulado 3, afirma-se que o estereótipo «role» não pode aparecer como superclasse de tipo. Também no OntoUML se estabelece a necessidade da existência de no mínimo uma associação (r) entre uma classe X estereotipada como «role» com outra classe, ou seja, #X.r$\geq$ 1.

A técnica VERONTO faz um mapeamento das propriedades Indivíduo (+U, +I), Identificável (-U, +I) e Inteiro (+U, -I) em construtores UML, impondo restrições sobre relacionamentos 'todo-parte' (Tabela 2.3). O Perfil OntoUML também aborda restrições referentes a relacionamentos 'todo-parte' em Guizzardi (2005).

Faz parte da técnica VERONTO regras para o uso de relacionamentos, baseadas nas regras propostas por Wand et al. (1999c) (Seção 2.3.1), que não são abordados no Perfil OntoUML. Um dos diferenciais do Perfil OntoUML é a definição do padrão de projeto para resolver problemas relativos à modelagem de papéis (Seção 2.3.2).

3.2 OntoCon - Proposta de combinação da técnica VERONTO com o Perfil OntoUML para Modelos Conceituais UML

Com base no estudo comparativo feito entre a técnica VERONTO e o Perfil OntoUML estruturou-se a técnica OntoCon, que combina características de ambas com o objetivo de propor uma técnica única que auxilie ainda mais no processo de validação de diagramas de classe UML. O principal objetivo da construção da OntoCon é a montagem de uma técnica com uma estrutura organizada, focada nas restrições que permitirão uma validação adequada de diagramas de classe UML no que diz respeito

a relacionamentos de generalização e especialização. Propicia-se desta forma atingir o objetivo da construção de um procedimento de uso da técnica proposta.

A Tabela 3.2 mostra também, os estereótipos definidos para a técnica OntoCon tendo como base a classificação das propriedades utilizadas na técnica VERONTO. Além disso, a Tabela 3.2 mostra o mapeamento desses estereótipos para construtores de modelagem conceitual.

Apesar de existir uma relação direta entre cada propriedade definida por Guarino and Welty (2000a) e os estereótipos definidos no Perfil OntoUML conforme análises mostradas na Seção 3.1, optou-se pela nomenclatura da VERONTO, por estar alinhada com o trabalho de pesquisa que tem sido desenvolvido no DPI (Departamento de Informática da Universidade Federal de Viçosa) e em função de que a adoção da visão de subkind em oposição à quase-tipo teria um forte impacto sobre o procedimento desenvolvido.

Como parte do mapeamento dos estereótipos em construtores de diagramas de classe UML foi acrescentado o construtor Interface. Interfaces executam um importante papel em sistemas bem estruturados proporcionando um caminho gerenciável para manutenção de sistemas. Uma interface não pode instanciar objetos reais, elas contêm somente operações abstratas (Eriksson et al., 2004).

Optou-se pelo Perfil OntoUML no uso da meta-propriedade dependência externa. Entende-se que apesar da diferença entre os conceitos de dependência externa atribuída em Guarino and Welty (2000a) e em Guizzardi (2005), a essência dos conceitos é a mesma. O fato de ser dependente ou parte inseparável acabará forçando a necessidade de uma associação entre classes para suprir a dependência externa ou a parte inseparável. Com essa decisão foi possível estabelecer de forma mais criteriosa todos os possíveis supertipos e subtipos de cada estereótipo, uma vez que cada um deles só poderá ser dependente (+D) ou independente (-D). Por exemplo, conforme Guizzardi (2005) o estereótipo «kind» e um «subkind» são independentes (-D) e logo, seguindo as restrições hierárquicas (Seção 2.2.2), um «kind» pode ser supertipo de um «subkind». Já conforme Guarino and Welty (2000a), um *Tipo* e um *Quase-tipo* podem ser dependentes (+D) ou independentes (-D), dependendo do contexto,e logo,

Tabela 3.2: Relação entre os estereótipos da OntoCon e as Propriedades definidas por Guarino and Welty (2000a). Mapeamento dos estereótipos a construtores UML, baseado em Villela (2004) e Guizzardi (2005).

Classificação da Propriedade	**Estereótipo**	**Meta-Propriedades**	**Construtor UML**
Tipo	«tipo»	+O +I +R -D	Classe abstrata ou classe concreta. Sempre será classe abstrata quando forma o todo das partições de estereótipos que sejam «fase».
Quase-Tipo	«quase-tipo»	-O +I +R -D	Classe abstrata ou classe concreta.
Papel Material	«papel material»	-O +I ˜R +D	Classe abstrata ou classe concreta.
Sortal com Fase	«fase»	-O +I ˜R -D	Classe abstrata ou classe concreta.
Categoria	«categoria»	-O -I +R -D	Interface ou classe abstrata
Papel Formal	« papel formal»	-O -I ˜R +D	Interface ou classe abstrata
Atribuição	-	-O -I ˜R -D -O -I ¬R	Atributo

Fonte: o autor.

seguindo as restrições hierárquicas, um elemento classificado como *Tipo* não poderá ser supertipo de um elemento classificado como *Quase-tipo* se o *Tipo* for -D e o subtipo for +D. A decisão de optar pelo conceito de Dependência Externa de Guizzardi (2005) torna mais simples a elaboração de um procedimento de uso da técnica OntoCon e conseqüentemente torna o procedimento mais aplicável.

Quando a Técnica VERONTO coloca que as propriedades 'sortais', por possuírem identidade (+I) devem ser mapeadas como classes concretas, torna a modelagem conceitual restrita. Com este mapeamento, uma classe Pessoa caracterizada como *Tipo* que possua as classes Masculino e Feminino como especializações e caracterizadas como *Quase-Tipo*, não poderia ser criada como uma classe abstrata. No entanto sabe-se que isso pode ser necessário numa modelagem conceitual.

O mapeamento definido para a técnica OntoCon é mais completo quando define que um fenômeno do domínio, classificado como «tipo», «quase-tipo», «papel formal» ou «fase», pode vir a ser uma Classe Concreta ou Abstrata. Tais considerações foram feitas uma vez que, segundo o Postulado 1, apresentado na Seção 2, todo objeto em um modelo conceitual tem que ser uma instância de «kind» direta ou indiretamente (Guizzardi, 2005). O fato do objeto poder ser uma instância indireta de «kind» permite afirmar que todos os estereótipos *sortals*, ou seja, que possuem uma identidade (+I) podem ser mapeados como classe abstrata e interface. Porém alguns cuidados devem ser tomados. Para garantir que o objeto será realmente instanciado, a hierarquia de classes abstratas deve culminar em uma classe concreta. Num diagrama de classe UML os estereótipos que estiverem na folha da árvore da hierarquia de relacionamentos deverão ser classes concretas, quando for objetivo da modelagem. Há casos que não existirão classes concretas, como por exemplo nas modelagens de *frameworks*.

Já os estereótipos «papel formal» e «categoria» não podem ser mapeados como classes concretas uma vez que são conceitos abstratos (dispersivos). Conceitos dispersivos não denotam 'sortais', portanto não podem ter instâncias diretas (Guizzardi et al., 2004a).

Na Tabela 3.3 são apresentadas as restrições herdadas da técnica VERONTO e do Perfil OntoUML. Os supertipos e subtipos definidos para a OntoCon se baseiam

na técnica VERONTO, seguindo a linha de pesquisa de (Guarino and Welty, 2000a). As análises feitas nas Seção 3.1 e os dados das Tabelas 2.2 e 2.4 foram considerados para se definir as relações hierárquicas permitidas entre os estereótipos da OntoCon. Todas as possibilidades de supertipos e subtipos de cada estereótipo da OntoCon segue rigorosamente as restrições impostas pelas meta-propriedades de identidade, rigidez e dependência, restrições estas citadas na Seção 2.2.2.

Tabela 3.3: Restrições taxonômicas da técnica OntoCon, herdadas da técnica VERONTO (Villela, 2004) e do Perfil OntoUML para Modelos Conceituais UML (Guizzardi, 2005).

Estereótipo	**Restrições Hierárquicas**
«tipo»	-Supertipos possíveis: «categoria», «tipo», «quase-tipo». -Subtipos possíveis: «tipo», «quase-tipo», «papel material», «fase». -Todo objeto tem que ser instância de um *Tipo* em modelagem conceitual (Postulado 1). Tal objeto não pode ser instância de mais de um *ultimate sortal*.
«quase-tipo»	-Supertipos possíveis: «tipo», «quase-tipo» e «categoria». -Subtipos possíveis: «tipo», «quase-tipo», «papel material», «fase». -Tem que ser subclasse de uma classe correspondente a «tipo» para herdar a condição de identidade.

Estereótipo	**Restrições Hierárquicas**
«papel material»	-Supertipos possíveis: «tipo», «quase-tipo», «fase», «papel material», «papel formal», «categoria». -Subtipos possíveis: «papel material». -Tem que ser subclasse de, no mínimo, uma classe correspondente a «tipo» para herdar a condição de identidade. -Por corresponder a um papel da superclasse em uma associação tem-se: seja X uma classe estereotipada como «papel material» e r seja uma associação representando condições de restrições de X. Então, #X.r$\geq$ 1.
«fase»	-Supertipos possíveis: «tipo», «quase-tipo», «fase», «categoria». -Subtipos possíveis: «fase», «papel material». -Tem que ser subclasse de uma classe correspondente a «tipo», para herdar a condição de identidade. -As fases P1 ... Pn que formam a partição de um *Tipo* K são definidas em UML como um conjunto disjunto e completamente generalizado.
«categoria»	-Supertipos possíveis:«categoria». -Subtipos possíveis: todos. -Normalmente, correspondem a classes de mais alto nível, dentro de uma hierarquia. -Classe utilizada para fins classificatórios.

Estereótipo	Restrições Hierárquicas
«papel formal»	-Supertipos possíveis:«categoria» e «papel formal». -Subtipos possíveis: «papel formal», «papel material». -Corresponde a um papel formal de uma classe em uma associação. Logo, seja X uma classe estereotipada como «papel formal» e r seja uma associação representando condições de restrições de X, então,#Xr≥1. -Classe utilizada para organizar a hierarquia de classes correspondentes a papéis.

Fonte: o autor.

Além das restrições de relacionamentos hierárquicos herdadas da técnica VERONTO e do Perfil OntoUML, a técnica OntoCon incorpora também o padrão de projeto definido no Perfil OntoUML conforme descrito na Seção 2.3.2. Alguns autores caracterizam o padrão de projeto proposto por Guizzardi (2005) como padrão de análise (Fowler, 1997) por ser utilizado numa fase de análise e não na fase de implementação como costumam ser a maioria dos padrões de projeto. O padrão de projeto definido por Guizzardi (2005) permite resolver de forma padronizada o problema da modelagem de papéis relacionados a múltiplos tipos disjuntos permitidos. Tal padrão trata dois casos, sendo o primeiro quando têm-se uma classe estereotipada como «role» sendo supertipo de classes estereotipadas como «kind» e o segundo quando têm-se uma classe «role» como subtipo de mais de uma classe estereotipada como «kind». A única alteração feita ao padrão de projeto definido por Guizzardi (2005) foi a mudança dos nomes dos estereótipos, utilizando os adotados na OntoCon. Tal alteração é possível, uma vez que existe uma relação direta entre os estereótipos da OntoCon com os estereótipos do Perfil OntoUML, conforme analisado na Seção 3.1.

É importante destacar que alguns problemas detectados na técnica VERONTO e no Perfil OntoUML são superados com a técnica OntoCon, como mostra a Tabela 3.4.

Tabela 3.4: Problemas detectados na técnica VERONTO e no Perfil OntoUML e melhorias proporcionadas pela OntoCon.

Problemas com as técnicas abordadas	**Melhorias agregadas com a OntoCon**
Na VERONTO, o mapeamento das propriedades *tipo, quase-tipo, papel material e sortal de fase* torna-se muito restritivo uma vez que tais propriedades são mapeadas somente para classe concreta. O Perfil OntoUML não destaca explicitamente nenhum mapeamento para os estereótipos «kind», «subkind», «role» e «phase».	Na OntoCon, o mapeamento dos estereótipos relativos a essas mesmas propriedades é mais abrangente: um possível mapeamento tanto para classe abstrata quanto para classe concreta é permitido.
Na VERONTO e no Perfil OntoUML, o mapeamento das propriedades categoria («category») e papel formal («rolemixin») é muito restritivo, uma vez se refere somente a classe abstrata.	Na OntoCon, os estereótipos relativos às propriedades categoria e papel formal podem ser mapeados tanto para interface quanto para classe abstrata.
As restrições de relacionamento entre classes e subclasses no que diz respeito à meta-propriedade dependência externa não são ressaltadas de forma clara na VERONTO.	Na OntoCon, todas as relações de supertipos e subtipos são tratadas respeitando a restrição de relacionamento da meta-propriedade dependência externa.
O Perfil OntoUML e a VERONTO não tratam explicitamente os subtipos possíveis de cada estereótipo e para cada tipo de propriedade, respectivamente.	A OntoCon trata todos os subtipos possíveis a cada estereótipo, o que proporciona uma validação mais fácil e completa de um diagrama de classe.

Problemas com as técnicas abordadas	**Melhorias agregadas com a OntoCon**
Tanto a VERONTO quanto o Perfil OntoUML não citam de forma explícita todos os possíveis supertipos para cada uma das propriedades e ou estereótipos.	A OntoCon cita explicitamente todos os possíveis supertipos para cada um dos estereótipos existentes, proporcionando uma maior segurança no uso da técnica.

Fonte: o autor.

Nem todos os recursos da Técnica VERONTO e do Perfil OntoUML foram contemplados pela Técnica OntoCon. A verificação de relacionamentos parte-todo e o uso adequado de alguns relacionamentos conforme regras propostas por Wand et al. (1999c) não estão presentes na técnica. Priorizou-se tratar os relacionamentos de herança e a modelagem de papéis de forma mais ampla, considerando e verificando diversos casos, sendo, assim, verificar e testar uma maior diversidade de situações. Os demais recursos poderão ser contemplados em versões posteriores.

Resultados da aplicação da técnica VERONTO e do Perfil OntoUML mostraram que os modelos conceituais validados tornaram-se mais genéricos e mais flexíveis a mudanças. Espera-se, então, que a aplicação da técnica OntoCon contribua de forma mais satisfatória, uma vez que resolve os problemas relatados na Tabela 3.4 e incorpora duas técnicas. Tal incorporação é positiva uma vez que: (i) associa características distintas de cada técnica; (ii) elimina redundâncias que iriam ocorrer caso as técnicas fossem aplicadas em separado; (iii) monta uma técnica prática para se construir o procedimento de uso da mesma.

Capítulo 4

PrOntoCon - Procedimento de Análise para Validação de Diagrama de Classes de Domínio Baseado em Modelagem Ontológica

Através de aplicações e análises feitas com a OntoCon pôde-se constatar os benefícios que tal técnica traz ao processo de verificação de Diagramas de Classe, gerando diagramas mais genéricos e flexíveis a mudanças. Certamente tais benefícios foram herdados das técnicas de origem, aliado ao uso de uma técnica única. Porém, pôde-se constatar também que a aplicação de todas as regras da OntoCon não é algo trivial. O uso da modelagem ontológica como ferramenta de validação de Diagramas de Classe UML requer do modelador uma capacidade de abstração e aplicação de conceitos filosóficos que não são assimilados de forma rápida. Tal necessidade chega, por vezes, a invalidar o uso da técnica, uma vez que a maioria dos modeladores desitem de aplicá-la devido à dificuldade de entender e usar suas regras, apesar da eficiência das mesmas. Em função disso o objetivo principal da pesquisa aqui realizada é a elaboração de um procedimento que facilite ao modelador a aplicação da técnica OntoCon. Tal procedimento, intitulado PrOntoCon - Procedimento para uso da Técnica OntoCon - tem por objetivo final conduzir à validação de Diagramas de Classes UML de forma que

as dificuldades trazidas pelos conceitos e interpretações filosóficas, inerentes à Análise Ontológica, se tornem o mais transparente possível ao aplicador da técnica.

Através da aplicação da Técnica OntoCon, três pontos chaves podem ser validados em um diagrama de classes: (i)relacionamentos de generalização/especialização; (ii) relacionamento de classes envolvidas na modelagem de papéis; e (iii) definição adequada de construtores UML (classe concreta, classe abstrata e inteface). Com base nesses pontos foram estruturadas as etapas do procedimento PrOntoCon.

Para que os relacionamentos de generalização/especialização possam ser validados é necessário, primeiramente, a identificação dos devidos estereótipos («tipo», «quase-tipo», «papel material», «fase», «papel formal», «categoria») associados a cada classe do diagrama, e conseqüentemente dos tipos de propriedades e das meta-propriedades definidas por Guarino and Welty (2000a,b,c,d). A primeira fase do PrOntoCon é denominada Identificação de Estereótipos e é descrita na Seção 4.1.

Após a identificação dos tipos de propriedades, tem-se todo subsídio para verificar se os relacionamentos de generalização/especialização estão obedecendo às restrições hierárquicas impostas pelo uso das meta-propriedades (Guarino and Welty, 2000a,b,c,d). Tal fase, intitulada Verificação Hierárquica, é descrita em detalhes na Seção 4.2. Relacionamentos das classes envolvidas na modelagem de papéis podem ser passíveis de erros e, portanto, necessitam passar pelas etapas da terceira fase do PrOntoCon: Aplicação do Padrão de Projeto. Nessa fase, descrita na Seção 4.3, aplica-se o padrão de projeto da OntoCon, herdado do Perfil OntoUML, que verifica dois casos de problemas envolvendo modelagem de papéis. O primeiro caso trata da não observância das restrições hierárquicas impostas pelo uso das meta-propriedades (Guarino and Welty, 2000a,b,c,d). O segundo caso trata problemas de propagação da meta-propriedade identidade (+I) nos casos de herança múltipla envolvendo classes estereotipadas como «tipo» e «papel material». Finalmente pode-se passar para a quarta e última fase, descrita na Seção 4.4 que é responsável por fazer uma verificação do uso adequado dos construtores UML: classe abstrata, classe concreta e interface, utilizados no diagrama. A Figura 4.1 mostra a sequência das quatro fases no procedimento de validação PrOntoCon que foi construído utilizando-se o SPEM -

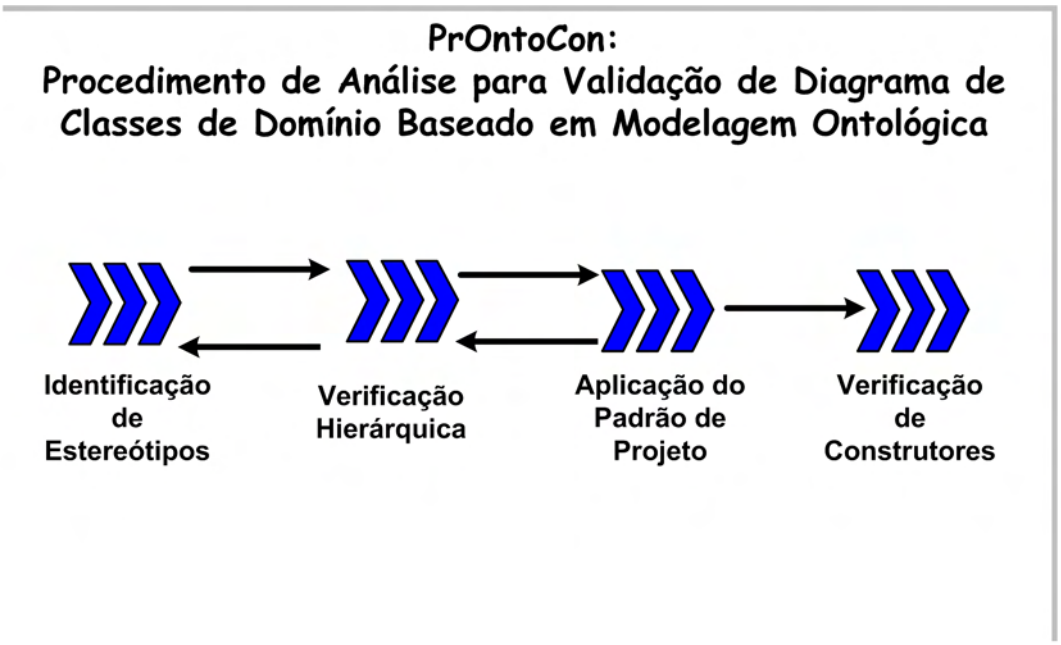


Figura 4.1: Fases do Procedimento de Validação de Diagramas de Classe - PrOntoCon

Software Process Engeneering Metamodel- um metamodelo para modelar processos e seus componentes (Anexo E).

A Figura 4.2 apresenta uma visão macro de todas as etapas do PrOntoCon com o objetivo de mostrar a ramificação das fases em subfases. Tal figura não representa a ordem que será seguida ao se utilizar o procedimento, uma vez que tal ordem dependerá de diversas particularidades do diagrama a ser validado.

Espera-se que com o PrOntoCon, os modeladores possam validar os Diagramas de Classes UML de forma mais intuitiva, usufruindo, assim, de todos os benefícios que a Modelagem Ontológica traz: modelos conceituais mais genéricos e mais fáceis de serem compreendidos, compartilhados e integrados. Tais benefícios são fato, uma vez que o uso da modelagem ontológica na análise conceitual evita a presença de informações redundantes no modelo e obriga que apenas as informações estáveis do domínio sejam representadas, ou seja, aqueles conceitos rígidos que deverão ser sempre modelados, independentes da aplicação.

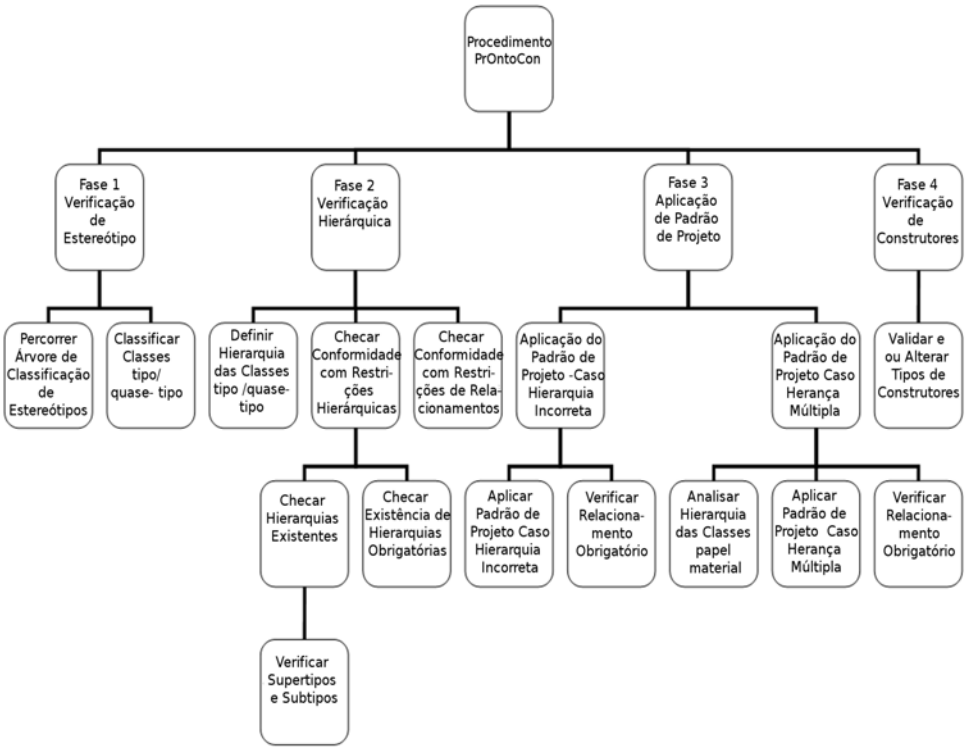


Figura 4.2: Fluxograma representativo das fases e subfases do procedimento PrOnto-Con, numa visão hierárquica.

4.1 Primeira Fase: Identificação de Estereótipos

Através da construção da OntoCon percebeu-se uma relação direta entre os estereótipos do Perfil OntoUML (Guizzardi, 2005) e os tipos de propriedades definidas por Guarino and Welty (2000a,b,c,d). Tal relação pode ser observada na Tabela 3.2. Tanto os estereótipos quanto os tipos de propriedades são definidos com base num agrupamento das meta-propriedades de rigidez, identidade e dependência. Para se classificar uma classe como sendo «papel material» por exemplo tem-se que identificar na mesma as meta-propriedades:-O, +I,˜R e +D, ou seja, não fornece uma identidade, possui identidade, é anti-rígida e dependente, respectivamente. Tal identificação é de certa forma imprescindível, uma vez que é com base nessas identificações que se torna possível checar as restrições hierárquicas impostas por tais meta-propriedades (Guarino and Welty, 2000a,b,c,d). Na Técnica VERONTO, cada classe passa por uma análise onde é verificada a existência de cada meta-propriedade. Somente após esta análise é possível concluir o tipo de propriedade que tal classe representa (Villela, 2004). Esse caminho é muito árduo para o modelador, uma vez que tem que passar a conhecer de forma mais detalhada os conceitos filosóficos nos quais se embasada a análise ontológica.

Visando minimizar as dificuldades inerentes ao processo de classificação das classes do diagrama quanto às meta-propriedades, percebeu-se que nem sempre é necessário investigar todas as meta-propriedades de uma classe para que a mesma seja classificada corretamente. Por exemplo, uma vez identificado que uma classe não possui identidade (-I) e além disso é rígida (+R), não é necessário verificar se fornece ou não uma identidade (-O ou + O) ou se é ou não dependente (+D ou -D), pois o único tipo de propriedade que é -I (não possui identidade) e +R (rígida) é o tipo *Categoria* («categoria»). Com base nesta forma de análise montou-se uma árvore de caminhamento para identificação dos estereótipos de cada classe de forma que o menor número de passos fossem necessários. Tal árvore é mostrada na Figura 4.3 e tem como ponto de partida a análise se a classe possui ou não identidade (+I ou -I), permitindo fazer um desmembramento entre as classes que se enquadrarão, segundo

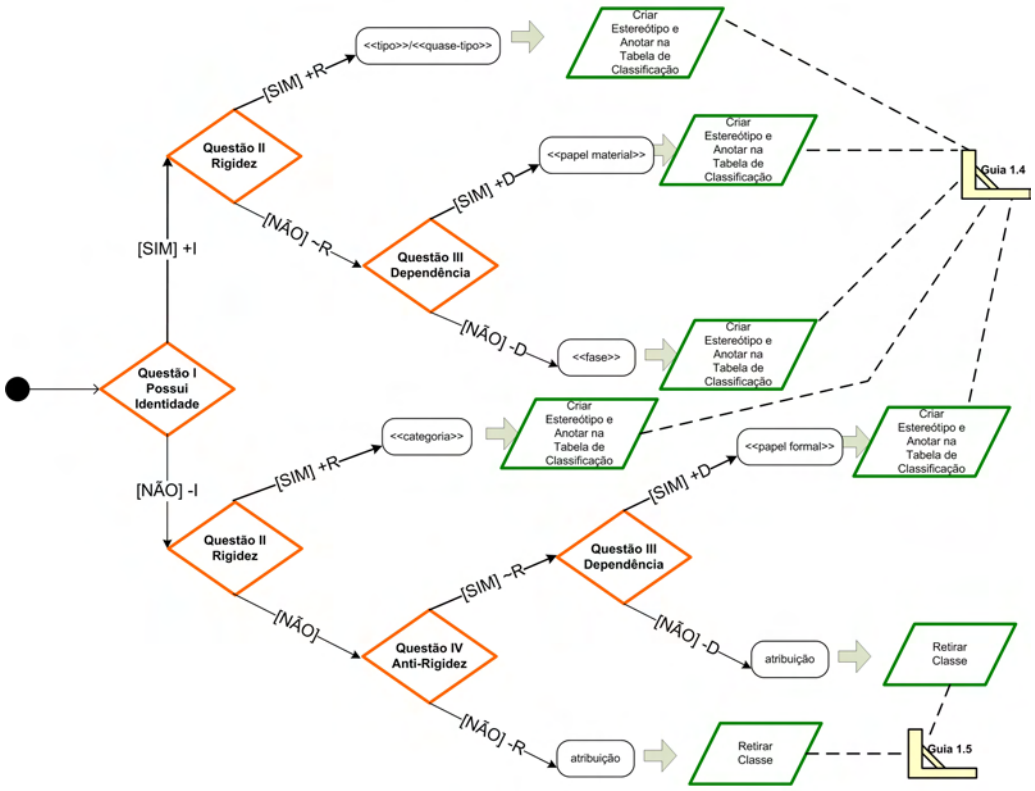


Figura 4.3: Árvore de decisão para identificação de estereótipo.

Guarino and Welty (2000a) em *sortal* ou não *sortal*.

O caminhamento na árvore ocorre conforme resposta dada às questões relacionadas em cada losango. Existem quatro tipos de perguntas. Cada um desses tipos foram elaborados de forma que o modelador possa entender o que se pretende identificar sem necessariamente se aprofundar no conceito filosófico em questão. Porém muitas vezes, uma pergunta isolada não é suficiente para fazer o modelador entender e identificar determinada propriedade na classe. Então, associado a cada pergunta existem exemplos e uma série de dicas que têm por objetivo conduzir ao entendimento correto do que se pretende identificar.

Iniciando um caminhamento na árvore apresentada na Figura 4.3, tem-se inicialmente um losango que levará à primeira pergunta, cujo texto e exemplos foram embasados em Guarino and Welty (2000c, 2001a); Welty and Guarino (2001). Tal pergunta tem o objetivo de identificar se a classe em análise possui ou não identidade.

A pergunta feita tem o seguinte texto:

'Todas as instâncias da classe em análise possuem uma característica comum cujo valor seja único? Exemplo: para a classe PESSOA existe a característica impressão digital e cada instância dessa classe possui uma impressão digital única. Por isso pode-se dizer que a classe PESSOA executa (possui) uma identidade (+I). Veja considerações importantes no suporte para análise da pergunta.

1. *Outros exemplos:*

   - *Classe LOCALIZAÇÃO: toda instância desta classe possuiu um valor único para a característica região do espaço. Essa classe possui então uma identidade (+I).*
   - *Classe ANIMAL: toda instância desta classe possuiu um valor único para a característica DNA, ou seja, todo animal tem um tipo de DNA e cada DNA é diferente um do outro. Essa classe possui então uma identidade (+I).*
   - *Classe ORGANIZAÇÃO: considere organização como sendo um grupo de pessoas com papéis que definem uma estrutura. Toda instância dessa classe possuiu um valor único para a característica missão, ou seja, cada instância de ORGANIZAÇÃO possui uma missão diferente em algum aspecto. Essa classe possui então uma identidade (+I).*

2. *Contra exemplos:*

   - *Classe ENTIDADE: entidade aqui representa tudo, tudo é uma entidade. Logo não existe nenhuma característica única que identifique individualmente cada instância da classe ENTIDADE.*
   - *Classe ENTIDADE SOCIAL: considere entidade social como sendo algo que representa um grupo de pessoas unidas por razões sociais. Normalmente não existe uma característica única que identifique cada instância desta classe. Então ela não possui uma identidade (-I).*

3. *É importante destacar que as chaves primárias são muitas vezes características criadas para fins de armazenamento de dados e muitas vezes não devem ser usadas como sendo a característica comum e de valor único investigada na questão sobre identidade. Exceto em situações onde a chave primária é uma informação que existe na realidade e faz parte da natureza do conceito em análise. Um exemplo seria o número de uma nota fiscal.'*

Quanto ao último item da Questão I, cabe fazer a seguinte ressalva: apesar da chave primária se basear em características extrínsecas e sociais, tal como CPF por exemplo, muitas vezes ela reflete a existência de um critério de identidade para a classe e está relacionada com características intrínsecas da mesma. Portanto, muitas vezes o modelador irá utilizar a chave primária como indicador da classe ter ou não identidade. Sabe-se que analisando do ponto de vista filosófico, não seria adequada tal escolha, porém forçar os desenvolvedores a entrar em questões filosóficas para desenvolver um sistema é extremamente complicado, uma vez que eles não têm um treinamento e uma visão adequada sobre o tema.

Dependendo da resposta dada à primeira pergunta o modelador caminhará para um determinado ramo da árvore e encontrará a próxima pergunta que terá o objetivo de detectar a existência ou não da meta-propriedade rigidez (Figura 4.4). Desta forma, respondendo a cada pergunta e caminhando adequadamente na árvore conforme resposta dada a cada pergunta, o modelador chegará a um nó folha da árvore. Neste ponto tem-se a definição do estereótipo da classe em análise. Nos nós folhas da árvore de identificação dos estereótipos encontram-se as ações que devem ser efetuadas ao se concluir o estereótipo representado pela classe em análise. Quando a classe é identificada como «tipo»/«quase-tipo», «papel material», «fase», «papel formal» ou «categoria», a ação a ser executada é alterar o diagrama de classe acrescentando-se na classe em análise o estereótipo identificado.

Quando a classe em análise é identificada como atribuição, deve-se retirá-la do diagrama e torná-la atributo das classe(s) pertinente(s). Tais procedimentos são necessários em função do mapeamento existente na Técnica OntoCon, herdada da

Questão II - Rigidez

Pergunta:

O objeto que representa a instância da classe em análise permanecerá sendo uma instância desta classe durante toda a sua existência em todos os possíveis domínios? Logo, tal objeto não deixará de pertencer a esta classe. Por exemplo, normalmente o objeto que representa uma instância da classe PESSOA não deixará de ser pessoa, durante toda a sua existência. Desta forma esta classe possui uma característica chamada rigidez (+R). Veja considerações importantes no suporte para análise da pergunta.

Suporte para análise da pergunta:

i)Quando a instância pode deixar de pertencer a classe dizemos que ela possui uma característica chamada anti-rigidez (~R)

ii) **Outros exemplos:**

- Classe LIVRO: normalmente, o objeto que representa uma instância da classe LIVRO permanecerá durante toda sua existência uma instância da classe LIVRO. Então LIVRO possui a propriedade rigidez (+R)
- Classe EXEMPLAR: significando exemplar de livro. Normalmente, o objeto que representa uma instância da classe EXEMPLAR permanecerá durante toda sua existência uma instância da classe EXEMPLAR. Então EXEMPLAR possui a propriedade rigidez (+R)
- Classe EMPRÉSTIMO-EXEMPLAR: normalmente, o objeto que representa uma instância da classe EMPRÉSTIMO-EXEMPLAR permanecerá durante toda sua existência, uma instância da classe EMPRÉSTIMO-EXEMPLAR (+R).

iii) **Contra exemplos**:

- Classe ITENS EMPRESTIMO-EXEMPLAR: normalmente, o objeto que representa uma instância da classe ITENS EMPRESTIMO-EXEMPLAR **não** permanecerá durante toda sua existência uma instância da classe ITENS EMPRESTIMO-EXEMPLAR. Isso ocorre porque um item pode ser retirado do empréstimo voltando a ser apenas uma instância de EXEMPLAR e não mais uma instância pertencente à classe ITENS EMPRESTIMO-EXEMPLAR.
- Classe ESTUDANTE: normalmente, o objeto que representa uma instância da classe ESTUDANTE **não** permanecerá durante toda sua existência uma instância da classe ESTUDANTE. Isto ocorre porque tal instância pode deixar de ser estudante, permanecendo apenas como instância da classe PESSOA.

iv) Quando uma resposta a pergunta sobre rigidez for negativa, não significa que o objeto que representa a instância deixará de existir. Na realidade esta instância deixa de pertencer à classe em análise e continua sendo uma instância da superclasse que nesses casos necessariamente irá existir. É o que acontece com uma instância pertencente à classe ESTUDANTE, quando esta instância deixa de pertencer a tal classe, obrigatoriamente continuará sendo uma instância da classe PESSOA.

v) Quando estiver na pergunta sobre rigidez originada do caminho "[NÃO] –I" da árvore e a classe em análise tem fins classificatórios, provavelmente instâncias da classe em análise permanecerão para sempre uma instância da mesma durante toda a sua existência.

Figura 4.4: Questão 2 - Rigidez. Pergunta utilizada no procedimento com o objetivo de descobrir se a classe é ou não uma classe rígida.

Técnica VERONTO, que diz que uma classe identificada como tipo atribuição deve ser mapeada como atributo de uma classe. Após feito isso é necessário verificar se as classes que receberam tal atributo não sofreram alteração na sua classificação, caso já tenham sido classificadas. Tal verificação deve ser feita percorrendo a árvore de decisão novamente para tais classes. Todo esse processo de caminhamento na árvore deve ser feio para cada classe do Diagrama de Classes, exceto para classes de associações.

Classes de associações são classes conectadas a uma associação. Trata-se de uma classe com características semelhantes às classes normais, com a diferença que surge do relacionamento de outras duas classes. Por exemplo, do relacionamento da classe Empresa com a classe Pessoa surge a classe Emprego. Cada instância da classe Pessoa que se relaciona com uma instância da classe Empresa gera uma instância da classe Emprego. Quando analisa-se uma classe de associação quanto a meta-propriedade identidade percebe-se que sua identidade está totalmente dependente das classes pertencentes à associação. No exemplo, a identidade da classe Emprego só consegue ser identificada analisando-se o contexto todo, ou seja, informações do conceito Empresa e Pessoa. Tal fato dificulta a análise ontológica de tal classe, uma vez que a identidade que se procura em uma classe não pode ser dependente de outras classes. Trabalhos futuros serão necessários para adaptar o procedimento PrOntoCon de forma validar as classes de associações. Poderia-se utilizar o conceito de *Relators* do Perfil OntoUML. *Relators* são indivíduos com o poder de ligar entidades; por exemplo, conexão de vôo é um relator que liga dois aeroportos, uma matrícula é um relator que liga um aluno com uma instituição de ensino (Guizzardi, 2005).

O uso da árvore de decisão é, na realidade, a primeira atividade da primeira fase do procedimento PrOntoCon. O diagrama de atividades mostrado na Figura 4.5 apresenta como deve acontecer a seqüência de atividades nesta primeira fase do procedimento, cuja objetivo é realizar a identificação dos estereótipos associados às classes do diagrama. Os guias 1.1 e 1.3 contém instruções do que deve ser realizado em cada atividade.

O modelador do sistema, tendo em mãos o diagrama de classes a ser validado, percorre a árvore de decisão classificando todas as classes do diagrama, associando a

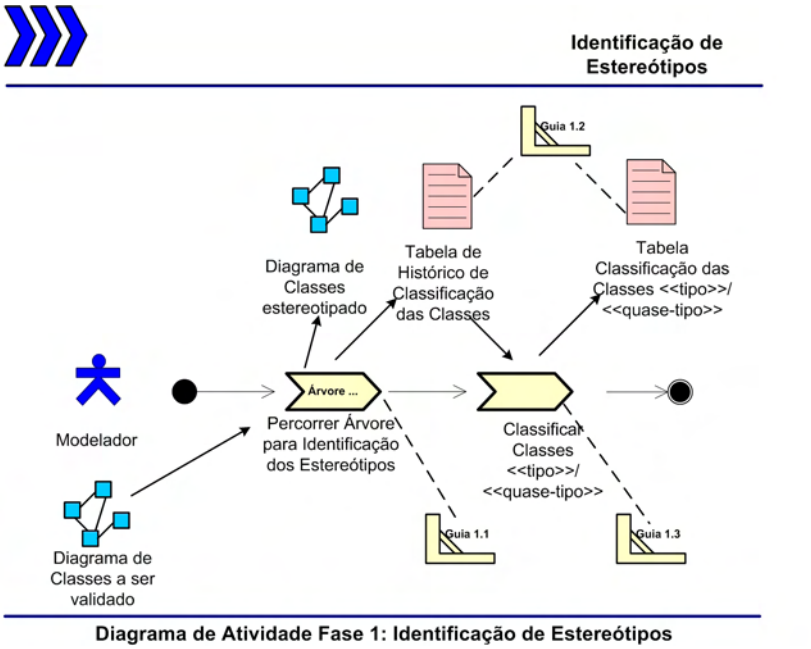


Figura 4.5: Diagrama de Atividades da Fase 1: Identificação de Estereótipos

tais classes os estereótipos identificados e gerando uma tabela que contém, para cada classe, um histórico do caminhamento na árvore. As informações desta tabela serão úteis caso o modelador necessite consultar como foi a classificação de uma determinada classe. Nessa tabela são anotados, para cada classe, qual foi a resposta a cada tipo de pergunta, bem como algumas informações como: qual a condição de identidade encontrada para as classes que possuem identidade (+I) e a qual classe é a dependência externa das classes que são dependentes (+D). Veja esboço da tabela na Figura A.9.

É importante verificar que optou-se em não distinguir nesta primeira fase do procedimento as classes que são classificadas como «tipo» e «quase-tipo». Percebe-se pela árvore de decisão que a classificação não foi definida, ficando identificada como «tipo»/«quase-tipo». A única diferença entre ser «tipo» e «quase-tipo» refere-se a identificar se a classe em análise fornece ou não uma identidade (+O, -O). Porém, concluir tal fato não é muito simples e estruturar uma pergunta que tornasse fácil ao modelador tal identificação mostrou-se inviável na prática.

A solução para tal problema foi buscar formas de conseguir classificar uma classe em «tipo» ou «quase-tipo» tendo como base uma ontologia topo. Buscou-se

auxílio na Ontologia Fundacional Unificada-UFO de Guizzardi et al. (2008) que tem por objetivo ser uma ontologia aplicável em quase todo tipo de domínio. A UFO, como descrito na Seção 2.2.1 é dividida em três partes, a UFO-A: ontologia de objetos; a UFO-B: ontologia de eventos; e a UFO-C: ontologia de entidades sociais. Analisou-se cada uma dessas três partes objetivando extrair categorias que fossem de fácil entendimento para o modelador e que juntas pudessem abranger grande parte das entidades modeladas num diagrama de classe. Percebeu-se então que, a maioria das classes de um domínio se enquadrariam em uma das seguintes categorias: '*Agente*', '*Objeto*', '*Evento*' ou '*Modo*'. Conforme descrito na Seção 2.2.1 a categoria *Agente* representa tanto entidades físicas quanto sociais, por exemplo, pessoa, organização. A categoria *Objetos* representa tanto objetos físicos quanto sociais, por exemplo: casa, carro, dinheiro, curso. A categoria *Evento* representa algo que sofre uma transformação, ou seja, algo que gera uma mudança de estado (Guizzardi and Wagner, 2004). Uma venda é algo que pertence à categoria *Evento*, pois quando ocorre uma venda ocorrem mudanças de estado: produtos são retirados do estoque de uma loja e passam a pertencer a um cliente e este transfere uma determinada quantia de dinheiro para a loja. Por fim a categoria *Modo* representa indivíduos que só podem existir em outros indivíduos, como por exemplo: cor, sintomas, etc.

Como o conjunto dessas categorias são bem abrangentes e como os exemplos de coisas pertencentes a cada uma delas deixa bem claro de que categoria se trata, espera-se que o modelador consiga enquadrar cada classe que tenha sido classificada como «tipo»/«quase-tipo» como sendo um '*Agente*', '*Objeto*', '*Evento*' ou '*Modo*'. Posteriormente, na fase 2, tal classificação permitirá distinguir uma classe «tipo»/«quase-tipo» em somente «tipo» ou somente «quase-tipo».

Voltando à Figura 4.5 tem-se a segunda atividade da primeira fase, intitulada Classificar as Classes «tipo»/«quase-tipo». Nesta atividade, cada classe previamente classificada como «tipo»/«quase-tipo» deverá ser enquadrada nas categorias: '*Agente*', '*Objeto*', '*Evento*', '*Modo*' ou '*Outros*', organizadas numa tabela como apresentado na Figura A.11. O grupo 'Outros' foi acrescentado com o objetivo de atender casos que o modelador não conseguir enquadrar a classe em algumas das catego-

rias escolhidas da Ontologia de Guizzardi et al. (2008). Com essa classificação, feita com base na Ontologia de Fundamentação Unificada, será possível levar o modelador a identificar de forma mais intuitiva quais de suas classes são «tipo» e quais são «quase-tipo», sem obrigá-lo a entender o significado do conceito filosófico de 'fornecer identidade (+O)'. Na Seção 4.2 será explicado em detalhes de que forma isso será possível.

4.2 Segunda Fase: Verificação Hierárquica para Relacionamentos de Generalização/Especialização

Ao concluir a primeira etapa do procedimento PrOntoCon têm-se em mãos uma tabela do histórico da classificação de cada classe, um Diagrama de Classes estereotipado, e uma tabela contendo uma organização das classes «tipo»/«quase-tipo» em cinco grupos: '*Agente*', '*Objeto*', '*Evento*', '*Modo*' e '*Outros*'. Tais informações são os artefatos de entrada para a segunda fase do procedimento PrOntoCon, intitulada Verificação Hierárquica. O objetivo principal desta fase é organizar e analisar todo o Diagrama de Classes de forma que todos os relacionamentos de generalização/especialização existentes estejam corretos. Além disso, verifica-se também nesta fase a presença obrigatória de determinados tipos de relacionamentos hierárquicos. Como produto final, tem-se um Diagrama de Classes completamente validado no que diz respeito a relacionamentos de generalização/especialização.

A Figura 4.6 representa a segunda fase do PrOntoCon, que é composta de três atividades. A primeira delas é responsável por definir a hierarquia e a classificação final das classes até então classificadas como «tipo»/«quase-tipo». Na segunda atividade deve-se checar se todos os relacionamentos hierárquicos existentes obedecem às restrições hierárquicas impostas pelas meta-propriedades e se não falta algum relacionamento hierárquico obrigatório conforme restrições impostas pela OntoCon. A terceira atividade da fase é responsável por verificar se outras restrições gerais de relacionamentos são devidamente obedecidas. Após passar pelas três atividades da etapa têm-se como artefato principal um Diagrama de Classes totalmente validado no que

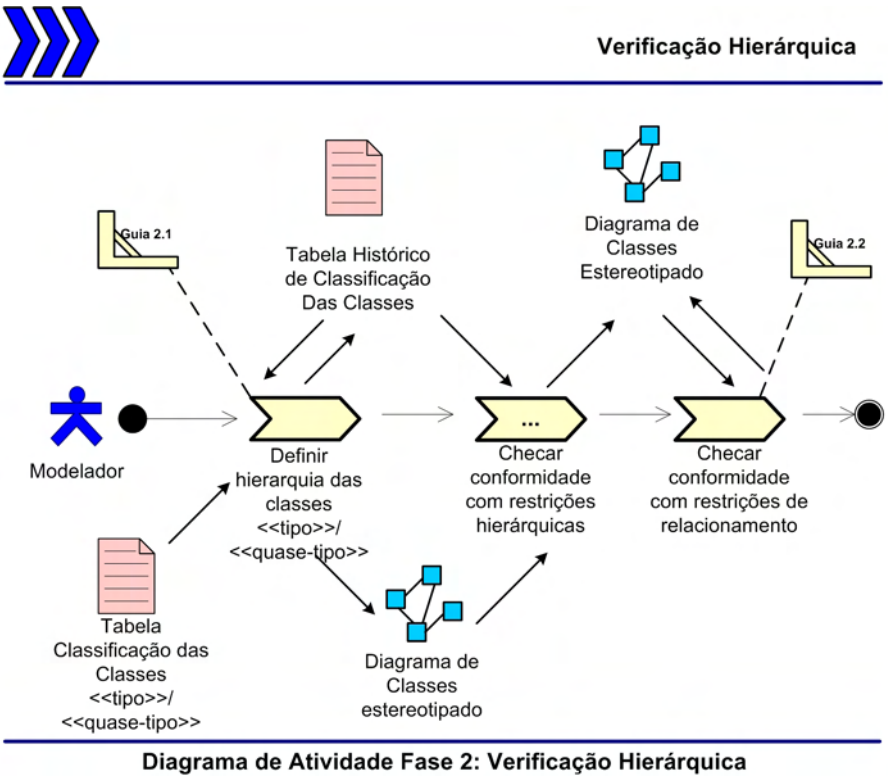


Figura 4.6: Diagrama de Atividades da Fase 2: Verificação Hierárquica dos relacionamentos de Generalização/Especialização.

se refere a relacionamentos de herança. Além disso a Tabela Histórico de Classificação das Classes é atualizada para obtenção de dados para fins históricos.

A primeira atividade desta fase: Definir Hierarquia das Classes «tipo»/«quase-tipo», é feita tendo em mãos dois artefatos produzidos na primeira fase do PrOntoCon: o Diagrama de Classes estereotipado e a Tabela de Classificação Classes «tipo»/«quase-tipo» que separa essas classes nos grupos: '*Agente*', '*Objeto*', '*Evento*', '*Modo*' e '*Outros*'. Para cada grupo desta tabela deve-se verificar se existem ou se deveriam existir, dentre as classes do grupo, relacionamentos caracterizados como generalização/especialização. Caso existam ou devam existir tais relacionamentos e eles possuírem mais de um nível hierárquico, as classes que ocuparem o nível mais alto da hierarquia deverão ser classificadas como «tipo». E a cada nível inferior, a classe que o ocupa será um «quase-tipo» quando sua identidade for a mesma da classe superior, caso contrário, será classificada como «tipo». Em situações onde houver apenas um nível hierárquico deve-se verificar primeiramente se a identidade da superclasse é a mesma das suas subclasses (identidade esta verificada na primeira fase do procedimento). Se isso acontecer, as subclasses deverão ser classificadas com o estereótipo «quase-tipo» e a superclasse deverá receber o estereótipo «tipo». Quando para um determinado grupo não existir ou não se detectar relacionamentos hierárquicos, significa que todas as classes devem ser classificadas como «tipo».

Todas as classes que foram classificadas como «tipo» são as fornecedoras de identidade (+O) e as subclasses («quase-tipo») apenas carregam consigo tal identidade sem a fornecer (-O). Percebe-se, então, que foi possível conduzir o modelador a diagnosticar se a classe fornece ou não identidade sem que se passasse de forma explícita pelo conceito de 'fornecer identidade', atingindo, portanto, o objetivo proposto quando se utilizou a Ontologia de Fundamentação Unificada de Guizzardi et al. (2008).

Todas as definições de classes «tipo» e classes «quase-tipo» devem ser anotadas no Diagrama de Classes, obtendo-se um diagrama devidamente estereotipado, seguindo os estereótipos presentes da Técnica OntoCon.

O próximo passo é realizar a verificação das restrições hierárquicas presentes

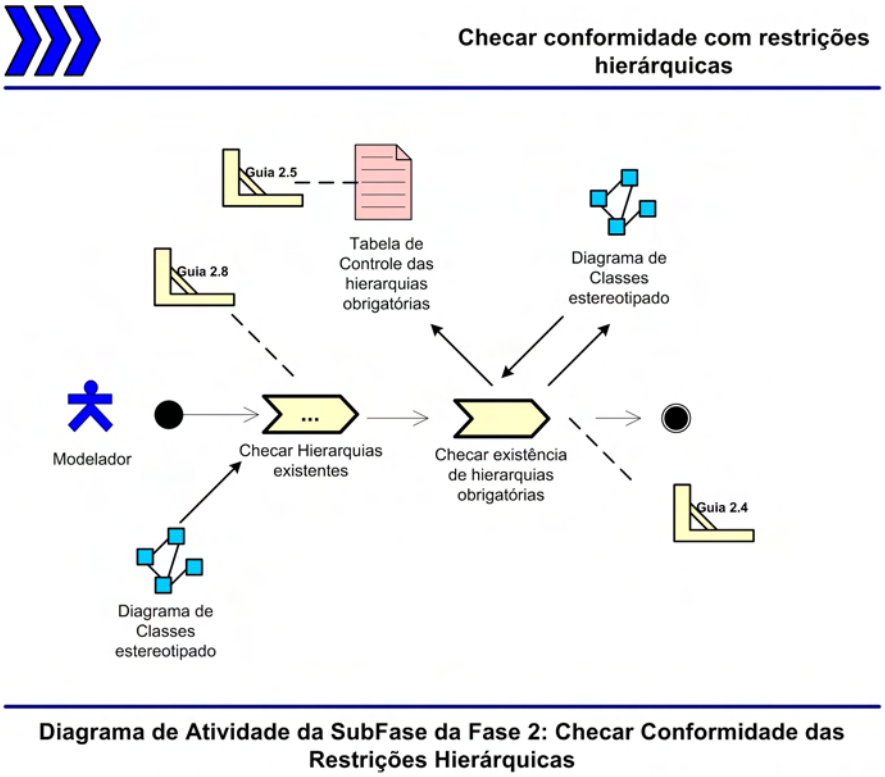


Figura 4.7: Diagrama de Atividade da SubFase da Fase 2: Checar Conformidade das Restrições Hierárquicas.

na OntoCon, herdadas da VERONTO e do Perfil OntoUML. A atividade identificada na Figura 4.6 como Checar Conformidade com Restrições Hierárquicas é responsável por tal tarefa. Tal atividade, quando acessada, direciona ao diagrama de atividades mostrado na Figura 4.7. Em tal diagrama tem-se inicialmente a atividade Checar Hierarquias Existentes.

O objetivo desta atividade é conduzir o modelador a checar se, para cada grupo de classes com relacionamento de generalização/especialização, as restrições hierárquicas foram devidamente seguidas. Caso seja detectado algum tipo de erro, o modelador é encaminhado a reavaliar seu modelo ou a aplicar uma solução para resolver o erro encontrado. A Figura 4.8 apresenta um fluxograma que deve ser seguido pelo modelador. Através do Guia 2.3 (Figura B.6 e B.7), relativo à atividade Verificar Supertipos e Subtipos, o modelador terá acesso a todos os tipos de hierarquias permitidas num diagrama de classes estereotipado com os estereótipos da técnica OntoCon. Caso

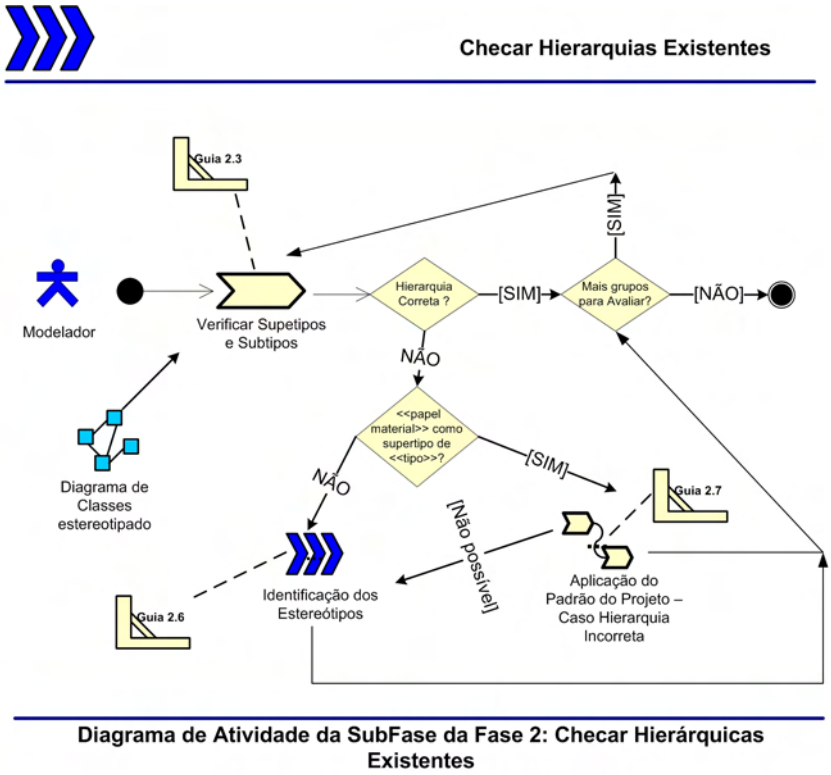


Figura 4.8: Diagrama de Atividade da SubFase da Fase 2: Checar Hierárquicas Existentes.

exista um erro de hierarquia deve-se verificar se foi por ter encontrado uma classe estereotipada como «papel material» sendo supertipo de uma classe estereotipada como «tipo». Se for esse o caso, deve-se passar para uma etapa da terceira fase do procedimento que trata da aplicação do Padrão de Projeto na modelagem de papéis proposto por Guizzardi (2005), para resolver este tipo de problema. O método para aplicação do padrão de projeto será explicado na Seção 4.3. Porém, se o erro encontrado ocorreu entre outros tipos de estereótipos que não sejam «tipo» e «papel material» ou não foi possível aplicar o padrão de projeto, deve-se retornar a primeira fase do procedimento PrOntoCon para tentar identificar um possível erro de classificação dos estereótipos das classes envolvidas. Veja Figura 4.8.

A atividade Checar Hierarquias Existentes só termina quando todos os grupos envolvidos com relacionamentos de generalização/especialização forem checados. Tem-se, então, que passar à atividade Checar Existência de Hierarquias Obrigatórias,

cujo objetivo é verificar se todos os relacionamentos hierárquicos obrigatórios foram criados. Tal atividade está indicada na Figura 4.7.

Na Técnica OntoCon, as restrições sobre relacionamentos hierárquivos obrigatórios são descritos na Tabela 3.3. Classes estereotipadas como «quase-tipo» têm, obrigatoriamente, que ser subclasse de uma classe estereotipada como «tipo», direta ou indiretamente. Caso exista alguma classe «quase-tipo» que não seja subclasse de um «tipo» deve-se criar tal classe e os relacionamentos da classe «quase-tipo» devem ser reavaliados. Em alguns casos tais relacionamentos passarão para a superclasse.

O mesmo acontece para classes estereotipadas como «fase» e «papel material». Essas classes têm, obrigatoriamente, que ser subclasse de uma classe estereotipada como «tipo», direta ou indiretamente. Caso exista alguma classe «fase» ou «papel material» que não seja subclasse de um «tipo», deve-se criar tal classe e os relacionamentos que pertencem à subclasse devem ser reavaliados. Em alguns casos tais relacionamentos passarão para a superclasse. Deve-se estar atento para o caso de existir mais de uma classe estereotipada como «fase». Verificar se elas possuem uma superclasse comum é uma boa prática para se evitar erros de modelagem. A mesma observação é válida quando houver mais de uma classe «papel material».

Percebe-se que em vários momentos da atividade Checar Existência de Hierarquias Obrigatórias, novas classes «tipo» serão necessárias. Porém, antes de criar uma superclasse «tipo» para classes «fase», «papel material» ou «quase-tipo», deve-se verificar com atenção todo o Diagrama de Classes, pois talvez já exista uma superclasse para a classe em análise e não se tenha estabelecido um relacionamento de generalização/especialização entre elas.

Ao terminar a atividade responsável por checar as heranças obrigatórias, conclui-se também toda a seqüência do diagrama de atividades da Figura 4.7, produzindo como artefato uma Tabela de Controle das Hierarquias Obrigatórias. O objetivo deste artefato é construir um histórico das alterações que são feitas pelo PrOntoCon no Diagrama de classes. O layout desta tabela pode ser visto na Figura B.11. Retorna-se, então, ao diagrama de atividades da Figura 4.6, tendo como próximo passo a execução da atividade de Checar Conformidade com Restrições de Relacionamento.

Tal atividade é responsável por garantir o cumprimento da dependência externa das classes «papel material». Toda classe estereotipada como «papel material» deve ter um relacionamento de dependência com outra classe. Tal relacionamento expressa a dependência da classe «papel material» com outra classe, ou seja, para que instâncias da classe estereotipada como «papel material» existam é necessário a existência de instâncias dessa outra classe. Caso não exista tal relacionamento deve existir uma superclasse da classe «papel material» classificada como «papel formal» e esta deverá possuir tal relacionamento de dependência. Além disso é preciso verificar a cardinalidade do relacionamento da classe «papel material» ou «papel formal» com tal classe, que deve ser de no mínimo 1 (um).

Outro objetivo desta última atividade da segunda fase é garantir que as classes classificadas como «fase» formem um conjunto disjunto. Desta forma, para relacionamentos de especialização onde as subclasses sejam estereótipos «fase» deve-se estar atento ao fato de que uma instância da superclasse não pode ser instância de mais de uma subclasse «fase» ao mesmo tempo.

4.3 Terceira Fase: Aplicação de Padrão de Projeto para Modelagem de Papéis

Papéis representam uma noção fundamental quando conceitua-se a realidade de um domínio e por isso tem recebido muita atenção por parte dos pesquisadores (Guizzardi, 2005), cujo objetivo é resolver alguns problemas envolvendo modelagem de papéis. Alguns desses problemas são denominados por Guizzardi (2005) como *o problema de papéis com múltiplos tipos disjuntos*. Na Seção 2.3.2 tais problemas foram detalhados.

A terceira fase do procedimento PrOntoCon foi criada com o objetivo de auxiliar o modelador a aplicar o padrão de projeto apresentado por Guizzardi (2005). Em dois momentos distintos do procedimento PrOntoCon o modelador pode detectar a necessidade de aplicar o padrão de projeto. O primeiro deles é quando se está na segunda fase do procedimento, realizando a atividade de Checar as Hierarquias Existentes, na qual se pode detectar o seguinte tipo de hierarquia incorreta: um «papel

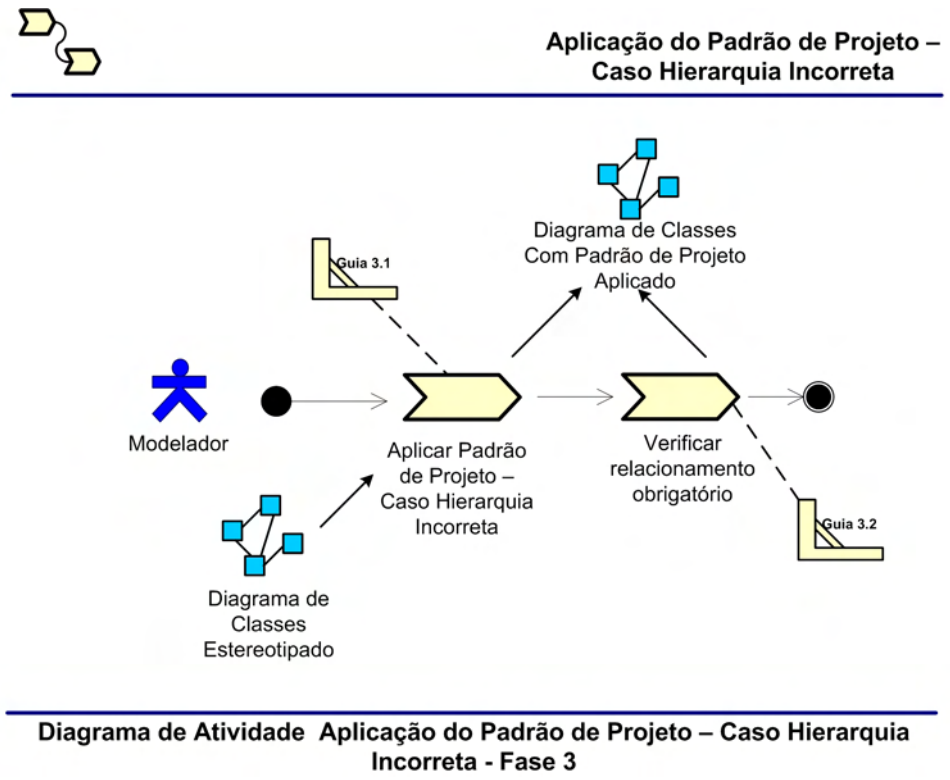


Figura 4.9: Diagrama de Atividades - Aplicação Padrão de Projeto - Caso Hierarquia Incorreta.

material» como supertipo de um «tipo». Tal momento pode ser observado na Figura 4.8, onde existe uma resposta afirmativa para um erro hierárquico entre «papel material» e «tipo». A partir desse ponto, o procedimento guia o modelador a aplicar o primeiro caso do uso do padrão de projeto proposto por Guizzardi (2005), representado no procedimento com um diagrama de atividades intitulado Aplicar Padrão de Projeto - Caso Hierarquia Incorreta. A Figura 4.9 mostra as duas atividades deste diagrama.

A primeira delas, Aplicar o Padrão de Projeto - Caso Hierarquia Incorreta, irá levar o modelador a observar as seguintes itens:

- Verificar se a classe A (Figura 4.10-a) estereotipada como «papel material» não deveria possuir, ou possui, vários subtipos que obedeçam a princípios de identidade diferentes. Por exemplo, uma classe Cliente pode ser uma superclasse das classes ClientePrivado e ClienteCorporativo que possuem princípios de identi-

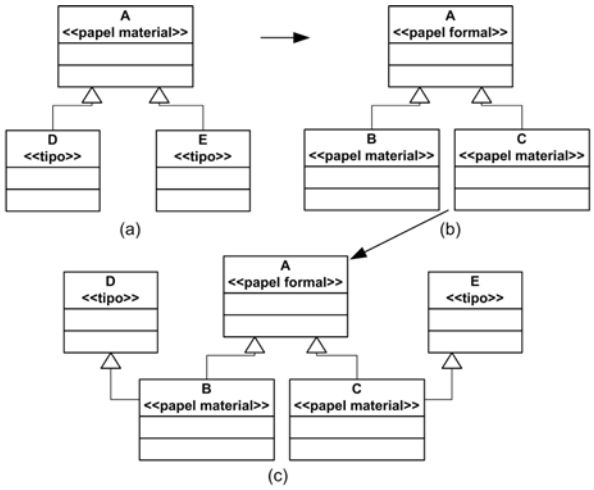


Figura 4.10: Transformações ocorridas na aplicação do Padrão de Projeto para o caso de hierarquia incorreta. Fonte: adaptada Guizzardi et al. (2004b).

dades herdados de outras classes. Ou seja, tais classes são identificadas de forma diferente.

- Caso esses subtipos não existam, deve-se retornar à segunda fase, no ponto onde se detectou o erro na hierarquia e considerar que não foi possível aplicar o padrão de projeto e então repetir o processo de Identificação de Estereótipos (Figura 4.8), pois provavelmente houve um erro de classificação.

- Caso o modelador detecte a existência desses subtipos, a classe A passa a ser uma generalização de seus subtipos. A classe A deve ser na realidade um «papel formal» e seus subtipos serão classificados como «papel material». Tal transformação pode ser observada na Figura 4.10-b, onde os subtipos estão identificados com B e C.

- Identificar quais são as classes que fornecem os princípios de identidade para as subclasses B e C. Provavelmente tais classes são aquelas que inicialmente estavam como subclasses da classe «papel material» (classes D e E). Uma vez que elas irão fornecer identidade, terão que ser classificados como «tipo». Tal transformação pode ser observada na Figura 4.10-c.

Na Figura 4.11 observa-se a aplicação do Padrão de Projeto utilizando-se um

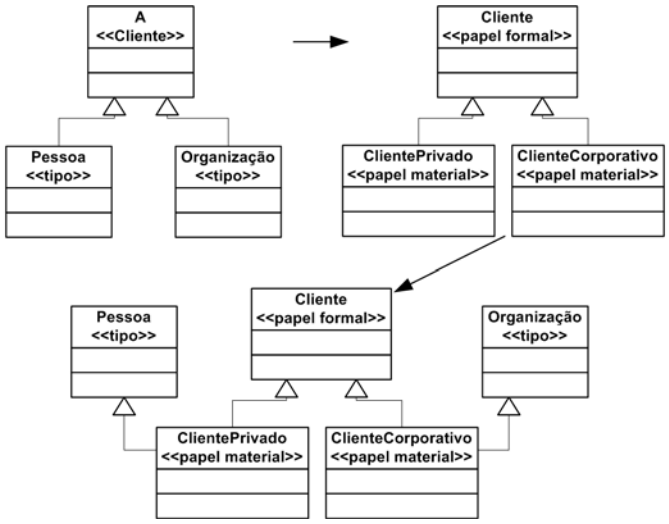


Figura 4.11: Um exemplo de aplicação do Padrão de Projeto para o Caso de Hierarquia Incorreta. Fonte: adaptada de Guizzardi et al. (2004b).

exemplo. Primeiramente é apresentado uma hierarquia incorreta entre as classes Cliente, Pessoa e Organização. Num segundo passo, são identificadas as possíveis suclasses de Cliente: ClientePrivado e ClienteCorporativo. Por fim cada classe «papel material» é associada à classe «tipo» correspondente, ficando a classe Pessoa como superclasse de ClientePrivado e logo provedora do princípio de identidade desta. Para a classe ClienteCorporativo tem-se a superclasse Organização como fornecedora de identidade.

O segundo caso da aplicação do Padrão de Projeto proposto por Guizzardi (2005) tem por objetivo resolver outro problema de modelagem de papéis, intitulado aqui de Caso Herança Múltipla. Este caso será utilizado quando o modelador concluir a segunda fase do PrOntoCon e passar para a terceira fase, acessando diretamente o diagrama de atividades Aplicação do Padrão de Projeto, como indicado na Figura 4.1, mostrada no início deste capítulo. Nesta fase, tendo em mãos o diagrama de classe estereotipado, dá-se início à realização do diagrama de atividades Aplicação do Padrão de Projeto - Caso Herança Múltipla cujas atividades são mostradas na Figura 4.12. Cada classe «papel material» deverá passar por uma análise de seus supertipos.

Para as classes «papel material» cuja análise de hierarquia múltipla com su-

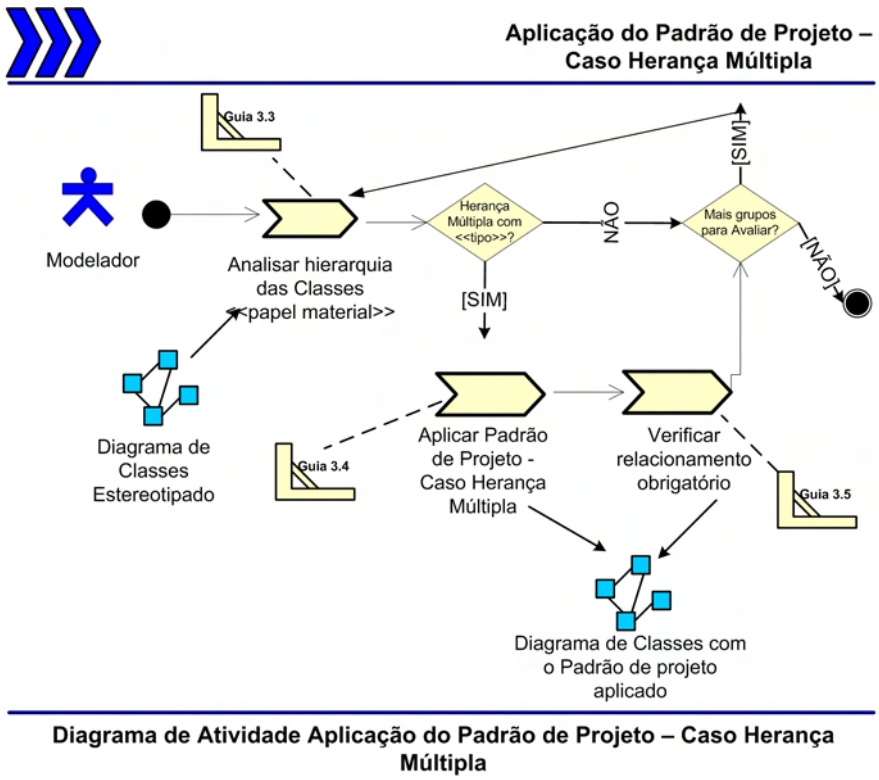


Figura 4.12: Diagrama de Atividade - Aplicação do Padrão de Projeto - Caso Herança Múltipla.

pertipo «tipo» for positiva deve-se executar a atividade Aplicar Padrão de Projeto - Caso Herança Múltipla, como observado no fluxo do Diagrama de Atividades da Figura 4.12. Nesta atividade tem-se que realizar as seguintes ações:

- A classe em análise, estereotipada como «papel material» deve ser transformada em um «papel formal» (Figura 2.7-a).
- Deverão ser criadas tantas subclasses «papel material» quantas forem as superclasses «tipo» da classe «papel material» inicial, caso já não existam tais classes (Figura 2.7-b).
- Cada classe «papel material», além de ser subclasse da classe «papel formal», será subclasse também da classe «tipo» correspondente (Figura 2.7-c).

Como exemplo deste segundo caso de aplicação do padrão de projeto pode-se ter em um diagrama a seguinte situação: uma classe «papel material» Cliente que seja subclasse das classes Pessoa e Organização. Seguindo as instruções do PrOntoCon consegue-se aplicar o padrão de projeto e tem-se um resultado idêntico ao mostrado na Figura 4.13.

Para concluir a aplicação do padrão de projeto, tanto para o caso Hierarquia Incorreta, quanto o caso Herança Múltipla, tem-se que realizar a atividade Verificar Relacionamento Obrigatório (Figuras 4.9 e 4.12). Tal atividade foi colocada para fazer com que o modelador garanta a existência de no mínimo um relacionamento de dependência para a classe «papel formal». Tal relacionamento é obrigatório, uma vez que está se trabalhando com classes que representam papéis, e tais classes possuem a propriedade dependência externa (+D), o que implica em ser dependente de outra classe para existir. Deve-se, então, checar a existência de um relacionamento de dependência entre a classe A «papel formal» e uma classe F qualquer, sendo a cardinalidade desse relacionamento de no mínimo 1. Se não existir tal relacionamento e tal classe, eles deverão ser criados. A Figura 4.14 mostra como fica o esquema final de uma aplicação do Padrão de Projeto da OntoCon para qualquer um dos casos e mostra também um exemplo final.

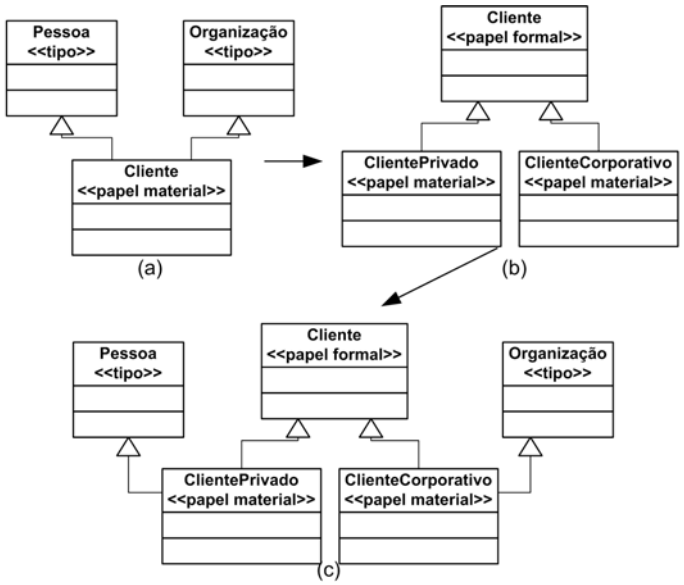


Figura 4.13: Um exemplo de aplicação do Padrão de Projeto para o Caso de Herança Múltipla.

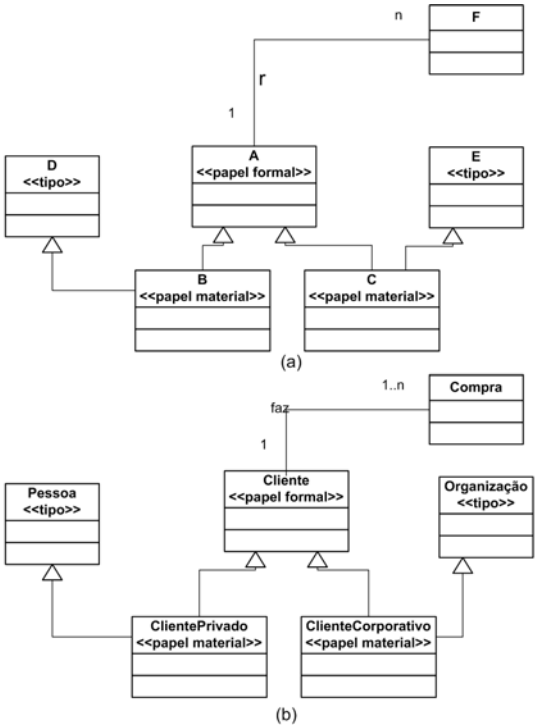


Figura 4.14: (a) Resultado final da aplicação do Padrão de Projeto para os dois casos: hierarquia incorreta e herança múltipla de «tipo» (Guizzardi, 2005). (b) Exemplo adaptado de Guizzardi et al. (2004b).

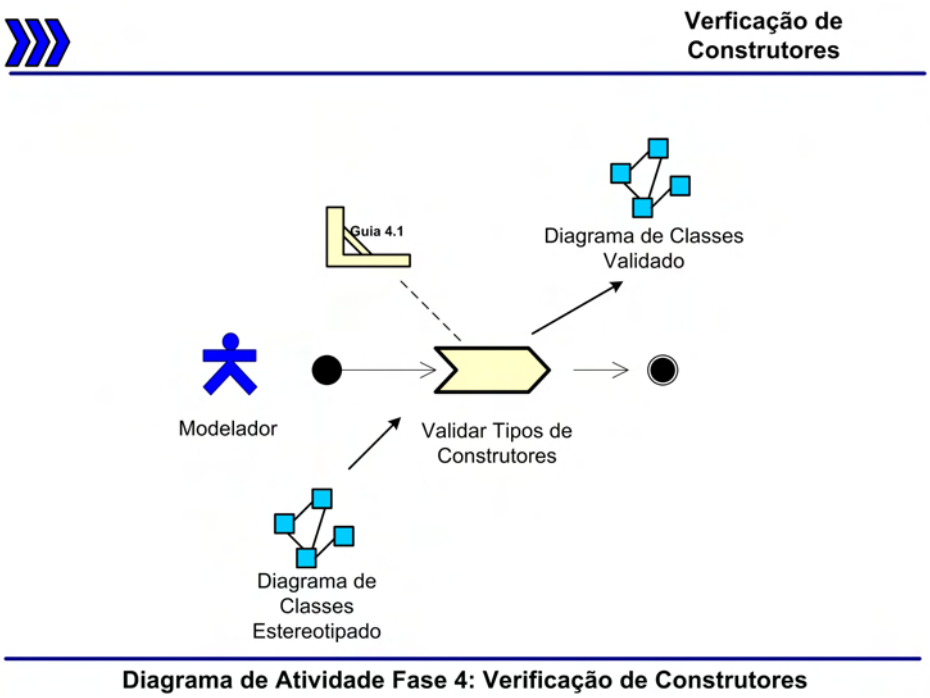


Figura 4.15: Diagrama de Atividades da Fase 4 - Verificação de Construtores UML.

4.4 Quarta Fase: Verificação de Construtores UML

A quarta e última fase do procedimento PrOntoCon diz respeito aos contrutores UML aplicados às classes do diagrama de classes. A identificação das classes como concreta, abstrata ou simplesmente como interface é extremamente relevante tanto para o modelador, quanto para o desenvolvedor, principalmente em se tratando de manutenibilidade do sistema. Justifica-se, então, estar atento a algumas restrições e possibilidades de uso desses construtores.

Nomeada de Verificação de Construtores, a quarta fase do PrOntoCon possui apenas uma atividade, como mostrado na Figura 4.15. Tal fluxo é bem simples e requer que o modelador tenha em mãos o Diagrama de Classes devidamente estereotipado. Na atividade Validar Tipos de Construtores, cada classe desse diagrama passa por uma verificação que segue as instruções da Tabela 4.1, embasadas na Técnica OntoCon. Gera-se, assim, o Diagrama de Classes validado conforme proposto no presente trabalho.

Tabela 4.1: Possibilidades e restrições quanto ao uso de construtores UML

Estereótipo	Construtor UML Permitido
«tipo»	-Classe Abstrata; -Classe Concreta; - Restrição: quando um elemento do diagrama de classe estiver classificado como «tipo» e possuir como especialização elementos classificados como «fase», tal elemento deverá ser uma classe abstrata ou uma interface.
«quase-tipo»	-Classe Abstrata; -Classe Concreta.
«papel material»	-Classe Abstrata; -Classe Concreta.
«fase»	-Classe Abstrata; -Classe Concreta.
«categoria»	-Interface; -Classe Abstrata.
«papel formal»	-Interface; -Classe Abstrata.

Capítulo 5

Estudo de Casos

Nesta seção serão abordados dois domínios diferentes já modelados conceitualmente por meio de diagramas de classes UML. A cada diagrama será aplicado o PrOntoCon com o objetivo de mostrar não só exemplos de uso do procedimento, bem como também melhorias proporcionadas pela aplicação do mesmo.

5.1 Domínio de Controle de Atividades de Estagiários em Psicologia

O primeiro domínio ao qual o procedimento PrOntoCon foi aplicado é o domínio de controle de atividades feitas por alunos estagiários do curso de Psicologia do Unileste-MG - Centro Universitário do Leste de Minas Gerais (Balbino, 2005). Tal escolha foi feita em função do domínio tratar de um assunto relacionado ao meio acadêmico, facilitando, assim, o entendimento do mesmo. Outro fator que determinou a escolha desse domínio foi o fato da modelagem ter sido feita por profissionais pertencentes à Fábrica de Software do Unileste-MG, tais profissionais possuem experiência tanto acadêmica quanto prática em modelagem conceitual utilizando diagramas de classe UML.

Primeiramente é importante descrever o domínio para que se possa entender corretamente a aplicação do PrOntoCon. Todo aluno do curso de Psicologia tem

que fazer atividades de estágio curricular. Tal estágio é desenvolvido na própria instituição de ensino através de projetos diversos e de atendimentos psicológicos à sociedade, com o suporte dos professores. Foi solicitado à Fábrica de Software do Unileste-MG um sistema que fizesse o controle dos alunos estagiários do curso no que diz respeito a atividades de estágio desenvolvidas. Professores e alunos podem formar grupos com o objetivo de desenvolverem projetos em instituições diversas com as quais são feitos convênios. Cada um desses grupos de projeto pode desenvolver várias atividades caracterizadas como atividades de estágio. Os alunos também podem fazer atendimentos a pacientes desde que sob a orientação de um professor. Para realizar esses atendimentos são feitos agendamentos conforme a demanda. A Figura 5.1 representa o diagrama de classes UML deste domínio.

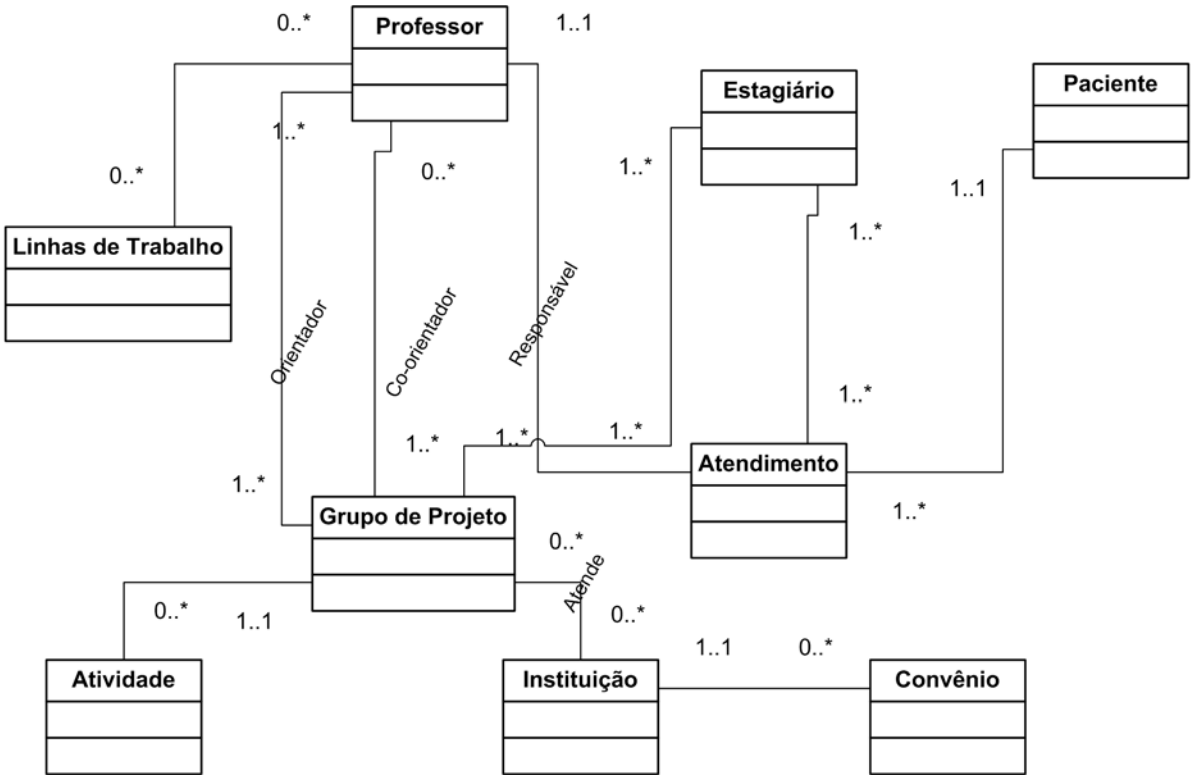


Figura 5.1: Diagrama de Classes UML original do domínio que representa o controle das atividades dos estagiários do curso de Psicologia do Unileste-MG.

Nas seções seguintes são mostradas a aplicação e os resultados de cada fase do procedimento PrOntoCon. Todas as fases, subfases e guias do procedimento estão

apresentadas nos Anexos A, B, C e D.

5.1.1 Aplicação da Fase 1 do PrOntoCon

A primeira fase do PrOntoCon possui um diagrama de atividade denominado Identificação de Estereótipos (Figura A.2), cuja primeira atividade é Percorrer Árvore para Identificação dos Estereótipos. Nessa atividade cada classe foi analisada com base nas perguntas existentes nos nós internos da árvore até se chegar à sua classificação final presente nos nós folhas (Figura A.4). Detalhes da análise de cada classe são apresentadas a seguir.

Sobre a Classe Professor:

A classe Professor representa as informações necessárias sobre os professores do curso de Psicologia que participem de quaisquer atividades relacionadas a estágio curricular.

Para a classe Professor foi respondido **sim** para a Questão I - Possui Identidade. Tal resposta foi dada com base em que cada objeto da classe Professor possui uma resposta única para alguma característica, tal como a impressão digital. Caminhando na árvore tem-se a próxima questão: Questão II - Rigidez, para a qual foi dada a resposta **não**, uma vez que um objeto que representa uma instância da classe em análise não permanecerá para sempre sendo uma instância desta classe, ou seja, uma instância da classe Professor não será necessariamente, um professor durante toda sua existência. Seguindo na árvore tem-se a Questão III -Dependência para a qual foi respondido **sim**, uma vez que para ser professor participante do controle de atividades de estágio deverá estar envolvido no mínimo com um grupo de projeto ou com um atendimento. Desta forma se chega a um nó folha da árvore de decisão com a classe classificada como **«papel material»**. Detalhes das perguntas relativas às questões I, II e III podem ser vistas nas Figuras A.5, 4.4 e A.7, respectivamente.

Sobre a Classe Estagiário:

A classe Estagiário representa as informações necessárias sobre os alunos esta-

giários do curso de Psicologia que participem de quaisquer atividades relacionadas a estágio curricular.

Para a classe Estagiário foi respondido **sim** para a Questão I - Possui Identidade. Tal resposta foi dada com base em que cada objeto da classe Estagiário possui uma resposta única para a característica impressão digital. Caminhando na árvore tem-se a próxima questão: Questão II - Rigidez, para a qual foi dada a resposta **não**, uma vez que um objeto que representa uma instância da classe em análise não permanecerá para sempre sendo uma instância desta classe, ou seja, uma instância da classe Estagiário não será necessariamente um estagiário durante toda sua existência. Seguindo na árvore tem-se a Questão III -Dependência para a qual foi respondido **sim**, uma vez que para ser estagiário participante do controle de atividades de estágio deverá estar envolvido no mínimo com um grupo de projeto ou com um atendimento. Desta forma se chega a um nó folha da árvore de decisão com a classe classificada como **«papel material»**. Detalhes das perguntas relativas às questões I, II e III podem ser vistas nas Figuras A.5, 4.4 e A.7, respectivamente.

Sobre a Classe Paciente:

A classe Paciente representa as informações necessárias sobre os pacientes que participam de atendimentos psicológicos realizados por estagiários do curso de Psicologia.

Para a classe Paciente foi respondido **sim** para a Questão I - Possui Identidade. Tal resposta foi dada com base em que cada objeto da classe Paciente possui uma resposta única para a característica impressão digital. Caminhando na árvore tem-se a próxima questão: Questão II - Rigidez, para a qual foi dada a resposta **não**, uma vez que um objeto que representa uma instância da classe em análise não permanecerá para sempre sendo uma instância desta classe, ou seja, uma instância da classe Paciente não será necessariamente um paciente durante toda sua existência. Seguindo na árvore tem-se a Questão III -Dependência para a qual foi respondido **sim**, uma vez que para ser um paciente participante do controle de atividades de estágio deverá ter participado de no mínimo um atendimento realizado por um estagiário do curso

de Psicologia. Desta forma se chega a um nó folha da árvore de decisão com a classe classificada como **«papel material»**. Detalhes das perguntas relativas às questões I, II e III podem ser vistas nas Figuras A.5, 4.4 e A.7, respectivamente.

Sobre a Classe Linhas de Trabalho:

A classe Linhas de Trabalho representa a descrição de todas as possíveis linhas de trabalho que possam ser seguidas pelos professores.

Para a classe Linhas de Trabalho foi respondido **sim** para a Questão I - Possui Identidade. Tal resposta foi dada com base em que cada objeto da classe Linhas de Trabalho possui uma resposta única para a característica campo de atuação. Entende-se que cada linha trabalho é criada para trabalhar em um determinado campo de atuação. Caminhando na árvore tem-se a próxima questão: Questão II - Rigidez, para a qual foi dada a resposta **sim**, uma vez que um objeto que representa uma instância da classe em análise permanecerá durante toda sua existência sendo uma instância desta classe. Desta forma se chega a um nó folha da árvore de decisão com a classe Linhas de Trabalho classificada como **«tipo»/«quase-tipo»**. Detalhes das perguntas relativas às questões I e II podem ser vistas nas Figuras A.5 e 4.4, respectivamente.

Sobre a Classe Grupo de Projeto:

A classe Grupo de Projeto representa os grupos que são formados a cada semestre tendo a participação de vários alunos, sob a orientação e co-orientação de professores. O objetivo desses grupos é desenvolver várias atividades em instituições diversas conveniadas com o Unileste-MG.

Para a classe Grupo de Projeto foi respondido **sim** para a Questão I - Possui Identidade. Tal resposta foi dada com base em que cada objeto da classe Grupo de Projeto possui um propósito único num determinado tempo e lugar, que o identifica dentre os demais grupos. Cabe nesse ponto uma consideração importante: muitas vezes pode-se entender que o número do grupo de projeto é a sua identidade. Na realidade, o número do grupo de projeto não é a identidade no sentido filosófico, mas não deixa de ser uma forma de representar a identidade de cada grupo do projeto.

Na maioria das vezes o modelador visualizará esta característica como sendo a identidade de tal classe. Caminhando na árvore tem-se a próxima questão: Questão II - Rigidez, para a qual foi dada a resposta **sim**, uma vez que um objeto que representa uma instância da classe em análise permanecerá durante toda sua existência sendo uma instância desta classe. Desta forma se chega a um nó folha da árvore de decisão com a classe Grupo de Projeto classificada como **«tipo»/«quase-tipo»**. Detalhes das perguntas relativas às questões I e II podem ser vistas nas Figuras A.5 e 4.4, respectivamente.

Sobre a Classe Atividade:

A classe Atividade representa as informações relativas às atividades realizadas por um determinado grupo de projeto. Data, descrição da atividade, tempo gasto na realização da atividade e tipo de atividade são os atributos de tal classe.

Para a classe Atividade foi respondido **sim** para a Questão I - Possui Identidade. Tal resposta foi dada com base em que cada objeto da classe Atividade possui um único propósito a ser atingido num determinado tempo e lugar. Caminhando na árvore tem-se a próxima questão: Questão II - Rigidez, para a qual foi dada a resposta **sim**, uma vez que um objeto que representa uma instância da classe em análise permanecerá durante toda sua existência sendo uma instância desta classe. Desta forma se chega a um nó folha da árvore de decisão com a classe Atividade classificada como **«tipo»/«quase-tipo»**. Detalhes das perguntas relativas às questões I e II podem ser vistas nas Figuras A.5 e 4.4, respectivamente.

Sobre a Classe Instituição:

A classe Instituição representa as informações relativas às instituições para as quais são prestadas as atividades realizadas pelos grupos de projeto.

Para a classe Instituição foi respondido **sim** para a Questão I - Possui Identidade. Tal resposta foi dada com base em que cada objeto da classe Instituição possui uma única missão.

Na maioria das vezes o modelador visualizará a característica CNPJ como

sendo a identidade de tal classe. Comumente poderá-se entender que o CNPJ de uma instituição é a sua identidade. Na realidade, o CNPJ não é a identidade no sentido filosófico, mas não deixa de ser uma forma de representar a identidade de cada instituição. Será, então, adotado neste trabalho esta forma de raciocínio em casos semelhantes e, doravante, não será mencionada a distinção do conceito filosófico da identidade e as convenções sociais que refletem sua existência.

Caminhando na árvore tem-se a próxima questão: Questão II - Rigidez, para a qual foi dada a resposta **sim**, uma vez que um objeto que representa uma instância da classe em análise permanecerá durante toda sua existência sendo uma instância desta classe. Desta forma se chega a um nó folha da árvore de decisão com a classe Instituição classificada como **«tipo»**/**«quase-tipo»**. Detalhes das perguntas relativas às questões I e II podem ser vistas nas Figuras A.5 e 4.4, respectivamente.

Sobre a Classe Convênio:

A classe Convênio representa as informações relativas aos convênios que são firmados com as instituições que irão receber as atividades desenvolvidas pelos grupos de pesquisa.

Para a classe Convênio foi respondido **sim** para a Questão I - Possui Identidade. Tal resposta foi dada com base em que cada objeto da classe Convênio possui um único propósito num determinado tempo e espaço. Caminhando na árvore tem-se a próxima questão: Questão II - Rigidez, para a qual foi dada a resposta **sim**, uma vez que um objeto que representa uma instância da classe em análise permanecerá durante toda sua existência sendo uma instância desta classe. Desta forma se chega a um nó folha da árvore de decisão com a classe Convênio classificada como **«tipo»**/**«quase-tipo»**. Detalhes das perguntas relativas às questões I e II podem ser vistas nas Figuras A.5 e 4.4, respectivamente.

Sobre a Classe Atendimento:

A classe Atendimento representa as informações relativas aos atendimentos feitos pelos estagiários a pacientes, sob a responsabilidade de um professor. Algumas

dessas informações são: data, descrição do atendimento e tempo gasto.

Para a classe Atendimento foi respondido **sim** para a Questão I - Possui Identidade. Tal resposta foi dada com base em que cada objeto da classe Atendimento é único, ou seja, nenhum atendimento será igual uma vez que se trata de um evento que é identificável dentro de um determinado tempo e espaço. Caminhando na árvore tem-se a próxima questão: Questão II - Rigidez, para a qual foi dada a resposta **sim**, uma vez que um objeto que representa uma instância da classe em análise permanecerá durante toda sua existência sendo uma instância desta classe. Desta forma se chega a um nó folha da árvore de decisão com a classe Atendimento classificada como **«tipo»/«quase-tipo»**. Detalhes das perguntas relativas às questões I e II podem ser vistas nas Figuras A.5 e 4.4, respectivamente.

Após concluída a verificação de todas as classes do diagrama de classes UML tem-se como resultado um diagrama de classes UML com os estereotipos até então detectados para cada classe e uma tabela de Histórico de Classificação das Classes, como a mostrada na Figura A.11, devidamente preenchida. Passa-se então à próxima atividade da Fase 1 do PrOntoCon: Classificar Classes «tipo»/«quase-tipo».

Em tal atividade, cada classe classificada como «tipo»/«quase-tipo» deverá ser distribuída nos seguintes grupos: *Agente*', '*Objeto*', '*Evento*', '*Modo*' e '*Outros*'. Seguindo as orientações do Guia 1.3 (Figura A.12) as classes Atendimento e Atividade foram classificadas como pertencentes ao grupo *Evento* por se tratarem de algo que gera uma transformação. As classes Grupo de Projeto, Instituição e Convênio foram classificadas como pertencentes ao grupo *Agente* por se tratarem de agentes sociais. A classe Linhas de Pesquisa foi classificada como pertencente ao grupo *Objeto*, por se tratar de um objeto social.

Conclui-se, assim, a primeira fase do procedimento PrOntoCon, tendo em mãos três artefatos: o diagrama de classes parcialmente estereotipado (Figura 5.2), a tabela de Histórico de Classificação das Classes e a tabela com Classificação das Classes «tipo»/«quase-tipo», como estrutura mostrada na Figura A.11.

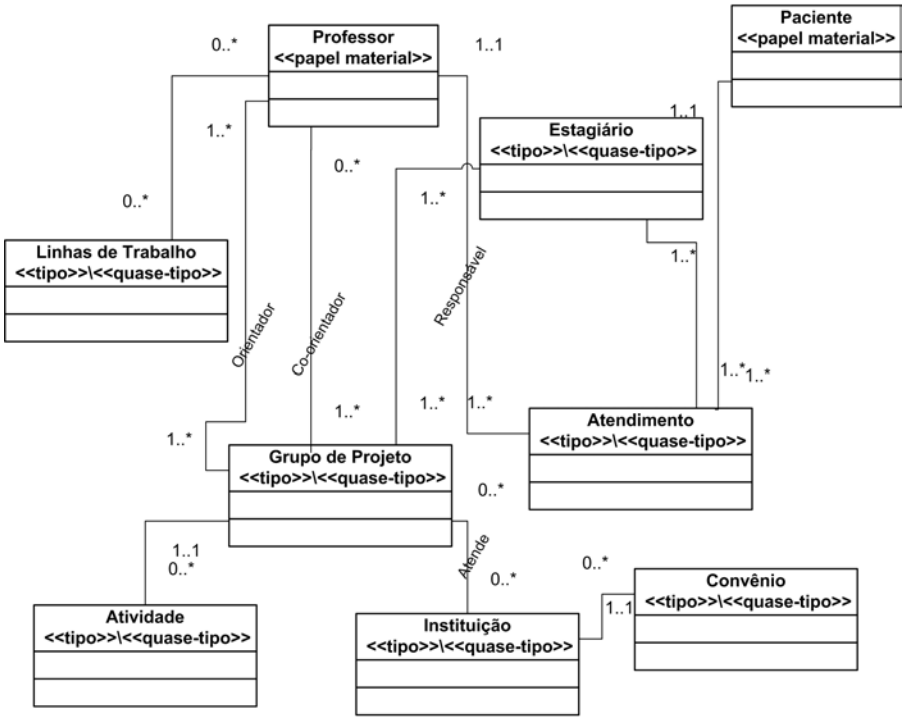


Figura 5.2: Diagrama parcialmente estereotipado após término da fase 1 do PrOnto-Con.

5.1.2 Aplicação da Fase 2 do PrOntoCon

A segunda fase do PrOntoCon, intitulada Verificação Hierárquica tem como principal objetivo fazer as verificações necessárias nas classes envolvidas com relacionamento de generalização/especialização. O diagrama de atividades desta fase pode ser analisado na Figura B.1.

A realização da primeira atividade desta fase, Definir Hierarquia das Classes «tipo»/«quase-tipo» , foi feita com base na distribuição das classes «tipo»/«quase-tipo» nos grupos '*Agente*', '*Objeto*', '*Evento*', '*Modo*' e '*Outros*' definida no final da fase 1 e com base nas instruções apresentadas no Guia 2.1 (Figura B.2).

Analisando o grupo *Evento*, que contém as classes Atendimento e Atividade não foi detectado nenhum relacionamento de generalização/especialização que exista ou deveria existir entre tais classes. Isto ocorre, uma vez que entre tais classes não existe uma classificação hierárquica, ou seja, não há uma classe geral que se relacione com classes específicas. Desta forma as classes Atendimento e Atividade foram classificadas como «tipo».

Analisando o grupo *Agente*, que contém as classes Grupo de Projeto, Instituição e Convênio não foi detectado nenhum relacionamento de generalização/especialização que exista ou deveria existir. Isso ocorre, uma vez que no domínio em análise tais classes não se relacionam hierarquicamente. Desta forma tais classes foram classificadas como «tipo».

Por fim, o grupo *Objeto* contém apenas a classe Linhas de Pesquisa que necessariamente será classificada com o estereótipo «tipo».

Todas essas alterações devem ser efetuadas no diagrama de classes.

Passa-se então para a segunda atividade da fase 2: Checar Conformidade com Restrições Hierárquicas onde todas as restrições ontológicas pertinentes a relacionamentos de generalização/especialização serão verificadas. Esta atividade, como pode ser visto no diagrama de atividades da Figura B.3, é composta por outras duas atividades: Checar Hierarquias Existentes e Checar Existência de Hierarquias Obrigatórias. Para a primeira delas, Checar Hierarquias Existentes, não foi necessário fazer qualquer tipo de verificação, uma vez que não existiam relacionamentos de generali-

zação/especialização no diagrama de classe UML original.

A outra atividade, Checar Existência de Hierarquias Obrigatórias, foi realizada seguindo as orientações do Guia 2.4 (Figura B.10). Detectou-se, então, que as classes Paciente, Estagiário e Professor, todas estereotipadas como «papel material» têm obrigatoriamente que ser subclasses de outra(s) classe(s) estereotipadas como «tipo». Para resolver tal problema, criou-se a classe Pessoa como sendo uma superclasse das classes Paciente, Estagiário e Professor, uma vez que a identidade executada pelas classes «papel material» é fornecida pela classe Pessoa.

Conclui-se assim o diagrama de atividade da SubFase da Fase 2: Checar Conformidade com Restrições Hierárquicas (Figura B.3) e retorna-se ao diagrama de atividade:Verificação Hierárquica (Figura B.1) onde resta apenas realizar a atividade Checar Conformidade Com Restrições de Relacionamento. Seguindo as orientações do Guia 2.2 (Figura B.12) observou-se que todos os relacionamentos de dependência entre as classes «papel material» e suas respectivas classes dependentes têm cardinalidade mínima 1, respeitando portanto as restrições de relacionamento. Ou seja, objetos da classe Paciente sempre se relacionarão no mínimo com um objeto da classe Atendimento, objetos da classe Estagiário e Professor se relacionarão no mínimo com um objeto da classe Atendimento ou Grupo de Projeto. Conclui-se, assim, a segunda fase do procedimento PrOntoCon.

5.1.3 Aplicação das Fases 3 e 4 do PrOntoCon

A terceira fase do procedimento PrOntoCon, Aplicação do Padrão de Projeto, tem por objetivo aplicar o padrão de projeto herdado do Perfil OntoUML para o caso de hierarquia múltipla entre classes «tipo» e «papel material». Como tal fato não ocorre no diagrama em análise, a fase 3 não foi aplicada neste estudo de caso.

A quarta fase do procedimento PrOntoCon, Verificação de Construtores, tem por objetivo verificar se os construtores UML: classe abstrata, concreta e interface, foram devidamente aplicados ao diagrama de classes. No diagrama de classes inicial, por default, todas as classes eram concretas. Analisando o Guia 4.1 da fase 4 do PrOntoCon (Figura D.2), concluiu-se que todas as classes obedecem as restrições

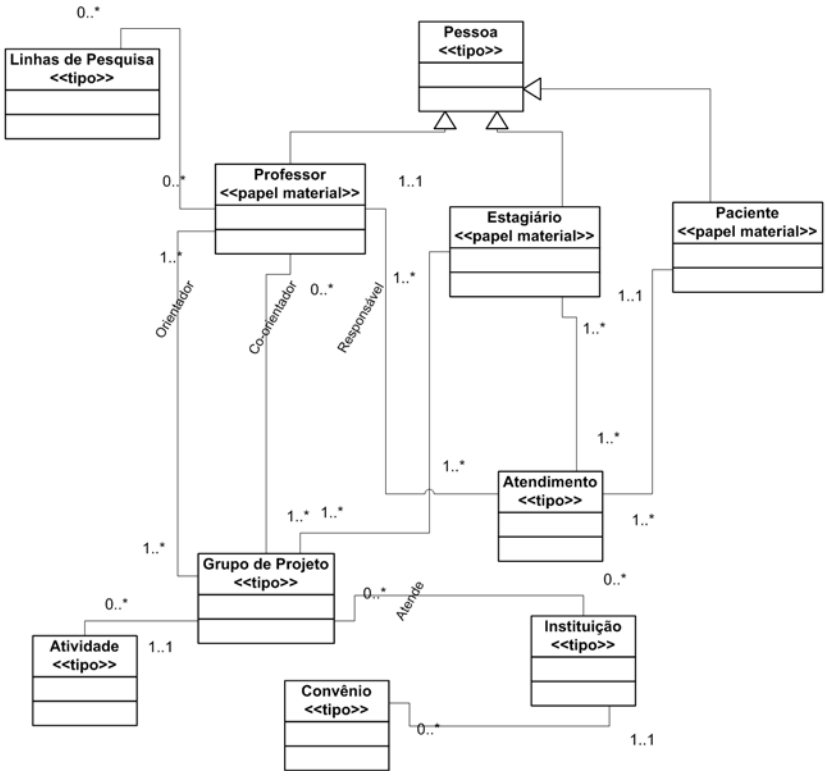


Figura 5.3: Diagrama de Classes do domínio de Controle de Atividades de Estagiários de Psicologia validado pelo procedimento PrOntoCon.

impostas às classes validadas com o PrOntoCon. Não se julgou necessária nenhuma alteração.

Como resultado final da aplicação do procedimento PrOntoCon tem-se o diagrama de classes apresentado na Figura 5.3 contendo as seguintes alterações:

- Classes devidamente estereotipadas seguindo as propriedades ontológicas de Guarino and Welty (2000a);
- Criação da classe Pessoa, que concentrará as informações comuns das classes Professor, Estagiário e Paciente, evitando redundância de dados;

O resultado obtido com este primeiro estudo de caso pode ser analisado comparando o diagrama de classes UML original da Figura 5.1, produzido sem o uso do PrOntoCon, com o diagrama de classes UML da Figura 5.3, alterado após aplicação do procedimento PrOntoCon. No diagrama validado com a técnica pode-se destacar

um aspecto que evidencia a melhora do mesmo em relação ao diagrama original. É a criação da classe rígida Pessoa como superclasse das classes Professor, Estagiário e Paciente. Com a criação de tal classe tem-se os seguintes benefícios:

- Diminuição da redundância de dados: na modelagem original da Figura 5.1, caso um indivíduo representante da classe Estagiário, viesse a ser também representante das classes Professor e Paciente, alguns de seus dados seriam replicados de forma desordenada, implicando numa dificuldade de mantê-los atualizados. Tal fato pode ser claramente percebido com os dados relativos a nome e CPF. Esses dados não dependem do papel exercido pelo objeto, trata-se de dados inerentes a uma classe mais estável (rígida). Com a nova modelagem, a criação da classe Pessoa como mostrado na Figura 5.3 resolve este problema, separando dados relacionados à instância da classe pessoa de dados relativos ao objeto enquanto executor de um determinado papel.

- Promoção da Escalabilidade: com a criação da classe rígida Pessoa, novos papéis que venham a surgir no domínio em questão serão facilmente integrados ao sistema;

- Melhora da integração de módulos: com a criação da classe rígida Pessoa, novos módulos de outros domínios que se comuniquem com o domínio em questão serão facilmente integrados uma vez que ambos terão dado prioridade à criação de classes rígidas, como a classe Pessoa, e estas serão pontos de contato entre os módulos. Por exemplo, se fosse necessário integrar ao sistema de Controle de Atividades de Estagiários ao módulo acadêmico da universidade à qual pertence o curso de Psicologia, as classes Pessoa, Funcionário, Aluno, Professor e outras, seriam facilmente integradas. A facilidade da integração dá-se pelo fato de existir nos dois módulos a classe rígida Pessoa, cuja criação é induzida pelo procedimento e sob a qual serão colocadas as respectivas subclasses.

  Na atual versão, o PrOntoCon não detectou a necessidade de se criar as classes «papel material» Orientador e Co-orientador, como subclasses da classe Professor. Para resolver essas questões seria necessário explorar mais a OntoUML,

investigando conceitos como *Relators*. A aplicaçõe desse tipo de conceito permitirá se ter controle de situações como por exemplo: 'Se um orientador deixar de ser orientador de um grupo, este grupo continua existindo ou para a ser outro grupo?'. Portanto, pesquisas futuras deverão considerar o conceito de *Relators*, objetivando aumentar ainda mais capacidade de se gerar diagramas de classes mais fidedignos com o uso do PrOntoCon.

5.2 Domínio de Controle de Contratos Financeiros para Cursos de Pós-Graduação

O segundo domínio ao qual o procedimento PrOntoCon foi aplicado, trata-se do domínio de controle de contratos financeiros feitos por pessoas físicas ou jurídicas no ato da matrícula de alunos nos cursos de pós-graduação do Unileste-MG - Centro Universitário do Leste de Minas Gerais (Moreira, 2007). Tal escolha foi feita em função da modelagem original apresentar problemas comumente cometidos por modeladores e que forão resolvidos com a aplicação do procedimento PrOntoCon.

Primeiramente é importante descrever o domínio para que se possa entender corretamente a aplicação do PrOntoCon. Para se fazer um curso de pós-graduação no Unileste-MG deve-se obrigatoriamente fazer um contrato financeiro no qual se estabelecerá o valor total do curso e a forma de pagamento do mesmo. Tal contrato pode ser realizado por uma pessoa física ou jurídica. Quando o contrato é firmado por uma pessoa física, esta pode ser o próprio aluno ou um responsável financeiro do mesmo. Um contrato financeiro feito por uma pessoa jurídica pode acontecer principalmente nos casos quando uma empresa resolve financiar a um grupo de funcionários um determinado curso de pós-graduação. Desta forma, na maioria das vezes tem-se vários alunos vinculados a um contrato financeiro feito com uma pessoa jurídica. Quando uma turma de um determinado curso de pós-graduação é criada vincula-se a esta os possíveis planos de negociação financeira existentes para a mesma, por exemplo, valor à vista com desconto, valor dividido em 12 vezes, etc. Existem planos específicos para contrato jurídico e para contratos físicos. Cada contrato estará vinculado

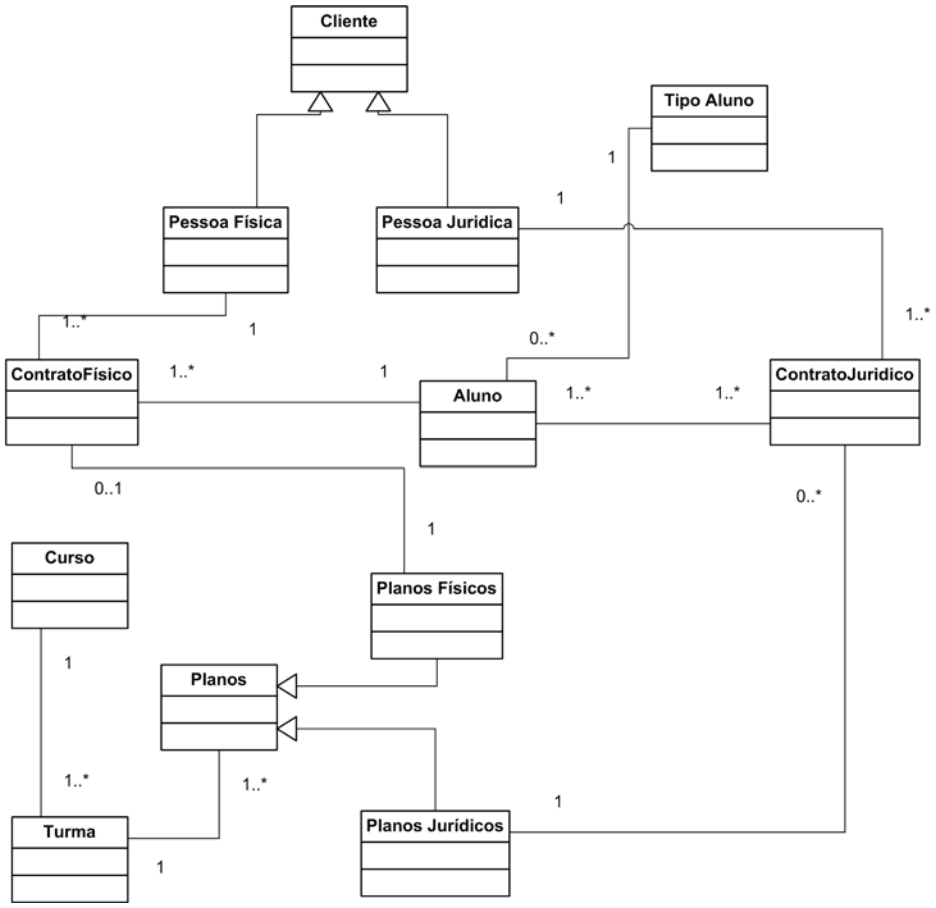


Figura 5.4: Diagrama de Classes UML original do domínio que representa o controle dos contratos financeiros para cursos de Pós-Graduação do Unileste-MG.

a um determinado plano. A Figura 5.4 representa o diagrama de classes UML deste domínio.

Nas seções seguintes mostra-se a aplicação e os resultados de cada fase do procedimento PrOntoCon. Aplicou-se o PrOntoCon da mesma forma que foi mostrado no primeiro estudo de caso. Porém, utilizou-se uma forma de apresentação tabular, visando ser mais breve, objetivo e claro. Todas as fases, subfases e guias do procedimento estão apresentadas nos Anexos A, B, C e D.

5.2.1 Aplicação da Fase 1 do PrOntoCon

A classe Cliente representa as informações necessárias sobre os clientes que fazem algum tipo de contrato com a Pós-Graduação do Unileste-MG. A tabela 5.1 mostra a

análise ontológica feita para se identificar o estereótipo dessa classe.

Classe Cliente		
Questão	Resposta	Justificativa
I Possui Identidade	S	Todo Cliente possui uma identificação, se jurídico possui um CNPJ, se físico possui RG.
II Rigidez	N	Um objeto que representa uma instância da classe em análise não permanecerá para sempre sendo uma instância desta classe, ou seja, uma instância da classe Cliente não será necessariamente um cliente durante toda sua existência.
III Dependência	S	Todo Cliente participante do controle de contratos financeiros da Pós-Graduação do Unileste-MG deverá estar envolvido no mínimo um contrato, quer seja jurídico ou físico.
Classificação: «papel material»		

Tabela 5.1: Análise Ontológica da Classe Cliente

A classe Pessoa Física representa as informações necessárias sobre as pessoas físicas que por ventura venham a ser clientes do Unileste-MG via curso de Pós-Graduação. A tabela 5.2 mostra a análise ontológica feita para se identificar o estereótipo dessa classe.

A classe Pessoa Jurídica representa as informações necessárias sobre pessoas jurídicas que por ventura venham a ser clientes do Unileste-MG via curso de Pós-Graduação. A tabela 5.3 mostra a análise ontológica feita para se identificar o estereótipo dessa classe.

A classe Contrato Jurídico representa as informações necessárias sobre os contratos jurídicos que por ventura venham a ser firmados entre pessoas jurídicas e o

Classe Pessoa Física		
Questão	Resposta	Justificativa
I Possui Identidade	S	Todo objeto da classe Pessoa Física possui uma resposta única para a característica impressão digital.
II Rigidez	S	Um objeto que representa uma instância da classe em análise permanecerá para sempre sendo uma instância desta classe, ou seja, uma instância da classe Pessoa Física será uma pessoa física durante toda sua existência.
Classificação: «tipo»/«quase-tipo»		

Tabela 5.2: Análise Ontológica da Classe Pessoa Física

Classe Pessoa Jurídica		
Questão	Resposta	Justificativa
I Possui Identidade	S	Todo objeto da classe Pessoa Jurídica possui uma resposta única para a característica documento de identificação (CNPJ).
II Rigidez	S	Um objeto que representa uma instância da classe em análise permanecerá para sempre sendo uma instância desta classe, ou seja, uma instância da classe Pessoa Jurídica será uma pessoa jurídica durante toda sua existência.
Classificação: «tipo»/«quase-tipo»		

Tabela 5.3: Análise Ontológica da Classe Pessoa Jurídica

Unileste-MG via curso de Pós-Graduação. A tabela 5.4 mostra a análise ontológica feita para se identificar o estereótipo dessa classe.

Classe Contrato Jurídico		
Questão	Resposta	Justificativa
I Possui Identidade	S	Todo objeto da classe Contrato Jurídico possui uma resposta única para a característica nº de contrato.
II Rigidez	S	Um objeto que representa uma instância da classe em análise permanecerá para sempre sendo uma instância desta classe, ou seja, uma instância da classe Contrato Jurídico será um contrato jurídico durante toda sua existência.
Classificação: «tipo»/«quase-tipo»		

Tabela 5.4: Análise Ontológica da Classe Contrato Jurídico

A classe Contrato Físico representa as informações necessárias sobre os contratos físicos que por ventura venham a ser firmados entre pessoas físicas e o Unileste-MG via curso de Pós-Graduação. A tabela 5.5 mostra a análise ontológica feita para se identificar o estereótipo dessa classe.

A classe Aluno representa as informações necessárias dos alunos que se matriculam em um determinado curso de Pós-Graduação do Unileste-MG, estando necessariamente vinculados a algum tipo de contrato. A tabela 5.6 mostra a análise ontológica feita para se identificar o estereótipo dessa classe.

A classe Tipo de Aluno representa as informações necessárias sobre uma classificação dos tipos de alunos considerados nos cursos de Pós-Graduação do Unileste-MG. Pode-se ter aluno EgressoI, EgressoII, EgressoIII ou Externo, cada um com suas características como por exemplo tempo de conclusão da graduação. A tabela 5.7 mostra a análise ontológica feita para se identificar o estereótipo dessa classe.

A classe Curso representa as informações necessárias aos cursos de Pós-Graduação

Classe Contrato Físico		
Questão	Resposta	Justificativa
I Possui Identidade	S	Todo objeto da classe Contrato Físico possui uma resposta única para a característica nº de contrato.
II Rigidez	S	Um objeto que representa uma instância da classe em análise permanecerá para sempre sendo uma instância desta classe, ou seja, uma instância da classe Contrato Físico será um contrato físico durante toda sua existência.
Classificação: «tipo»/«quase-tipo»		

Tabela 5.5: Análise Ontológica da Classe Contrato Físico

oferecidos pelo Unileste-MG. A tabela 5.8 mostra a análise ontológica feita para se identificar o estereótipo dessa classe.

A classe Turma representa as informações necessárias sobre as turmas que são formadas dos cursos de Pós-Graduação oferecidos pelo Unileste-MG. A tabela 5.9 mostra a análise ontológica feita para se identificar o estereótipo dessa classe.

A classe Planos representa as informações necessárias sobre as possíveis formas de negociação de pagamento dos contratos firmados entre o Unileste-MG e seus clientes de cursos de Pós-Graduação. A tabela 5.10 mostra a análise ontológica feita para se identificar o estereótipo dessa classe.

A classe Planos Físicos representa as informações necessárias sobre as possíveis formas de negociação de pagamento dos contratos físicos firmados entre o Unileste-MG e seus clientes físicos de cursos de Pós-Graduação. A tabela 5.11 mostra a análise ontológica feita para se identificar o estereótipo dessa classe.

A classe Planos Jurídicos representa as informações necessárias sobre as possíveis formas de negociação de pagamento dos contratos jurídicos firmados entre o Unileste-MG e seus clientes jurídicos de cursos de Pós-Graduação.A tabela 5.12 mos-

Classe Aluno		
Questão	Resposta	Justificativa
I Possui Identidade	S	Todo objeto da classe Aluno possui um valor único para a característica impressão digital.
II Rigidez	N	Um objeto que representa uma instância da Aluno em análise não permanecerá para sempre sendo uma instância desta classe, ou seja, uma instância da classe Aluno não será necessariamente um aluno durante toda sua existência.
III Dependência	S	Para ser aluno participante do controle de contratos financeiros da Pós-Graduação do Unileste-MG deverá estar envolvido em no mínimo um contrato, quer seja jurídico ou físico.
Classificação: «papel material»		

Tabela 5.6: Análise Ontológica da Classe Aluno

Classe Tipo de Aluno		
Questão	Resposta	Justificativa
I Possui Identidade	S	Todo objeto da classe Tipo de Aluno possui um código único que o identifica.
II Rigidez	S	Um objeto que representa uma instância da classe em análise permanecerá para sempre sendo uma instância desta classe, ou seja, uma instância da classe Tipo de Aluno será um tipo de aluno durante toda sua existência.
Classificação: «tipo»/«quase-tipo»		

Tabela 5.7: Análise Ontológica da Classe Tipo de Aluno

Classe Curso		
Questão	Resposta	Justificativa
I Possui Identidade	S	Todo objeto da classe Curso possui um código único que o identifica.
II Rigidez	S	Um objeto que representa uma instância da classe em análise permanecerá para sempre sendo uma instância desta classe, ou seja, uma instância da classe Curso será um curso durante toda sua existência.
Classificação: «tipo»/«quase-tipo»		

Tabela 5.8: Análise Ontológica da Classe Curso

Classe Turma		
Questão	Resposta	Justificativa
I Possui Identidade	S	Todo objeto da classe Turma possui um código único que o identifica.
II Rigidez	S	Um objeto que representa uma instância da classe em análise permanecerá para sempre sendo uma instância desta classe, ou seja, uma instância da classe Turma será uma turma durante toda sua existência.
Classificação: «tipo»/«quase-tipo»		

Tabela 5.9: Análise Ontológica da Classe Turma

Classe Planos		
Questão	Resposta	Justificativa
I Possui Identidade	S	Todo objeto da classe Planos possui um código único que o identifica.
II Rigidez	S	Um objeto que representa uma instância da classe em análise permanecerá para sempre sendo uma instância desta classe, ou seja, uma instância da classe Planos será um plano durante toda sua existência.
Classificação: «tipo»/«quase-tipo»		

Tabela 5.10: Análise Ontológica da Classe Planos

Classe Planos Físicos		
Questão	Resposta	Justificativa
I Possui Identidade	S	Todo objeto da classe Planos Físicos possui um código único que o identifica.
II Rigidez	S	Um objeto que representa uma instância da classe em análise permanecerá para sempre sendo uma instância desta classe, ou seja, uma instância da classe Planos Físicos será um plano físico durante toda sua existência.
Classificação: «tipo»/«quase-tipo»		

Tabela 5.11: Análise Ontológica da Classe Planos Físicos

tra a análise ontológica feita para se identificar o estereótipo dessa classe.

Após concluída a verificação de todas as classes do diagrama de classes UML passa-se então à próxima atividade da Fase 1 do PrOntoCon: Classificar Classes «tipo»/«quase-tipo».

Obtendo-se a classificação como mostrada na Tabela 5.13.

Conclui-se assim a primeira fase do procedimento PrOntoCon. A Figura 5.5 apresenta o diagrama de classes parcialmente estereotipado após conclusão da primeira fase.

5.2.2 Aplicação da Fase 2 do PrOntoCon

Na segunda fase do PrOntoCon, realiza-se a atividade Verificação Hierárquica. Analisando o grupo *Evento*, que contém as classes Contrato Jurídico, Contrato Físico e Turma detectou-se que as classes Contrato Físico e Contrato Júridico, por se tratrarem de contratos possuem muitas características semelhantes, com algumas especificidades de cada caso. Logo, deveria existir um relacionamento de generalização/especialização envolvendo tais classes. Criou-se, então, uma superclasse Contrato, que foi classificada como «tipo». As classes Contrato Jurídico e Contrato Físico, sendo subclasses,

Classe Planos Jurídicos		
Questão	Resposta	Justificativa
I Possui Identidade	S	Todo objeto da classe Planos Jurídicos possui um código único que o identifica.
II Rigidez	S	Um objeto que representa uma instância da classe em análise permanecerá para sempre sendo uma instância desta classe, ou seja, uma instância da classe Planos Jurídicos será um plano jurídico durante toda sua existência.
Classificação: «tipo»/«quase-tipo»		

Tabela 5.12: Análise Ontológica da Classe Planos Jurídicos

Classe «tipo»/«quase-tipo»	Categoria UFO
Contrato Jurídico	Evento
Contrato Físico	Evento
Pessoa Física	Agente
Pessoa Jurídica	Agente
Turma	Objeto
Tipo Aluno	Objeto
Curso	Objeto
Planos	Objeto
Planos Físicos	Objeto
Planos Jurídicos	Objeto

Tabela 5.13: Classificação das classes «tipo»/«quase-tipo» nas categorias UFO

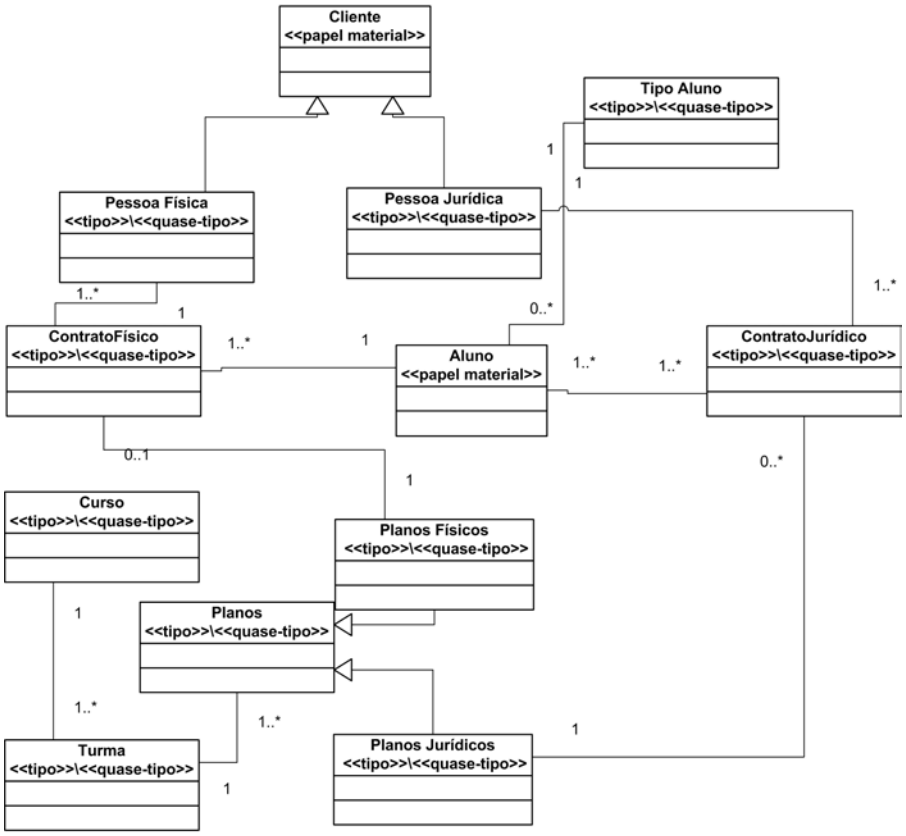


Figura 5.5: Diagrama de Classes parcialmente estereotipado após execução da fase 1 do PrOntoCon

foram classificadas como «quase-tipo» e por fim, a classe Turma foi classificada como «tipo».

No grupo *Agente* tem-se a classe Pessoa Física e Pessoa Jurídica. Não foi detectado nenhum relacionamento de generalização/especialização que exista entre tais classes. Portanto, ambas foram classificadas como «tipo». Mas, poderia ser uma decisão de modelagem correta a criação de uma superclasse Pessoa, com o objetivo de conter atributos comuns como endereço por exemplo.

Analisando o grupo *Objeto*, que contém as classes Tipo Aluno, Curso, Planos, Planos Físicos e Planos Jurídicos detectou-se apenas um relacionamento de generalização/especialização que já existia entre a superclasse Planos e as subclasses Planos Jurídicos e Planos Físicos. Desta forma a classe Planos é classificada como «tipo» e as subclasses Planos Jurídicos e Planos Físicos são classificadas como «quase-tipo». As demais classes: Tipo Aluno e Curso são classificadas como «tipo». Todas essas alterações devem ser efetuadas no diagrama de classes.

Passa-se então para a segunda atividade da fase 2: Checar Conformidade com Restrições Hierárquicas. Checou-se o relacionamento hierárquico entre as classes Planos e Planos Jurídicos e Planos Físicos não constatando nenhuma irregularidade, uma vez que classes «tipo» podem ser superclasses de classes «quase-tipo».

Outro relacionamento de generalização/especialização existente no diagrama de classes já estereotipado é o relacionamento entre a superclasse Contrato e as subclasses Contrato Físico e Contrato Jurídico. Esse relacionamento foi criado pelo próprio PrOntoCon e, portanto, obedece todas as restrições de relacionamento hierárquicos.

Por fim tem-se o relacionamento hierárquico entre a superclasse Cliente e as subclasses Pessoa Física e Pessoa Jurídica. Executando a subatividade Verificar Supertipos e Subtipos conforme orientações do Guia 2.3 detecta-se uma violação da restrição hierárquica que diz que classes estereotipadas com «papel material» não podem ser supertipo de classes «tipo». Deve-se, então, neste caso, executar o diagrama de atividades: Aplicação do Padrão de Projeto - Caso Hierarquia Incorreta.

Aplicação do Padrão de Projeto - Caso Hierarquia Incorreta

Seguindo a sequência do diagrama de atividades Aplicação do Padrão de Projeto - Caso Hierarquia Incorreta, realizou-se as instruções do Guia 3.1 (Figura C.2) para executar a atividade Aplicar Padrão de Projeto - Caso Hierarquia Incorreta. Tais instruções levaram às transformações das classes envolvidas no erro hierárquico, como mostra a Figura 5.6(a) e 5.6(b). Foram criados os subtipos Cliente Físico e Cliente Jurídico para a classe Cliente. Sendo então a classe Cliente reclassificada, assumindo o estereótipo de «papel formal» e suas subclasses foram estereotipadas como «papel material». Sabe-se que uma classe estereotipada como «papel material» obrigatoriamente tem que ter um supertipo de uma classe estereotipada como «tipo». Então, identificou-se no diagrama que as classes Pessoa Física e Pessoa Jurídica são respectivamente os supertipos das classes Cliente Físico e Cliente Jurídico, fornecendo às mesmas a identidade que tais classes possuem.

Para concluir a Aplicação do Padrão de Projeto - Caso Hierarquia Incorreta, realizou-se a atividade Verificar Relacionamento Obrigatório seguindo as orientações do Guia 3.2 (Figura C.3). Percebe-se que não existe o relacionamento de dependência obrigatório na classe «papel formal» Cliente. Porém, as classes «papel material» Cliente Físico e Cliente Jurídico se relacionam respectivamente com as classes Contrato Físico e Contrato Jurídico. Analisando tal situação, estes dois relacionamentos podem ser substituídos por um relacionamento entre as classes Cliente e Contrato. A Figura 5.7 apresenta o diagrama de classes com as alterações até então realizadas.

Após término da aplicação do padrão de projeto, retorna-se à fase 2 do PrOntoCon concluindo-se assim a subatividade Checar Hierarquias existentes. Retorna-se então ao diagrama de atividades da Figura B.3, para executar a atividade: Checar Existência de Hierarquias Obrigatórias. Seguindo as orientações do Guia 2.4 (Figura B.10) detectou-se que a classe Aluno, estereotipada como «papel material» têm obrigatoriamente que ser subclasse de outra classe estereotipada como «tipo». Analisando o diagrama de classes atual, detectou-se que a superclasse para a classe Aluno deveria ser a classe Pessoa Física, uma vez que a identidade executada pela classe «papel material» Aluno é fornecida pela classe «tipo» Pessoa Física.

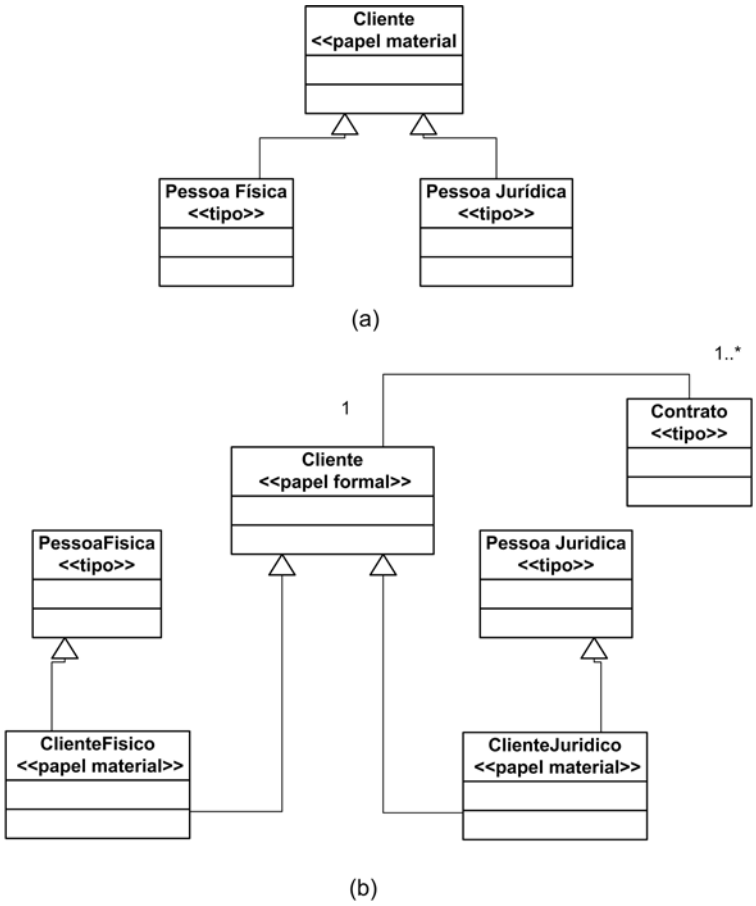


Figura 5.6: Aplicação do Padrão de Projeto - Caso Hierarquia Incorreta no domínio de Controle de Contratos Financeiros de Cursos de Pós-Graduação do Unileste-MG.

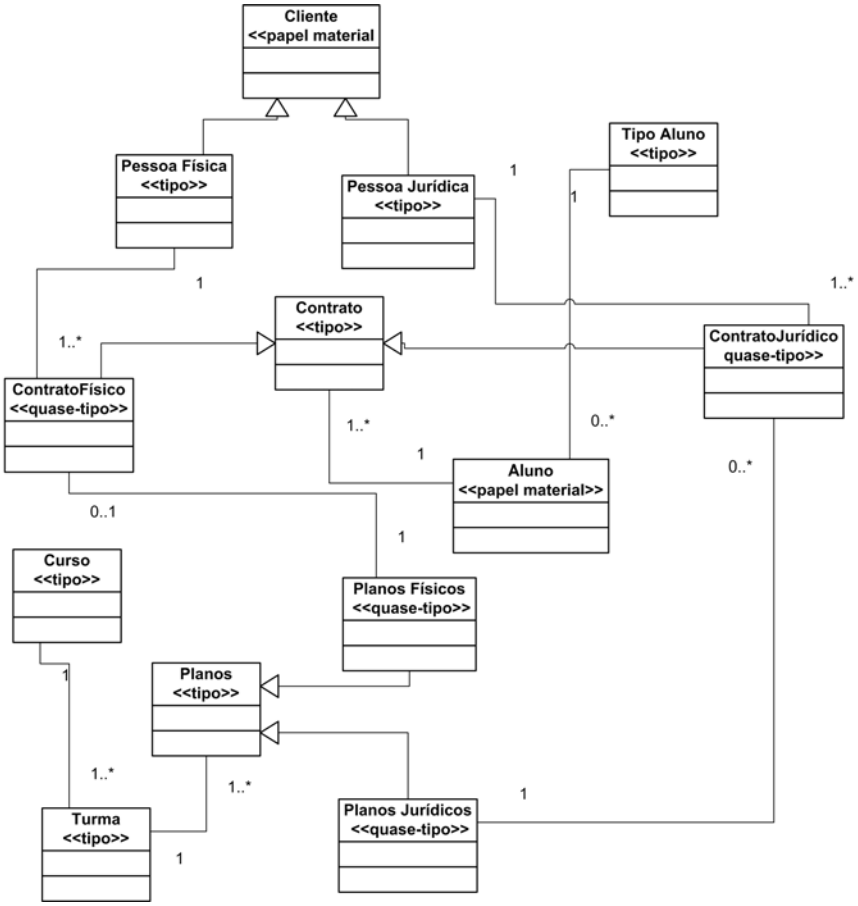


Figura 5.7: Diagrama de Classes estereotipado após término da subatividade Checar Hierarquias existentes.

Conclui-se, assim, o diagrama de atividade da SubFase da Fase 2: Checar Conformidade com Restrições Hierárquicas (Figura B.3) e retorna-se ao diagrama de atividade: Verificação Hierárquica (Figura B.1) onde resta apenas realizar a atividade Checar Conformidade Com Restrições de Relacionamento. Seguindo as orientações do Guia 2.2 (Figura B.12) observou-se que o relacionamento de dependência entre a classe «papel formal» Cliente e sua respectiva classe dependente Contrato tem cardinalidade mínima 1 (um), respeitando, portanto, as restrições de relacionamento. O mesmo foi observado para a classe «papel material» Aluno, pois objetos da classe Aluno sempre se relacionarão no mínimo com um objeto da classe Contrato. A conclusão da segunda fase do procedimento PrOntoCon gera uma diagrama de classes hierarquicamente validado, restando apenas a execução de parte da fase 3 e da fase 4 do procedimento.

5.2.3 Aplicação das Fases 3 e 4 do PrOntoCon

A terceira fase do procedimento PrOntoCon, Aplicação do Padrão de Projeto, tem por objetivo aplicar o padrão de projeto proposto por Guizzardi et al. (2004b). O primeiro caso possível de aplicação do padrão de projeto já foi verificado na segunda fase e aplicado a partir dela. Outro possível caso de aplicação do padrão de projetos é detectado quando existe casos de hierarquia múltipla entre classes «tipo» e «papel material». Como este fato não ocorre no diagrama em análise, tal parte da fase 3 não foi aplicada neste estudo de caso.

A quarta fase do procedimento PrOntoCon, Verificação de Construtores, tem por objetivo verificar se os construtores UML: classe abstrata, concreta e interface, foram devidamente aplicados ao diagrama de classes. Analisando o Guia 4.1 da fase 4 do PrOntoCon (Figura D.2), concluiu-se que todas as classes obedecem as restrições impostas às classes validadas com o PrOntoCon. Não se julgou necessária nenhuma alteração.

Como resultado final da aplicação do procedimento PrOntoCon, tem-se o diagrama de classes apresentado no Figura 5.8 contendo as seguintes alterações:

- Classes devidamente estereotipadas seguindo as propriedades ontológicas de Guarino and Welty (2000a);

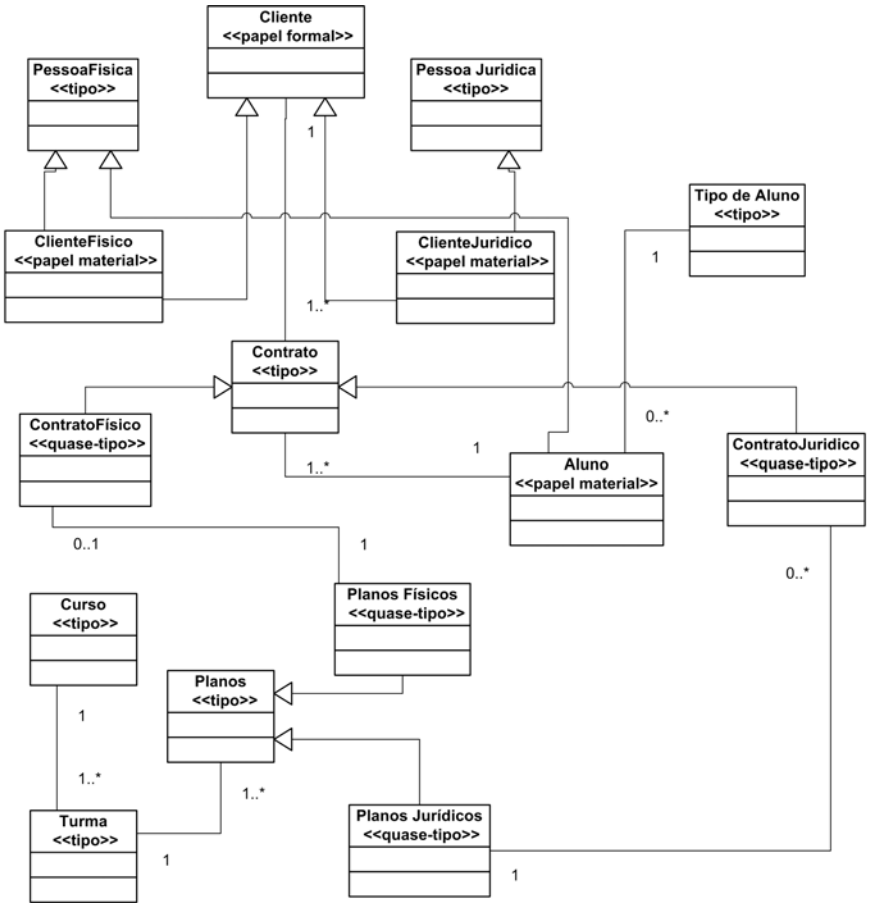


Figura 5.8: Diagrama de Classes do domínio de Controle de Contratos Financeiros para Cursos de Pós-Graduação validado pelo procedimento PrOntoCon.

- Criação da classe Contrato como generalização das classses Contrato Físico e Contrato Jurídico;

- Aplicação do padrão de projeto proposto em Guizzardi (2005) proporcionando a correção de um erro de modelagem do papel representado pela classe Cliente. Como pode ser visto na Figura 5.8, foram criadas as classes Cliente Físico e Cliente Jurídico, dois papéis devidamente associados às superclasses Pessoa Física e Pessoa Jurídica respectivamente, de onde obtêm suas devidas identidades.

- Relacionamento de generalização/especialização entre a classe Pessoa Física e Aluno evitando principalmente redundância de dados;

O resultado obtido com este segundo estudo de caso pode ser analisado com-

parando o diagrama de classes UML original da Figura 5.4, alterado sem o uso do PrOntoCon, com o diagrama de classes UML da Figura 5.8 produzido com o uso do PrOntoCon. No diagrama validado com a técnica pode-se destacar dois pontos que evidenciam a melhora do mesmo em relação ao diagrama original.

O primeiro deles é a solução de um erro comum em modelagem conceitual: a inversão de classes num relacionamento de generalização/especialização, tendo como superclasses classes que representam papéis e como subclasses, classes rígidas. Tal fato ocorreu neste estudo de caso quando o modelador fez um relacionamento de generalização/especialização entre as classes Cliente e as classes Pessoa Física e Pessoa Jurídica, sendo a primeira superclasse e as outras duas subclasses. Esse erro representa de forma inadequada o domínio do problema pois permite que uma classe rígida (Pessoa Física), cujos objetos serão para sempre pertencentes a esta classe, seja uma subclasse de uma classe não rígida (Cliente), cujos objetos não serão para sempre pertencentes a tal classe. Um objeto representante da classe Cliente que por ventura deixar de representar tal classe, deveria automaticamente deixar de ser representante das suas subclasses. Porém, quando um indivíduo deixa de ser cliente, não deixa de ser pessoa. Portanto, o diagrama de classes UML representava uma situação que não condiz com a realidade.

Com a solução gerada através da aplicação do Padrão de Projeto de Guizzardi (2005), o relacionamento hierárquico incorreto é desfeito, como pode ser observado na Figura 5.6(b), aumentando-se a escalabilidade do sistema, uma vez que novos tipos de clientes que possam surgir serão facilmente inseridos no modelo sem causar grandes alterações.

A criação de diferentes visões é facilmente obtida uma vez que o relacionamento entre os vários papéis de clientes e as classes rígidas que lhes fornecem identidade está claramente separado e bem estruturado: Pessoa Física como superclasse de Cliente Físico e Pessoa Jurídica como superclasse de Cliente Jurídico, bem como os seus devidos relacionamentos de dependência.

No diagrama original do domínio de Controle de Contratos Financeiros a classe Aluno não estava vinculada a nenhuma outra classe através de um relacionamento de

especialização. Desta forma o sistema permitia redundância de dados nos casos onde um aluno fosse fazer um contrato físico em seu nome. Nesta situação dados básicos referentes a esse aluno (nome, endereço, telefone) seriam duplicados na base de dados, nas classes Pessoa Física e Aluno, causando grandes transtornos para atualização dos mesmos. Com o uso da OntoCon detectou-se que existe, na realidade, um relacionamento hierárquico entre tais classes, diminuindo conseqüentemente a redundância de dados.

Fato semelhante ao analisado em relação à classe Aluno foi proporcionado também com a criação da classe Contrato, como superclasse das classes Contrato Físico e Contrato Jurídico, ambas estereotipadas como «quase-tipo».

A aplicação da técnica OntoCon, através do uso do PrOntoCon, conduz o modelador a detectar problemas de redundância de dados, uma vez que não permite que classes que representem papéis, como a classe Aluno, ou classes que representem fases e quase-tipos, estejam desassociadas de uma classe rígida que lhes forneça identidade. Obtêm-se, então, sistemas com menor redundância de dados e conseqüentemente com menos inconsistências.

Com as análises dos resultados obtidos nos dois diagramas de classe UML validados com o PrOntoCon fica claro que a aplicação da técnica OntoCon, que faz uso da análise ontológica na modelagem conceitual, proporciona o aumento de características muito importantes a um modelo conceitual: escalabilidade, diminuição da redundância de dados, facilidade de integração de módulos e facilidade de criação de visões.

Capítulo 6

Conclusões

Os estudos, análises e testes de técnicas como a VERONTO e o Perfil OntoUML, feitos no decorrer deste trabalho, evidenciaram a importância e os benefícios que tais técnicas são capazes de trazer para a modelagem conceitual. Porém, ficou claro também que o uso de tais técnicas é extremamente difícil, uma vez que envolve conceitos filosóficos, exigindo do modelador conhecimento e maturidade em análises filosóficas com as quais provavelmente não está acostumado. Ficou evidente, então, a necessidade de se criar algum mecanismo que minimizasse tal problema. Por esse motivo, o principal objetivo proposto no presente trabalho foi a construção de um procedimento de análise de conceitos que permita guiar o modelador na validação das classes e dos relacionamentos expressos em um modelo de classes do domínio de uma aplicação, através do uso de técnicas de análise ontológica.

Para se atingir o objetivo principal, outros objetivos secundários tiveram que ser realizados. O primeiro objetivo proposto foi montar uma técnica que fizesse uso da modelagem ontológica como apoio à modelagem conceitual utilizando características da técnica VERONTO (Villela, 2004) e do Perfil OntoUML definido por Guizzardi (2005). Para conseguir alcançar tal objetivo, um estudo cuidadoso da VERONTO e do Perfil OntoUML foi feito na Seção 3.1. Tal estudo deu origem à OntoCon, uma técnica que utiliza como base as propriedades definidas por Guarino and Welty (2000a), propriedades essas presentes na VERONTO, e utiliza também várias características do Perfil OntoUML, sendo a principal delas, o padrão de projeto para modelagem de

papéis.

O segundo, e principal objetivo proposto, foi a construção de um procedimento que auxiliasse na utilização da técnica OntoCon. Tal procedimento, intitulado PrOntoCon - Procedimento de uso da Técnica OntoCon, foi desenvolvido utilizando como linguagem de representação o SPEM. Todas as suas fases foram discutidas no Capítulo 4 e o procedimento, na íntegra, se encontra nos Anexos A, B, C e D.

A aplicação do procedimento PrOntoCon e a avaliação do benefício de uso do mesmo também foram objetivos propostos no trabalho. A aplicação do PrOntoCon se deu através da validação de dois diagramas de classes UML já existentes, cada um envolvendo domínios diferentes. Os resultados foram satisfatórios, permitindo perceber que os diagramas validados trouxeram alterações relevantes no que diz respeito à qualidade de manutenibilidade do sistema (Capítulo 5).

As conclusões obtidas com o desenvolvimento do trabalho proposto têm dois focos principais de análise. O primeiro trata-se do uso do procedimento como mecanismo facilitador da aplicação da análise ontológica na modelagem conceitual. O segundo foco trata dos benefícios obtidos com a aplicação do procedimento PrOntoCon em diagramas de classe UML.

Tendo em mente o primeiro foco de conclusão, através do uso do PrOntoCon na validação de diagramas de classes UML pode-se perceber que o modelador não necessitará de ter um profundo conhecimento dos conceitos filosóficos envolvidos na análise ontológica. Tal fato é possível principalmente pelo uso de perguntas direcionadas na árvore de caminhamento para definição dos estereótipos (Figura A.4). Em cada uma das quatro questões da árvore de caminhamento, existem, além da pergunta, uma série de exemplos e contra exemplos que levam o modelador a identificar as meta-propriedades existentes na classe em análise sem mesmo saber explicitamente quais são elas e qual é o conceito filosófico envolvido em cada uma. Outro ponto muito relevante a ser analisado é a facilidade de uso do procedimento, proporcionado pelo uso da linguagem de modelagem de processos: SPEM. Os diagramas de atividades montados em cada fase ou subfase do procedimento apresentam de forma clara o ponto de partida e finalização, a seqüência das atividades e os artefatos utilizados,

gerados e/ou alterados por cada atividade. Para cada atividade existe um guia que relata o que se deve fazer naquela atividade e como se deve fazer. Entende-se que outro fator que propiciou a boa estruturação do procedimento PrOntoCon foi a criação da técnica OntoCon. Com a criação da mesma centralizou-se todos os pontos e características que objetivava serem cumpridos pelo procedimento. Se o procedimento não tivesse uma base única sobre a qual se estruturar, ou seja, se fosse se basear ora na VERONTO, ora no Perfil OntoUML, acredita-se que o mesmo não seria de tão fácil entendimento e aplicabilidade.

Para o segundo foco de conclusão deste trabalho, os benefícios obtidos com a aplicação do procedimento PrOntoCon, tem-se como ponto de suporte a análise e discussão dos resultados obtidos com os estudos de caso realizados no Capítulo 5. Por esses resultados, acredita-se que o uso do PrOntoCon propicia alguns benefícios muito importantes: o aumento da escalablidade dos sistemas, ou seja, tornam-se mais preparados para o crescimento, a diminuição da redundância de dados na base de dados dos sistemas e a facilidade na integração de módulos dos sistemas. Percebe-se também que, em termos de complexidade, os diagramas de classes validados são sempre maiores e mais complexos, o que pode erroneamente levar o modelador a deduzir que a aplicação do PrOntoCon não é benéfica. É importante ressaltar que na maioria das vezes haverá um aumento de classes no diagrama de classes validado com o procedimento, pois o mesmo conduz sempre à criação de classes rígidas do domínio representado. Explicitando assim conhecimentos antes implícitos.

Objetivando uma análise correta dos resultados obtidos com a aplicação do PrOntoCon, é necessário que membros da equipe de modelagem tenham o conhecimento da importância e da necessidade de se ter modelos mais fidedignos ao domínio, mesmo que tais modelos sejam mais complexos, pois os benefícios futuros em termos de manutebilidade do sistema são compensadores e superam uma implementação inicialmente mais trabalhosa.

6.1 Trabalhos Futuros

Nem todos os recursos da Técnica VERONTO e do Perfil OntoUML estão presentes na Técnica OntoCon. Dentre eles podem-se destacar as restrições sobre relacionamentos 'todo-parte', que devem ser contemplados em versões posteriores da OntoCon e do PrOntoCon.

Encontra-se em fase de desenvolvimento, em de um projeto de iniciação científica (Oliveira and Biasutti, 2007), uma ferramenta que permite automatizar partes do PrOntoCon, conduzindo o modelador a um uso mais fácil do procedimento. É importante que tal projeto seja concluído futuramente, contemplando o PrOntoCon na sua totalidade.

O PrOntoCon foi criado com o propósito de validar diagramas de classe UML já existentes. É importante também que se estruture um procedimento para guiar a construção de um diagrama de classe respeitando as restrições hierárquicas de relacionamentos de generalização/especialização e relacionamentos 'todo-parte'.

O PrOntoCon não valida classes de associações. Trabalhos futuros serão necessários para adaptar o procedimento PrOntoCon de forma validar esse tipo de classe. Sugere-se o uso de *Relators* do perfil OntoUML.

Explorar alguns conceitos da OntoUML, tais como: *relators, highOrderType e mode*(Guizzardi, 2005), objetivando agregar melhorias ao PrOntoCon de forma que os diagramas de classes validados sejam ainda mais fidedignos.

A análise dos resultados obtidos com a aplicação do PrOntoCon foi feita através de uma comparação não formal dos diagramas de classes originais com os diagramas de classes validados. Para uma validação mais formal deve-se futuramente trabalhar com equipes de profissionais modeladores, comparando os resultados de diagramas produzidos com o uso do procedimento PrOntoCon com diagramas produzidos sem o uso do procedimento. Para realizar tal comparação deverão ser estabelecidas métricas e critérios objetivos de avaliação, permitindo, assim, concluir de forma mais precisa a eficácia do uso do PrOntoCon.

Apêndice A

O procedimento PrOntoCon - Fase 1

O procedimento PrOntoCon foi constituído utilizando uma ferramenta que permitiu gerar uma sequência de páginas *web*, representando-se assim a sequência dos passsos a serem seguidos quando utiliza-se o procedimento. Duas observações são importantes para o entendimento da ligação entre as páginas: (i) cada guia existente nessas páginas possui um link e (ii) cada símbolo representante de uma atividade que conster três pontos (...) é um link que direciona a outro diagrama de atividades.

Neste anexo, apresenta-se cada uma das páginas referentes à Fase 1 do PrOntoCon, além da página inical do procedimento (Figura A.1).

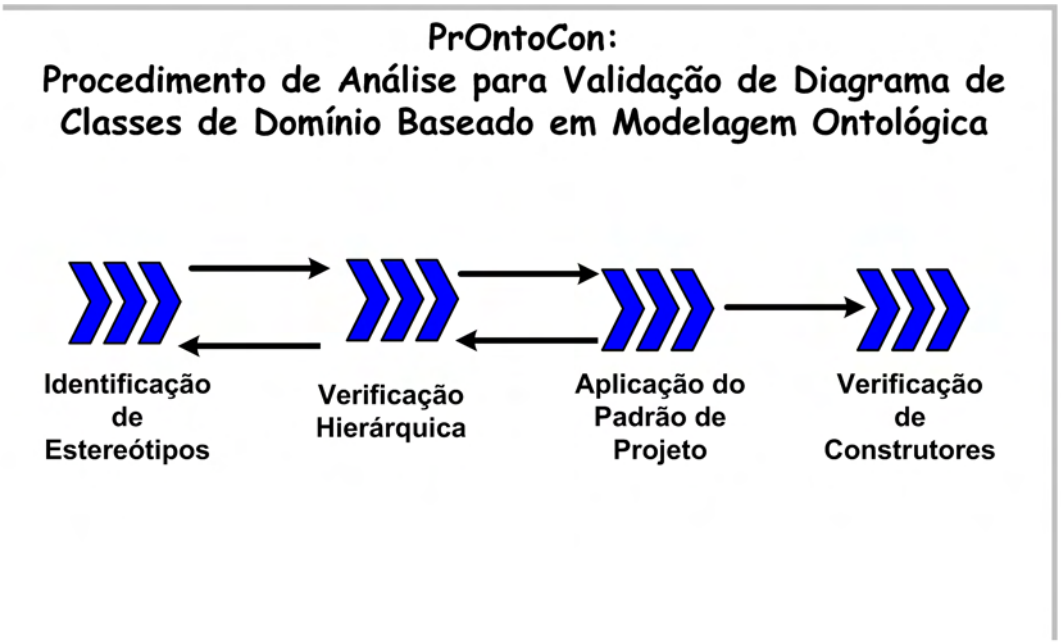


Figura A.1: Fases do Procedimento de Validação de Diagramas de Classe UML - PrOntoCon.

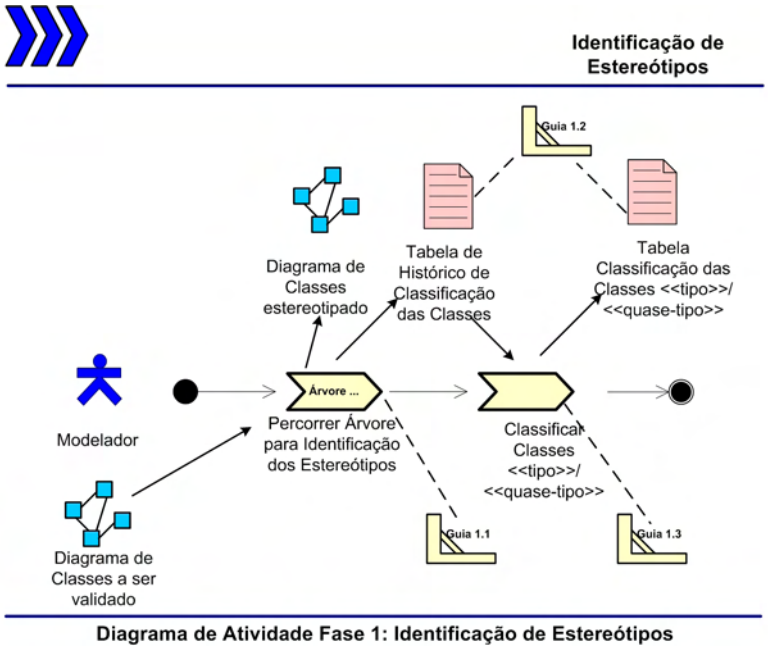


Figura A.2: Diagrama de Atividades da Fase 1: Identificação de Estereótipos

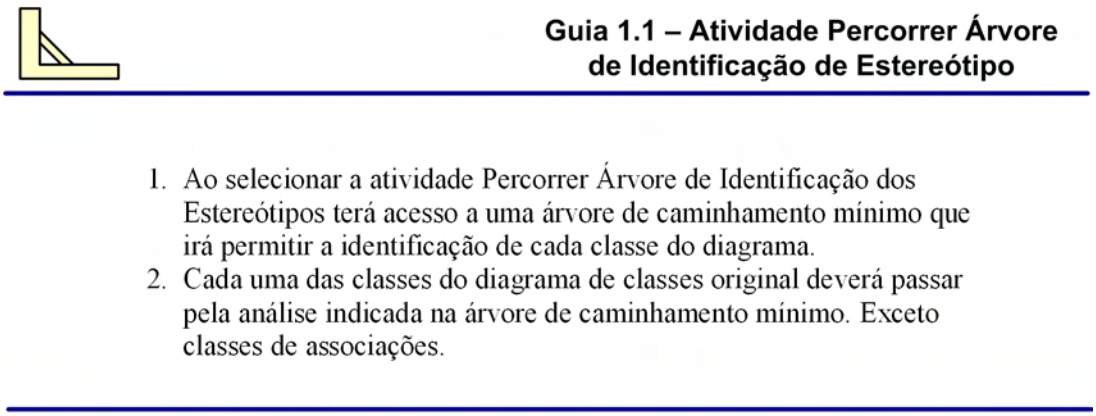


Figura A.3: Guia 1.1 - Atividade Percorrer Árvore de Identificação de Estereótipo

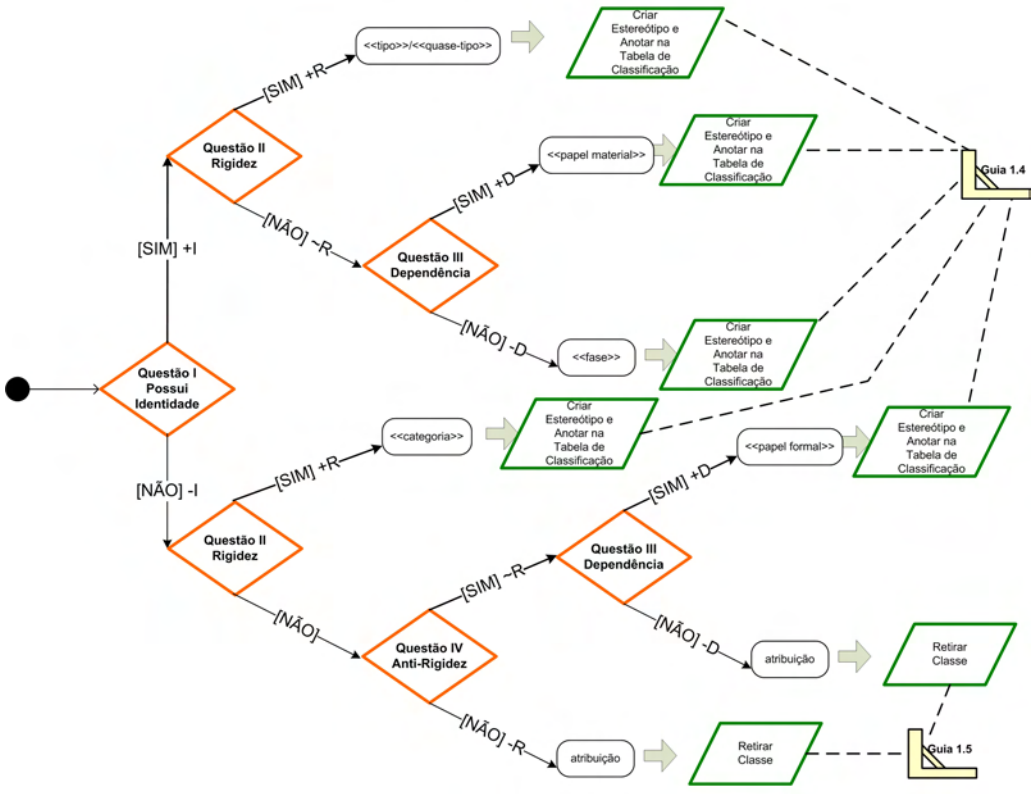
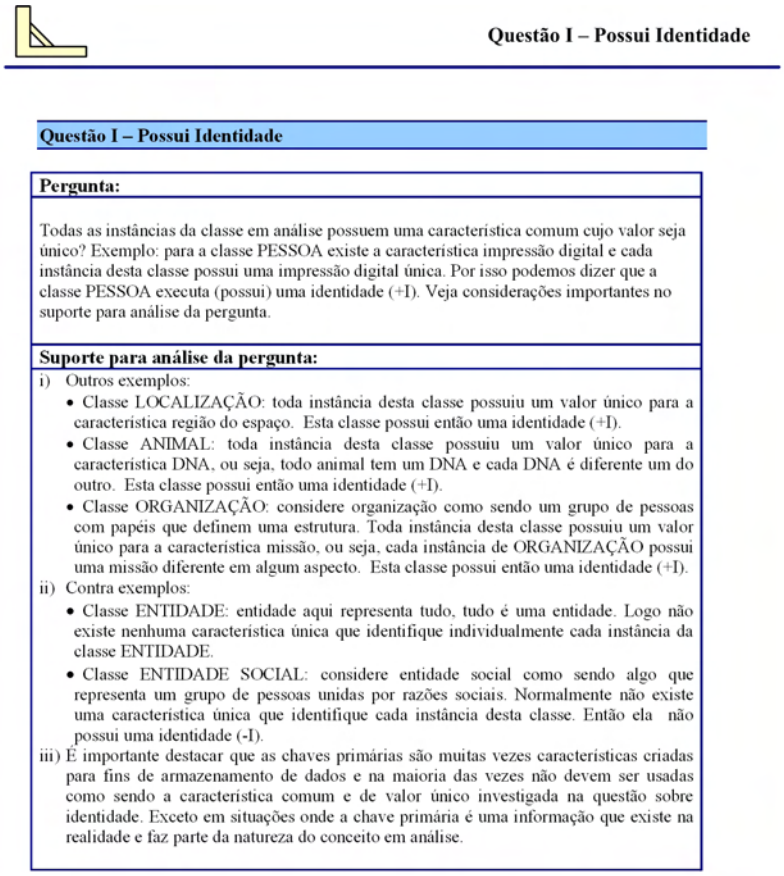


Figura A.4: Árvore de caminhamento para identificação dos estereótipos.



Questão I – Possui Identidade

Questão I – Possui Identidade

Pergunta:

Todas as instâncias da classe em análise possuem uma característica comum cujo valor seja único? Exemplo: para a classe PESSOA existe a característica impressão digital e cada instância desta classe possui uma impressão digital única. Por isso podemos dizer que a classe PESSOA executa (possui) uma identidade (+I). Veja considerações importantes no suporte para análise da pergunta.

Suporte para análise da pergunta:

i) Outros exemplos:
- Classe LOCALIZAÇÃO: toda instância desta classe possuiu um valor único para a característica região do espaço. Esta classe possui então uma identidade (+I).
- Classe ANIMAL: toda instância desta classe possuiu um valor único para a característica DNA, ou seja, todo animal tem um DNA e cada DNA é diferente um do outro. Esta classe possui então uma identidade (+I).
- Classe ORGANIZAÇÃO: considere organização como sendo um grupo de pessoas com papéis que definem uma estrutura. Toda instância desta classe possuiu um valor único para a característica missão, ou seja, cada instância de ORGANIZAÇÃO possui uma missão diferente em algum aspecto. Esta classe possui então uma identidade (+I).

ii) Contra exemplos:
- Classe ENTIDADE: entidade aqui representa tudo, tudo é uma entidade. Logo não existe nenhuma característica única que identifique individualmente cada instância da classe ENTIDADE.
- Classe ENTIDADE SOCIAL: considere entidade social como sendo algo que representa um grupo de pessoas unidas por razões sociais. Normalmente não existe uma característica única que identifique cada instância desta classe. Então ela não possui uma identidade (-I).

iii) É importante destacar que as chaves primárias são muitas vezes características criadas para fins de armazenamento de dados e na maioria das vezes não devem ser usadas como sendo a característica comum e de valor único investigada na questão sobre identidade. Exceto em situações onde a chave primária é uma informação que existe na realidade e faz parte da natureza do conceito em análise.

Figura A.5: Questão 1 - Possui Identidade. Pergunta utilizada no procedimento com o objetivo de descobrir se a classe possui ou não uma identidade.

Questão II - Rigidez

Pergunta:

O objeto que representa a instância da classe em análise permanecerá sendo uma instância desta classe durante toda a sua existência em todos os possíveis domínios? Logo, tal objeto não deixará de pertencer a esta classe. Por exemplo, normalmente o objeto que representa uma instância da classe PESSOA não deixará de ser pessoa, durante toda a sua existência. Desta forma esta classe possui uma característica chamada rigidez (+R). Veja considerações importantes no suporte para análise da pergunta.

Suporte para análise da pergunta:

i)Quando a instância pode deixar de pertencer a classe dizemos que ela possui uma característica chamada anti-rigidez (~R)

ii) **Outros exemplos:**

- Classe LIVRO: normalmente, o objeto que representa uma instância da classe LIVRO permanecerá durante toda sua existência uma instância da classe LIVRO. Então LIVRO possui a propriedade rigidez (+R)
- Classe EXEMPLAR: significando exemplar de livro. Normalmente, o objeto que representa uma instância da classe EXEMPLAR permanecerá durante toda sua existência uma instância da classe EXEMPLAR. Então EXEMPLAR possui a propriedade rigidez (+R)
- Classe EMPRÉSTIMO-EXEMPLAR: normalmente, o objeto que representa uma instância da classe EMPRÉSTIMO-EXEMPLAR permanecerá durante toda sua existência, uma instância da classe EMPRÉSTIMO-EXEMPLAR (+R).

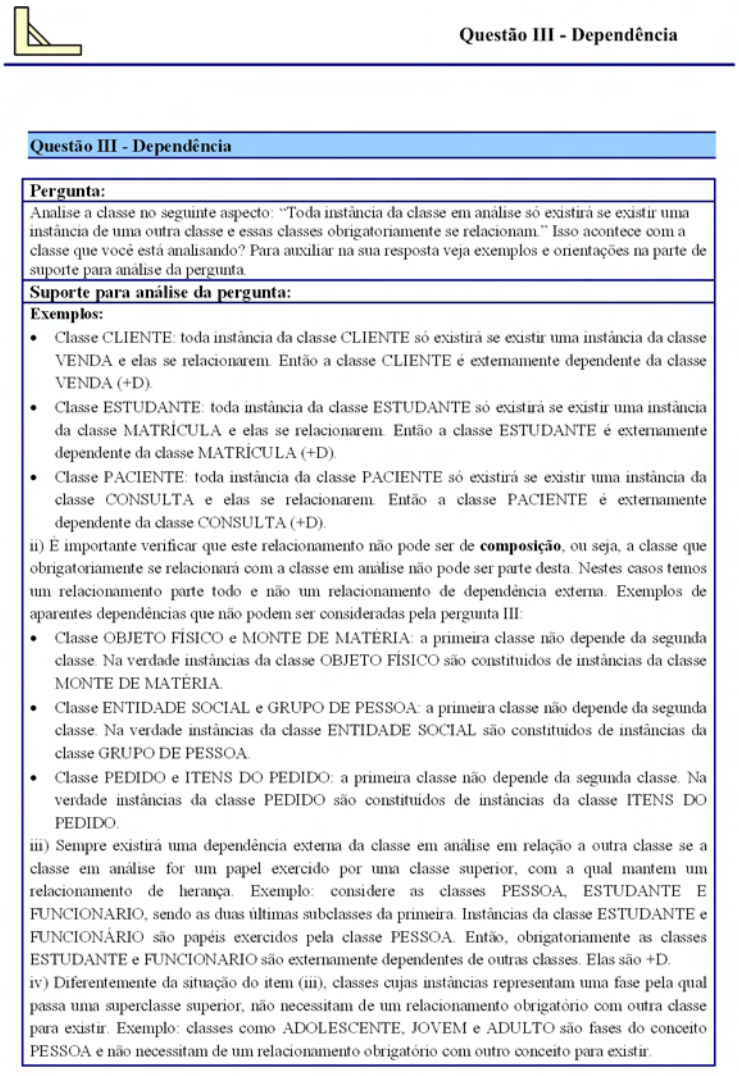
iii) **Contra exemplos**:

- Classe ITENS EMPRESTIMO-EXEMPLAR: normalmente, o objeto que representa uma instância da classe ITENS EMPRESTIMO-EXEMPLAR **não** permanecerá durante toda sua existência uma instância da classe ITENS EMPRESTIMO-EXEMPLAR. Isso ocorre porque um item pode ser retirado do empréstimo voltando a ser apenas uma instância de EXEMPLAR e não mais uma instância pertencente à classe ITENS EMPRESTIMO-EXEMPLAR.
- Classe ESTUDANTE: normalmente, o objeto que representa uma instância da classe ESTUDANTE **não** permanecerá durante toda sua existência uma instância da classe ESTUDANTE. Isto ocorre porque tal instância pode deixar de ser estudante, permanecendo apenas como instância da classe PESSOA.

iv) Quando uma resposta a pergunta sobre rigidez for negativa, não significa que o objeto que representa a instância deixará de existir. Na realidade esta instância deixa de pertencer à classe em análise e continua sendo uma instância da superclasse que nesses casos necessariamente irá existir. É o que acontece com uma instância pertencente à classe ESTUDANTE, quando esta instância deixa de pertencer a tal classe, obrigatoriamente continuará sendo uma instância da classe PESSOA.

v) Quando estiver na pergunta sobre rigidez originada do caminho "[NÃO] –I" da árvore e a classe em análise tem fins classificatórios, provavelmente instâncias da classe em análise permanecerão para sempre uma instância da mesma durante toda a sua existência.

Figura A.6: Questão 2 - Rigidez. Pergunta utilizada no procedimento com o objetivo de descobrir se a classe é ou não uma classe rígida.



Questão III - Dependência

Questão III - Dependência

Pergunta:

Analise a classe no seguinte aspecto: "Toda instância da classe em análise só existirá se existir uma instância de uma outra classe e essas classes obrigatoriamente se relacionam." Isso acontece com a classe que você está analisando? Para auxiliar na sua resposta veja exemplos e orientações na parte de suporte para análise da pergunta.

Suporte para análise da pergunta:

Exemplos:

- Classe CLIENTE: toda instância da classe CLIENTE só existirá se existir uma instância da classe VENDA e elas se relacionarem. Então a classe CLIENTE é externamente dependente da classe VENDA (+D).
- Classe ESTUDANTE: toda instância da classe ESTUDANTE só existirá se existir uma instância da classe MATRÍCULA e elas se relacionarem. Então a classe ESTUDANTE é externamente dependente da classe MATRÍCULA (+D).
- Classe PACIENTE: toda instância da classe PACIENTE só existirá se existir uma instância da classe CONSULTA e elas se relacionarem. Então a classe PACIENTE é externamente dependente da classe CONSULTA (+D).

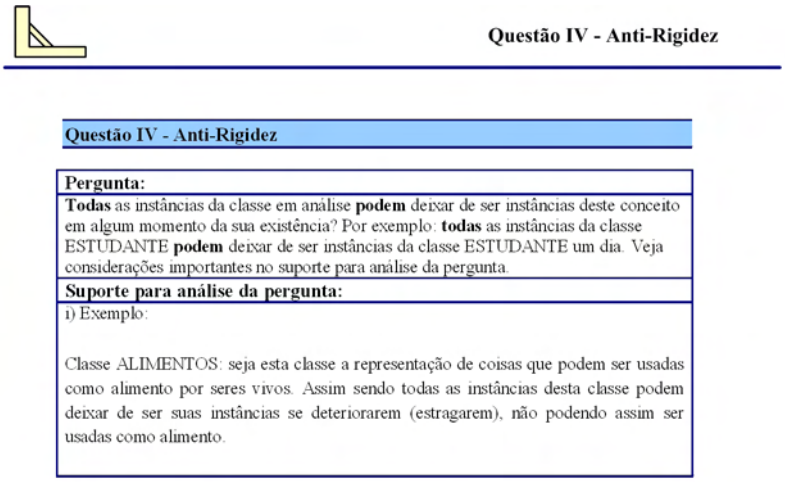
ii) É importante verificar que este relacionamento não pode ser de **composição**, ou seja, a classe que obrigatoriamente se relacionará com a classe em análise não pode ser parte desta. Nestes casos temos um relacionamento parte todo e não um relacionamento de dependência externa. Exemplos de aparentes dependências que não podem ser consideradas pela pergunta III:

- Classe OBJETO FÍSICO e MONTE DE MATÉRIA: a primeira classe não depende da segunda classe. Na verdade instâncias da classe OBJETO FÍSICO são constituídos de instâncias da classe MONTE DE MATÉRIA.
- Classe ENTIDADE SOCIAL e GRUPO DE PESSOA: a primeira classe não depende da segunda classe. Na verdade instâncias da classe ENTIDADE SOCIAL são constituídos de instâncias da classe GRUPO DE PESSOA.
- Classe PEDIDO e ITENS DO PEDIDO: a primeira classe não depende da segunda classe. Na verdade instâncias da classe PEDIDO são constituídos de instâncias da classe ITENS DO PEDIDO.

iii) Sempre existirá uma dependência externa da classe em análise em relação a outra classe se a classe em análise for um papel exercido por uma classe superior, com a qual mantem um relacionamento de herança. Exemplo: considere as classes PESSOA, ESTUDANTE E FUNCIONARIO, sendo as duas últimas subclasses da primeira. Instâncias da classe ESTUDANTE e FUNCIONÁRIO são papéis exercidos pela classe PESSOA. Então, obrigatoriamente as classes ESTUDANTE e FUNCIONARIO são externamente dependentes de outras classes. Elas são +D.

iv) Diferentemente da situação do item (iii), classes cujas instâncias representam uma fase pela qual passa uma superclasse superior, não necessitam de um relacionamento obrigatório com outra classe para existir. Exemplo: classes como ADOLESCENTE, JOVEM e ADULTO são fases do conceito PESSOA e não necessitam de um relacionamento obrigatório com outro conceito para existir.

Figura A.7: Questão 3 - Dependência. Pergunta utilizada no procedimento com o objetivo de descobrir se a classe é ou não uma classe que possui dependência externa em relação a outra classe do modelo.

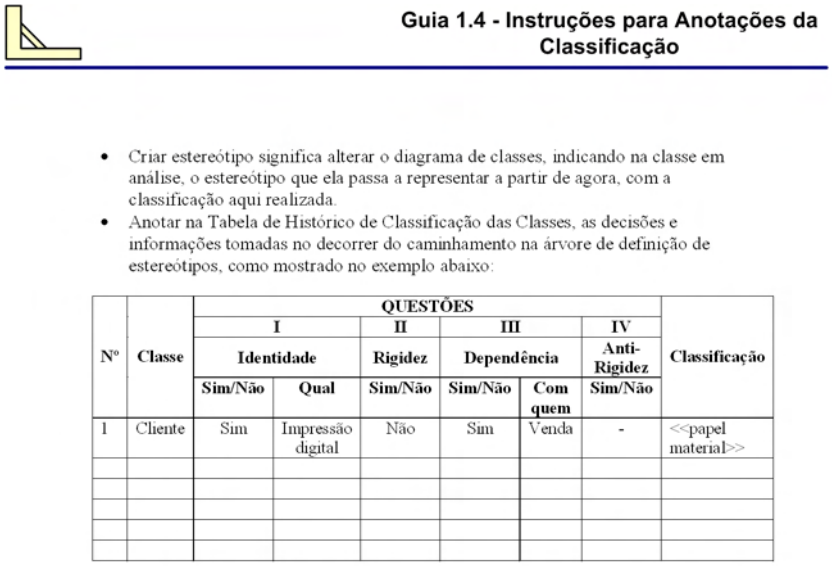


Questão IV - Anti-Rigidez

Questão IV - Anti-Rigidez

Pergunta:
Todas as instâncias da classe em análise **podem** deixar de ser instâncias deste conceito em algum momento da sua existência? Por exemplo: **todas** as instâncias da classe ESTUDANTE **podem** deixar de ser instâncias da classe ESTUDANTE um dia. Veja considerações importantes no suporte para análise da pergunta.
Suporte para análise da pergunta:
1) Exemplo: Classe ALIMENTOS: seja esta classe a representação de coisas que podem ser usadas como alimento por seres vivos. Assim sendo todas as instâncias desta classe podem deixar de ser suas instâncias se deteriorarem (estragarem), não podendo assim ser usadas como alimento.

Figura A.8: Questão 4 - Anti-Rigidez. Pergunta utilizada no procedimento com o objetivo de descobrir se a classe é ou não uma classe anti-rígida.

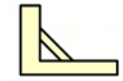


Guia 1.4 - Instruções para Anotações da Classificação

- Criar estereótipo significa alterar o diagrama de classes, indicando na classe em análise, o estereótipo que ela passa a representar a partir de agora, com a classificação aqui realizada.
- Anotar na Tabela de Histórico de Classificação das Classes, as decisões e informações tomadas no decorrer do caminhamento na árvore de definição de estereótipos, como mostrado no exemplo abaixo:

Nº	Classe	QUESTÕES						Classificação
		I		II	III		IV	
		Identidade		Rigidez	Dependência		Anti-Rigidez	
		Sim/Não	Qual	Sim/Não	Sim/Não	Com quem	Sim/Não	
1	Cliente	Sim	Impressão digital	Não	Sim	Venda	-	<<papel material>>

Figura A.9: Guia 1.4 - Instruções para Anotações da Classificação. Guia contendo instruções sobre como o estereótipo diagnosticado para a classe será acrescentado ao diagrama original.



Guia 1.5 - Retirar Classes Classificadas como Atribuição

Caso uma classe seja classificada como atribuição, deve-se:

- Retirar tal classe do diagrama;
- Tal classe passará a ser atributo da(s) classe(s) que o modelador julgar pertinente;
- Para cada classe que receber o novo atributo, verificar se não sofreu alteração em sua classificação.

Figura A.10: Guia 1.5 - Retirar Classes Classificadas como Atribuição. Guia contendo instruções sobre como retirar uma classse que foi classificada como Atribuição.

Guia 1.2 - Tabela de Histórico de Classificação de Classes

Nº	Classe	QUESTÕES						Classificação
		I		II	III		IV	
		Identidade		Rigidez	Dependência		Anti-Rigidez	
		Sim/Não	Qual	Sim/Não	Sim/Não	Com quem	Sim/Não	

Guia 1.2 - Tabela Com Classificação das Classes <<tipo>>/<<quase-tipo>>

Agente	Objeto	Evento	Momento	Outros
<lista dos nomes das classes>	<lista dos nomes das classes>	<lista dos nomes das classes>	<lista dos nomes das classes>	<lista dos nomes das classes>

Figura A.11: Guia 1.2 - Tabela de Histórico de Classificação de Classes e Tabela Com Classificação das Classes «tipo»/«quase-tipo». Apresentação do layout de tais tabelas.

Guia 1.3 - Classificação das Classes <<tipo>>/<<quase-tipo>>

Cada classe classificada como <<tipo>>/<<quase-tipo>> deverá ser enquadrada em um dos 5 (cinco) grupos: AGENTE, OBJETO, EVENTO, MOMENTO ou OUTROS.

- AGENTE: podem representar "coisas" que são agentes físicos, como por exemplo pessoas, animais; ou agentes sociais, como por exemplo uma organização, uma sociedade..
- OBJETO: podem representar "coisas" que são objetos físicos, como por exemplo um carro, uma casa, um produto; ou objetos sociais, como por exemplo dinheiro, idioma, um curso.
- EVENTO: algo que provoca uma transformação, que gera uma mudança de estado das partes envolvidas. Por exemplo: um jogo de futebol, uma venda, uma matrícula num curso de faculdade, uma conversa entre pessoas, uma aula ministrada, etc.
- MODO: representam "coisas" que só podem existir em outras "coisas". Por exemplo: cor, sintomas, carga elétrica, temperatura.
- OUTROS: qualquer classe que não se enquadrar em nenhum dos 4 (quatro) grupos anteriores.

Figura A.12: Guia 1.3 - Classificação das Classes «tipo»/«quase-tipo». Orientações de como se proceder para agrupar as classes «tipo»/«quase-tipo».

Apêndice B

O procedimento PrOntoCon - Fase 2

Anexo contendo as páginas referentes à Fase 2 do procedimento PrOntoCon: Verificação Hierárquica.

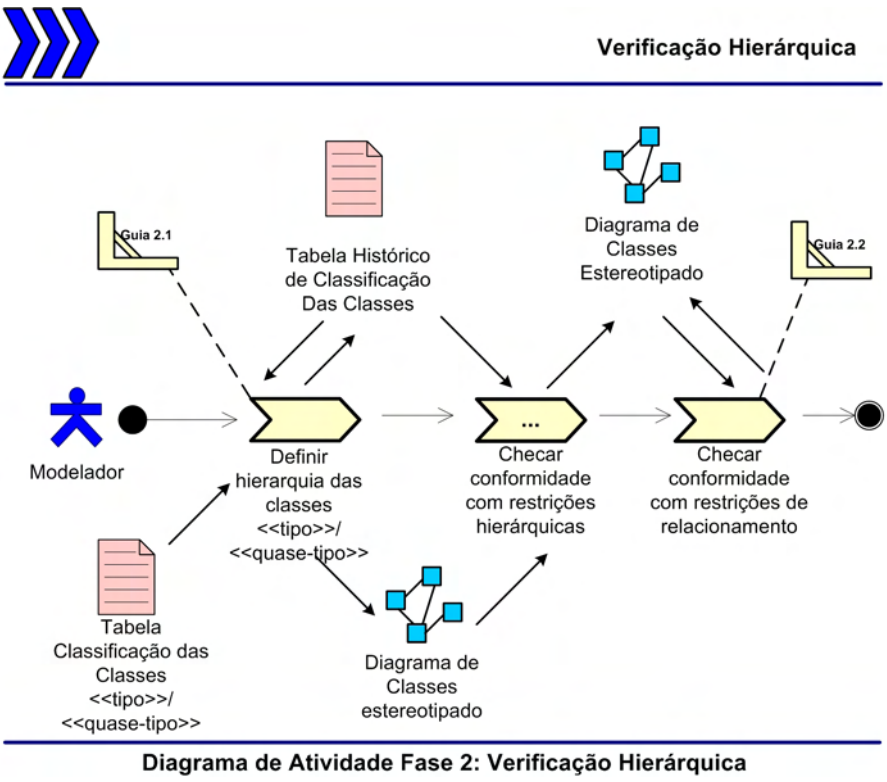
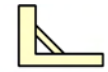


Figura B.1: Diagrama de atividade da Fase 2 do procedimento PrOntoCon - Verificação Hierárquica.



Guia 2.1 - Definição Hierárquica das Classes <tipo>>/<<quase-tipo>>

1. **Para cada um dos grupos (Agente, Objeto, Evento, Momento ou Outros), existentes na Tabela Classificação de Classes <<tipo>>/<<quase-tipo>>, faça a seguinte análise:**
   - 1.1. Verificar se existe ou deveria existir dentre as classes deste grupo relacionamentos caracterizados como generalização/especialização.
     - 1.1.1. Quando tal generalização possuir apenas um nível hierárquico:
       - 1.1.1.1. Verificar se a identidade da superclasse é a mesma das suas subclasses (identidade esta verificada na primeira fase do procedimento). Se isso acontecer, as subclasses deverão ser classificadas com o estereótipo <<quase-tipo>> e a superclasse deverá receber o estereótipo <<tipo>>.
     - 1.1.2. Quando tal generalização possuir mais de um nível hierárquico:
       - 1.1.2.1. A classe que ocupar o nível mais alto da hierarquia será um <<tipo>>.
       - 1.1.2.2. Para cada nível inferior, a classe será um <<quase-tipo>> quando a identidade de tal classe for a mesma da classe superior, caso contrário, será classificada como <<tipo>>;
   - 1.2. Todas as classes que não se enquadrarem nas situações anteriores, independentes do grupo ao qual pertençam, deverão ser estereotipadas como <<tipo>>.

Figura B.2: Guia 2.1 - Definição Hierárquica das Classes <tipo»/«quase-tipo». Guia para orientar o modelador na classificação final das classes até então classificadas como «tipo»/«quase-tipo».

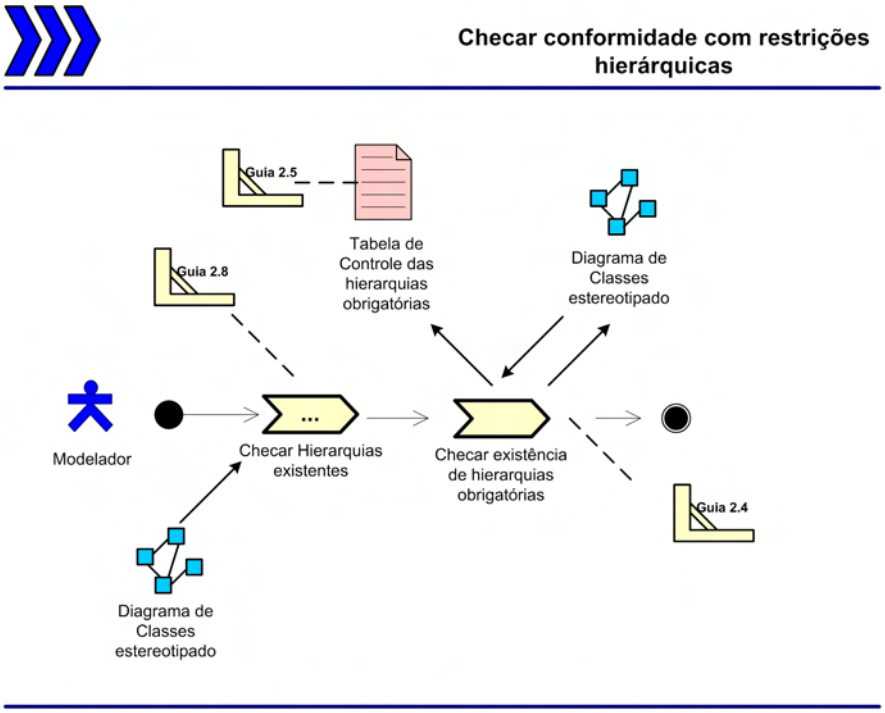


Figura B.3: Diagrama de Atividade da SubFase da Fase 2: Checar Conformidade das Restrições Hierárquicas. Objetiva conduzir o modelador a (i) checar se aos relacionamentos hierárquicos existentes estão corretos e (ii) a avaliar as hierarquias obrigatórias.

A atividade Checar Hierarquias Existentes deve ser realizada somente se existirem relacionamentos de generalização/especialização no diagrama de classes em análise. Casos de generalização/especialização que foram criados durante a atividade Definir Hierarquia das Classes <<tipo>>/<<quase-tipo>>, não precisam ser checados uma vez que já foram cridos seguindo todas as restrições necessárias.

Figura B.4: Guia 2.8 - Orientações sobre Atividade Checar Hierarquias Existentes.

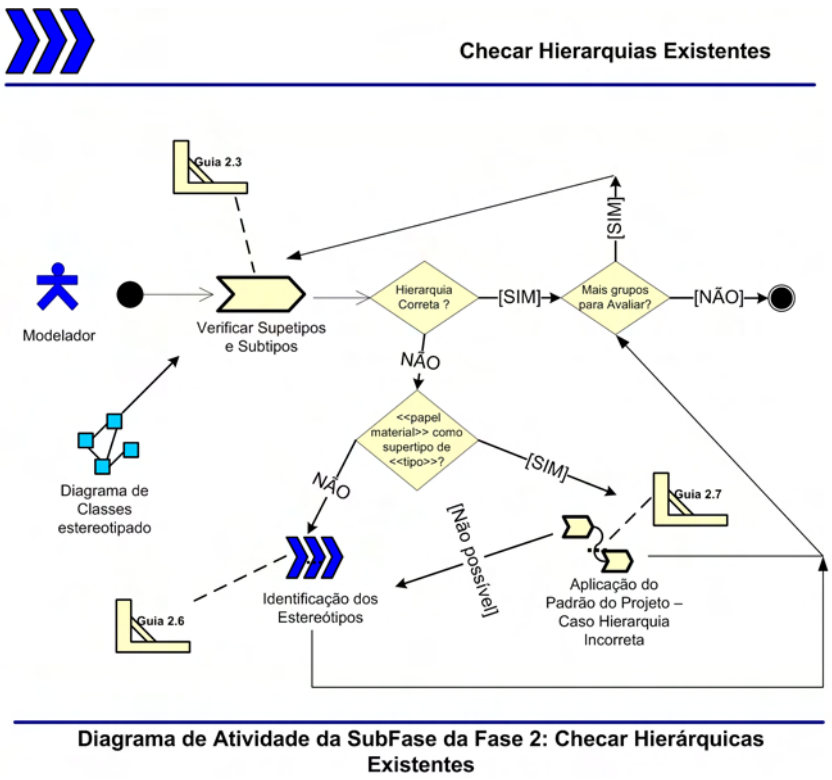


Figura B.5: Diagrama de Atividade da SubFase da Fase 2: Checar Hierárquicas Existentes. Tem por objetivo conduzir o modelador na verificação das hierarquias já existentes, detectando possíveis erros.

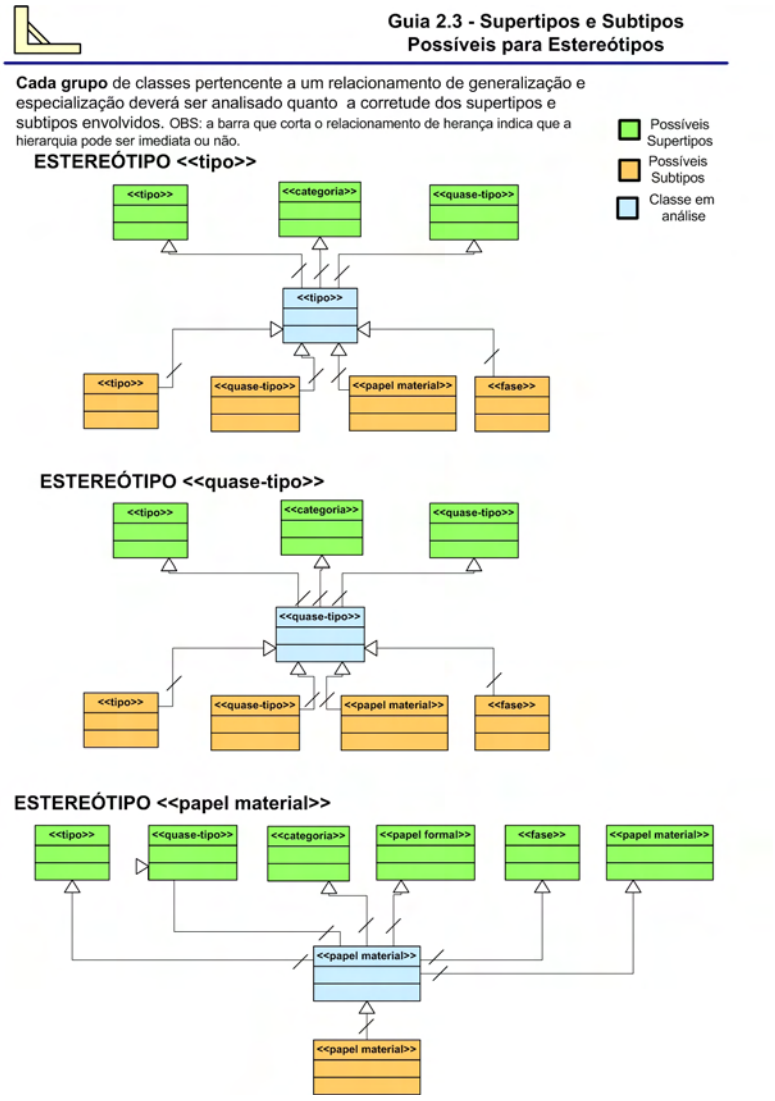


Figura B.6: Guia 2.3 - Supertipos e Subtipos Possíveis para Estereótipos. Utilizado na atividade de Verificar Supertipo e Subtipos permitindo detectar erros hierárquicos.

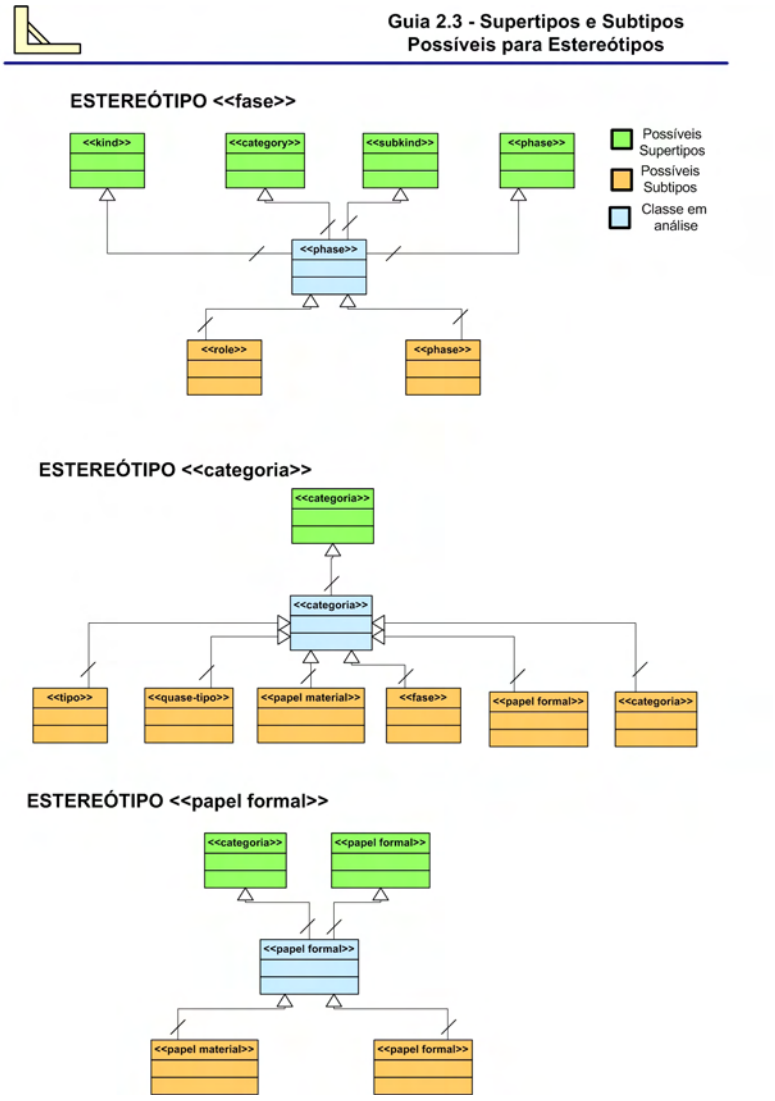
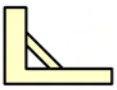


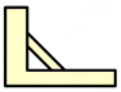
Figura B.7: Continuação: Guia 2.3 - Supertipos e Subtipos Possíveis para Estereótipos.



Guia 2.7 – Condução Aplicação do Padrão de Projeto

Neste ponto do procedimento é necessário ir para a Fase 3 – Aplicação de Padrão de Projeto, especificamente para o Caso de Hierarquia Incorreta, em função da detecção de um erro de hierarquia: uma classe <<papel material>> como supertipo de uma classe <<tipo>>

Figura B.8: Guia 2.7 - Condução à Aplicação do Padrão de Projeto.



Guia 2.6 - Condução a Reavaliação dos Estereótipos

Neste ponto do procedimento pode ser necessário retornar à fase 1 – Identificação de Estereótipos por dois motivos distintos:

1. Quando ocorrer um erro de hierarquia e este não for entre classes estereotipadas com <<papel material>> e <<tipo>>. Retorna-se, então, à fase 1 com o objetivo de detectar algum possível erro de classificação das classes envolvidas no erro hierárquico diagnosticado.
2. Quando não foi possível aplicar o Padrão de Projeto – Caso Hierarquia Incorreta. Então, retorna-se à fase 1 com o objetivo de detectar algum possível erro de classificação das classes envolvidas no erro hierárquico diagnosticado.

Figura B.9: Guia 2.6 - Condução à Reavaliação dos Estereótipos.

Guia 2.4 - Checar Existência de Hierarquias Obrigatórias

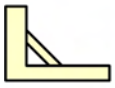
Verificações quanto a presença obrigatória de superclasses <<tipo>>:

1. Preencher a tabela abaixo para conter uma visualização geral das decisões tomadas nos itens 2, 3 e 4 a seguir:

Classe	Estereótipo	Supertipo			Relacionamentos Alterados
		Já Existente	Encontrado	Criado	
Cliente	<<papel material>>	Sim. Pessoa	-	-	Não
Criança	<<fase>>	Não	Sim. Pessoa	-	

2. Para classes estereotipadas como <<quase-tipo>> deverá existir uma superclasse imediata ou não, classificada como <<tipo>>.
   2.1. Caso não exista, uma superclasse da classe em análise deve ser criada.
   2.2. Ou talvez já exista uma classe que deva ser superclasse da classe em análise. E o relacionamento estabelecido está errado, devendo ser de herança.
   2.3. Os relacionamentos existentes nas subclasses deverão ser analisados. Em alguns casos tais relacionamentos passarão a se referir à superclasse criada.
3. Para classes estereotipadas como <<fase >> deverá existir uma superclasse imediata ou não, classificada como <<tipo>>.
   3.1. Caso não exista, uma superclasse da classe em análise deve ser criada.
   3.2. Ou talvez já exista uma classe que deva ser superclasse da classe em análise. E o relacionamento estabelecido está errado, devendo ser de herança.
   3.3. No caso da existência de mais de uma classe classificada como <<fase>> verificar se tais classes podem ter a mesma classe superior em comum.
   3.4. Os relacionamentos existentes nas subclasses deverão ser analisados. Em alguns casos tais relacionamentos passarão a se referir à superclasse criada.
4. Para classes estereotipadas como <<papel material>> deverá existir uma superclasse imediata ou não, classificada como <<tipo>>.
   4.1. Caso não exista, uma superclasse da classe em análise deve ser criada.
   4.2. Ou talvez já exista uma classe que deva ser superclasse da classe em análise. E o relacionamento estabelecido está errado, devendo ser de herança.
   4.3. No caso da existência de mais de uma classe classificada como <<papel material>> verificar se tais classes podem ter a mesma classe superior em comum.
   4.4. Os relacionamentos existentes nas subclasses deverão ser analisados. Em alguns casos tais relacionamentos passarão a se referir à superclasse criada.

Figura B.10: Guia 2.4 - Checar Existência de Hierarquias Obrigatórias. Orientações que conduzem o modelador na análise de hierarquias obrigatórias no diagrama.



Guia 2.5 - Tabela de Controle das Hierarquias Obrigatórias

Classe	Estereótipo	Supertipo			Relacionamentos Alterados
		Já existente	Encontrado	Criado	

Figura B.11: Guia 2.5 - Tabela de Controle das Hierarquias Obrigatórias. Apresentação do layout de tal tabela.

Guia 2.2 - Checar Coformidade com Restrições de Relacionamento

Checar conformidades de relacionamento

1. No mínimo um relacionamento deverá permanecer na classe <<papel material>>. Toda classe classificada como <<papel material>> deve obrigatoriamente possuir um relacionamento com outra classe A. Tal relacionamento expressa a dependência da classe <<papel material>> com outra classe, ou seja, para que instâncias da classe <<papel material>> existam é necessário a existência de instâncias relacionadas na classe A.

1.1. Caso não exista tal relacionamento deve existir uma superclasse de <<papel material>> classificada como <<papel formal>> e esta deverá possuir tal relacionamento de dependência com a classe A.

1.2. Se não existir tal relacionamento, identificar no diagrama de classes, uma classe que deveria ter um relacionamento de dependência com a classe <<papel material>> e por algum motivo não o possui no momento.

1.3. Verificar a cardinalidade do relacionamento da classe <<papel material>> ou <<papel formal>> com a classe A, que deve ser de no mínimo 1.

2. Para relacionamentos de especialização onde as subclasses sejam estereótipos <<fase>> deve-se estar atento ao fato de que uma instância da superclasse não pode ser instância de mais de uma subclasse <<fase>> ao mesmo tempo.

Figura B.12: Guia 2.2 - Checar Conformidade com Restrições de Relacionamento. Orientações sobre análise de relacionamentos envolvendo classes «papel material», «papel formal» ou «fase».

Apêndice C

O procedimento PrOntoCon - Fase 3

Anexo contendo as páginas referentes à Fase 3 do procedimento PrOntoCon: Aplicação do Padrão de Projeto.

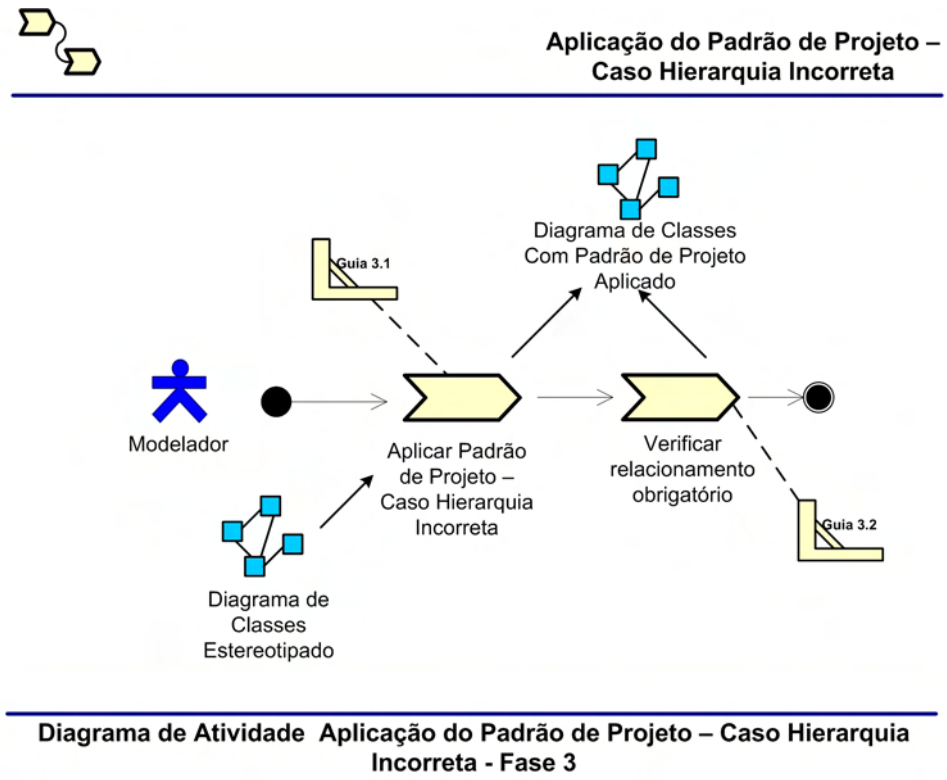


Figura C.1: Diagrama de Atividade-Aplicação do Padrão de Projeto: Caso Hierarquia Incorreta - Fase 3.

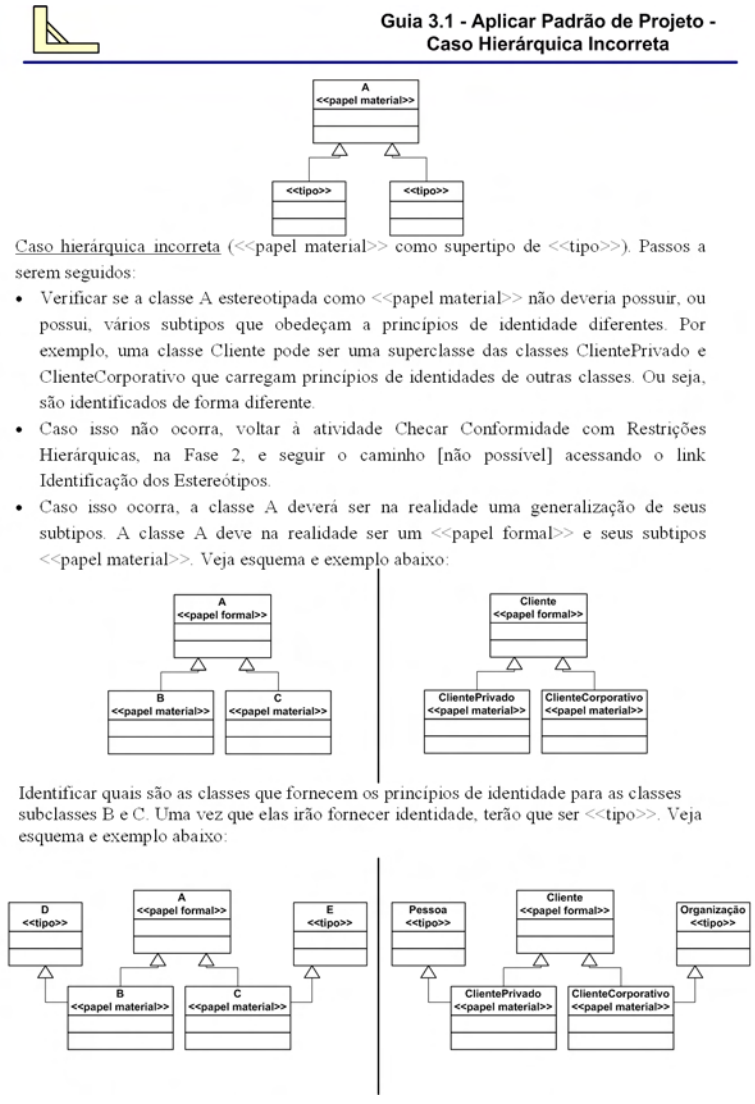


Figura C.2: Guia 3.1 - Aplicar Padrão de Projeto: Caso Hierárquica Incorreta. Orientações para aplicação do padrão de projeto.

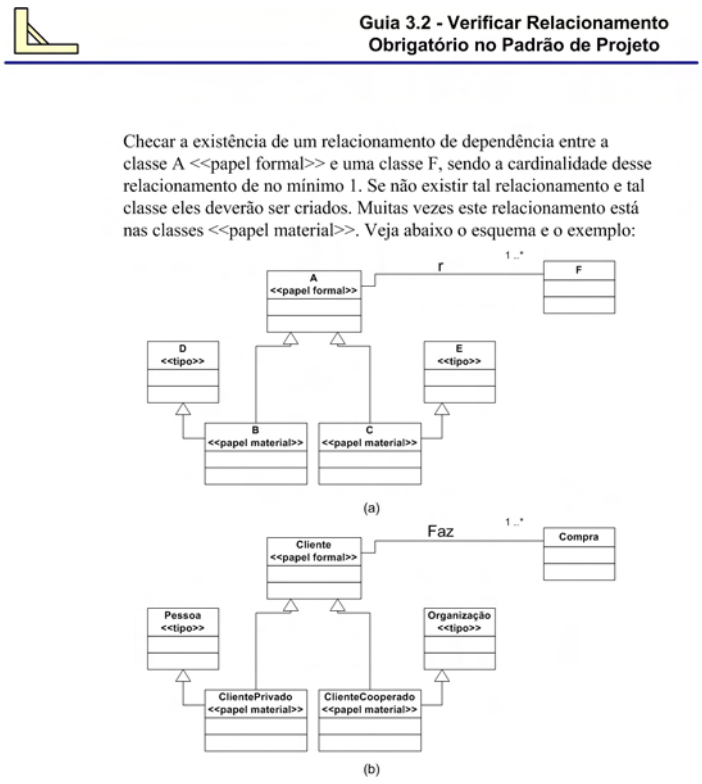


Figura C.3: Guia 3.2 - Verificar Relacionamento Obrigatório no Padrão de Projeto. Orientações para verificação de relacionamentos obrigatórios.

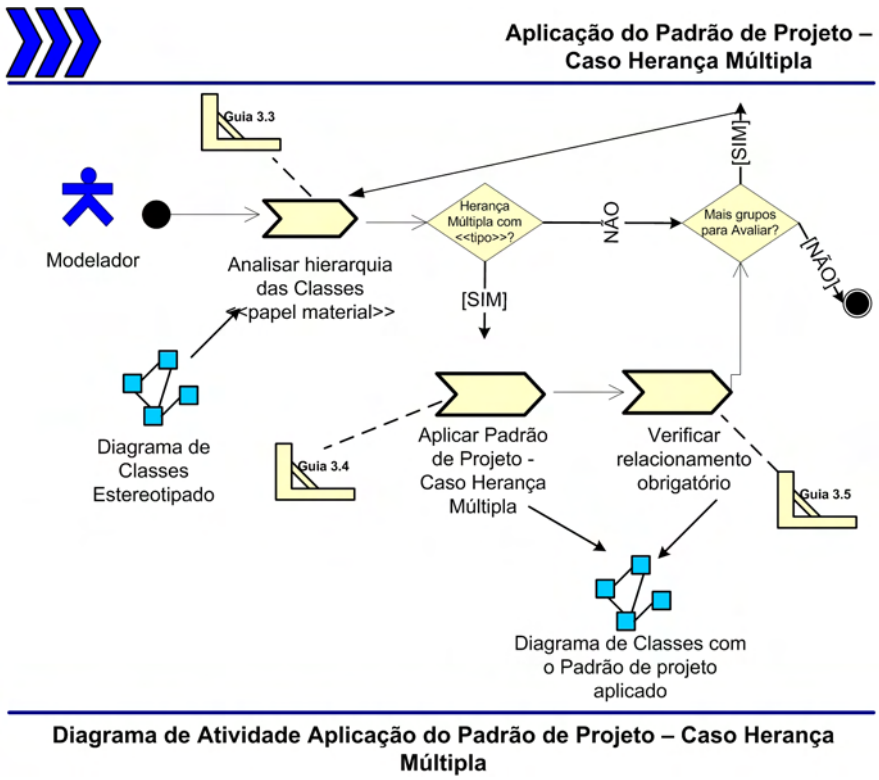


Figura C.4: Diagrama de atividades-Aplicação do Padrão de Projeto: Caso Herança Múltipla. Fase 3.

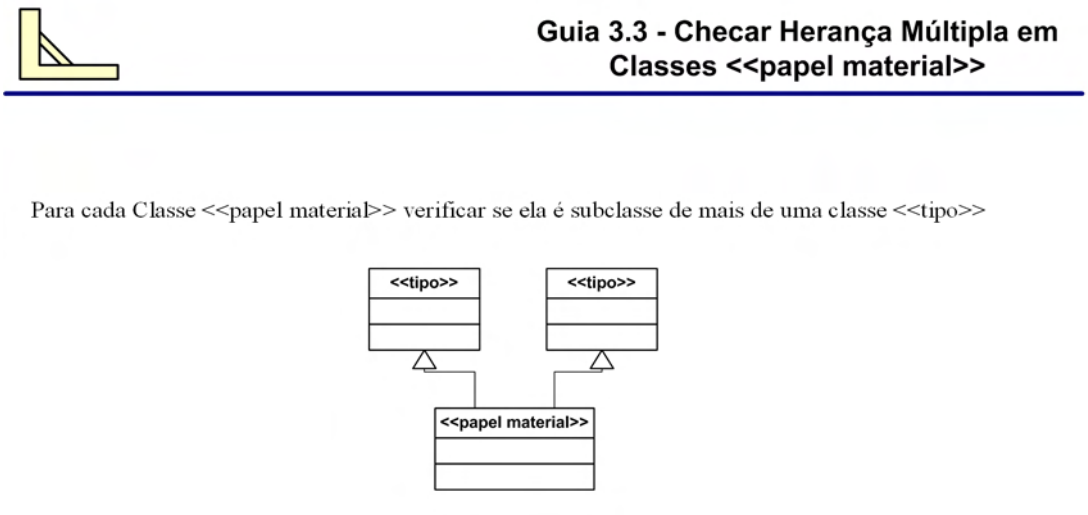


Figura C.5: Guia 3.3 - Checar Herança Múltipla em Classes «papel material».

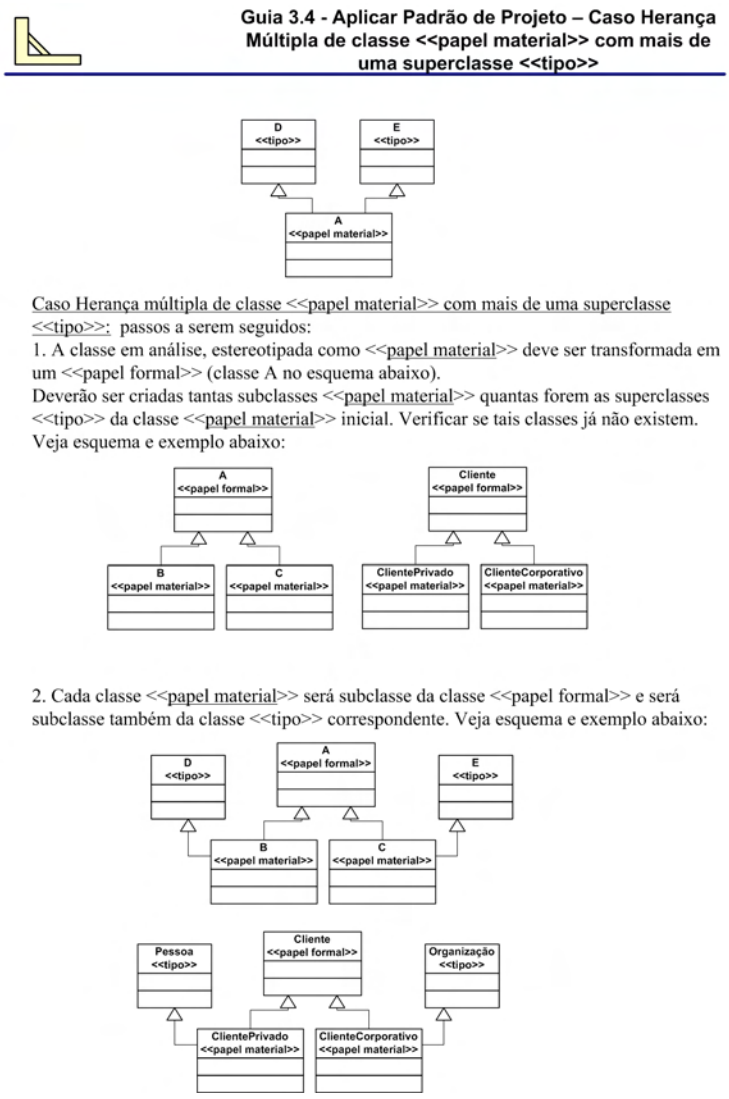


Figura C.6: Guia 3.4 - Aplicar Padrão de Projeto: Caso Herança Múltipla de classe «papel material» com mais de uma superclasse «tipo». Orientações para a aplicação do padrão de projeto.

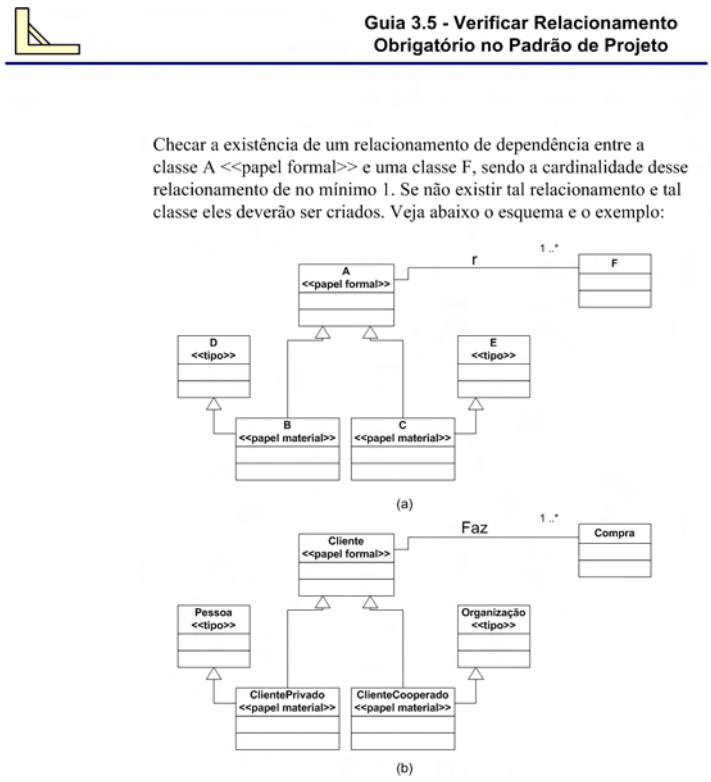


Figura C.7: Guia 3.5 - Verificar Relacionamento Obrigatório no Padrão de Projeto.

Apêndice D

O procedimento PrOntoCon - Fase 4

Anexo contendo as páginas referentes à Fase 4 do procedimento PrOntoCon: Verificação de Construtores.

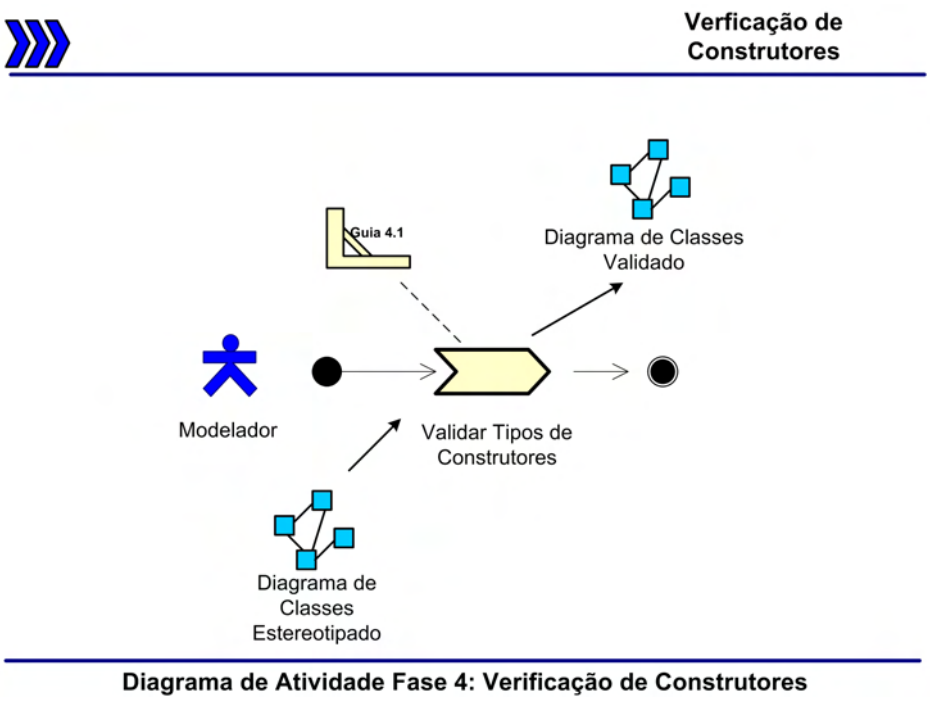


Figura D.1: Diagrama de Atividade Fase 4: Verificação de Construtores.

Guia 4.1 - Tabela de Possibilidades de Construtores

Para cada elemento do diagrama de classe, verificar se respeita ou não as restrições de tipos de construtores UML, conforme tabela mostrada a seguir. Alterar quando necessário.

Estereótipo	Construtor UML Permitido
<<tipo>>	• Interface; • Classe abstrata; • Classe concreta. **OBS**: Quando um elemento do diagrama de classe estiver classificado como <<tipo>> e possuir como especialização elementos classificados como <<fase>>, tal elemento deverá ser uma classe abstrata ou uma interface.
<<quase-tipo>>	• Interface; • Classe abstrata; • Classe concreta;
<<papel material>>	• Interface; • Classe abstrata; • Classe concreta;
<<fase>>	• Interface; • Classe abstrata; • Classe concreta;
<<categoria>>	• Interface; • Classe abstrata;
<<papel formal>>	• Interface; • Classe abstrata;

Figura D.2: Guia 4.1 - Tabela de Possibilidades de Construtores. Orientações para checar uso correto dos construtores UML no diagrama de classes.

Apêndice E

SPEM - Software Process Engineering Metamodel

Em função do crescente número de metodologias para modelagem de processos a OMG - *Object Management Group* publicou a UMP - *Unified Process Model*, uma proposta de unificação entre as diferentes metodologias existentes. Tal proposta evoluiu para o SPEM - *Software Process Engineering Metamodel Specification* que é o metamodelo para definição de processos e seus componentes. Em 2005 a primeira versão passou por uma revisão e tem-se atualmente a versão 2.0 (Instituite, 2008).

O SPEM - *Software Process Enginnering Metamodel* é um meta-modelo que pode ser usado para descrever um processo concreto ou uma família de processos de desenvolvimento de sofware relacionados. Ferramentas baseadas no SPEM são ferramentas para autoria e customização de processo. Não faz parte do escopo do SPEM a parte de execução de um projeto usando um processo modelado em SPEM.

A linguagem SPEM utiliza uma abordagem orientada a objetos e a UML (Unified Modeling Language). Possui um conjunto mínimo de objetos de modelagem que são necessários para se estruturar um processo de desenvolvimento de software. A expressividade da UML e a capacidade de modelagem dedicada ao domínio do processo de software são fatores favoráveis ao uso do SPEM na definição de processos. Desta forma é possível reutilizar os conhecimentos de modelagem de software utilizando UML na modelagem de processos de software.

Para se definir um processo, o SPEM utiliza vários elementos que têm por objetivo auxiliar na demonstração de como o processo será executado. Tais elementos podem descrever ou restringir o comportamento geral do processo a ser executado permitindo um melhor planejamento, execução e monitoramento do mesmo. Na Figura E.1 são apresentados os ícones e estereótipos dos principais elementos do SPEM. Em Lahoz (2004) encontra-se uma descrição de cada estereótipo do SPEM, conforme relacionado a seguir:

- *WorkProduct*: estereótipo utilizado com o objetivo de representar um artefato, algo que tenha sido produzido ou consumido pelo processo. Alguns exemplos de *WorkProducts*: modelos, planos, código, documentos, base de dados, etc. Os tipos de artefatos de um processo variam muito dependendo do domínio.

- *WorkDefinition*: é um tipo de *Operation* que descreve o trabalho executado no processo. Logo, descreve a execução, as operações e as transformações que os papéis operam nos *WorkProducts*. As subclasses de *WorkDefinition* são: *Activity*, *Phase*, *Iteration* e *LifeCycle*.

- *Guindance*: estereótipo que deve estar associado aos principais elementos do modelo com o objetivo de conter e descrever informações adicionais tais como técnicas, orientações, procedimentos, padrões, modelos de documentos, etc. Os tipos possíveis são: *Guideline*, *Tool*, *Checklist*, *Template*, *UMLProfile*,*Technique*.

- *Activity*: tem por objetivo descrever o que um *ProcessRole* executa. Num processo, as atividades são os principais elementos do trabalho e descrevem as tarefas, as operações e ações que são executadas por um papel. Uma *Activity* pode consistir de elementos atômicos chamados *Steps*.

- *ProcessPerformer*: utilizado para definir um executante para um conjunto de *WorkDefinition* em um processo. É usado para *WorkDefinitions* que não podem ser associadas a *ProcessRoles* individuais.

- *ProcessRole*: descreve os papéis, as responsabilidades e competências de um indivíduo na execução de *Activities* em um processo e a responsabilidade sobre

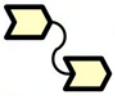
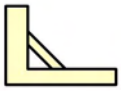
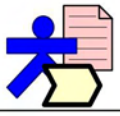
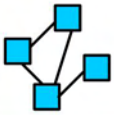
Estereótipo	Notação
WorkProduct	
WorkDefinition	
Guindance	
Activity	
ProcessPerformer	
ProcessRole	
ProcessPackage	
Phase	
Process	
Document	
UML Model	

Figura E.1: Principais estereótipos com respectivos ícones do SPEM. Fonte: Lahoz (2004).

certos *WorkProducts*. Pode estabelecer uma relação de dependência entre uma *WorkDefinition* e outra para indicar dependências início-início, fim-início ou fim-fim entre elas.

- *ProcessPackage*: representa um pacote do SPEM. *Package* é um recipiente que pode tanto possuir como importar elementos de definição de processo. Um *Package* pode ser especializado em *Diciplines*, cujo propósito é reunir atividades do processo que possuem um tema em comum.

- *Phase*: é uma especialização de um *WorkDefinition* com critério de entrada definido por pré-condição e critério de saída definido por seu objetivo (ou marco), com restrição de sequência. Logo, suas atividades são executadas de acordo com marcos distribuídos no tempo.

- *Process*: Descreve completamente um processo de engenharia de software, em termos de *ProcessPerformers*, *ProcessRoles*, *WorkDefinitions*, *WorkProducts* e *Guidances* associados.

- *Document*: Diferentes tipos de *WorkProduct* como por exemplo Documento Texto, um Modelo UML, Executável, Biblioteca de Código, etc.

- *UML Model*: utilizado quando existe um modelo UML representado no processo.

Com o objetivo de exemplicar o uso do SPEM na modelagem de processos, Gengivir et al. (2003) escolheram o processo de Engenharia de Requisitos e mais especificamente o subprocesso de Elicidação. Cada subprocesso pode ser representado como um ProcessPackage conforme Figura E.2.

O subprocesso de Elicidação é uma das disciplinas da Engenharia de Requisitos, que por sua vez é composta por quatro *WorkDefinition*: Compreender o Domínio da Aplicação, Compreender o Problema a Ser Resolvido, Compreender o Contexto do Negócio e Comprender as Necessidades dos Usuários.

A Figura E.3 apresenta um Diagrama de Atividades da *WorkDefinition* Compreender o Domínio da Aplicação utilizando SPEM. Tal diagrama permite visualizar a sequência, a ordem e quais *ProcessRoles* executam cada uma das *Activities*.

Figura E.2: Pacote representante da *Discipline* Elicidação. Um Processo pode ser composto por várias *Disciplines*. Fonte: Gengivir et al. (2003).

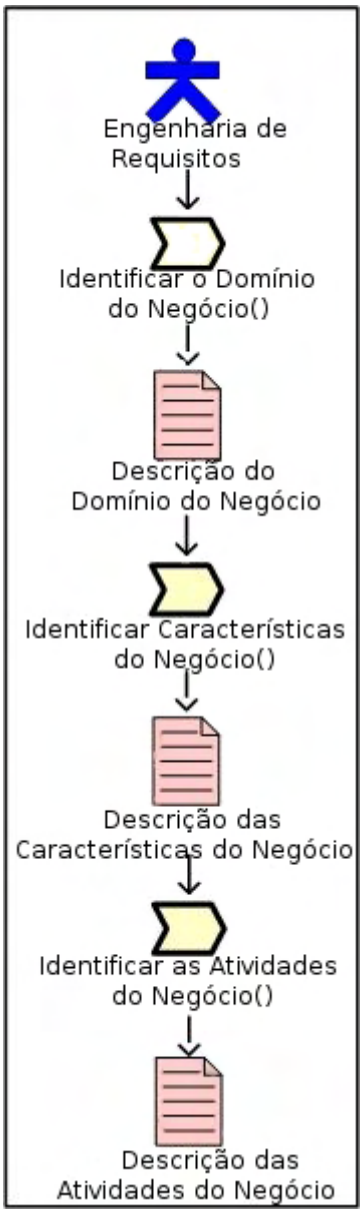


Figura E.3: Diagrama de Atividades do WorkDefinition Compreender Domínio da Aplicação. Fonte: Gengivir et al. (2003).

Referências Bibliográficas

Balbino, M. S. (2005). Sistema de controle de atividades de estagiários de psicologia.

Bunge, M., editor (1977). *Treatise on Basic Philosofy: Vol. 3: Ontology I: The Furniture of the World*. Reidel Publishing Company.

Bunge, M., editor (1979). *Treatise on Basic Philosophy, Volume 4, Ontology II: A World of Systems*. Reidel Publishing Company.

Elmastri, R. and Navathe, S. B., editors (2000). *Fundamentals of database systems*. Addison-Wesley.

Eriksson, H.-E., Penker, M., Lyons, B., and Fado, D., editors (2004). *UML 2 Toolkit*. Wiley Publishing, Inc, Indianapolis, Indiana.

Fowler, M., editor (1997). *Analysis Patterns: Reusable Object Patterns*. Addison-Wesley.

Gengivir, E. C., Sant'ana, N., and Filho, L. F. B. (2003). Modelando processos de software através do spem - software process engineering metamodel - conceitos e aplicação. *Workshop dos Cursos de Computação Aplicada do INPE*.

Guarino, N. (1998). Formal ontology and information systems. *In: Proceedings of the First International Conference on Formal Ontology in Information Systems*.

Guarino, N. and Welty, C. (2000a). A formal ontology of properties. *In R. Dieng, Ed., Proceedings of 12th Int. Conf. On Knowledge Engineering and Knowledge Management*.

Guarino, N. and Welty, C. (2000b). Identity, unity, and individuality: Towards a formal toolkit for ontological analysis. *In Proceedings of the ECAI-2000: The European Conference on Artificial Intelligence.*

Guarino, N. and Welty, C. (2000c). Ontological analysis of taxonomic relationships. *Proceedings of ER-2000: The International Conference on Conceptual Modeling.*

Guarino, N. and Welty, C. (2000d). Towards a methodology for ontology based model engineering. *In Proceedings of the ECOOP-2000 Workshop on Model Engineering.*

Guarino, N. and Welty, C. (2001a). Conceptual modeling and ontological analysis. Technical report.

Guarino, N. and Welty, C. (2001b). Identity and subsumption. *LADSEB-CNR Internal Report 01/2001.*

Guarino, N. and Welty, C. (2002). Evaluating ontological decisions with ontoclean. *Communications of the ACM*, 5(2).

Guizzardi, G., editor (2005). *Ontological Foundations for Structural Conceptual Models.* Universal Press, The Netherlands.

Guizzardi, G., Falbo, R., and Guizzardi., R. (2008). A importância de ontologias de fundamentação para a engenharia de ontologias de domínio: o caso do domínio de processos de software. *IEEE Latino-americano.*

Guizzardi, G. and Wagner, G. (2004). On a unified foundational ontology and some applications of it in business modeling. *Open INTEROP Workshop on Enterprise Modelling and Ontologies for Interoperability, 16th International Conference on Advances in Information Systems Engineering.*

Guizzardi, G., Wagner, G., and Nicola Guarino, M. v. S. (2004a). An ontologically well-founded profile for uml conceptual models. *In Proceedings of the 16th International Conference on Advanced Information Systems Engineering.*

Guizzardi, G., Wagner, G., and van Sinderen, M. (2004b). A formal theory of conceptual modeling universal. *Proceedings of the International Workshop on Philosophy.*

Instituite, O. M. (2008). Software and systems process engineering meta-model specification. Technical report.

Lahoz, C. H. N. (2004). Uma abordagem para a gerÊncia da qualidade em um ambiente de engenharia de software centrado em processo. Master's thesis, Instituto Nacional de Pesquisas Espaciais, INPE - São José dos Campos - SP - Brasil. Dissertação de Mestrado.

Moreira, J. S. (2007). Sistema de controle de contratos para fgpa:fundação geraldo perlingeiro de abreu.

Oliveira, A. P. and Biasutti, O. H. S. (2007). Procedimento de análise para validação de diagrama de classes de domínio baseado em modelagem ontológica.

Paula, W. P., editor (2003). *Engenharia de Software: Fundamentos, Métodos e Padrões.* LTC Editora, Rio de Janeiro, RJ, Brasil.

Steimann, F. (2000). On the representation of roles in object-oriented and conceptual modeling. *Data E Knowledge Engineering.*

Villela, M. L. B. (2004). Validação de diagramas de classe por meio de propriedades ontológicas. Master's thesis, Universidade Federal de Minas Gerais, UFMG - Belo Horizonte - MG - Brasil. Dissertação de Mestrado.

Villela, M. L. B., de Paiva Oliveira, A., and Braga, J. L. (2004). Modelagem ontológica no apoio à modelagem conceitual. *XVIII Simpósio Brasileiro de Engenharia de Software.*

Wand, Y., Storey, V. C., and Weber, R. (1999a). An ontological analysis of the relationship construct in conceptual modeling. *ACM Trans. Database Syst.*, 24(4):494–528.

Wand, Y., Storey, V. C., and Weber, R. (1999b). An ontological analysis of the relationship construct in conceptual modeling. *ACM Trans. Database Syst.*, 24(4):494–528.

Wand, Y., Storey, V. C., and Weber, R. (1999c). An ontological analysis of the relationship construct in modeling conceptual. *ACM Transactions on Database Systems*, 24(4).

Wazlawick, R., editor (2004). *Análise e projeto de sistemas de informação orientados a objetos*. Elsevier, Rio de Janeiro, RJ, Brasil.

Welty, C. and Guarino, N. (2001). Suporting ontological analysis of taxonomic relationships. *Data e Knowledge Engineering*, 39(4).