UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

MODELAGEM CONCEITUAL DE REGRAS DE NEGÓCIO BASEADA EM ONTOLOGIA DE FUNDAMENTAÇÃO

Mauro Moura Gomes Lopes

Orientador
Fernanda Araujo Baião Amorim
Co-orientador
Sean Wolfgand Matsui Siqueira

Rio de Janeiro, RJ – Brasil
Setembro de 2011

MODELAGEM CONCEITUAL DE REGRAS DE NEGÓCIO BASEADA EM ONTOLOGIA DE FUNDAMENTAÇÃO

Mauro Moura Gomes Lopes

DISSERTAÇÃO APRESENTADA COMO REQUISITO PARCIAL PARA OBTENÇÃO DO TÍTULO DE MESTRE PELO PROGRAMA DE PÓSGRADUAÇÃO EM INFORMÁTICA DA UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO (UNIRIO). APROVADA PELA COMISSÃO EXAMINADORA ABAIXO ASSINADA.

Aprovada por:

_____

Fernanda Araujo Baião Amorim, D. Sc. – UNIRIO

_____

Sean Wolfgand Matsui Siqueira, D. Sc. – UNIRIO

_____

Ricardo de Almeida Falbo, D. Sc. - UFES

_____

Flávia Maria Santoro, D. Sc. - UNIRIO

Rio de Janeiro, RJ - Brasil

Setembro de 2011

Lopes, Mauro Moura Gomes.

L864 Modelagem conceitual de regras de negócio baseada em ontologia de fundamentação / Mauro Moura Gomes Lopes, 2011.

223f.

Orientador: Fernanda Araújo Baião Amorim.

Co-orientador: Sean Wolfgand Matsui Siqueira.

Dissertação (Mestrado em Informática) – Universidade Federal do Estado do Rio de Janeiro, Rio de Janeiro, 2011.

1. Modelagem conceitual. 2. Ontologia (Informática). 3. Regras de negócio.

Dedicatória

À minha Linda, meu pai, minha mãe, Sadinha e minhas avós.

Agradecimentos

Agradeço à Cássia, que tomou conta do meu coração, espírito e saúde, durante esses dois anos e meio. Seus sorrisos me salvaram nos momentos mais conturbados, sorrisos inesquecíveis e essenciais para mim. Sua atenção em nossas conversas, em meus questionamentos, trazendo ideias diferentes das minhas, enriquecendo meu pensamento e visão do mundo. Sua preocupação, que me encheu de um sentimento sem palavras. Seu entendimento do que é isso tudo, por já ter passado pela experiência do mestrado, ajudando-me a entendê-lo melhor. Esse trabalho tem partes dela, transferidos por mim nessas letras. Na verdade, não tenho como agradecer ao seu companheirismo. Só consigo retribuir com meu amor.

Agradeço à minha família, que entendeu a minha ausência. Em especial, meu pai Jayme, mãe Gilberta, irmã Sara e avó Sara que são o meu lar de todo dia. Não tomaram conta de mim, pois eu estava escondido, fechado nas minhas atividades. Peço desculpas. Estou voltando.

Agradeço aos meus orientadores, Fernanda Baião e Sean Siqueira, que me guiaram durante todo esse tempo. Agradeço a forma como me preparam, sejam com ideias, revisões, escrita de texto, apresentações. Agradeço as oportunidades que me deram de viajar o mundo. Em especial, agradeço à Fernanda, que me acolheu faz mais de três anos. Aprendi muito para descrever em um parágrafo. Sou uma cria da UNIRIO.

Agradeço aos professores Ricardo Falbo e Flávia que aceitaram participar de minha banca. O frio da barriga de defender a minha dissertação para professores tão graduados é indescritível.

Agradeço aos “participantes” da minha validação, as moças que me ajudaram muito: Juliana França, Lúcia Castro e Natália Padilha. As horas que passaram me ajudando (sejam na UNIRIO ou fora dela) e a dedicação tornaram possível avaliar a proposta desse trabalho. Estou eternamente em débito com elas.

Agradeço também ao Rafael Lage, que em momento essencial do trabalho, me ajudou a esclarecer muitas coisas com a sua visão imparcial das ideias.

Agradeço aos mestres da UNIRIO. Tenho o orgulho de encontrá-los nos corredores, e cumprimentar sempre com um “Oi, professor”.

Agradeço às moças da NP2Tec, abrindo suas portas para que eu participasse de pesquisas e começasse a vida acadêmica. Os projetos que participei tornaram esse trabalho possível.

Agradeço ao pessoal da secretária da UNIRIO, a quem perturbo faz mais de sete anos.

Agradeço aos meus colegas de trabalho, que, em tão pouco tempo, me cativaram imensamente. Em especial ao Sérgio Kikkawa, Julliana Vieira e Marcos Sobral, que aceitou me dividir com o mestrado durante esse tempo.

E, finalmente, agradeço aos meus amigos, tantos que não tenho como citar todos os nomes sem exagerar no número de páginas ou esquecer alguém. Desde Eulina Ribeiro, dicas musicais, Pedro II, CEFET, Ponto de Ensino, Unirio... Não vejo muitos faz tempo, mas isso vai mudar.

Agradeço a esses dois anos e meio de experiência, que me trouxeram tantos aprendizados, de tantas formas. Principalmente, um autoconhecimento que levarei para o resto da vida.

LOPES, Mauro Moura Gomes. **Modelagem Conceitual de Regras de Negócio baseada em Ontologia de Fundamentação**. UNIRIO, 2011. 223 páginas. Dissertação de Mestrado. Departamento de Informática Aplicada, UNIRIO.

RESUMO

Regras de Negócio são declarações que definem ou restringem algum aspecto de uma organização. Esse tipo de regra é uma fonte importante de conhecimento sobre a conceitualização[1] de um domínio, especialmente quando consideradas em seu nível conceitual, desprendidas de características ou aspectos tecnológicos ou de implementação. A perspectiva de regras de negócio contempla tanto a visão comportamental quanto a visão estrutural de um domínio. Essas duas conceituações complementares compreendem o conhecimento necessário para definir a essência do domínio ou negócio sendo explorado.

Entretanto, as linguagens de representação de regras de negócio atuais não possibilitam uma representação precisa e livre de ambiguidades. Com isso, podem gerar mais de uma interpretação a partir de um mesmo modelo. A má interpretação do domínio pode gerar diversos problemas na aplicação das regras de negócio, ocorrendo eventos que desobedecessem a essas regras que regem a dinâmica do negócio.

Pressupõe-se que, se a linguagem representação de regras for baseada em ontologias de fundamentação, então o modelo resultante da sua utilização pode ser mais completo, válido e preciso do que uma representação não baseada em uma ontologia de fundamentação. Para confirmar esta hipótese, este trabalho propõe o R-OntoUML, um perfil de representação para a modelagem de regras de negócio do tipo Derivação e Reação. O perfil R-OntoUML é baseado na ontologia de fundamentação UFO e compatível com a sua linguagem de representação subjacente, OntoUML. Ainda, R-OntoUML acrescenta construtos modificados da URML e OCL para tratar os elementos da representação de Regras de Negócio que a OntoUML não considera. A proposta foi avaliada em um estudo de caso, onde foram obtidos modelos de regras de negócio com maior precisão.

[1] do inglês, *conceptualization*.

Palavras-chave: Modelagem Conceitual; Regras de Negócio; Ontologia; Linguagem de Modelagem Baseada em Ontologia de Fundamentação; OntoUML.

ABSTRACT

Business Rules are statements that define or restrict some aspect of an organization. They consist of an important source of knowledge about a domain, especially when considered in the conceptual level, detached from any technological or implementation characteristics.

The Business Rules perspective covers both the behavioral and structural view of a domain. These two complementary conceptualizations comprise the required knowledge to define the essence of the domain or business being explored.

However, current Business Rules representation languages do not allow a precise representation. Using these languages, the resulting models are subject to ambiguous interpretations of the same domain. The misinterpretation of the domain can generate a wrong enforcement of the Business Rules, leading to erroneous and inconsistent implementations to conduct business dynamics.

We argue that if the rule representation language is based on a foundational ontology, then the resulting models will be more complete, valid and precise than the ones produced using a representation language that is not based on a foundational ontology. To corroborate this hypothesis, this work proposes R-OntoUML, a UML profile to represent derivation and reaction business rules. The R-OntoUML profile is based on the UFO foundational ontology and is compatible with its underlying representation language, OntoUML. Moreover, R-OntoUML adds modified constructs from URML and OCL to address some representation elements that are not covered by OntoUML. The proposal was evaluated through a case study, where business rules models with greater precision were obtained.

Keywords: Conceptual Modeling; Business Rules; Ontology; Well-founded Ontology Modeling Language; OntoUML.

Índice

Índice de Figuras

1 Introdução

Modelagem conceitual consiste em uma representação mais abstrata (ou alto nível) que deve refletir a realidade de um domínio (ELMASRI e NAVATHE, 2002). Representa a forma como os conceitos de um domínio são classificados, especificando hierarquias e relações entre esses conceitos, assim como a perspectiva comportamental do domínio, explorando a dinâmica entre os conceitos e restrições que regem o comportamento desses conceitos.

Por ser computacionalmente independente, a modelagem conceitual de um domínio representa a conceitualização deste domínio usando uma linguagem com menor dependência dos elementos específicos do vocabulário de sistemas de informação, usados pelos seus especialistas. Devido a essa maior universalidade na sua representação, um modelo conceitual facilita a comunicação com especialistas do domínio sendo representado, mas que desconhecem as linguagens usadas em sistemas de informação. Dessa forma, a discussão da conceitualização do domínio torna-se mais intuitiva para os interessados no modelo conceitual resultante.

No entanto, para que possa ser compartilhada, a conceitualização sobre uma realidade precisa ser especificada e externalizada segundo uma linguagem de modelagem. Conforme discutido em (GUIZZARDI, 2005), a relação entre a linguagem e a realidade de um domínio é intermediada por uma conceitualização. A Figura 1 apresenta as diferenças entre os elementos que fazem parte dessa relação.

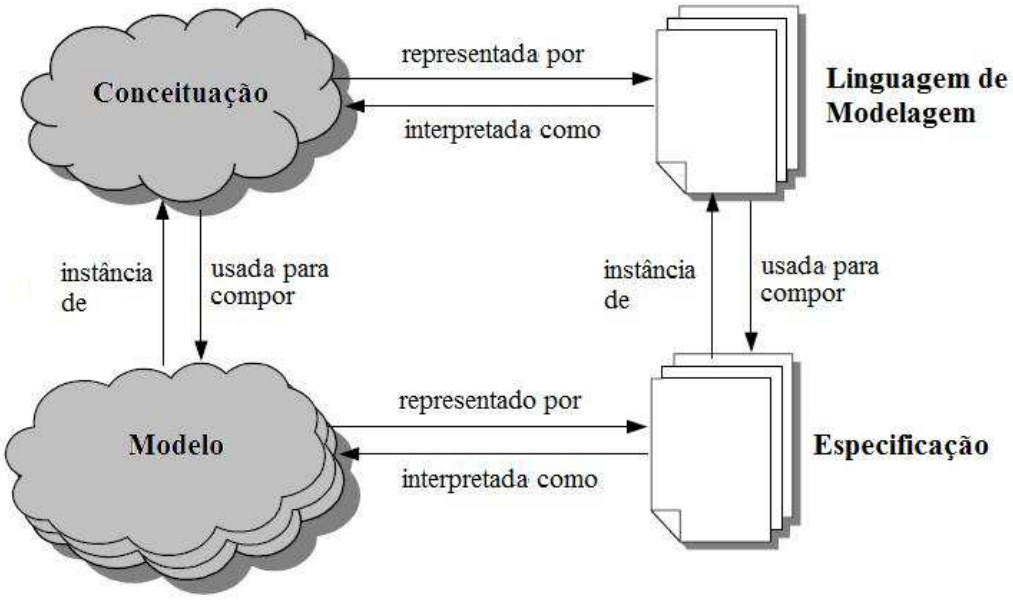


Figura 1 – Relação entre Conceitualização, Modelo, Linguagem de Modelagem e Especificação (GUIZZARDI, 2005).

A Conceitualização do domínio identifica como os conceitos de um domínio são classificados mentalmente por humanos, sendo usada para compor o Modelo. Um Modelo descreve o domínio, instanciando sua Conceitualização, sendo representado por uma Especificação. A Conceitualização é representada por uma Linguagem de Modelagem, um conjunto de construtos com meta-propriedades implícitas que classificam os elementos da Conceitualização. A Linguagem é usada para compor uma Especificação; o Modelo da Conceitualização é representado por uma Especificação. Essa Especificação é interpretada como um Modelo. Essas relações destacam como uma Conceitualização é descrita em um Modelo, que usa uma Especificação, cuja base é uma Linguagem de Modelagem.

Essas relações explicitam que o conhecimento sobre um domínio deve, idealmente, ser completamente refletido em modelos conceituais cuja linguagem de representação seja isomorfa em relação à especificação do domínio.

Uma fonte importante de conhecimento sobre a conceitualização de um domínio, especialmente quando consideradas em seu nível conceitual, desprendidas de características ou aspectos tecnológicos ou de implementação, é a representação de regras de negócio. Regras de Negócio (BRG, 2000) são declarações que definem ou restringem algum aspecto de uma organização.

O conjunto de Regras de Negócio contempla tanto a visão comportamental quanto a visão estrutural de um domínio. Essas duas conceituações complementares

compreendem o conhecimento necessário para definir a essência do domínio ou negócio sendo explorado.

Algumas linguagens focam na representação da perspectiva estrutural de um domínio, por exemplo, o diagrama de classes da UML (OMG, 2009), o diagrama entidade-relacionamento (CHEN, 1976) e ORM (HALPIN, 1995). Contudo, outros trabalhos visam à representação mais precisa desta perspectiva, como por exemplo, a OntoUML (GUIZZARDI, 2005).

Com objetivo de representar a perspectiva dinâmica das regras de negócio, existem linguagens como URML (WAGNER *et al*., 2004) e RuleML (HALPIN, 1996). Existem trabalhos que abordam a questão da dinâmica de um domínio dentro da modelagem conceitual, como (HEUSER *et al*., 1993), que poderia ser usada para a representação de regras de negócio.

Entretanto, as linguagens de representação de regras de negócio atuais não possuem uma representação precisa, especialmente no que se refere à perspectiva comportamental. Com isso, as regras podem gerar mais de uma interpretação de um mesmo domínio. A má interpretação do domínio poderia levar à aplicação das regras de negócio de forma errada, possibilitando o registro e a implementação de eventos que violam essas regras que regem a dinâmica do negócio. Tomemos o exemplo do modelo abaixo, que mostra a representação de um domínio de casamento, usando UML (OMG, 2009) e URML (WAGNER *et al*., 2004).

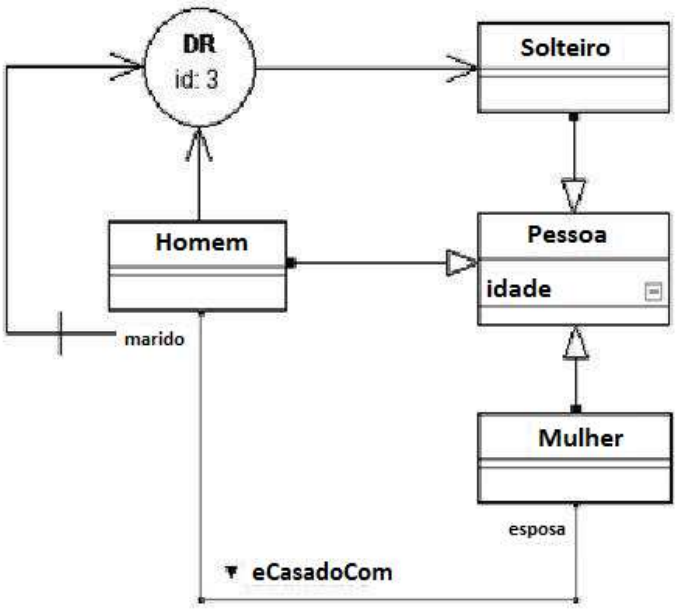


Figura 2 – Exemplo de Regra de Negócio em UML e URML. Traduzido de (REWERSE, 2011).

Nesse modelo, uma regra de negócio é representada, indicando que uma pessoa (“Pessoa”) do sexo masculino (“Home”), quando não é marido (“marido”), é solteira (“Solteiro”). Entretanto, não está explícito o que é solteiro, se é algo passageiro, quando uma pessoa é solteira ou não, como funciona o relacionamento do casamento (“eCasadoCom”). Esse conjunto de informações não está explícito, o que pode gerar mais de uma interpretação do modelo. Por exemplo, fica a dúvida se uma pessoa do sexo masculino pode voltar a ser solteira depois de casado. As meta-propriedades das classes não diferenciam pessoas do sexo masculino, feminino ou solteiros: a diferença está nos relacionamentos desses conceitos com o restante do domínio. Uma representação mais precisa deveria identificar as diferenças semânticas entre esses conceitos.

1.1 Problema

O problema a ser tratado no presente trabalho é a ambiguidade gerada pela representação a partir das linguagens atuais de modelagem conceitual, contemplando as perspectivas estrutural e comportamental de um domínio.

1.2 Hipótese e Proposta de Solução

A hipótese desse trabalho é resumida na seguinte frase:

“*se a linguagem representação de regras de negócio for baseada em ontologias de fundamentação, então o modelo resultante da sua utilização será mais completo, mais válido e mais preciso do que uma representação não baseada ontologia de fundamentação”*.

Este trabalho propõe uma solução para atender essa hipótese. A representação de um modelo conceitual de forma mais precisa é buscada utilizando uma linguagem baseada em ontologias de fundamentação, como defendido em (BAIÃO *et al*. 2008) e (GUIZZARDI *et al*. 2010). Ontologias de fundamentação são ontologias de alto nível que representam conceitos globais, não restritos a um domínio de representação. Esse tipo de ontologia pode ser usado para a representação das meta-propriedades de uma linguagem para modelagem conceitual.

A representação de regras de negócio usando modelos conceituais baseados em ontologias de fundamentação resulta em um modelo mais completo (quando maior quantidade dos conceitos presentes do domínio é representada), mais válido (quando a representação usada representa corretamente as meta-propriedades do conceito sendo modelado) e mais preciso (quando a representação não acrescenta meta-propriedades à conceitualização do domínio) que modelos construídos com linguagens não baseadas em ontologia de fundamentação. Essa representação de regras de negócio não é explorada em (BAIÃO *et al*. 2008) e (GUIZZARDI *et al*. 2010).

A ontologia de fundamentação escolhida para ser utilizada é a Ontologia de Fundamentação Unificada (*Unified Foundational Ontology* – UFO) (GUIZZARDI, 2005), uma ontologia com base filosófica e cognitiva. UFO tem como objetivo apresentar construtos para representação de um domínio de forma precisa. As meta-propriedades desses construtos auxiliam o modelador a identificar e classificar os conceitos de um domínio.

A UFO é dividida em UFO-A, UFO-B e UFO-C. A UFO-A define o núcleo da UFO, em sua maior parte composto por conceitos estáticos, ou *endurants*, diferente da UFO-B, que representa eventos com ideia de continuidade, composta por *perdurants* (como são chamados os eventos). Por fim, a UFO-C representa conceitos relacionados com questões sociais e intenções, incluindo conceitos linguísticos (GUIZZARDI e WAGNER, 2005).

A OntoUML, a linguagem baseada na ontologia de fundamentação UFO-A, escolhida para este trabalho, não tem como objetivo representar certos conjuntos de regras. Devido a este fato, os elementos necessários para a modelagem dessas regras devem ser identificados, para que um perfil da UML seja proposto, integrado aos elementos da UFO e OntoUML. Perfil UML é uma extensão da UML padrão, acrescentando semântica aos construtos da UML. No caso, os construtos do diagrama de classes da UML terão as suas meta-propriedades estendidas para atender a proposta desse trabalho.

Usando o exemplo da Figura 2, o mesmo modelo anterior foi construído utilizando os construtos da OntoUML, junto com a representação da regra de negócio de derivação da URML. O resultado é apresentado na Figura 3.

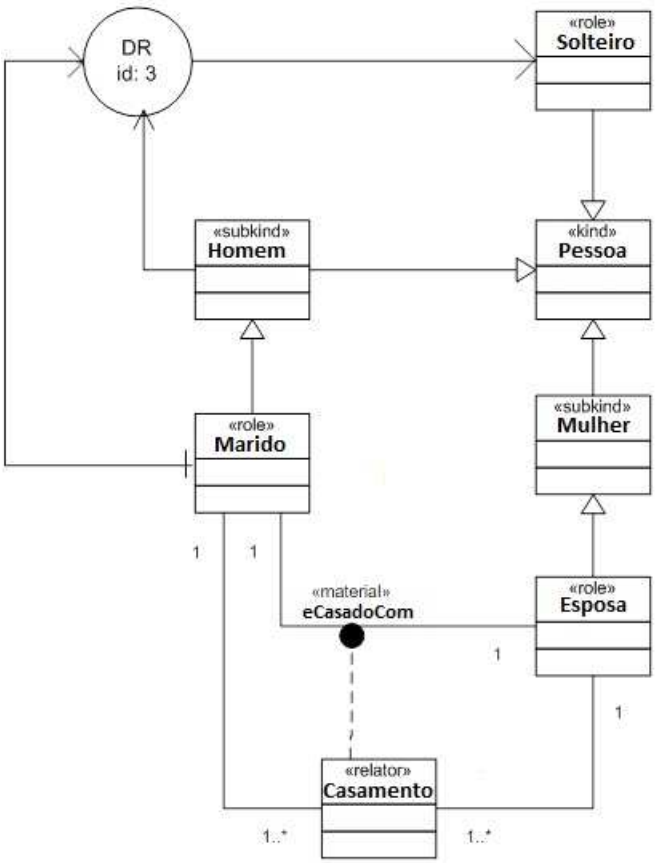


Figura 3 – Exemplo de Regra de Negócio da figura 2, acrescentando semântica da UFO.

Com os elementos da OntoUML acrescentados, algumas das questões que o modelo da Figura 2 levantou são respondidas. Como o conceito de solteiro é do tipo “role”, isso indica que uma pessoa do sexo masculino pode assumir o papel de solteiro, mas que ele pode ser passageiro e que pode ser assumido mais de uma vez durante sua existência, respeitando as Regras de Negócio do domínio: uma pessoa com papel de casado (“Marido” ou “Esposa”) trocará para o papel solteiro (no exemplo, somente é abordado o caso em que o marido vira solteiro). Outro elemento que não estava presente é matrimônio (“Casamento”), um conceito do tipo “relator”, que não estava presente na representação inicial. Esse conceito indica que o relacionamento de casamento existe somente quando há o casamento entre Pessoas com os papéis Marido e Mulher, sendo que cada um pode participar de mais um casamento durante sua existência.

Este trabalho propõe o R-OntoUML, um perfil UML baseado na ontologia de fundamentação UFO (considerando os construtos não usados na OntoUML, ou seja, explorando as UFO-B e UFO-C) e a linguagem de representação de regras URML, por ser visual e ser uma extensão da UML, assim como a OntoUML. Deve gerar uma

estrutura e construtos que, usados em conjunto da linguagem OntoUML, permitam a construção de um modelo sem ambiguidade, mais completo e correto.

Para a definição do perfil R-OntoUML serão vislumbradas possíveis extensões às linguagens de modelagem conceitual baseadas em ontologias de fundamentação existentes, especificamente a OntoUML, que sejam necessárias para a modelagem dos elementos de uma regra de negócio.

1.3 Objetivo

O objetivo desse trabalho é a proposta de uma linguagem de modelagem conceitual que permitisse a especificação de modelos mais completos, mais precisos e mais válidos.

Para atingir esse objetivo geral, outros objetivos específicos devem ser considerados. Há a necessidade de desenvolver um perfil UML baseado em uma ontologia de fundamentação chamada UFO, que seja mais completo, mais válido e mais precisa do que as formas de representação de regras de negócio existentes. Esse perfil deve ser testado, usando um estudo de caso para a sua avaliação, com a análise qualitativa dos resultados.

1.4 Metodologia científica

O Método de pesquisa usado é uma avaliação qualitativa, a partir de um estudo de caso explanatório, visando avaliar o perfil UML proposto para representação de regras de negócio em modelos conceituais. A avaliação qualitativa foi escolhido devido à subjetividade inerente às características da qualidade de um modelo, que é passível de uma avaliação quantitativa. Outro elemento do estudo de caso que levou ao uso da avaliação qualitativa foi a dificuldade de encontrar um número considerável de especialistas em OntoUML e Regras de Negócio.

A variável dependente desse trabalho é a percepção dos participantes sobre a proposta, avaliando a completude, precisão e validade dos modelos gerados. A variável independente é a linguagem para modelagem disponibilizada ao participante. No primeiro momento, a linguagem escolhida para comparação é utilizada, para o domínio do estudo de caso ser modelado novamente usando a proposta.

A coleta de dados foi proveniente de encontros com os participantes do estudo de caso e os resultados gerados pelos mesmos, na construção de modelos e respostas a questionário desenvolvido para a atividade. Esses encontros visaram à avaliação da proposta, com um treinamento inicial, explicando o básico da linguagem escolhida para a comparação, a primeira etapa de obtenção de dados. Em seguida a explicação da proposta, com a construção de outro modelo e questionário respondido. Em seguida, os participantes avaliaram modelos de seus pares, respondendo a novo questionário.

A avalição qualitativa dos resultados teve como base as respostas dos participantes, com o objetivo de identificar a percepção dos participantes de seus modelos, considerando a completude, precisão e validade dos mesmos. Anterior a essa avaliação qualitativa, os modelos construídos foram analisados e detalhados, permitindo o entendimento das respostas dos questionários.

Como resultados esperados, o modelo resultante da modelagem OntoUML junto com perfil proposto neste trabalho deve ser mais completo, mais válido e mais preciso na visão do grupo participante, em comparação com a UML junto com a URML. Caso contrário, a linguagem proposta não contempla as características necessárias para ser mais completa, mais válida e mais precisa.

1.5 Organização do Texto

Este trabalho é divido em seis capítulos. O segundo capítulo apresenta ontologias e linguagens de representação, com enfoque na UFO e OntoUML, seguido do terceiro capítulo, que aborda regras de negócio e suas linguagens de representação. O quarto capítulo aborda a interpretação de regras de negócio que será usada neste trabalho, uniformizando os tipos de regras que serão abordados e suas interpretações, junto com o perfil proposto neste trabalho, enquanto o quinto capítulo aborda uma avaliação dessa proposta e seus resultados. O sexto e último capítulo apresenta a conclusão. Os apêndices contêm os documentos criados para apoiar a avaliação da proposta. Em anexo, são encontrados os questionários e modelos usados no processo de avaliação.

2 Modelagem Conceitual e Ontologias

Em (CORCHO *et al*., 2003b), a definição da palavra ontologia é explicitada, vindo da filosofia, onde significa uma explicação sistemática do ser. A definição de (GRUBER, 1993) é uma das mais referenciadas na literatura: ontologia é "uma especificação explícita de uma conceitualização". Modificando essa definição, BORST (1997) disse que "ontologias são definidas como uma especificação formal de uma conceitualização compartilhada".

Como ontologias são utilizadas para satisfazer diferentes propósitos (processamento de linguagem natural, gerência de conhecimento, comércio eletrônico, integração inteligente de informação, web semântica etc.) em diferentes comunidades (engenharia de conhecimento, inteligência artificial, banco de dados, engenharia de software etc.), USCHOLD e JASPER (1999) deram outra definição para ontologias: "uma ontologia pode ter uma variedade de formas, mas necessariamente inclui um vocabulário de termos e alguma especificação de seu significado. Isto inclui definições e uma indicação de como conceitos estão inter-relacionados, que coletivamente impõem uma estrutura no domínio e restringem as interpretações possíveis dos termos". Posteriormente, a definição de ontologia foi revista por GRUBER (2009) para enfatizar sua implementação através de sistemas de informação. Uma ontologia é uma estrutura composta por quatro conjuntos (conceitos, relações, atributos e tipos de dados), uma hierarquia de conceitos (taxonomia de conceitos) e uma hierarquia de relações (taxonomia de relações).

As ontologias são divididas em classificações diferentes como apresentadas em (GUARINO, 1997), que se relacionam segundo os níveis da Figura 4:

- Ontologias de alto nível, que incluem vocabulário relacionado a conceitos globais, não pertencendo a um único tema ou domínio. Esses conceitos são genéricos e podem ser especificados por ontologias de domínio ou tarefa.

- Ontologias de domínio, que representam um determinado domínio, podendo ser reutilizáveis dentro deste domínio. Os conceitos desse domínio e seus relacionamentos com as atividades relacionadas desse domínio são representados.
- Ontologias de tarefa, que representam conjunto de conceitos usados para resolver problemas que podem ser ou não de um mesmo domínio. Dessa forma, inclui nomes de conceitos genéricos, podendo ser reaproveitados em diferentes domínios.
- Ontologias de aplicação, que são ontologias criadas a partir de conceitos de um domínio particular, em certos casos, da aplicação de uma tarefa em particular em um domínio específico.

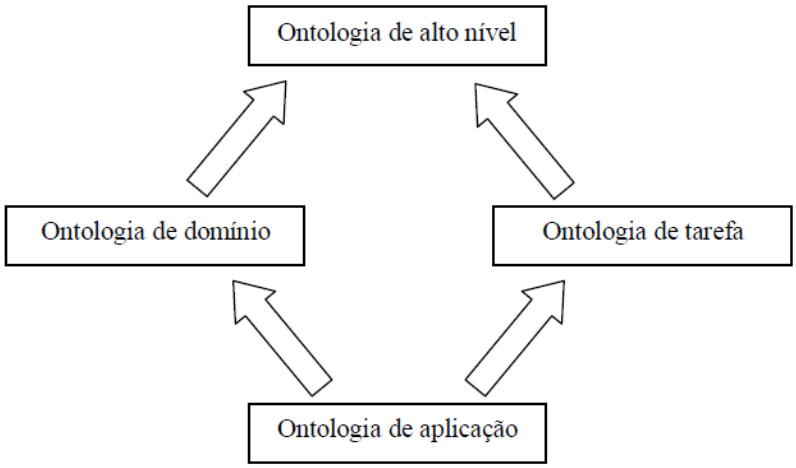


Figura 4 - Tipos de ontologias, traduzido de (GUARINO, 1997).

2.1 Linguagens de representação de ontologias

Existem linguagens criadas com o objetivo de representar ontologias, cada uma com um objetivo diferente.

Existem linguagens usadas para a representação do aspecto estrutural do domínio, explorando a hierarquia dos conceitos e seus relacionamentos. Segundo (CORCHO *et al.*, 2003a), linguagens de implementação de ontologias baseadas em inteligência artificial foram criadas, no início da década de 90. Entre essas linguagens, algumas eram baseadas em lógica de primeira ordem, como KIF (GENESERETH e FIKES, 1992), em frames combinados com lógica de primeira ordem, como Ontolingua

(GRUBER, 1992) e OCML (MOTTA, 1999), ou em lógica descritiva, como Loom (MACGREGOR, 1991).

Com o advento da internet, linguagens de ontologia foram criadas com objetivo de explorar as suas características na Web. Essas linguagens são conhecidas como linguagens de ontologias baseadas na web ou linguagens de marcação de ontologias. Como exemplos, tem-se RDF (LASSILA e SWICK, 1999); RDF Schema, uma recomendação da W3C, assim como RDF (W3C, 2004b); e OWL (DEAN *et al*., 2006), (W3C, 2004a).

Existem outras linguagens para representar conceitos e seus relacionamentos de uma forma mais abstrata (em modelos conceituais), como diagramas de Entidades-Relacionamentos, proposta por (CHEN, 1976). O diagrama de classes da UML (OMG, 2009) também é uma forma de representação dos conceitos e seus relacionamentos. Outras linguagens têm como objetivo representar o aspecto dinâmico de um domínio, explicitando seu comportamento, tal como a extensão da proposta de Chen desenvolvida em (HEUSER *et al*., 1993), que acrescenta representação de redes Petri ao diagrama Entidade-Relacionamento, visando à representação das propriedades estáticas e dinâmicas do sistema. Em (HALPIN, 1995) e (HALPIN, 1997), a ORM (Object-Role Modeling) é apresentada, uma linguagem e método orientado a fatos para realizar análise de informação a nível conceitual. No trabalho, a relação da ORM com Regras de Negócio é explorada, apresentando um conjunto de possibilidades de uso da linguagem para representação de Regras (HALPIN, 1996).

Entretanto, as linguagens citadas não têm como base uma ontologia de fundamentação, que é uma ontologia de alto nível. Esse tipo de linguagem reflete as meta-propriedades dos conceitos da ontologia de fundamentação, que agregam riqueza semântica aos modelos resultantes, conforme defendido em (GUIZZARDI *et al*., 2010).

Em (LOPES *et al*., 2009), um breve estudo da representação de modelos conceituais é realizado, comparando os construtos do Diagrama de Entidades-Relacionamentos (assim como os elementos do Diagrama de Entidades-Relacionamentos Estendido) e a OWL. Todos os construtos dos Diagramas de Entidades-Relacionamentos e do diagrama estendido são passíveis de representação na OWL, com o diferencial na OWL de, por exemplo, representar algumas meta-propriedades dos relacionamentos e operações de conjunto (que especificam o conjunto de instâncias de uma classe).

Em (GUIZZARDI *et al*., 2010), um mesmo domínio, parte do universo de exploração e produção de petróleo, é modelado em OWL-DL e OntoUML, linguagem proposta por Guizzardi (2005), baseada na ontologia de fundamentação UFO. Nesse trabalho são discutidos os conceitos do domínio que são explicitados pelo uso da OntoUML, antes implícitos na representação em OWL. As diferenças entre os modelos resultantes são apresentadas e comparadas através das meta-propriedades dos construtos UFO usados na OntoUML, os elementos que motivam essa explicitação de mais conceitos. A conclusão indica a necessidade de duas fases na modelagem de uma ontologia. Na etapa inicial, a OntoUML é usada para a modelagem conceitual do domínio com uma maior riqueza semântica, independente de mecanismos de inferência usados na Web Semântica. Na etapa final, o modelo em OntoUML seria fonte para a construção de um modelo em OWL, voltado para programação e uso de mecanismos de inferência para manipulação do modelo criado.

A representação de regras de negócio, por compreender um conhecimento rico e amplo sobre uma realidade, tanto sob a perspectiva estrutural quanto sob a perspectiva dinâmica, também deve ter as mesmas características de qualidade que qualquer modelo conceitual de dados, e que, portanto, deveria poder representar as regras de forma precisa. Com o uso de uma ontologia de fundamentação categorizada como uma ontologia de alto nível na hierarquia definida por GUARINO (1997), a riqueza semântica proporcionada faz com que a representação conceitual de Regras de Negócio seja mais precisa, mais completa e mais válida.

Portanto, a proposta deste trabalho utiliza como base a UFO (GUIZZARDI, 2005), que é uma ontologia de fundamentação que apresenta um conjunto de construtos que serão a base para a proposta de representação de Regras de Negócio em um modelo conceitual que seja mais completo, mais válido e mais preciso.

2.2 Ontologia de Fundamentação Unificada (*Unified Foundational Ontology* – UFO)

A UFO é uma ontologia de fundamentação. Ela explora os aspectos cognitivos e filosóficos para definir conceitos que se aproximam da forma que as conceituações são definidas, aproximando a linguagem de modelagem, o modelo e a especificação dessa conceitualização.

A UFO é dividida em UFO-A, UFO-B e UFO-C. A UFO-A define o núcleo da UFO, em sua maior parte composto por conceitos estáticos, ou *Endurants*, diferente da UFO-B, que é um incremento da UFO-A, representando eventos, definidos a partir de suas partes temporais, com ideia de continuidade, composta por *Perdurants*. Por fim, a UFO-C, que é um incremento das UFO-A e UFO-B, representa termos relacionados com questões sociais e intenções (assim como ações geradas a partir dessas intenções) (GUIZZARDI e WAGNER, 2005). Todos os conceitos de um domínio devem ser classificados em algum tipo de categoria definido na UFO. Este trabalho utiliza a UFO-A, UFO-B e parte da UFO-C, apresentadas a seguir.

2.2.1 UFO-A e OntoUML

A UFO-A define um conjunto de conceitos que são apresentados na Figura 5 (GONÇALVES *et al*., 2007). Por questões de simplificação, alguns construtos não são apresentados, mas serão abordados durante o texto.

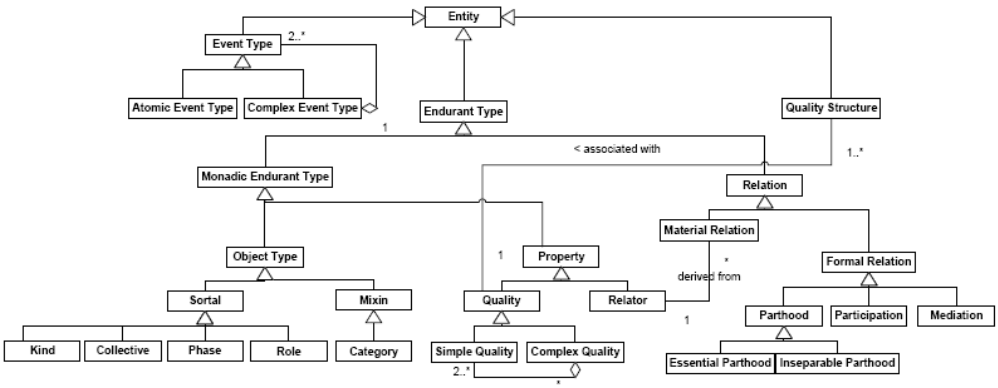


Figura 5 - Estrutura dos construtos da UFO-A. Extraído de (GONÇALVES *et al*., 2007).

A UFO é considerada uma ontologia de *universals*. *Universal* classifica um conjunto de entidades, que representam conceitos da realidade, em um dado mundo. Esses conceitos são compostos principalmente de *endurants* e *events*. *Endurants* mantém sua identidade durante a passagem do tempo, enquanto *events* são definidos por um momento em que algo acontece, podendo acontecer múltiplas vezes. Os *endurant* podem ser entidades *monadic* (que classificam um conjunto de instâncias substanciais, que persistem durante o tempo, mantendo suas identidades), que podem ser subdivididas em *object types* ou *properties*.

Instâncias de *object types* são entidades que possuem identidade completa, portanto, não são essencialmente dependentes de outras entidades, por exemplo, um doente existe em um dado mundo, independente de outras entidades presentes neste. Doente tem suas características definidas pela doença que o aflige. Nesse caso, a doença que aflige o doente. A doença é um *moment universal*, que tem suas características definidas por um período de tempo, existencialmente dependente de outros conceitos do domínio.

Properties são os atributos presentes no meta-modelo UML, representando propriedades dos conceitos representados. Instâncias de *properties* são entidades essencialmente dependentes de outras entidades.

2.2.1.1 Categorias de classes

Objetos podem ser categorizados em diferentes tipos, ou categorias. Alguns desses tipos instanciam os objetos em todas suas situações possíveis e definem o que é o objeto, de um ponto de vista metafísico: Tipo (*kind*), que define objetos gerais, e Coletivo (*collective*), para conjunto de entidades. Além desses elementos, há o Subtipo (*subkind*), que assume as características da classe que especializa (independente do tipo dela). Entretanto, existem tipos que instanciam objetos somente em certas circunstâncias: *phases* e *roles*.

Fase (*Phase*) instancia objetos durante um determinado período, mas não todos necessariamente, no mínimo e no máximo uma Fase. Por exemplo, as fases da vida de uma pessoa, adolescência e fase adulta: a pessoa não é adolescente e adulto ao mesmo tempo, assumindo essas fases durante períodos da sua vida.

A Figura 6 apresenta um exemplo de Fase, indicando o conjunto de fases que uma pessoa assume: criança, adolescente e adulto. Esse conjunto de fases é disjunto e completo, indicando que uma pessoa não pode ser criança e adulto ao mesmo tempo e que não existem outras fases para pessoa no universo sendo modelado. Como homem e mulher não possuem características que os classifiquem como outros tipos de construtos, eles são categorizados como *subkinds* de pessoa.

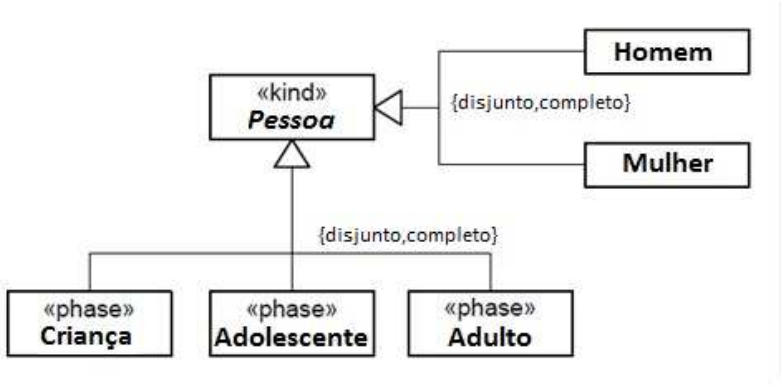


Figura 6 - Exemplo de *Phase*. Traduzido de (GUIZZARDI, 2005).

Papel (*Role*) instancia objetos em um contexto específico, durante um evento ou quando participa de um determinado relacionamento, podendo assumir mais de um papel ou nenhum, dependendo da situação do domínio. Por exemplo, o papel de uma pessoa: uma pessoa não é estudante durante sua vida toda, podendo deixar de ser um e voltar posteriormente, ou até assumir dois papéis, como estudante e professor; outro exemplo é o papel de doente, que uma pessoa assume somente quando possui uma doença.

A Figura 7 apresenta um exemplo de papel, em que uma pessoa pode assumir o papel de cliente pessoal e uma organização pode assumir o papel de cliente corporativo e fornecedor. Esses papéis não são disjuntos, pois uma organização pode assumir o papel de cliente em uma transação, enquanto é o fornecedor em outra, pode não estar executando nenhuma transação ou assumindo nenhum papel, assim como uma pessoa, quando não está fazendo compras.

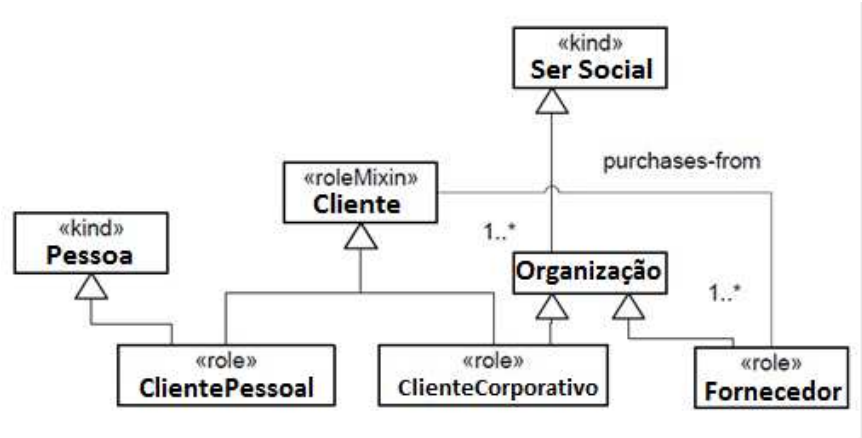


Figura 7 - Exemplo de *Role* e *RoleMixin*. Traduzido de (GUIZZARDI, 2005).

Outra categoria de classe apresentada na Figura 7 é *roleMixin*, que define um conjunto de características que são comuns entre vários *roles*. Na Figura 7, o *roleMixin*

Cliente possui características que são comuns aos papéis cliente pessoal e cliente corporativo, mas não dá identidade a nenhuma instância.

Mixin define propriedades que são essenciais para algumas de suas instâncias e acidentais para outras. A Figura 8 apresenta um exemplo de *mixin*, onde a característica de se poder sentar em algo é essencial pra uma cadeira, mas acidental para um engradado sólido: ele pode quebrar, deixando de ser um engradado sólido e passando a ser um engradado quebrado, não deixando de possuir a identidade de engradado, mas perdendo a identidade de ser algo em que se pode sentar.

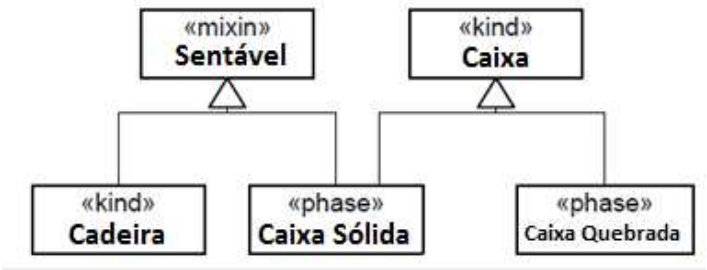


Figura 8 - Exemplo de *Mixin*. Traduzido de (GUIZZARDI, 2005).

Categoria (*Category*) classifica entidades que pertencem a tipos diferentes, mas que dividem uma propriedade essencial em comum. A Figura 9 apresenta um exemplo de *category*, representada pelo conceito entidade racional, que é abstrata, pois não pode dar identidade completa a um conceito. Dessa forma, os *kinds* pessoa e agente artificial dão identidade completa a conjuntos de instâncias diferentes, porém herdam as características de uma entidade racional, que são comuns aos dois *kinds*.

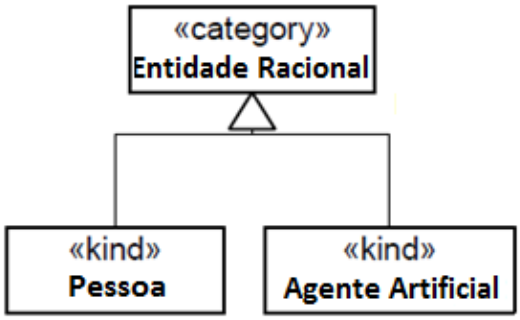


Figura 9 - Exemplo de *Category*. Traduzido de (GUIZZARDI, 2005).

Quantidade (*Quantity*) fornece princípios de identidade para conceitos representados por quantidades, que são definidos por sua massa ou espaço que ocupam. A Figura 10 apresenta um exemplo de *quantity*, representado pelos conceitos álcool e vinho que estão contidos em objetos do tipo *container* (presente na UML), constituem

uma safra de vinho e tem uma fase em que está armazenado em um tanque, no caso, um tanque de vinho (um tipo de *container*).

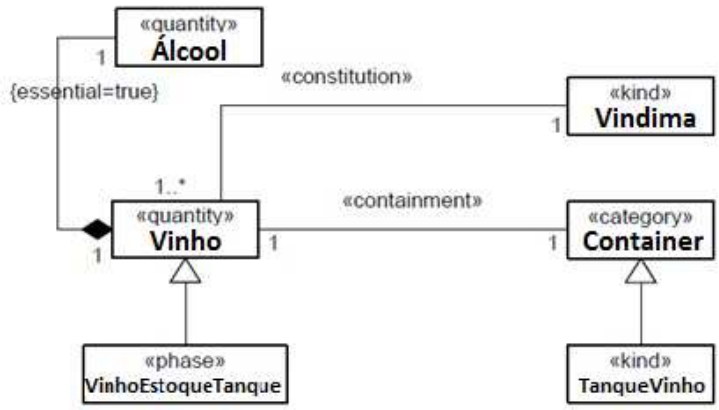


Figura 10 - Exemplo de *Quantity*. Traduzido de (GUIZZARDI, 2005).

A Figura 11 apresenta a estrutura taxonômica das categorias de classes.

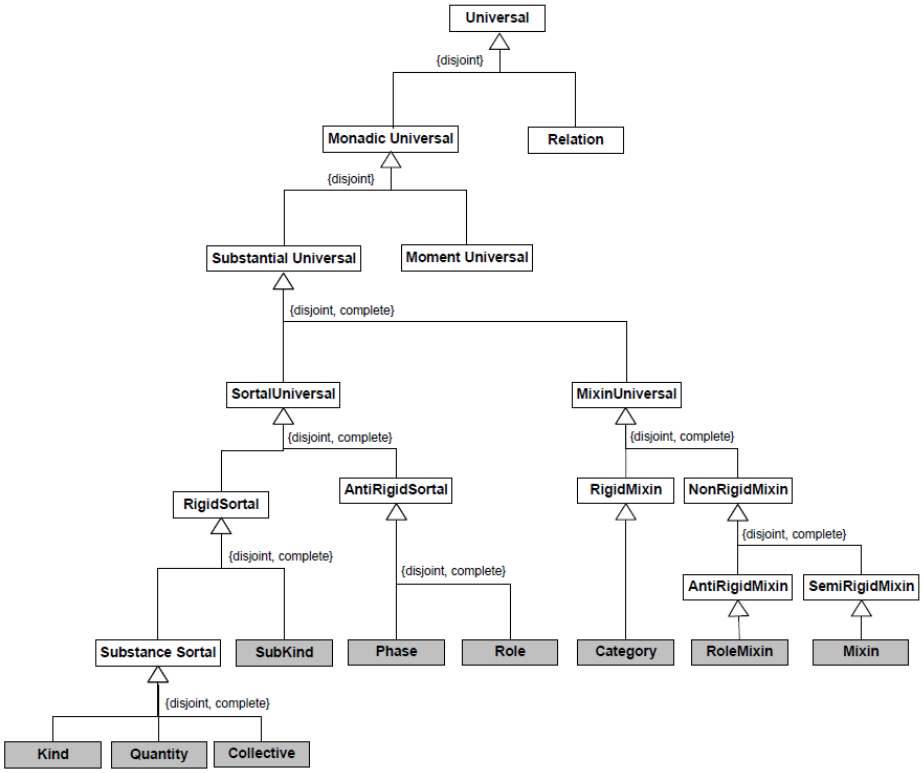


Figura 11 - Estrutura das categorias de classes (GUIZZARDI, 2005).

Outra categoria de classe é *mode*, que, assim como o *relator*, é um elemento que é definido por um momento específico do domínio sendo retratado. *Mode* provê mais características a um conceito em um dado momento sendo existencialmente dependente dele. *Relator*, que foi abordado anteriormente, é existencialmente dependente de ao menos duas outras entidades. O *Relator* representa um acontecimento que ocorre entre

duas ou mais entidades, identificando a existência de um relacionamento entre essas entidades. Esse relacionamento é do tipo *material*. A Figura 12 apresenta a taxonomia dessas categorias.

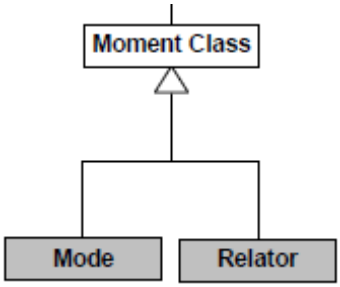


Figura 12 - Classificação de *Mode* e *Relator* (GUIZZARDI, 2005).

A Figura 13 apresenta um exemplo de *Mode*, um sintoma de um paciente, que é dependente da existência do paciente, e explicita sintomas de um paciente, que lhe conferem mais características que o definem enquanto está com o sintoma.

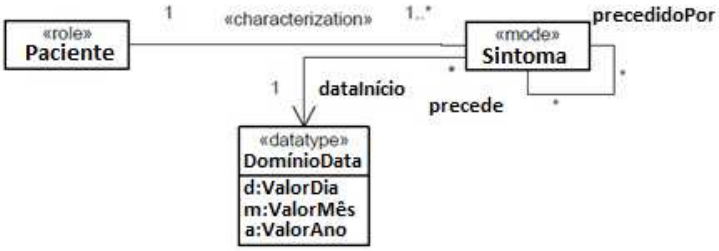


Figura 13 - Exemplo de *Mode*. Traduzido de (GUIZZARDI, 2005).

A Figura 14 apresenta um exemplo de *Relator*, que é o acontecimento de um tratamento realizado em uma unidade médica em um paciente. O relacionamento *material* "é tratado em" (*TreatedIn*) entre unidade médica e paciente é derivado do *relator* tratamento.

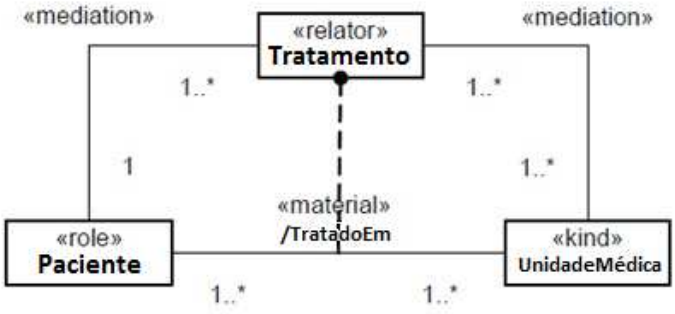


Figura 14 - Exemplo de *Relator*. Traduzido de (GUIZZARDI, 2005).

Este é o conjunto de categorias que podem identificar uma classe dentro da estrutura da UFO-A. Com esse conjunto de opções de classificação dos elementos do domínio, a escolha de qual categoria de classe se enquadra deve levar em consideração todas as propriedades dos conceitos do domínio.

2.2.1.2 Categorias de relacionamentos

Os objetos do domínio se relacionam de várias formas e um conjunto de tipos de relacionamentos é apresentado na UFO-A. Todo relacionamento do modelo deve ser justificado.

O relacionamento *formal* liga duas entidades de forma direta, apresentando uma relação interna ou uma relação de comparação, como, por exemplo, o fato de uma pessoa ser mais velha que outra.

Relacionamento de *characterization* é uma relação que identifica que uma classe do tipo *mode* está acrescentando características que identificam um conceito, em um dado momento, durante certo período. A Figura 13 apresenta um exemplo desse relacionamento, que ocorre entre o *role* paciente e o *mode* sintoma.

Os tipos de relacionamento *material*, *mediation* e *derivation* fazem parte de uma estrutura específica. Eles complementam a semântica necessária para representar um *Relator* e os relacionamentos com as entidades de que ele é essencialmente dependente. O relacionamento *material* representa um relacionamento material entre duas entidades ligadas a um *Relator* (esta ligação é representada pelo relacionamento *mediation*), que é derivado a partir da existência deste *Relator*. O relacionamento *derivation* identifica que o relacionamento *material* é derivado de um *relator*, e é identificado por uma linha pontilhada com um círculo preto colado ao *relator*, ligando o *relator* e o relacionamento *material* derivado. O relacionamento *mediation* liga um *relator* e cada entidade que faz parte do evento que o *Relator* representa.

A Figura 14 apresenta essa estrutura, que esses relacionamentos fazem parte. Enquanto o relacionamento *material* “é tratado em” explicita quantos pacientes podem receber tratamento em uma unidade médica e quantas unidades médicas podem tratar um paciente, os relacionamentos *mediation* entre o *relator* Tratamento e o papel Paciente, e o tipo Unidade Médica explicitam, respectivamente, quantos tratamentos são aplicados a um paciente e quantos pacientes podem ser tratados por um tratamento em

um dado momento, e quantos tratamentos podem ser executados em uma unidade médica e quantas unidades médicas podem executar um tratamento em um dado momento. A explicitação das possibilidades das cardinalidades dos relacionamentos do tipo *mediation* é explorada em (GUIZZARDI *et al*., 2010).

2.2.1.3 Relacionamentos parte-todo (Meronímia)

UFO-A permite a classificação de conceitos da realidade em várias categorias, e provê relacionamentos de parte-todo entre esses conceitos. O relacionamento *componentOf* representa um relacionamento de parte-todo entre conceitos complexos, como Tipo e Papel. A Figura 15 apresenta um exemplo do relacionamento *componentOf*, entre uma pessoa, que é composta por um cérebro, que é composto por um cerebelo. Esse relacionamento é modelado com um losango preto. As características *essential* e *inseparable* explicitam as propriedades dos relacionamentos por serem essenciais, em que pessoa não existe sem um cérebro e que um cérebro não existe sem um cerebelo, e inseparáveis, em que um cerebelo é existencialmente dependente de um cérebro e um cérebro é existencialmente dependente de uma pessoa.

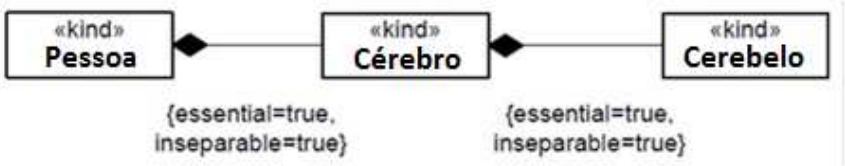


Figura 15 - Exemplo de relacionamento *componentOf*. Traduzido de (GUIZZARDI, 2005).

O relacionamento *subQuantityOf* representa um relacionamento parte-todo entre quantidades, sempre não compartilhável e limitado a cardinalidade de no máximo um no fim da associação. A Figura 17 apresenta um exemplo real do *subQuantityOf*, onde um container contém vinho, que é composto por suco de uva. A Figura 16 apresenta um exemplo de *subQuantityOf* de forma genérica. Esse relacionamento é modelado com um losango preto e a letra "Q" branca.

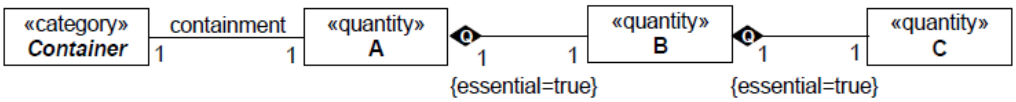


Figura 16 - Exemplo de *subQuantityOf*. Retirado de (GUIZZARDI, 2005).

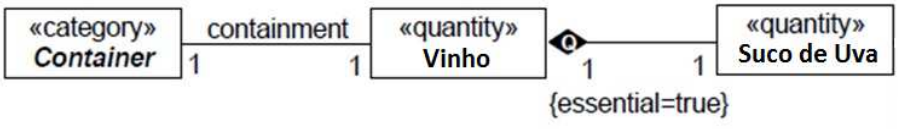


Figura 17 - Exemplo real de *subQuantityOf*.

O relacionamento *subCollectionOf* representa um relacionamento parte-todo entre *collectives*, limitado a cardinalidade de no máximo um no fim da associação. A Figura 19 apresenta um exemplo real do subCollectionOf, onde uma coleção de atletas é composta de uma coleção de atletas de esportes em grupo, composta por atletas de futebol, que assumem o papel de jogadores do meio-campo. A Figura 18 apresenta um exemplo de *subCollectionOf* de forma genérica. Esse relacionamento é modelado com um losango branco e a letra "C" preta.

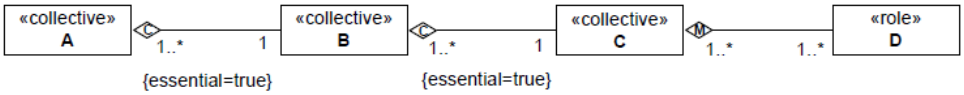


Figura 18 - Exemplo de *subCollectionOf*. Retirado de (GUIZZARDI, 2005).

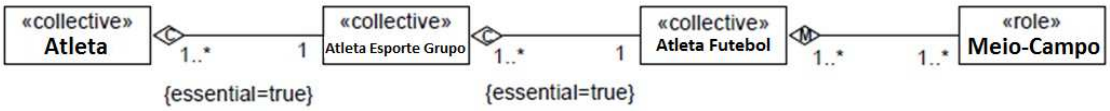


Figura 19 - Exemplo real de *subCollectionOf*.

O relacionamento *memberOf* representa um relacionamento parte-todo entre um conceito complexo (como um *kind* ou *role*) ou um *collective* (como uma parte) e outro *collective* (como um todo). A Figura 20 apresenta um exemplo de *memberOf*, onde um membro de um clube, que é uma pessoa, tem relacionamento *memberOf* (membro de um clube específico) com clube, que é membro de um corpo internacional. Esse relacionamento é modelado com um losango branco e a letra "M" preta.

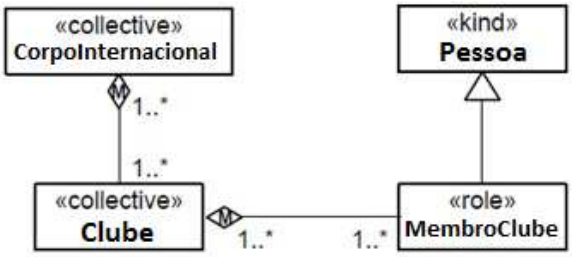


Figura 20 - Exemplo de memberOf. Traduzido de (GUIZZARDI, 2005).

Enquanto a UFO-A está formalizada em uma linguagem, a OntoUML, usando as meta-propriedades dos conceitos apresentados, a UFO-B e a UFO-C não estão

formalizadas em uma linguagem. A UFO-B e UFO-C são relevantes e, também, serão base para este trabalho. A seguir, os elementos dessas duas ontologias de fundamentação relevantes ao trabalho apresentado nesta dissertação são explorados. Foram usados como referência para definição dessas ontologias os trabalhos (GUIZZARDI e GUIZZARDI, 2010), (CARDOSO *et al*., 2010), (MARTINS *et al*., 2011) e (MARTINS, 2009).

2.2.2 Eventos e intervalos temporais: UFO-B

A UFO-B tem como objetivo representar os *Perdurants*, elementos que têm a sua existência definida a partir de um intervalo de tempo e suas relações com os *Endurants*. Enquanto que os indivíduos duradouros (*Endurants*) mantêm suas identidades durante toda a sua existência, os Eventos (*Events* - *Perdurants*) são definidos a partir do período de sua existência. Essa relação é onde a UFO-A se expande e se conecta com a UFO-B. Exemplos de eventos são partidas de futebol, a transmissão de uma sessão de um filme, um abraço e uma chuva. Esses eventos podem ser divididos em partes menores, possuindo identidade em um período de tempo pré-definido, sejam as partes que compõem o Evento, assim como o próprio Evento composto por suas subpartes. A Figura 21 apresenta o meta-modelo da UFO-B.

Existem dois conceitos com características semelhantes na UFO que podem gerar confusão: *Event* e *Relator*. Além de um *Event* poder ser a fundação de (relacionamento "*foundation of*") um *Relator*, a diferença entre esses conceitos é proveniente da dependência existencial do intervalo de tempo em que o *Event* acontece, enquanto que o *Relator* não tem sua identidade dependente de intervalos de tempo. No exemplo do tratamento de um doente (apresentado na Figura 14), o tratamento mantém a sua identidade, mesmo que o Evento que gerou esse conceito, como uma consulta realizada pela unidade médica, já tenha terminado: o tratamento altera as características do paciente e está sempre presente nele. O exemplo de casamento como contrato, que é o *Relator*, gerado a partir do casamento como festa e cerimônia civil, que é o Evento, é mais claro para identificar a diferença: o casamento como contrato é mantido até um possível divórcio, ao passo que o casamento como festa termina quando o horário pré-estabelecido para tal chega.

Os Eventos (*Events* ou *Perdurants*) alteram uma Situação (*Situation*), intervindo no estado das coisas do domínio sendo representado. A relação de Evento e os *Endurants* advêm dessa intervenção nas Situações do domínio, onde no mínimo um *Endurant* está presente (denotado pelo relacionamento "*is present in*", na Figura 21). O Evento altera um Pré-Estado (*pre-state*) para um Pós-Estado (*pos-state*).

Além dessa relação com a Situação, Eventos são existencialmente dependentes dos seus participantes, no sentido de que não existiriam caso não houvesse a participação de *Substantials*. Essa Participação (*Participation*) é um Evento, sendo individual para cada *Substantial* participante (cada Participação está ligada somente a um *Substantial*).

Eventos podem ser Eventos Complexos (*Complex Event*) ou Eventos Atômicos (*Atomic Event*). Eventos Complexos são Eventos compostos por no mínimo outros dois Eventos, sejam Atômicos ou Complexos, e as suas partes trazem identidade ao Evento Complexo. Eventos Atômicos são indivisíveis, não no sentido temporal, mas seguindo a semântica do domínio, ou seja: um Evento Atômico pode se estender por um período de tempo longo, assim como um Evento Complexo pode ser instantâneo, composto por outros Eventos Atômicos instantâneos.

Eventos são delimitados por Intervalos de Tempo (*Time Interval*), que são Estruturas Temporais (*Temporal Structure*), apresentando qualidades (ou propriedades inerentes ao conceito, a Estrutura de Qualidade – *Quality Structure*) a cada Evento. Intervalos de Tempo possuem subintervalos (relacionamento "*is subinterval of*"), e possuem um Ponto no Tempo (*Time Point*) de início (relacionamento "*begin*") e fim (relacionamento "*end*"). Um Ponto no Tempo é uma *Quale*, que provê as características do Evento ao definirem o início e fim do Intervalo de Tempo que delimita o Evento.

Um Intervalo de Tempo pode possuir uma ou mais Relações de Intervalos de Tempo (*Time Interval Relation*), onde pode ser fonte (relacionamento *source*) ou alvo (relacionamento *target*). As Relações de Intervalos de Tempo são Relações Formais (Formal Relation), definidas em 2.2.1.2.

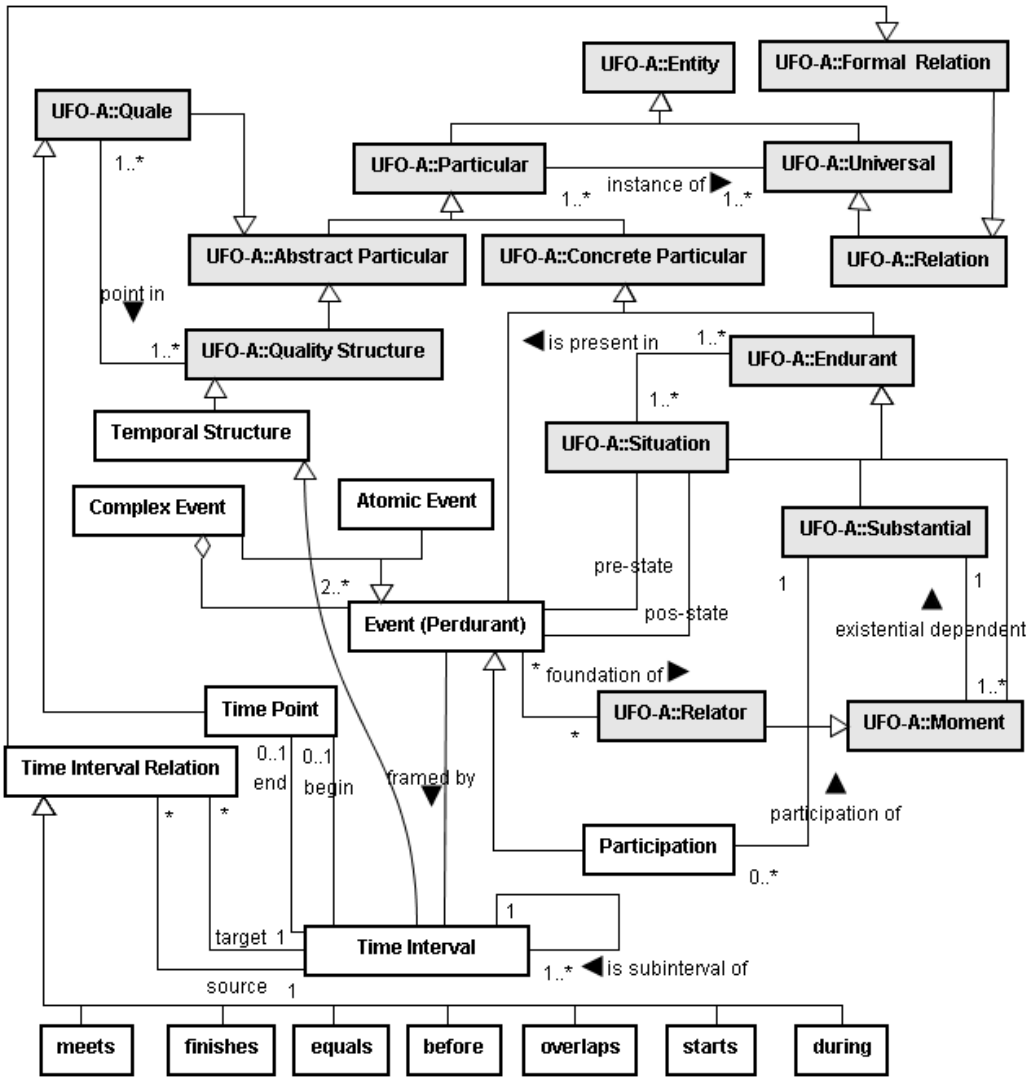


Figura 21 - Meta-modelo da UFO-B (MARTINS, 2009).

Essas Relações de Intervalo de Tempo da UFO são, conforme explicitado em (MARTINS, 2009), “definidas a partir das *relações entre intervalos de Allen* (ALLEN, 1983), a partir das quais as relações correspondentes entre eventos podem ser derivadas (GUIZZARDI *et al*., 2008)”. Essas relações são apresentadas na Figura 22, onde cada um dos tipos de Relação de Intervalo de Tempo é graficamente comparado a uma atividade referência para a definição das relações. A relação Antes (*before*) ocorre antes da atividade de referência, terminando antes do início da atividade. A relação Encontra (*meets*) também ocorre antes da atividade, terminando quando a atividade começa. A relação Sobrepõe (*overlaps*) inicia antes e termina no meio da atividade de referência. A relação Começa (*starts*) inicia junto, mas termina antes do término da atividade. A relação Durante (*during*) ocorre no meio da atividade, ou seja, começa depois do início

da atividade e finaliza antes do término da mesma. A relação Termina (*finishes*) começa no meio e termina junto da atividade. Por fim, a relação Igual (*Equals*) tem seu início e fim iguais à atividade de referência.

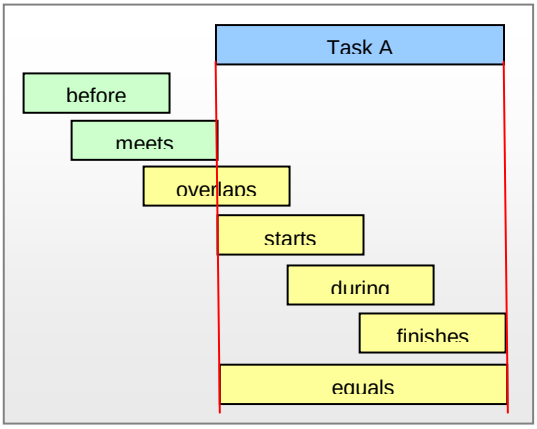


Figura 22 – Relações de Allen (apud MARTINS, 2009).

Com a definição da UFO-B, os conceitos da UFO-C relevantes ao trabalho apresentado nesta dissertação também serão explicados.

2.2.3 Aspectos sociais, intenções e ações: UFO-C

A UFO-C é a extensão da UFO-A e UFO-B, que tem como objetivo representar questões sociais, assim como pensamentos, de um domínio. Enquanto a UFO-B se caracteriza por expressar somente *Perdurants*, a UFO-C estende algumas definições dos eventos da UFO-B e elementos da UFO-A. A Figura 23 apresenta os construtos da UFO-C necessários para o trabalho apresentado nesta dissertação.

O conceito principal dessa ontologia para este trabalho é a Ação (*Action*), um subtipo de Evento, que se diferencia do Evento por ser causada (relacionamento "*caused by*") pela Intenção (*Intention*) de um Agente (*Agent*). Um Evento possui a Participação (*Participation*) de *Substantials*, porém não é iniciado por esses participantes.

Uma Intenção representa um desejo de um Agente de atender (relacionamento "*proposition content of*") a um Objetivo (*Goal*), um subtipo de Proposição (*Proposition*), visando alterar a Situação do domínio para atender às suas necessidades. Uma Intenção é um subtipo de Modo Mental (*Mental Moment*), que é um subtipo de Modo Intencional (*Intentional Moment*).

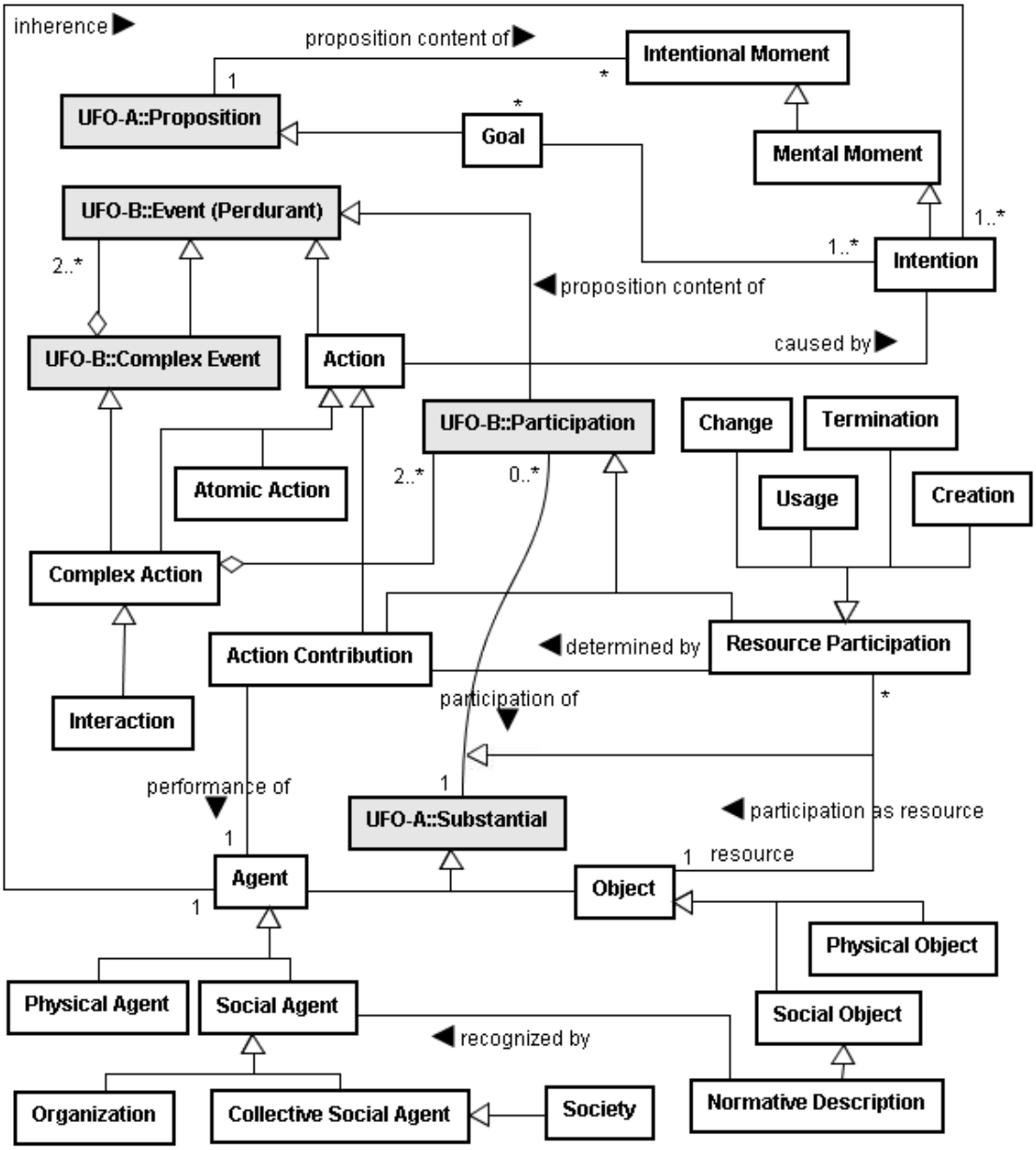


Figura 23 - Meta-modelo do subconjunto da UFO-C necessário para este trabalho (MARTINS, 2009).

Um Agente pode ser um Agente Físico (*Physical Agent*), como uma pessoa, ou um Agente Social (*Social Agent*), que pode ser uma Organização (*Organization*) ou uma Sociedade (*Society*), que é subtipo de um Coletivo de Agentes Sociais (*Collective Social Agent*).

De forma semelhante a um evento, uma Ação pode ser Ação Complexa (*Complex Action*) ou Ação Atômica (*Atomic Action*): uma Ação Complexa é composta por duas ou mais Participações, diferente de uma Ação Atômica.

Existe também a Participação desempenhada (relacionamento "*performance of*") pelo Agente (*Agent*), que possui a Intenção (relacionamento *inherence* com o conceito

Intention) de executar a ação. Esse subtipo de Participação também é um subtipo de Ação, chamado de Contribuição de Ação (*Action Contribution*).

Uma Contribuição de Ação determina (relacionamento "*determined by*") a Participação de um Recurso (*Resource Participation*), que é uma participação como recurso (relacionamento "*participation as resource*") de um *Substantial*, definido pelo seu subtipo Objeto (*Object*). Esses Objetos são *Substantials* inanimados, que podem ser Objetos Físicos (*Physical Object*) ou Objetos Sociais (*Social Object*). O subtipo de Objeto Social chamado Descrição Normativa (*Normative Description*) são reconhecidas por Agentes Sociais, que definem regras ou normas, por exemplo, uma lei ou um contrato.

Existem diferentes maneiras em que um Objeto pode ter uma Participação como Recurso em uma ação. A Criação (*Creation*) ocorre quando um objeto é criado depois da execução de uma Ação, enquanto que a Terminação (*Termination*) ocorre quando um objeto deixa de existir depois da execução de uma Ação. A Alteração (*Change*) acontece quando um Objeto é modificado durante uma Ação. Uso (*Usage*) é um tipo de Participação de Recurso diferente das anteriores, quando um Objeto participa em uma Ação, mas não é alterado, criado ou eliminado durante essa Ação.

2.3 Conclusão

Existem diversas abordagens para representação de ontologias que seguem estratégias diferentes umas das outras. A linguagem OntoUML, junto com a UFO-A e as extensões UFO-B e UFO-C, foram selecionadas para serem abordadas neste trabalho devido à fundamentação que proveem à modelagem conceitual.

3 Regras de Negócio

Na literatura, regra de negócio é definida de várias formas, sendo mais conhecidas as definições de (Halle, 2002) e (BRG, 2000), que partem de uma base semelhante do que é uma regra de negócio, porém, estruturam e conceituam regras de negócio e seus tipos de forma diferente.

Segundo BRG (2000), regra de negócio é uma declaração que define ou restringe algum aspecto de uma organização, sendo atômica, de forma que não pode ser dividida. Tem como objetivo afirmar a estrutura de um negócio ou controlar ou influenciar o comportamento deste. Dessa forma, é possível afirmar que regras de negócio podem ser usadas em conjunto aos modelos de processos, ajudando a defini-los. Segundo Halle (2002), as regras de negócio são as decisões que regem uma organização, compreendidas por políticas recomendadas e obrigatórias que governam a interação entre empregados, clientes, fornecedores e sistemas automatizados. Estas são as condições que governam os eventos do negócio de maneira que eles ocorram de forma aceitável para a organização.

Em modelagem de processos de negócio, regras de negócio podem estar atreladas à definição de atividades ou permeadas pelos processos, apoiando a execução destes, ao mesmo tempo em que os restringe, a partir das regras definidas no negócio da organização.

A definição encontrada em (BRG, 2000) é utilizada neste trabalho, sendo a base da linguagem de representação de regras de negócio SBVR (OMG, 2008).

Regras de negócio podem ser representadas formalmente ou informalmente. Para declarar as regras de negócio de uma organização formal e rigorosamente, regras de negócio, segundo (BRG, 2000), devem seguir os princípios de:

- Expressão explícita, seja graficamente ou em linguagem formal, baseada em lógica, como o uso de um modelo conceitual estrutural para representar regras estruturais;
- Representação coerente, com uma forma única de representação de regras de negócio de uma organização, para que possam fazer parte de sistemas de análise de processos;
- Extensão evolucionária, independente da forma como uma regra possa ser representada, deve permitir a evolução dessa representação, com a facilidade de acrescentar novos elementos para a construção de regras de negócio;
- Natureza declarativa, ou seja, uma regra de negócio não tem um formato de procedimento, pois descreve um estado sugerido, requerido ou proibido – podem ser condicionais, mas não descrevem os passos para realizar as transições de um estado para outro.

3.1 Formalizando regras de negócio

Uma regra de negócio é baseada em políticas internas e externas de uma organização. Usualmente, essas políticas são descritas de forma geral e informal, sendo necessária a tradução de um funcionário para a identificação de uma afirmação do que deve ser feito. Entretanto, essas afirmações podem ser claras ou ambíguas, e na maioria das vezes contêm mais de uma ideia (Halle, 1994).

A responsabilidade do analista especialista em regras de negócio é decompor essas políticas em regras de negócio atômicas, que tratem somente um termo, fato, derivação ou restrição do negócio, em seguida identificando qual das categorias é tratada pela regra. Termos, fatos e algumas regras podem ser representados em modelos conceituais estruturais, enquanto que o restante das regras e derivações possui uma conceitualização dinâmica e, para sua representação, modelos conceituais com construtos que permitam a representação da dinâmica de um domínio devem ser usados.

Uma parte do modelo conceitual de regras de negócio da BRG (2000) é apresentada na Figura 24 e descrita a seguir.

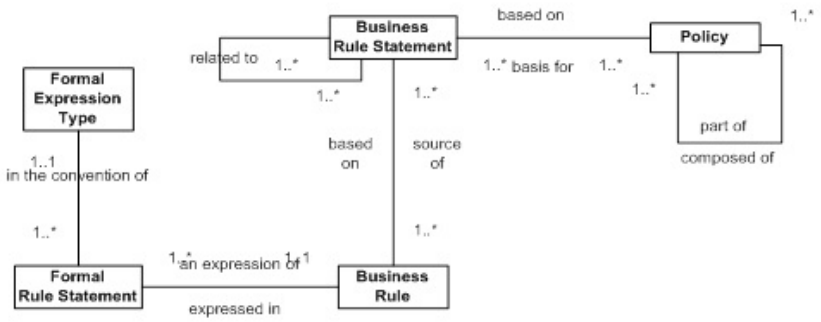


Figura 24 - Modelo conceitual da origem da regra de negócio (BRG, 2000).

Policy, ou a política de uma organização, é uma afirmação sem formalismos de regras internas a uma organização. Pode ser composta por uma ou mais políticas ou fazer parte de uma ou mais políticas, e é base para uma ou mais *Business Rule Statement*, ou uma afirmação de uma regra de negócio, que por sua vez pode ser baseada em uma ou mais políticas. As regras de negócio são afirmações declarativas de sua estrutura ou das restrições que uma organização aplica a si.

Uma afirmação de uma regra de negócio pode ser a fonte de uma ou mais regras de negócio atômicas. Semelhante a uma afirmação de regra de negócio, uma *Business Rule*, ou regra de negócio, é uma declaração que define ou restringe um aspecto de um negócio, porém não pode ser quebrada ou decomposta em regras mais detalhadas e, caso fosse reduzida, haveria perda de informações importantes ao negócio. Por estes motivos, uma regra de negócio é atômica. Regras de negócio se aplicam a uma organização ou domínio, independentemente da forma como são expressas.

Cada regra de negócio pode ser expressa em uma ou mais *Formal Rule Statement*, uma afirmação formal de uma regra, porém, deve ser uma expressão de somente uma regra de negócio atômica, em uma gramática formal específica. Por sua vez, cada afirmação formal de uma regra deve obedecer às convenções de uma *Formal Expression Type*, um tipo de expressão formal, que é uma gramática (linguagem de representação) para representar regras de negócio.

3.2 Tipos de regras de negócio

Segundo (BRG, 2000), regras de negócio são divididas entre os seguintes tipos:

- *Structural Assertion* ou Sentença Estrutural – uma conceitualização ou afirmação definida de um fato que expressa algum aspecto estrutural de uma organização. Engloba os termos e fatos que envolvem esses termos.
- *Action Assertion* ou Sentença de Ação – uma afirmação de uma regra ou condição que limita ou controla as ações dentro de uma organização.
- *Derivation* ou Derivação – uma afirmação de um conhecimento que é derivado de outro conhecimento dentro da organização.

  A seguir, a Figura 25 apresenta esses tipos de regras de negócio:

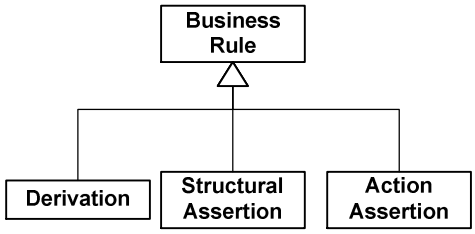


Figura 25 - Tipos de regras de negócio (BRG, 2000).

A seguir, cada tipo de regras de negócio é descrito.

3.2.1 Sentença estrutural

Sentenças estruturais são afirmações sobre algo que tem importância para o negócio, como um conceito importante ou um relacionamento entre conceitos de interesse para o negócio. Representam um aspecto estático e específico do negócio, apresentando coisas conhecidas e como elas estão estruturadas no universo de discurso. Podem ser representadas como modelos conceituais estruturais. Sentenças estruturais são divididas em dois tipos, apresentados na Figura 26: *terms* (termos) e *facts* (fatos).

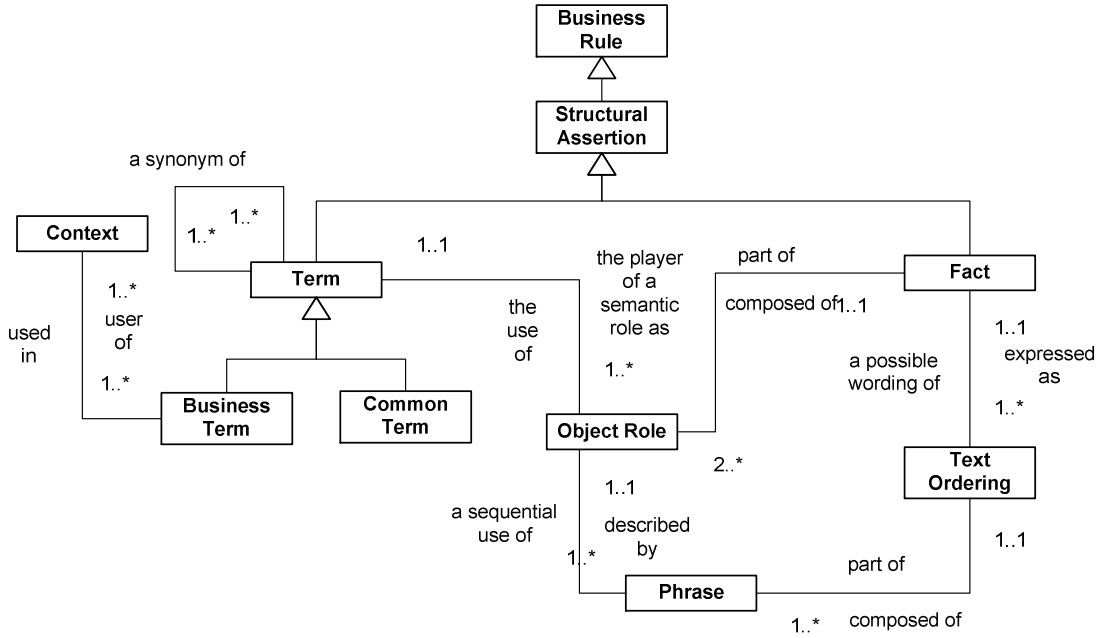


Figura 26 - Modelo conceitual de Sentença Estrutural (BRG, 2000).

Term, ou termo, é uma palavra ou expressão com significado para o negócio (semelhante a uma classe em um modelo conceitual estrutural). Esses termos são divididos em dois tipos diferentes: *business terms* (termos de negócio) e *common terms* (termos comuns). Cada termo de negócio deve ser usado em pelo menos um contexto, que pode usar um ou mais termos de negócio com sentido acrescido. Termos comuns são palavras usadas no dia-a-dia e com significado amplamente aceito. A diferença entre os dois é a necessidade explícita do termo de negócio estar ligado a um ou mais contextos (*Context*), e cada contexto ou domínio que as regras de negócio abordam poder ser usado na definição de um ou mais termos de negócio, enquanto que o termo comum não necessita uma definição explícita.

Fact, ou fato, define uma associação entre dois ou mais termos, expressando o relacionamento entre estes. Um fato envolve um ou mais termos e um termo pode estar em um ou mais fatos.

Outro elemento presente é o *object role*, ou papel de um objeto, que descreve o papel de um termo em um fato. Um termo pode ser o executor (semelhante a um sujeito em uma oração) de um papel semântico em um ou mais papéis de um objeto que deve ser usado em um fato. Um termo também pode ser usado como um ou mais papéis de objeto em um fato, semelhante a um predicado em uma oração.

Um fato possui várias formas de ser descrito e um fato entre dois termos pode possuir duas expressões para descrevê-lo, uma para cada direção do relacionamento.

Isso é representado pela entidade *text ordering*, ou ordenação do texto, explicitando que cada fato pode ser expresso por uma ou mais ordenações do texto, e uma ordenação de texto representa uma possível definição do fato. Cada ordenação de texto deve ser composta por uma ou mais *phrases*, ou frases, onde cada frase deve ser o uso sequencial de um papel de objeto que faz parte de um fato e o inverso se aplica. Além de apresentar a sequência dos papéis de objetos na ordenação do texto, a frase especifica o papel sintático (sujeito e predicado, por exemplo) dos papéis de objeto.

Apesar de não estar explicitamente descrito no modelo, deve existir exatamente uma frase que é parte de uma ordenação de texto para cada papel de objeto, que é parte de um fato expresso como uma ordenação de texto.

Termos podem ser classificados de forma diferente do que foi mostrado na Figura 26. Um termo pode ser classificado como um *Type*, ou tipo, que define categorias abstratas de coisas, ou como *Literal*, ou literal, que descreve instâncias dessas coisas ou valores de suas propriedades: um tipo define um conjunto de literais. Termos usados em regras de negócio são normalmente tipos. Porém, em certas situações, é necessária a referência a uma instância específica. Essas classificações são apresentadas na Figura 27.

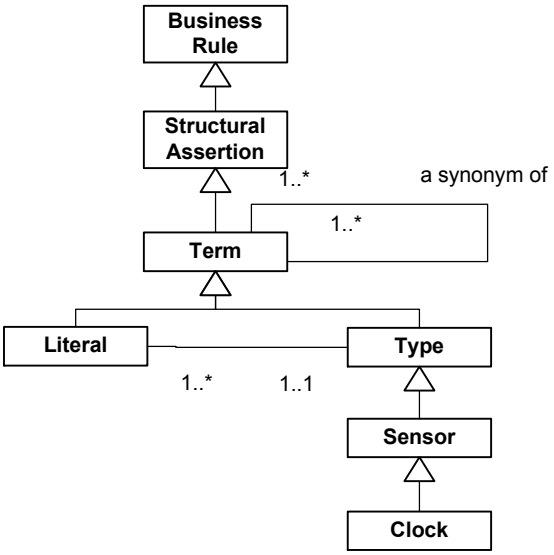


Figura 27 - Tipos de termo (BRG, 2000).

Uma categoria específica de tipos é *sensor*, cuja tradução para o português é direta. Sensor é algo que detecta e reporta periodicamente mudanças no mundo real. Uma especialização de sensor é *clock*, ou relógio, que reporta passagem do tempo.

Fatos podem ser classificados de duas formas. A primeira classificação divide fato em *base fact*, ou fato base (que é definido de forma simples), e *derived fact*, ou fato derivado (que é definido a partir de outras regras de negócio). A segunda classificação distingue as categorias *attribute*, ou atributo, *participation*, ou participação, e *generalization*, ou generalização. Esta classificação é apresentada na Figura 28.

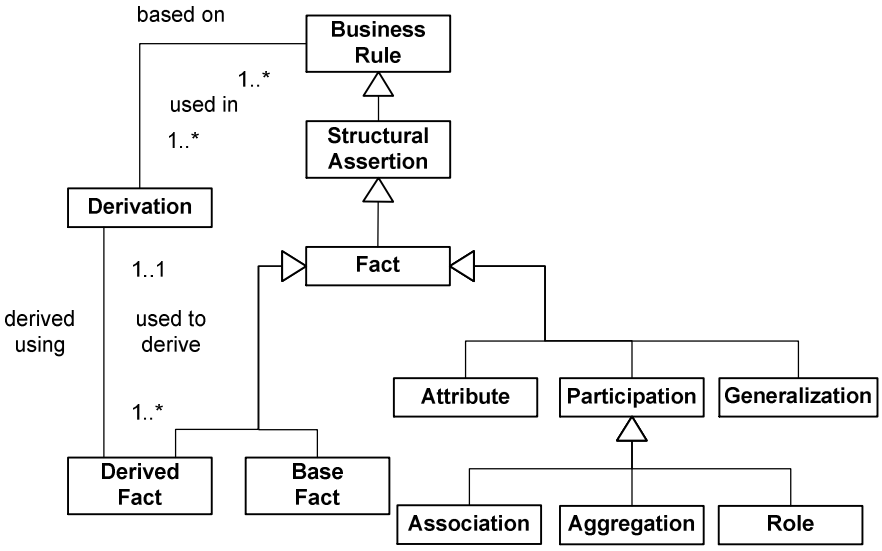


Figura 28 - Tipos de fato (BRG, 2000).

Um fato base é um registro de algo na realidade. Um fato derivado é uma assertiva que é construída a partir de outras assertivas. Pode ser um valor computado ou uma "visão". Cada fato derivado deve ser baseado em uma ou mais regras de negócio, e uma regra de negócio pode ser usada em qualquer quantidade de derivações.

Por outro lado, um atributo expressa um fato em que um termo descreve um aspecto (propriedade intrínseca) de outro termo. Uma generalização expressa a definição de um fato onde um termo (descrito como um tipo) caracteriza um subconjunto de ocorrências (como instâncias) de outro termo (também um tipo). Uma participação expressa um fato em que termos são associados entre si, que é um relacionamento semelhante ao que aparece em modelos de entidades e relacionamentos.

Uma participação, como apresentado na Figura 28, pode ser classificada em um dos três subtipos:

- Uma *aggregation*, ou agregação, é um relacionamento "parte-todo", que indica a composição de conceitos em suas partes: *aggregation* relaciona quaisquer termos entre si;

- Um *role*, ou papel, descreve como um termo age como um ator (que é outro termo) pelas interações com seu ambiente, diferente da definição da UFO, que agrega outras meta-propriedades ao construto *Role*, como a possibilidade de um conceito assumir mais de um papel ou nenhum em um dado momento, de forma distinta a uma *Phase*, conceito que não aparece na definição de regras de negócio da BRG (2000);
- Uma *association*, ou associação, é qualquer outro tipo de relacionamento.

3.2.2 Sentenças de ação

Sentença de ação é uma afirmação a respeito de algum aspecto dinâmico do negócio. Enquanto as sentenças estruturais focam na perspectiva estática de um domínio ou negócio, as sentenças de ação exploram a dinâmica do negócio, descrevendo as restrições que regem essa dinâmica. Especificam restrições nos resultados das ações, que são descritas de forma não processual, relacionadas com outras regras de negócio atômicas. Enquanto sentenças estruturais descrevem possibilidades dentro de um domínio (estados possíveis), sentenças de ação impõem restrições ao comportamento dinâmico do negócio, do tipo “deve”, “deveria”, “não deve” ou “não deveria”. O modelo conceitual das sentenças de ação é apresentado na Figura 29.

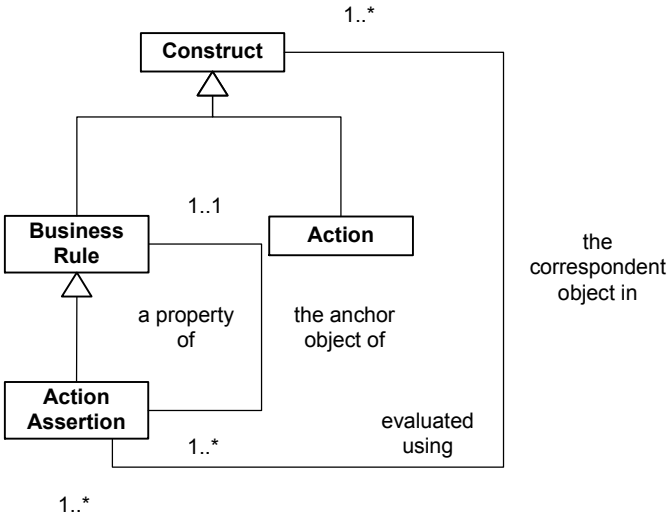


Figura 29 - Modelo conceitual da sentença de ação (BRG, 2000).

Uma sentença de ação é composta por um *anchor object*, ou objeto âncora, que será o objeto alvo da regra e pode ser qualquer tipo de regra de negócio (em uma frase “Se ... então ...”, o objeto âncora viria depois do “Se”), e um ou mais objetos

correspondentes (em uma frase “Se ... então ...”, os objetos correspondentes viriam depois do “então”). Uma sentença de ação deve ser a propriedade de alguma outra regra de negócio e uma regra de negócio pode ser o objeto âncora de uma ou mais sentenças de ação. O objeto âncora é normalmente uma sentença estrutural, porém pode ser uma sentença de ação. O objeto correspondente pode ser uma regra de negócio ou uma *action*, ou ação, representados por um *construct*, ou construto. Uma sentença de ação é avaliada usando um ou mais construtos, descrevendo as condições da restrição. Por exemplo, na regra de negócio do tipo de sentença de ação “Um carro deve ter um número de registro, quando for vendido”, “carro” (que é um termo) é o objeto âncora, e a regra de negócio que expressa a obrigação de um carro possuir um número de registro (que é um fato) é o objeto correspondente.

Sentenças de ação podem ser classificadas de três formas, que são ortogonais. Inicialmente, podem ser classificadas em classes, como apresentado na Figura 30, identificando se são *condition* (condição), *integrity constraint* (restrição de integridade) ou *authorization* (autorização). Outra forma de classificação é por tipos de sentenças de ação, como apresentado na Figura 31, identificando se são *enabler* (habilitador), *timer* (temporizador) ou *executive* (executor). E, finalmente, pode ser classificado em *Action Controlling Assertion*, ou sentenças de controle de ação, e *Action Influencing Assertion*, ou sentenças de influência de ação, como apresentado na Figura 32.

3.2.2.1 Classes de Sentenças de Ação

Sentenças de ação podem ser classificadas em várias classes (BRG, 2000), apresentadas na Figura 30.

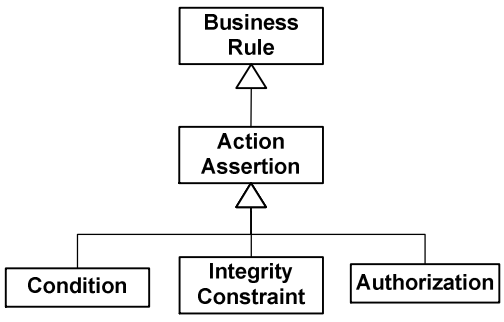


Figura 30 - Classes de Sentenças de Ação (BRG, 2000).

C*ondition*, ou condição, é uma classe de sentença de ação onde **se** algo é verdadeiro, **então** uma regra de negócio se aplica: pode ser entendido com um teste, onde, se algo é verdadeiro, então é a base para forçar ou testar outra regra de negócio.

Integrity constraint, ou restrição de integridade, é uma classe de sentença que deve ser sempre verdadeira: proíbe ações que contrariam a regra definida na sentença. Diferente de uma condição, que define um teste, a declaração de uma regra de integridade não pode ser violada.

Authorization, ou autorização, é uma classe de sentença que define um privilégio a respeito de um construto. É uma sentença definida por um predicado: (somente) *x* pode *y*, onde, usualmente, *x* é um usuário e *y* é uma ação. Autorizações são definidas somente para tipos capazes de realizar uma atividade, como uma pessoa, um departamento, etc.

3.2.2.2 Tipos de Sentenças de Ação

Sentenças de ação podem ser classificadas em vários tipos (BRG, 2000), apresentados na Figura 31.

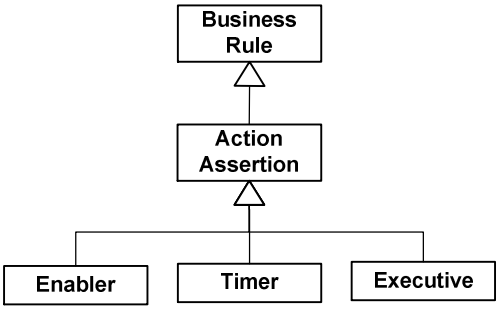


Figura 31 - Tipos de Sentenças de Ação (BRG, 2000).

O tipo *Enabler*, ou habilitador, permite ou conduz à existência do objeto correspondente na sentença, caso seja verdadeiro. A sentença é verdadeira caso o objeto âncora seja verdadeiro. Esse tipo de sentença de ação possui várias interpretações dependendo da natureza do objeto correspondente:

- Caso o objeto correspondente seja uma sentença estrutural, essa sentença de ação permite a criação de uma nova instância da sentença estrutural.
- Caso o objeto correspondente seja uma sentença de ação, essa sentença de ação permite que a outra sentença de ação seja acionada.

- Caso o objeto correspondente seja uma ação, essa sentença de ação permite a execução da ação.

O tipo *Timer*, ou temporizador, permite que um alarme definido pela sentença seja acionado pelo tempo ou que uma contagem regressiva definida pela sentença seja terminada. Caso o objeto âncora seja uma sentença de ação, o temporizador irá iniciar a sentença. Caso seja uma sentença estrutural, o temporizador irá ajustar ou testar o valor definido.

O tipo *Executive*, ou executor, causa a execução de uma ou mais ações. Como esses tipos de sentenças de ação podem ser usados em conjunto para formar uma sentença de ação, o tipo executor normalmente é usado com o objetivo de indicar o que deve ser realizado, caso as condições inicias tenham sido atendidas.

3.2.2.3 Sentenças de Ação de Controle e de Influência

Os tipos e classes de sentenças de ação se destinam a definir algo que deve ou não deve existir ou acontecer. Sentenças de ação criadas dessa forma são exemplos de *Action Controlling Assertion*, ou sentenças de ação de controle. Porém, sentenças de ação que definem coisas que deveriam ou não deveriam existir ou acontecer são *Action Influencing Assertion*, ou sentenças de ação de influência: são sentenças que as organizações tendem a seguir, mas que não serão, necessariamente, incluídas em sistemas. Indicam uma orientação para suporte de decisão, por exemplo, “não mais que 10% de aluguéis podem ter desconto por mês”, que pode ocorrer, apesar da orientação contrária. Sentenças de ação de influência podem ser úteis caso a organização deseje ser avisada pelos sistemas se alguma dessas sentenças for violada, ou caso queiram que as atividades humanas sejam guiadas por essas regras, quer tenham suporte automatizado ou não.

A Figura 32 apresenta as sentenças de ação de controle e de influência.

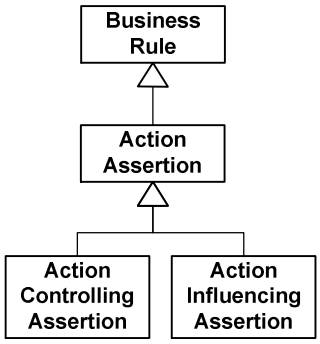


Figura 32 - Sentenças de Ação de Controle e de Influência (BRG, 2000).

Sentenças de ação de influência necessitam de um suporte da organização para ajudar nas informações e tomadas de decisão, caso uma regra desse tipo seja violada.

3.2.3 Derivações

Um fato base é um fato que existe no mundo sendo modelado e é guardado em um sistema. Um fato derivado é criado por inferência ou cálculo matemático de termos, fatos, outras derivações ou sentenças de ação.

A Figura 33 mostra o modelo conceitual das *derivations*, ou derivações. Como mostrado no modelo das sentenças estruturais, um fato pode ser um fato derivado ou um fato base. Uma derivação deve ser baseada em um ou mais fatos derivados, enquanto que um fato derivado deve ser derivado de somente uma derivação.

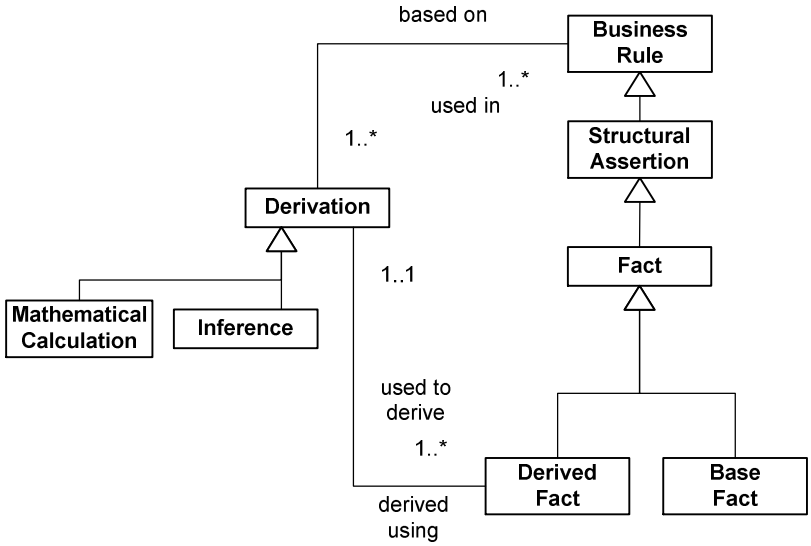


Figura 33 - Modelo conceitual de derivações (BRG, 2000).

Uma derivação é um tipo de regra de negócio e pode ser um cálculo matemático, que é definido através de um algoritmo matemático, ou uma inferência, que é produzida através de lógica indutiva (de particulares) ou dedutiva (de princípios gerais).

3.3 Definição de regras dividida em níveis de abstração

Nos trabalhos (WAGNER *et al*., 2006) e (WAGNER *et al*., 2004), cinco tipos de regras são listados, divididos em três níveis de abstração. Esses níveis de abstração são baseados na Arquitetura Direcionada a Modelos (*Model-Driven Architecture*, ou MDA, definido em (OMG, 2003)), que diferencia vários níveis de abstração computacional. A Figura 34 apresenta a hierarquia das regras, dividida nos níveis de abstração.

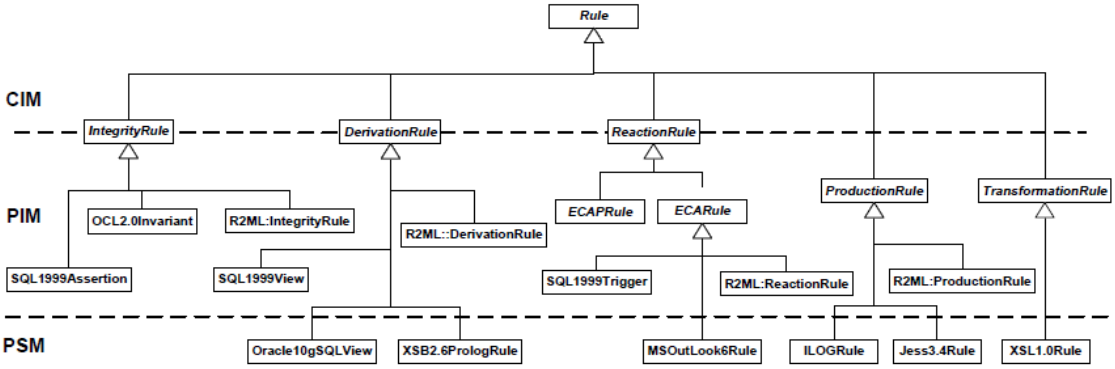


Figura 34 - Tipos de regras em diferentes níveis de abstração (WAGNER *et al*., 2006).

Cada nível de abstração apresenta uma definição para os tipos de regras que o compõem:

- no Nível do Domínio de Negócio Computacionalmente Independente (*Computational-Independent Business Domain Level*, CIM), regras são afirmações que expressam (partes de) uma política do domínio/negócio, de forma declarativa, normalmente usando linguagem natural ou visual, definindo termos do domínio ou restringindo suas operações;
- no Nível de Design Operacional Independente de Plataforma (*Plataform-Independent Operational Design Level*, PIM), regras são afirmações formais, expressas em algum formalismo ou paradigma computacional, que pode ser diretamente mapeado para afirmações executáveis de um sistema;

- no Nível de Implementação Específico de Plataforma *(Plataform-Specific Implementational Level*, PSM) (WAGNER *et al*., 2006), regras são afirmações em uma linguagem de um ambiente de execução específico.

Além da definição de Wagner et al., a OMG define CIM como uma visão de um ponto de vista independente de computação, sem mostrar os detalhes da estrutura computacional. A OMG assume que o usuário primário de CIM, o profissional que trabalha dentro do domínio, não conhece modelos e artefatos usados para compreender a funcionalidade para os quais os requisitos foram enunciados no CIM. Então, esse nível tem um papel importante em diminuir a lacuna entre os especialistas no domínio e seus requisitos e os especialistas na modelagem e construção dos artefatos que juntos satisfazem os requisitos do domínio.

Levando esses elementos em consideração, os tipos de regras que devem ser tratados na modelagem conceitual são os do nível Independente de Computação, a saber: Regra de Integridade (*IntegrityRule*), Regra de Derivação (*DerivationRule*) e Regra de Reação (*Reaction Rule*). As Regras de Produção (*Production Rule*) e Regras de Transformação (*TransformationRule*) fazem parte do nível dependente de computação (PIM) e não serão explorados neste trabalho, pois saem do escopo do mesmo.

As definições seguintes foram retiradas de (WAGNER *et al*., 2004) e (GIURCA *et al*., 2006):

- Regra de Integridade (IntegrityRule): consiste em uma afirmação de uma restrição, que é uma sentença em uma linguagem lógica, que expressa afirmações que devem ser verdadeiras nos estados e transições de estados do sistema discreto dinâmico definido;
- Regra de Derivação (DerivationRule): regras que possuem condições e conclusões (expressões sem condições ou sem conclusões não devem ser chamados de “regras”; deveriam ser chamados de “fatos” ou “restrições de negação”) e que explicam como um elemento do modelo pode ser derivado;
- Regra de Reação (ReactionRule): também conhecida como regra Evento-Condição-Ação (regra ECA, *Event-Condition-Action*), são afirmações que, no caso de ocorrência de um evento disparador ou no caso de um conjunto de condições ser satisfeito, especificam a execução de uma ou mais ações. Opcionalmente, depois da execução de uma ação, pós-condições podem ser verdadeiras.

- Regra de Transformação (TransformationRule): definem regras de transformações de conteúdo.

Com a definição de regras de negócio descrita, as linguagens de representação são apresentadas na seção 3.4.

3.4 Linguagens de representação

Com o objetivo de representar regras de negócio, algumas linguagens foram desenvolvidas. Entre elas, encontram-se a ORM, *Object Rule Modeling* (Halpin, 1995), com uma visão orientada a objeto e construção visual semelhante a um diagrama entidade-relacionamento; a SBVR, *Semantics of Business Vocabulary and Business Rules,* (OMG, 2008); a RuleML, *Rule Markup Language*, (Hirtle *et Al*., 2006); e URML, *UML-Based Rule Modeling Language*, (LUKICHEV e WAGNER, 2007). Outras linguagens podem ser utilizadas para representar regras de negócio, como a OCL, *Object Constraint Language*, (OMG, 2006a), apesar de seu foco ser regras, não especificamente regras de negócio.

Neste trabalho, duas características são consideradas fundamentais para as linguagens de representação de regras (ser baseada na UML e ter uma representação visual). Assim, as linguagens de representação de regras exploradas neste trabalho são a URML e OCL. A URML foi escolhida por ser uma representação visual e uma extensão da UML, características que as outras linguagens não possuem. Foram considerados também a R2ML, que é base da URML. Entretanto, a URML não possui construtos para a representação de regras de integridade. Com isso, a OCL, por ser uma extensão da UML, foi escolhida para preencher essa lacuna, apesar de não ser visual.

3.5 URML

UML-Based Rule Modeling Language (URML, Linguagem de Modelagem de Regras baseada em UML) é uma extensão da UML e provê uma representação visual de regras. Também é possível usar OCL para representar regras de integridade ou derivação, no entanto sua representação é puramente textual. A representação de regras de forma visual facilita o entendimento de um modelo por um especialista do

domínio/negócio modelado, permitindo uma melhor comunicação entre analistas e especialistas.

O meta-modelo da URML está disponível na web em (REWERSE, 2011). Alguns trabalhos disponíveis na literatura explicam esse meta-modelo (LUKICHEV e WAGNER, 2007), (GIURCA *et al*., 2006) e (LUKICHEV e JARRAR, 2009) e, como a URML foi criada baseada na R2ML, alguns trabalhos foram usados para melhor entender elementos do meta-modelo (como *Atom* ou *Term*): (WAGNER *et al*., 2006) e (WAGNER *et al*., 2004).

O meta-modelo dos tipos de regras da URML é apresentado na Figura 35. Como uma extensão da UML (OMG, 2009), uma Regra (*Rule*) é uma especialização de Elemento Nomeado (*NamedElement*), algo no modelo que possui um nome, e *Namespace*, que é identificado por um nome e contém um conjunto Elementos Nomeados: uma Regra é composta por outros elementos, de forma semelhante a um *Namespace*.

Os elementos que compõem uma regra são uma ou várias Condições (*Condition*); um ou vários Átomos de Igualdade de Objetos (*ObjectEqualityAtom*), que são as Condições de Junção (*joinConditions*) quando mais de uma Condição está envolvida na Regra; e Variáveis de Objeto (*ObjectVariable*), que são as variáveis tratadas na Regra (o relacionamento derivado "*ruleVariables*").

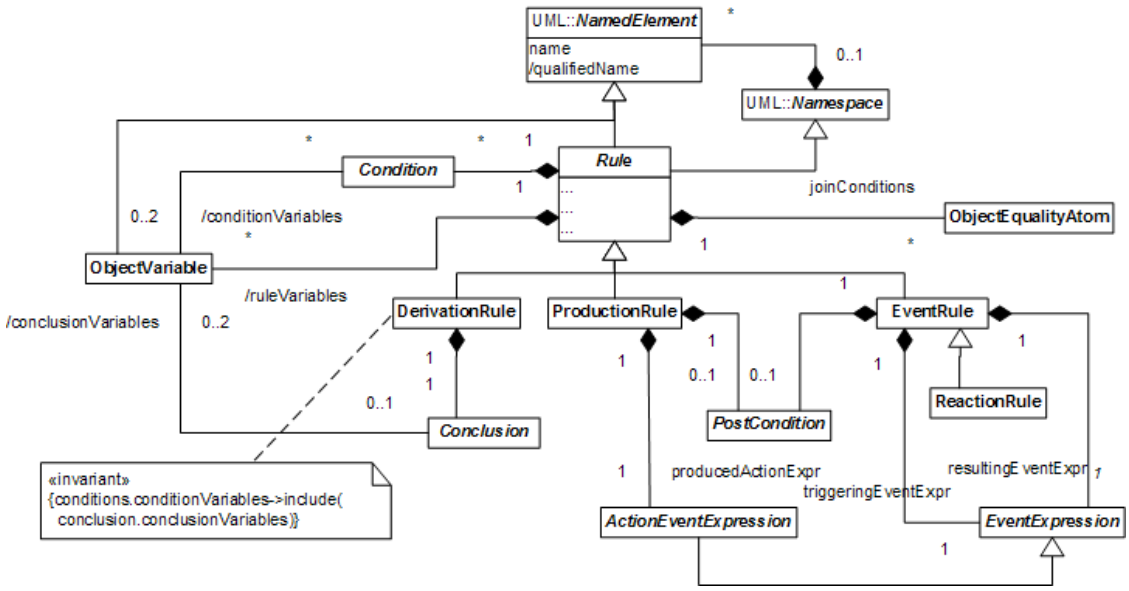


Figura 35 - Meta-modelo URML para os tipos de regras (REWERSE, 2011).

Um Átomo de Igualdade de Objetos (*ObjectEqualityAtom*) é um tipo de Átomo (*Atom*), composto de um ou mais Termos (*Terms*). Átomo é uma Fórmula Atômica, o

elemento básico que constitui uma regra, representando a formalização de uma regra indivisível. Os outros tipos de Átomos da URML serão abordados nos meta-modelos de Condição (Figura 36) e Conclusão (Figura 40).

Uma Variável de Objeto (*ObjectVariable*) é uma especialização de Termo de Objeto (*ObjectTerm*), conforme apresentado na Figura 39, e representa uma variável que será manipulada na Regra (relacionamento derivado "/*ruleVariables*") ou na Condição (relacionamento derivado "/*conditionVariables*"). São elementos testados ou alterados em uma Regra. Um Termo de Objeto é a representação de um objeto do domínio.

Uma Condição (*Condition*) corresponde a fórmulas lógicas sem quantificadores, que representam o elemento que é pré-requisito para a aplicação de uma regra. A Figura 36 apresenta o meta-modelo da Condição.

Condição é uma expressão booleana, que pode ser negativa (onde o teste da expressão é feito quando esta não é verdade), composta por um Filtro (*Filter*), que pode ser escrito em OCL (*OCLFilter*) ou um *OpaqueFilter*, para ser usado em uma implementação específica de um vendedor.

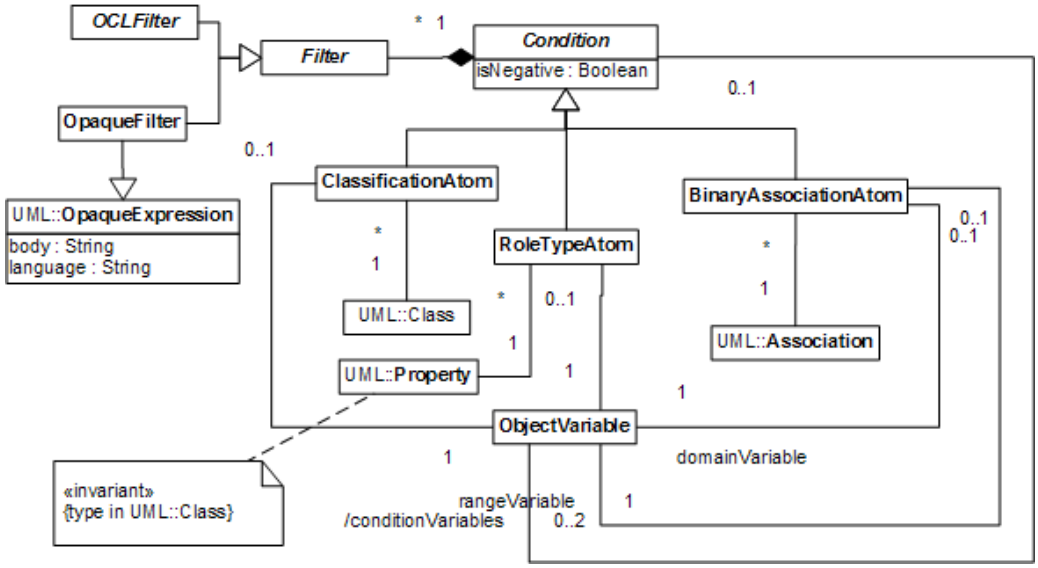


Figura 36 - Meta-modelo da URML para o elemento *Condition* (REWERSE, 2011).

Filtro OCL (*OCLFilter*) é baseado em uma parte da OCL que representa expressões lógicas. É dividido em subtipos: Filtros Conjuntivos (*ConjunctiveFilter*), com a mesma definição que um E (*and*) lógico, contendo dois ou mais Filtros OCL; Filtros Disjuntivos (*Disjunctive Filter*), com a mesma definição que um OU (*or*) lógico, contendo dois ou mais Filtros OCL; Átomo de Igualdade de Objeto

(*ObjectEqualityAtom*), em que se compara se um *Classifier* (no caso um Termo de Objeto - *ObjetcTerm*) está atendendo uma condição, descrita pelo Átomo relacionado a um Objeto; Átomo de Desigualdade de Objeto (*ObjectInequalityAtom*), em que se compara se um *Classifier* (no caso um termo de Objeto - *ObjetcTerm*) está atendendo à negação de uma condição, descrita pelo Átomo relacionado a um Objeto; e Átomo de Predicado de Tipo de Dado (*DatatypePredicateAtom*), se refere a um Predicado de Tipo de Dado (*DataypePredicate*), composto por Termos de dados (*DataTerm*), testando esses Termos com comparações de seus valores, como, por exemplo, “maior que”, ou “menor ou igual a”.

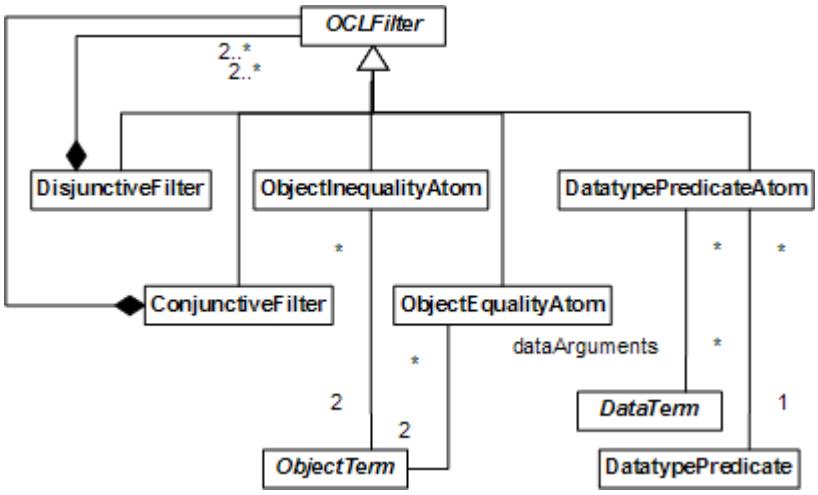


Figura 37 - Meta-modelo da URML para o elemento *OCL Filter* (REWERSE, 2011).

A Figura 38 apresenta o meta-modelo de Termo de Dado (*DataTerm*). Um Termo de Dado é usado para representar tipo de dados primitivos ou valores de dados. Existem três tipos de Termos de Dado: Variável de Dado (*DataVariable*), que representa uma variável relacionada a um dado; Literal de Dado (*DataLiteral*), que representa um valor de dado; e Termo de Função de Dado (*DataFunctionTerm*), que pode ser dos tipos:

- Termo de Função de Atributo (AttributeFunctionTerm) representando expressões aritméticas, relacionadas com Atributo Funcional (*FunctionalAttribute*).
- Termo de Função de Tipo de Dado (*DatatypeFunctionTerm*) representando a Função de Tipo de Dado (*FunctionDatatype*), que retorna o valor de um atributo de um objeto.

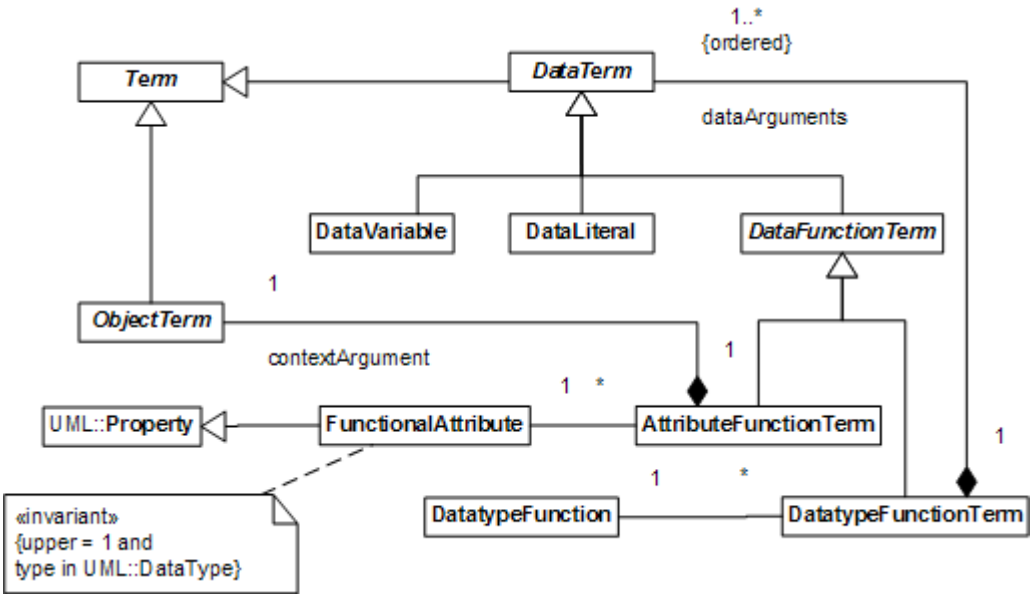


Figura 38 - Definição de *DataTerm* (REWERSE, 2011).

Um Termo de Objeto representa variáveis que podem ser instanciadas por valores ou constantes de objetos (*ObjectVariable*), apresentada na Figura 39. Termo de Função de Propriedade de Referência (*ReferencePropertyFunctionTerm*) representa o elemento que está associado a uma das pontas de uma Associação em UML (um papel), relacionado à Propriedade de Referência Funcional (*FunctionalReferenceProperty*).

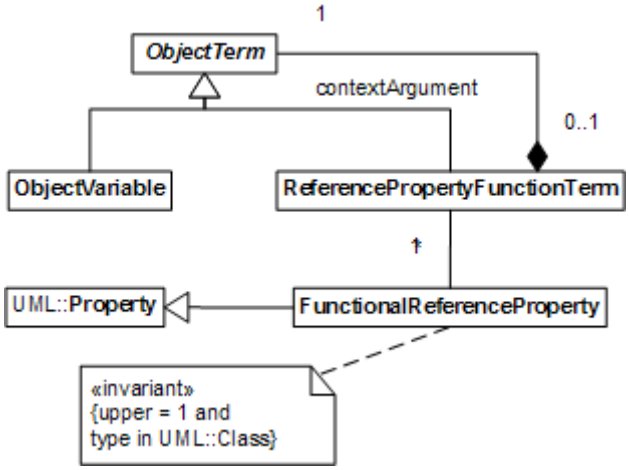


Figura 39 - Definição de *ObjectTerm* (REWERSE, 2011).

3.5.1 Tipo de Regras na URML

Na Figura 35, os tipos de regras são apresentados: Regras de Derivação (*DerivationRule*), Regras de Reação (*ReactionRule*) e Regras de Produção (*ProductionRule*).

Uma Regra de Derivação é composta por uma Condição que, caso seja satisfeita, apresenta um novo conceito ao domínio (como esse conceito é inferido, classificado como tal). A indicação desse novo conceito é feita a partir da Conclusão (*Conclusion*), que compõe uma Derivação. A definição de Regra de Derivação da URML é semelhante a da BRG (2000), apresentada na seção 3.2.3. BRG identifica que um conhecimento é derivado de outro conhecimento já existente. A URML diferencia esses conhecimentos, indicando que uma condição deve ser satisfeita para que uma conclusão seja feita, identificando novos elementos do domínio.

A Figura 40 apresenta o meta-modelo da Conclusão da URML. Uma Conclusão pode conter ou não Variáveis de Objetos, tratadas como Variáveis de Conclusão (relacionamento derivado "/*conclusionVariables*").

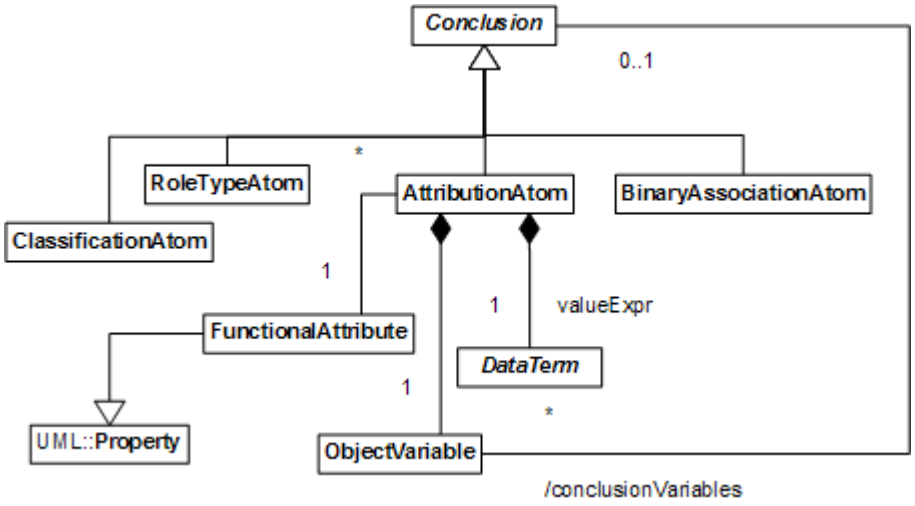


Figura 40 - Meta-modelo da URML para o elemento *Conclusion* (REWERSE, 2011).

Uma Regra de Reação ocorre quando a Condição é o pré-requisito para o acontecimento de um evento, em reação a essa Condição. Também é conhecida como regras Evento-Condição-Ação (regra ECA), onde um evento leva a satisfação de uma condição, disparando uma ação. Regra de Reação é um tipo de Regra de Evento (*EventRule*), que é composta pelos seguintes elementos:

- Evento de disparo (triggeringEvent) é uma Expressão de Evento (*EventExpression*), que pode ser atômico ou composto, iniciando a regra. A definição de Expressão de Evento é semelhante a da UFO: algo que ocorre em um determinado intervalo de tempo.
- Evento Resultante (*resultingEvent*) é uma Expressão de Evento, mas que ocorre depois que o Evento de Disparo acontece.

- Uma Pós-Condição (*Post-Condition*) opcional especifica uma mudança de estado, de forma semelhante a uma Condição.

Uma Regra de Produção tem uma estrutura e definição semelhante a uma Regra de Reação, mas ao invés de disparar uma Expressão de Evento, dispara uma Expressão de Evento Ação (*ActionEventExpression*). A Expressão de Evento Ação se diferencia de uma Expressão de Evento por executar uma alteração no domínio representado no modelo, podendo ser uma ação de invocação (que invoca uma operação em UML - como uma atividade de um diagrama de Atividades), uma ação de atribuição de valor a uma variável, ou uma ação que atualiza o estado do domínio, modificando as propriedades de seus conceitos. De forma semelhante a uma Regra de Reação, a Regra de Produção é composta por uma Pós-Condição.

3.5.2 Sintaxe da URML

Em URML, as regras são representadas por círculos. Condição é representada por uma Seta de Condição (*Condition Arrow*), ligada a um elemento do modelo, e um *Classifier*, como uma classe ou relacionamento. O Filtro seleciona instâncias da extensão do *Classifier* presente na condição. A Condição pode ser dividida nos seguintes tipos:

- Condição de Classificação (*Classification condition*) é representada como uma seta saindo de uma Classe para o círculo da Regra. Na ponta próxima da Classe, a seta é anotada com uma variável de Regra referente à Classe. A Figura 41 ilustra a sua representação gráfica.

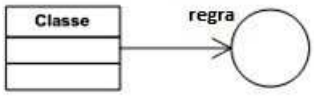


Figura 41 - Representação gráfica de Condição de Classificação.

- Condição de Regra (*Rule condition*) é representada como uma seta saindo de uma das pontas de uma Associação para o círculo da Regra, anotada por uma variável de Regra referente à Classe que está na ponta da Associação. A Figura 42 ilustra a sua representação gráfica.

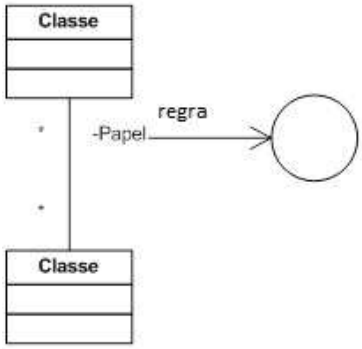


Figura 42 - Representação gráfica de Condição de Regra.

- Condição de Associação (*Association condition*) é representada como uma seta de uma Associação para o círculo da Regra, anotada com duas variáveis de Regras, uma para cada Classe nas pontas da Associação. A Figura 43 apresenta ilustra a sua representação gráfica.

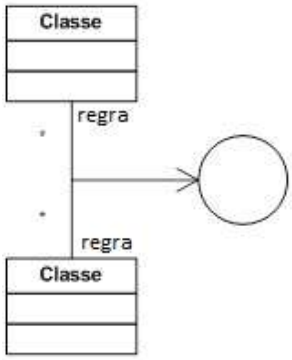


Figura 43 - Representação gráfica de Condição de Associação.

Uma Conclusão é representada por Setas de Conclusão (*Conclusion Arrow*), que de forma semelhante a uma Seta de Condição, está ligada a um elemento do modelo, indicando que o estado que o predicado descrito na conclusão é aplicado para todas as instâncias que satisfazem as condições das regras. Pode ser classificada como um dos seguintes tipos:

- Conclusão de Classificação (*Classification Atom*), que deriva uma nova Classe no domínio, composta por um Átomo de Classificação. Esse tipo de Conclusão é representado como uma seta que sai do círculo da Regra para a Classe derivada, que pode ser anotada como uma variável da Classe derivada. A Figura 44 ilustra a sua representação gráfica.

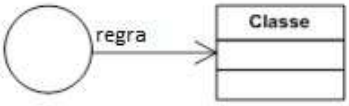


Figura 44 - Representação gráfica de Conclusão de Classificação.

- Conclusão de Papel (*Role Type Atom*), que deriva um novo Papel no domínio, representada como uma seta partindo do círculo de Regra para a ponta da Associação que foi derivada, podendo conter anotações com uma variável da Classe relacionada com a ponta da Associação. A Figura 45 ilustra a sua representação gráfica.

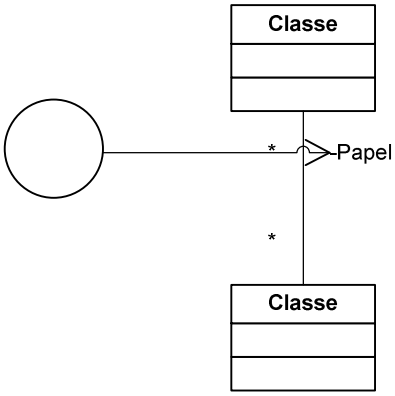


Figura 45 - Representação gráfica de Conclusão de Papel.

- Conclusão de Atribuição (*Attribution Atom*), que deriva o valor de um atributo, representada como uma seta saindo do círculo de Regra para a Classe que contém o Atributo derivado. A seta é anotada com a variável da classe e com o valor da propriedade a ser especificado. A Figura 46 ilustra a sua representação gráfica.

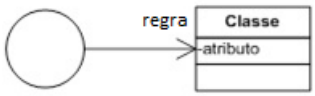


Figura 46 - Representação gráfica de Conclusão de Atribuição.

- Conclusão de Associação Binária (*Binary Association Atom*), representada como uma seta saindo de um círculo de Regra para a Associação derivada, anotada com duas variáveis, para cada uma das Classes nas pontas da Associação. A Figura 47 ilustra a sua representação gráfica.

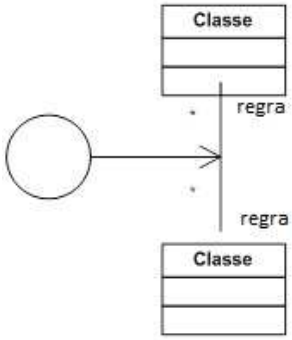


Figura 47 - Representação gráfica de Conclusão de Associação.

As Expressões de Evento (as definições das regras, com suas variáveis) são interligadas a representação gráfica da regra a partir de uma Seta de Evento (*Event Arrow*), que pode estar ligada a uma Regra ou a uma Classe.

3.6 OCL

OCL (*Object Constraint Language*) (OMG, 2006) é uma linguagem desenvolvida pela IBM para representação de regras baseadas em diagramas UML. Seu objetivo é tratar de aspectos de uma especificação que os diagramas não conseguem representar. A linguagem é textual e formal, não permitindo ambiguidades. Os exemplos de Regras OCL apresentados neste trabalho foram retirados de (OMG, 2006). As palavras reservadas estão em negrito.

Uma expressão em OCL é composta de um contexto (*context*), o elemento do modelo ao qual a regra se aplica. Ela contém tipos de restrição que podem ser Invariante (*invariant*), Pré-condição (*precondition*) e Pós-condição (*postcondition*).

O tipo de restrição Invariante (escrita com o elemento “inv”) aplica a regra para qualquer instância do contexto a qualquer momento de existência do domínio. Uma restrição Invariante é do tipo booleano. No exemplo abaixo, no contexto de uma Empresa, o número de empregados (representado por “self.numeroDeEmpregados”, onde *self* representa o contexto, no caso “Empresa”) não pode ser menor ou igual a 50, o que é caracterizado pela expressão *inv*.

context Empresa **inv**:
self.numeroDeEmpregados > 50

Os tipos pré e pós-condições determinam o estado do sistema para que uma operação ocorra e o estado do sistema após a execução da operação. Na especificação

abaixo, que apresenta a estrutura de uma regra OCL contendo os elementos de pré e pós-condição, são apresentados os elementos que declaram a regra: o nome do tipo (*Typename*), o nome da operação (*operationName*), que tem parâmetros (*param1*, *param2*, *paramN*) definidos por tipos (*Type1*, *Type2*, *TypeN*), podendo retornar um tipo de dado específico (*ReturnType*). Na pré-condição, apresentada depois do elemento "pre:", um teste é feito com um parâmetro, levando a uma pós-condição, apresentada pelo elemento "post", onde algum valor é atribuído a um resultado (*result*).

context Typename::operationName(param1 : Type1, ...): ReturnType
pre : param1 > ...
post: result = ...

A OCL possui outras expressões que também podem ser usadas para acrescentar semântica às regras. A operação *Body* é usada para indicar o resultado de uma operação de consulta. Em OCL, o elemento "--" indica um comentário.

context Typename::operationName(param1 : Type1, ...): ReturnType
body: -- Alguma expressão

Os valores iniciais (escritos como elemento "init") e derivados (escritos como elemento "derive") são usados para indicar um valor inicial e final para um atributo ou ponta de uma Associação, e podem ser indicados em uma regra OCL.

context Typename::attributeName: Type
init: -- Alguma expressão representando o valor inicial

context Typename::assocRoleName: Type
derive: -- Alguma expressão representando a regra de Derivação

O uso da OCL neste trabalho será feito para a escrita de fórmulas, aproveitando da característica da OCL de permitir que os elementos de um modelo em UML sejam referenciados e que podem ser comparados, entre si ou com um literal.

A comparação dos elementos é feita a partir do uso de operadores infixos, sendo usados entre os elementos da regra: "+" (soma de elementos do modelo), "-" (subtração de elementos do modelo), "*" (multiplicação de elementos do modelo), "/" (divisão de elementos do modelo), "<" (comparação "menor que" entre elementos do modelo), ">" (comparação "maior que" entre elementos do modelo), "<>" (comparação "diferente de" entre elementos do modelo), "<=" (comparação "menor ou igual a" entre elementos

do modelo), “>=” (comparação “maior ou igual a” entre elementos do modelo), e “=” (comparação “igual a” entre elementos do modelo). Esses elementos possuem a seguinte precedência entre si:

- “*”, “/”
- “+”, “-”
- “<“, “>”, “<=”, “>=”
- “=”, “<>”

A navegabilidade presente em OCL ocorre devido ao uso do ponto (“.”) para separar os elementos do modelo e referenciar partes dele. Pode-se usar essa estrutura para, a partir de um contexto (que é um elemento do modelo, mas que pode ser referenciado por “self” ou o nome do elemento), referenciar um atributo do elemento ou referenciar um atributo de um elemento que possui relacionamento com o contexto, como no exemplo (lembrando que o elemento “--” indica um comentário):

```
context Elemento inv:
self.Atributo > -- Valor
self.relacionamento.Elemento.Atributo = -- Valor
```

3.7 Conclusão

A modelagem de regras de negócio explicita como uma organização deve se comportar, restringindo a estrutura dinâmica da mesma. As regras de negócio fazem parte de uma organização, estejam elas descritas informalmente ou em representações específicas e formais. A URML e a OCL proveem um conjunto de construtos e uma sintaxe subjacente capazes de documentar regras de negócio, permitindo a descrição formal dos tipos diferentes de regras.

4 R-OntoUML: meta-modelo e perfil UML para representação de regras de negócio baseada em ontologia de fundamentação

No trabalho apresentado nesta dissertação, usa-se a definição de regras proposta em (WAGNER *et al*., 2006) e (WAGNER *et al*., 2004), que lista cinco tipos de regras, divididos em três níveis de abstração.

O objetivo deste trabalho é a representação conceitual das regras, abordando o nível que a OMG e (WAGNER *et al*., 2006) definem como CIM (*Computational Independent Model*).

Essas regras têm uma relação direta com alguns tipos de regras específicos definidos pela (BRG, 2000). A Regra de Integridade tem a mesma definição que a *Integrity Constraint* (Restrição de Integridade), a Regra de Reação tem a mesma definição que a *Condition* (Condição) e a Regra de Derivação tem a mesma definição que a *Derivation* (Derivação).

Como o objetivo deste trabalho é definir regras de negócio em modelagem conceitual, com uma linguagem baseada em uma ontologia de fundamentação, regras devem ser tratadas no nível computacionalmente independente (CIM). Entretanto, não se deseja discutir a interpretação ontológica desses tipos de regras, mas sim apresentar uma forma de representação das regras de negócio dos tipos de Integridade, Derivação e Reação, estendendo a OntoUML com a representação visual desses tipos de regras, com o objetivo de representar domínios de forma mais precisa. No trabalho de (WAGNER *et al*., 2006) e (GIURCA *et al*., 2006), os metamodelos desses tipos de regras em R2ML são apresentados, entretanto, como o objetivo deste trabalho é uma representação gráfica desses modelos, a URML (LUKICHEV e WAGNER, 2007) será utilizada como base. A URML é uma linguagem que, segundo (GIURCA *et al*., 2006), pode ser

considerada derivada da R2ML com o objetivo de prover modelagem de regras baseadas em UML, enquanto que a R2ML é uma linguagem de *markup*. Com isso, se tem uma relação mais próxima com a OntoUML, por ser um perfil UML.

A proposta é uma representação de regras de negócio que seja mais expressiva e precisa. Para isso pretende-se enriquecer a representação atualmente fornecida pela URML, pensando em novos elementos de representação de regras que estendem a semântica dos elementos URML e também estendem a semântica dos construtos de uma ontologia de fundamentação.

A única exceção de tipo de regra que não é tratada pela URML é a regra de Integridade. Esse tipo de regra é tratado na linguagem R2ML, que é usada como formato de troca de informação entre as linguagens *Rule Markup Language* (RuleML), a *Object Constraint Language* (OCL) e *Semantic Web Rule Language* (SWRL) da W3C (2004c). A R2ML será a base para a representação do metamodelo das regras de Integridade, porém não haverá uma representação visual desse tipo de regra para basear esta proposta. A linguagem OCL será utilizada para a representação desse tipo de regra.

Na proposta apresentada nesta dissertação, foi definida a Regra de Negócio de fundamentação (*FoundationalBusinessRule*), afirmações que definem ou restringem aspectos de um negócio/domínio, computacionalmente independente.

A Figura 48 apresenta o metamodelo proposto, com a Regra de Negócio de fundamentação e seus subtipos.

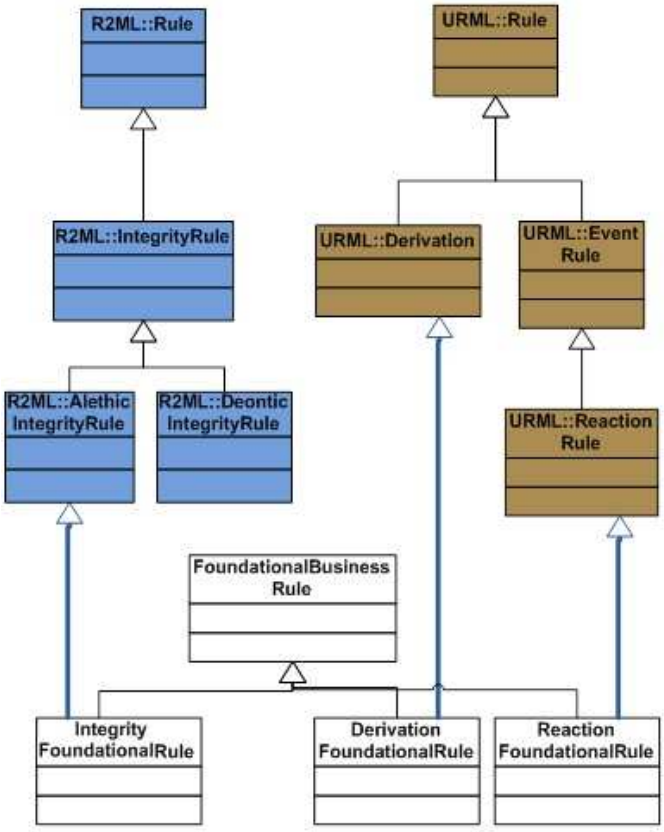


Figura 48 - Meta-modelo da proposta deste trabalho e sua relação com URML e R2ML.

A Regra de Fundamentação de Integridade (*IntegrityFoundationalRule*) é uma afirmação que rege a dinâmica do domínio e não pode ser violada. Possui aplicação imediata porque proíbe qualquer ação que resultaria em valor falso da afirmação. Sua definição é baseada na Restrição de Integridade (*Integrity Constraint*) da R2ML.

A Regra de Fundamentação de Derivação (*DerivationFoundationalRule*) é uma afirmação de um conhecimento que é derivado de outro conhecimento já existente no negócio/domínio. Sua definição é baseada na Derivação (*Derivation*) da URML.

A Regra de Fundamentação de Reação (*ReactionFoundationalRule*) é uma afirmação em que, se uma condição é satisfeita ou um evento ocorre, é disparada uma reação, que pode ser um evento ou ação. Sua definição é baseada na Regra de Reação (*ReactionRule*) da URML.

No decorrer deste capítulo, aborda-se mais precisamente a semântica de cada tipo proposto de regra de negócio.

4.1 Regra de Fundamentação de Integridade

Regra de Fundamentação de Integridade representa elementos no domínio que restringem algo do mesmo, como uma cardinalidade ou restrição aplicada a um relacionamento (por exemplo, “somente 10% dos empregados podem ser gerentes”). Esse tipo de regra é estrutural: não altera o domínio, não cria novo evento ou ação que altere o mesmo, somente restringe algo já existente entre os conceitos. A Regra de Integridade tratada neste trabalho é do tipo *Alethic*, conforme definido em (HALPIN, 2006) e (WAGNER *et al*., 2006). Esse tipo de regra impõe necessidade, que não pode, mesmo em princípio, ser violada pelo negócio. Em (HALPIN, 2006), o tipo de regra chamado *Deontic* também é explorada: regras do tipo *Deontic* representam obrigações, que podem ser violadas, mesmo que isso não devesse acontecer. O trabalho de Halpin apresenta uma formalização para a representação de regras do tipo *Deontic*, que utiliza operadores modais, entretanto esse tipo de regra está fora do escopo deste trabalho: os tipos de regras considerados nesse trabalho são as que não podem ser violadas, mesmo em princípio. A situação onde a regra é burlada não é abordada.

O meta-modelo da Regra de Fundamentação de Integridade é baseado totalmente na R2ML. A Figura 49 apresenta o meta-modelo de Regra de Fundamentação de Integridade.

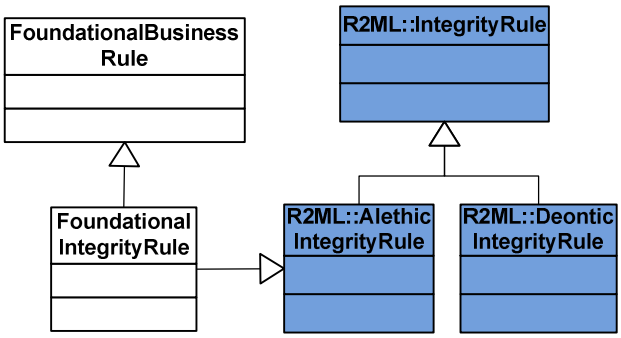


Figura 49 - Meta-modelo proposta desse trabalho da Regra de Fundamentação de Integridade.

As sentenças estruturais, definidas pela BRG (2000), se diferenciam das Regras de Integridade, apesar de parecerem semelhantes. Sentenças estruturais são compostas de Fatos e Termos, sem tratar as definições que restringem o comportamento do negócio. Os conceitos do domínio que se aproximam da definição de sentenças estruturais devem ser modeladas em OntoUML (considerando os tipos de classes e

relacionamentos descritos em 2.2), enquanto que as Regras de Integridade devem ser representados usando a proposta desse trabalho.

4.2 Regra de Fundamentação de Derivação

Regras de Fundamentação de Derivação representam a derivação de novos conceitos no domínio a partir de conhecimento já existente no domínio sendo modelado. Existe uma condição anterior à derivação. Quando o estado do domínio satisfizer essa condição, uma conclusão acontecerá, acrescentando um novo elemento ao domínio.

O meta-modelo da URML apresenta os elementos que formam uma regra de Derivação (*Derivation*). Todas as regras de URML são formadas por uma ou mais Condições (*Condition*), e a Derivação é composta por mais um elemento, uma Conclusão (*Conclusion*). A Figura 50 apresenta o meta-modelo URML para a Derivação.

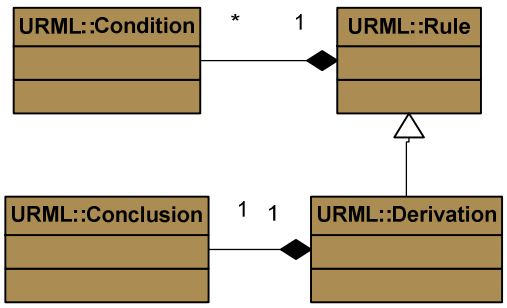


Figura 50 - Meta-modelo URML para a regra do tipo Derivação.

A UFO possui elementos que possuem significado semelhante às meta-propriedades de Regra de Derivação. Uma Condição corresponde a uma Situação (*Situation*) necessária no domínio para o acontecimento da Regra (Situação Pré-estado – *pre-state*), enquanto que a Conclusão corresponde à Situação Pós-estado (*pos-state*) da Regra. Situação representa parte do estado das coisas do domínio. Quando um Evento (*Event*) acontece, altera o estado das coisas, mudando a situação pré-estado para a situação pós-estado (*pos-state*). De forma semelhante ocorre com a Regra de Derivação. Quando uma condição da Regra de Derivação é satisfeita, a Regra de Derivação altera o domínio e o conhecimento representado no mesmo. Entretanto, uma Regra de Derivação não é um Evento, pois os elementos que compõem a Regra de Derivação, a Condição (que se relaciona com o Pré-estado) e a Conclusão (que se relaciona com o Pós-estado), definem um novo conceito no domínio, enquanto que o evento altera o

estado das coisas. A Figura 51 apresenta o meta-modelo da Regra de Fundamentação de Derivação.

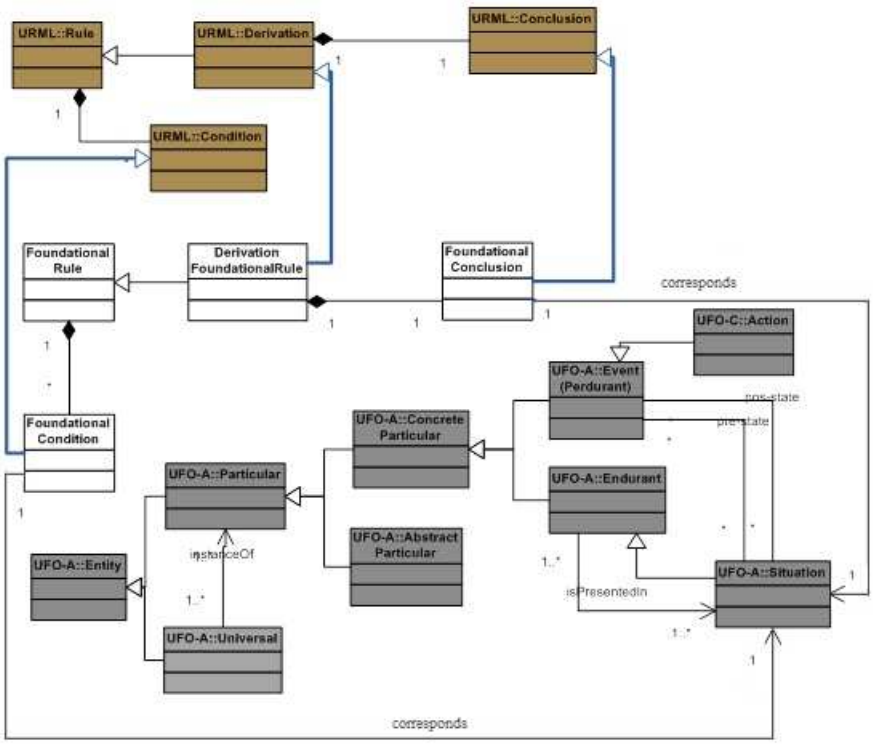


Figura 51 - Meta-modelo proposta desse trabalho da Regra de Fundamentação de Derivação.

4.3 Regra de Fundamentação de Reação

Regra de Fundamentação de Reação representa uma restrição no comportamento do domínio/negócio. Quando um acontecimento no domínio ocorre, caso uma condição pré-definida da Regra de Reação seja satisfeita, um evento de reação a essa condição é disparado, indicando o que deve ser feito no domínio para manter íntegra a restrição do comportamento.

O meta-modelo da URML apresenta os elementos que formam uma Regra de Reação (*ReactionRule*). Todas as regras de URML são formadas por uma Condição (*Condition*) e a Regra de Reação é composta por uma Pós-condição (*Post-Condition*), herdada de seu supertipo Regra de Evento (*EventRule*), e Expressão de Evento (*EventExpression*), que tem dois papéis na composição da Regra de Reação: como o acontecimento disparador da regra, assume o papel da Expressão de Evento Disparador (*triggeringEventExpression*) e, como o acontecimento resultante da regra, assume o

papel da Expressão de Evento Resultante (*resultingEventExpression*). A Figura 52 apresenta o meta-modelo da URML para Regra de Reação.

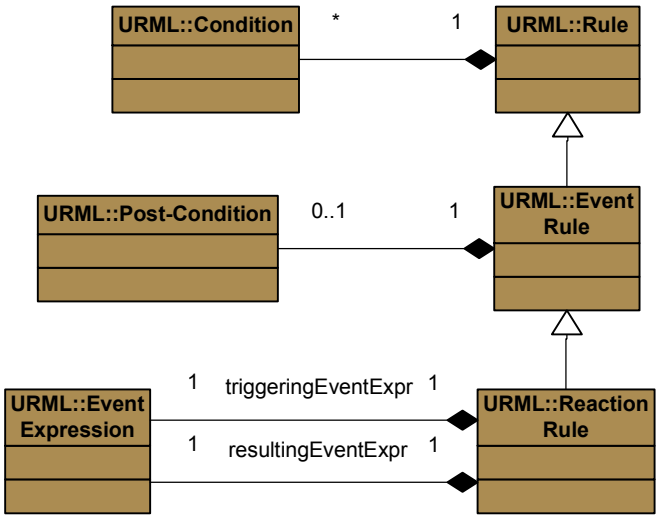


Figura 52 - Meta-modelo URML para a regra do tipo Regra de Reação.

A UFO possui elementos com significado semelhante às meta propriedades de Regra de Reação. Uma Condição corresponde a uma Situação (*Situation*) necessária no domínio para o acontecimento da Regra (Situação Pré-estado – *pre-state*), enquanto que a Conclusão corresponde à Situação Pós-estado (*pos-state*) da Regra. Quando um Evento (*Event*) acontece, altera o estado das coisas, mudando o pré-estado para o pós-estado. Nesse ponto, a Regra de Reação pode ser mais precisamente especificada se agregarmos a semântica da Regra de Reação da URML aos elementos da UFO, pois além da Condição e Pós-condição, a Regra de Reação é composta por um Evento, que ocorre depois que a Condição é satisfeita, alterando o domínio, no mesmo sentido do Evento da UFO. Na UFO, não há o Evento Disparador, somente o Evento Resultante: evento possui um pré-estado e um pós-estado. Com isso, somente a Condição (que é a Situação Pré-estado) é necessária na Regra de Reação. Como se deseja que a linguagem tenha base em uma Ontologia de Fundamentação, a Expressão de Evento Disparador (*triggeringEventExpression*) não é obrigatória na Regra de Fundamentação de Derivação. Considera-se somente a Expressão de Evento Resultante (*resultingEventExpression*), que terá impacto no domínio. A Figura 53 apresenta o meta-modelo da proposta deste trabalho para Regra de Fundamentação de Derivação.

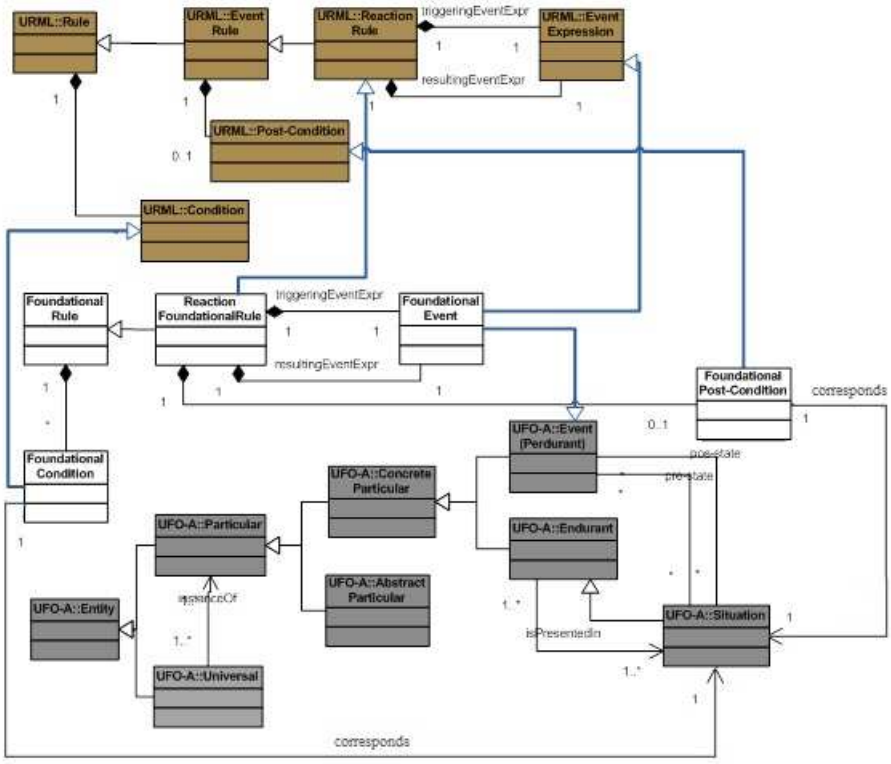


Figura 53 - Meta-modelo proposto da Regra de Fundamentação de Reação.

4.4 R-OntoUML

Nesta seção é apresentado um perfil UML para ser utilizado como linguagem de representação dos três tipos de Regras de Negócio definidos anteriormente: Regras de Fundamentação de Integridade, Regras de Fundamentação de Derivação e Regras de Fundamentação de Reação. Esta proposta de perfil UML é baseada em uma ontologia de fundamentação. Visando o uso desse perfil de forma complementar a linguagem OntoUML, que já é baseada em uma Ontologia de Fundamentação, a UFO-A, conceitos da UFO-B e UFO-C que acrescentariam semântica à representação de Regras de Negócio foram considerados. Também foram propostos construtos de representação de Regras de negócio, que estendem a URML, adicionando meta-propriedades de construtos UFO.

O perfil da UML criado foi chamado de R-OntoUML (a letra “R” é inspirada na palavra regra). A R-OntoUML pode ser usada em modelos conceituais em UML, de forma complementar à OntoUML.

Como o objetivo deste trabalho é modelar conceitualmente regras de negócio usado uma linguagem baseada em uma ontologia de fundamentação (uma linguagem ontologicamente bem-fundamentada) e a linguagem escolhida é a OntoUML, a escolha de qual das linguagens de representação de regras de negócio usar se baseou em uma maior compatibilidade com a OntoUML e integração com a UML, que a OntoUML estende, assim como uma representação visual das regras. A representação visual das regras permitiria uma integração direta com um modelo criado em OntoUML. As possíveis escolhas seriam OCL e URML, devido à integração que essas linguagens possuem com a UML. A URML foi escolhida para uso por possuir construtos visuais (que facilitam o entendimento do domínio) para a representação de regras, tendo compatibilidade maior com a OntoUML, que também é uma linguagem visual. Apesar de a OCL ser usada pela URML, ela é completamente textual, o que necessitaria de uma lista de regras fora do modelo conceitual para descrever as regras de negócio, apresentando pouca compatibilidade com a natureza visual da OntoUML. OCL foi usada para a representação de um conjunto específico de regras (que tem como meta-modelo a definição de regra da R2ML), que a URML não se propôs a representar.

4.5 Novos construtos a partir dos conceitos da UFO-B

A UFO-B representa os conceitos do mundo que são definidos pelo intervalo de tempo em que ocorrem. Conforme apresentado na Figura 21, Evento (*Event*) é o elemento central dessa ontologia, e é essencial para a representação de Regras de Fundamentação de Reação (*ReactionFoundationalRule*), conforme discutido na Seção 4.3.

O meta-modelo da URML apresentado na Figura 35 mostra que a Regra de Reação (*ReactionRule*) é composta por Expressões de Evento (*EventExpression*), que pode disparar ou ser o resultado da Regra. Na R2ML (que, como explicitado na Seção 3.5, é a base da URML), existe uma estrutura semelhante ao Evento da UFO, com mesmo nome, possuindo propriedades que indicam a dinâmica de um elemento no domínio, como pode ser visto em (BADICA *et al.*, 2006). Em (LUKICHEV e JARRAR, 2009), exemplos de modelagem de regras usando um estereótipo com o nome *Event* são representados. Com isso, foi identificado que Evento da UFO-B deve

fazer parte da R-OntoUML para representar os elementos que compõem uma Regra de Reação.

Considerando o que foi abordado na Seção 2.2.2, existem dois elementos que deveriam ser usados para representar um Evento, pois o mesmo é abstrato: Evento Atômico (*AtomicEvent*) e Evento Complexo (*ComplexEvent*). Enquanto Evento Atômico é um evento indivisível, Evento Complexo é composto por outros Eventos. Esses dois conceitos devem ser usados para representar classes em um domínio na R-OntoUML, sendo estereótipos a serem usados em um modelo, com os seguintes textos, respectivamente: <<atomicEvent>> e <<complexEvent>>. Um Evento Complexo deve ser composto por, no mínimo, dois outros Eventos. O conceito de Evento não será usado para representar elementos no modelo, pois é abstrato. A Figura 54 apresenta um exemplo de uso desses construtos, mostrando que um Evento Atômico compõe um Evento Complexo, onde um Acidente entre carros é composto por várias Batidas.

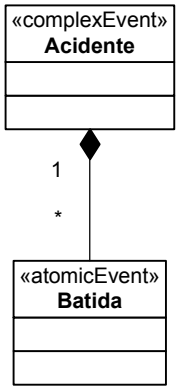


Figura 54 - Exemplo de Eventos Atômicos e Complexos.

Outro elemento da UFO-B que deve fazer parte da R-OntoUML é a Participação (*Participation*) de um *Substantial*, representado no meta-modelo da Figura 21 como o relacionamento “participationOf”. Esse elemento indica quais conceitos do domínio participaram de um Evento. Apesar de não estar explicitamente presente nas definições de regras da URML ou R2ML, a informação de quais elementos do domínio participaram de um Evento acrescenta informações ao modelo. Portanto, o relacionamento de Participação de um *Substantial* deve ser usado como estereótipo de relacionamentos da R-OntoUML entre *Substantials* e Eventos, sejam eles Atômicos ou Complexos, com o texto <<participationOf>>. Outra meta-propriedade que deve ser considerada durante a modelagem de uma Participação: os Substantials de um modelo podem não participar de nenhum evento, como também podem participar de vários. A

Figura 55 apresenta um exemplo de uso do relacionamento de participação, que representa o *Kind* Carro participando do Evento Complexo Acidente, usando o relacionamento com estereótipo <<participationOf>>.

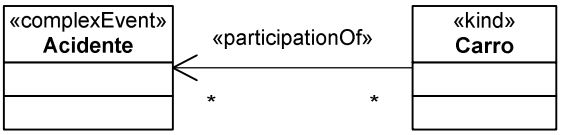


Figura 55 - Exemplo de uso do relacionamento do tipo *partcipationOf*.

Além da relação de Participação de um *Substantial* com Evento, outro elemento da UFO-A também é relacionado diretamente ao conceito Evento: o *Relator*. Um Evento pode ser a fundação de (relacionamento "*foundation of*") de um *Relator* do domínio. Esse relacionamento deve fazer parte dos conceitos usados no perfil R-OntoUML, principalmente, para que a extensão da UFO-A para a UFO-B seja mantida. Ele deve ser usado como estereótipo de relacionamentos da R-OntoUML entre Eventos e *Relators*, sejam eles Atômicos ou Complexos, com o texto <<foundationOf>>. A Figura 56 apresenta um exemplo de uso do <<foundationOf>>, que mostra que a ocorrência de um Evento Atômico Acidente é a fundação para a existência do *Relator* Boletim de Ocorrência.

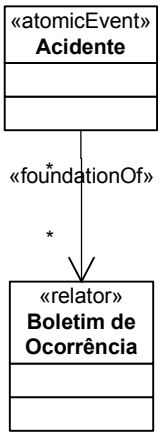


Figura 56 - Exemplo de uso do relacionamento do tipo *foundationOf*.

Existe outra relação entre a UFO-B e as meta-propriedades da URML. Na UFO-B, um Evento altera uma Situação (*Situation*) de um pré-estado para um pós-estado (relacionamentos *pre-state* e *pos-state*). Existem meta-propriedades semelhantes em uma Regra de Reação da URML. Há uma Condição, que é atendida por um estado no domínio (como uma Situação Pré-estado em UFO), iniciando a reação da Regra, que

tem uma Expressão de Evento resultante. Pode haver ou não uma Pós-Condição compondo a Regra de Reação na URML, como um Pós-estado em UFO, mas não é obrigatório. A diferença está na cardinalidade da Pós-Condição na URML, que pode existir ou não, enquanto que na UFO, um evento sempre altera a Situação para um Pós-estado. Na R-OntoUML, a definição da UFO será levada em consideração: a ocorrência da Regra, quando a Condição é atendida, sempre altera o domínio, pois sempre existirá um Evento disparado na Regra de Fundamentação de Reação (e esse Evento altera a Situação do domínio).

Temos uma diferença entre o meta-modelo da URML e da R-OntoUML. Enquanto Evento é parte obrigatória da Regra de Reação na URML como uma Expressão de Evento que dispara uma Regra desse tipo (relacionamento *triggeringEventExpression*), a R-OntoUML não terá essa obrigação. Enquanto que é obrigatória a ocorrência de um Evento resultante da Regra de Fundamentação de Reação , caso obrigássemos a existência de um Evento para disparar esse tipo de Regra, alguns cenários forçariam a criação de um Evento desnecessário. Por exemplo, considerando a condição que inicia uma Regra de Fundamentação de Reação onde o atributo “Dano Material” do “Carro Acidentado” seja maior ou igual a 50%, dispare o Evento subsequente “Perda Total do Veículo”: nessa Regra, a sua Condição é uma regra matemática, onde um Evento disparador é desnecessário. Todas as Regras em URML possuem uma Condição, como será na R-OntoUML, e uma Condição é o suficiente para que uma Regra seja disparada. Um Evento ainda pode disparar uma Regra de Fundamentação de Reação, mas não obrigatoriamente.

Esse conceitos da UFO-B serão aproveitados no perfil R-OntoUML. Novas meta-propriedades não estão sendo criadas: são utilizados os conceitos da Ontologia de Fundamentação UFO para criar o perfil R-OntoUML que busca ter interoperabilidade e compatibilidade com a OntoUML atual, que possui somente os conceitos da UFO-A como construtos.

4.6 Novos construtos a partir dos conceitos da UFO-C

A UFO-C representa os conceitos do mundo relacionados a questões sociais e intenções, assim como as ações que são geradas pelas intenções dos agentes que as executam. Conforme apresentado na Seção 2.2.3, a UFO-C é uma extensão da UFO-A e

da UFO-B, e parte dessa ontologia que é essencial para a representação de Regras de Negócio é apresentada.

Na UFO-C, Ação (*Action*), uma especialização de Evento, pode ser uma Ação Atômica (*AtomicAction*) ou Ação Complexa (*ComplexAction*). Quando uma Ação tem duas ou mais Participações (*Participation*), sejam Participações de Agentes (*Agent*) ou Objetos (*Object*), ela é uma Ação Complexa. Esses dois conceitos devem ser usados para representar classes em um domínio na R-OntoUML, sendo estereótipos a serem usados em um modelo, com os seguintes textos, respectivamente: <<atomicAction>> e <<complexAction>>. Uma Ação não será usada para representar conceitos no modelo, pois é abstrata. A Figura 57 apresenta um exemplo onde uma Ação Atômica compõe uma Ação Complexa, onde a Aceleração é composta por Troca de marcha.

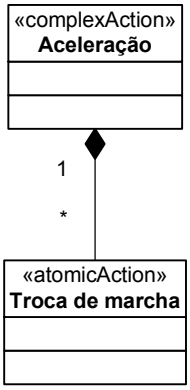


Figura 57 - Exemplo de Ações Atômicas e Complexas.

O conceito Ação pode acrescentar informações ao domínio que não seriam explicitadas. Um Evento possui a Participação de *Substantials* quando ocorre, enquanto que uma Ação tem a participação de um Agente (com o nome "action contribution"), sendo desempenhada por ("*performance of*") esse Agente e com a participação de um Objeto ("participation as resource"). Um Agente executa uma Ação devido a uma Intenção (*Intention*): um Agente tem um objetivo a atingir na execução da Ação, e esse objetivo advém de uma Intenção. Uma Ação também pode ter a Participação de Recursos (*ResourceParticipation*) que são Objetos manipulados durante a Ação. Essas informações enriquecem a representação do domínio, pois, em uma Regra de Fundamentação de Reação, a reação resultante pode ser um Evento ou uma Ação executada por um conceito do domínio com interesse no resultado dessa reação. Essa diferença não seria explicitada, caso ficássemos limitados ao conceito Evento e não explorássemos os conceitos da UFO-C, especificamente o conceito Ação. A execução

de uma Ação por um Agente é representada em R-OntoUML com o estereótipo <<performanceOf>>. A Figura 58 apresenta um exemplo de uso desse estereótipo, onde o Papel Motorista desempenha a Ação Atômica Direção de Carro.

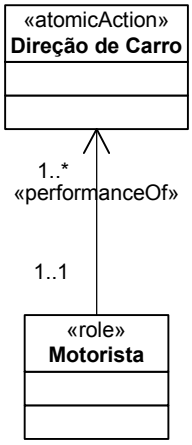


Figura 58 - Exemplo de uso do relacionamento do tipo *performanceOf*.

Em URML, conforme ilustrada na Figura 35, uma Expressão de Evento Ação (*ActionEventExpression*) é uma especialização da Expressão de Evento (*EventExpression*). A Expressão de Evento Ação compõe uma Regra de Produção (*ProductionRule*), podendo criar, eliminar, atribuir e usar objetos ou invocar uma atividade. Na UFO-C, o relacionamento de Participação de Recursos pode se especializar em tipos semelhantes (conforme visto na Seção 2.2.3): Criação (*Creation*) ocorre quanto um objeto é criado depois da execução de uma Ação; Terminação (*Termination*) ocorre quando um objeto deixa de existir depois da execução de uma Ação; e, Alteração (*Change*) acontece quando um Objeto é modificado durante uma Ação. Na representação proposta esses tipos de Participação de Recursos não são utilizados, pois se aproximaria da forma que a Expressão de Evento Ação é usada em Regras de Produção de URML, tipo de Regra que não será explorada neste trabalho, pois está no nível de representação dependente computacional. Todas as Participações de Recursos serão do tipo Uso (*Usage*), que ocorre quando um Objeto participa em uma Ação. No modelo usando o perfil R-OntoUML, o relacionamento Participação de Recurso deve ser representado como estereótipo <<participationAsResource>>. A Figura 59 apresenta o uso desse estereótipo, uma extensão da Figura 58, onde o *Kind* Carro participa como recurso da Ação Complexa da Direção de Carro. O Papel Motorista, como o executor da Ação, usa o Carro para desempenhar a Direção de Carro.

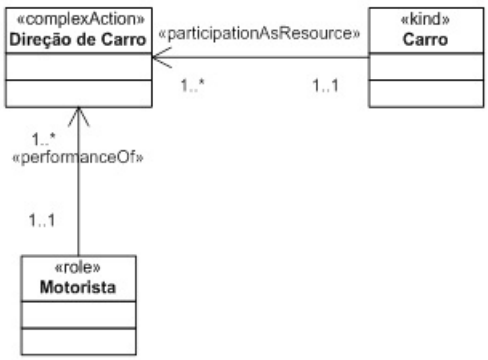


Figura 59 - Exemplo de uso do relacionamento do tipo *participationAsResource*.

Esse conceitos da UFO-C serão aproveitados no perfil R-OntoUML. Novas meta-propriedades não estão sendo criadas: os conceitos da Ontologia de Fundamentação UFO são utilizados para criar o perfil R-OntoUML que busca ter interoperabilidade e compatibilidade com a OntoUML atual, que possui somente os conceitos da UFO-A como construtos, assim como os conceitos da UFO-B usados na R-OntoUML.

4.7 Construtos da R-OntoUML a partir da URML

Com os elementos explorados na UFO, não foram encontrados construtos suficientes para representar os tipos de regras propostas: Regras de Fundamentação de Integridade, Regras de Fundamentação de Derivação e Regras de Fundamentação de Reação. Como a URML apresenta construtos para a representação de Regras de Derivação e Regras de Reação, estes construtos foram acrescentados na R-OntoUML.

Os elementos básicos da URML foram mantidos na R-OntoUML, por exemplo, Termos de Objetos (*ObjectTerm*), Variáveis de Objetos (*ObjectVariable*), Filtro OCL (*OCLFilter*) e Átomos (*Atom*).

Como boa parte dos elementos usados em URML são elementos já existentes em UML (por exemplo, Classes, Associações ou Atributos), durante a construção das regras nos modelos, devem-se considerar as meta-propriedades dos construtos da UFO (por exemplo, os estereótipos usados nas Classes alteram a semântica do conceito, o que não havia antes, pois a Classe não teria um estereótipo), que também é uma extensão da

UML, sejam esses elementos da UFO os já definidos na linguagem OntoUML ou os acrescentados nesse trabalho através do perfil R-OntoUML.

Em R-OntoUML, quando está sendo representada uma Variável de Regra (relacionamento "*/ruleVariables*", na Figura 35), as meta-propriedades do Termo de Objeto que está sendo usado como essa Variável (que será ou fará parte de um elemento representado com as meta-propriedades de um conceito UFO) devem ser levadas em conta para a sua ligação à regra.

Qualquer elemento no domínio pode ser a condição (com o uso de um Seta de Condição - *ConditionArrow*) de uma Regra, independentemente de qual tipo de classificação na UFO que ele tenha recebido (o que é traduzido na aplicação de estereótipos nos conceitos). Em R-OntoUML, uma Seta de Condição refere-se a um elemento de condição do modelo, que é um *classifier* como uma classe ou associação. Pode possuir uma *expressão de filtro* que seleciona instâncias do *classifier*, escrita em OCL. Esse relacionamento, saindo de um conceito do domínio em direção ao elemento que representa a Regra, deve ser representado como estereótipo <<conditionArrow>>. As Setas de Condição Negada também são usadas em R-OntoUML. Elas são cruzadas na origem da seta. Elas denotam a negação da condição que deve ser conjunta com uma ou mais setas de condição positivas para que suas variáveis sejam cobertas por elas. A Figura 60 apresenta uma notação de uso da Seta de Condição, relacionado a uma regra com nome "Regra de Derivação", do tipo Regra de Fundamentação de Derivação.

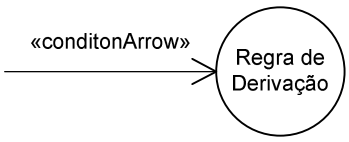


Figura 60 – Notação da *conditionArrow*.

4.7.1 Descrição da Regra de Fundamentação de Derivação

Conforme descrito na Seção 4.24.2, Regra de Fundamentação de Derivação representa a derivação de novos conceitos no domínio a partir de conhecimento já existente. Existe uma condição anterior à derivação e, quando o estado do domínio satisfizer essa condição, uma conclusão acontecerá, definindo como um elemento do domínio é identificado.

De forma análoga a URML, uma Regra de Fundamentação de Derivação é composta por uma Condição que, caso seja satisfeita, apresenta um novo conceito ao domínio. A indicação desse novo conceito é feita a partir da Conclusão de Fundamentação (*FoundationalConclusion*), que compõe uma Derivação. Uma Conclusão é representada por uma Seta de Conclusão (*Conclusion Arrow*), que, de forma semelhante a uma Seta de Condição, está ligada a um elemento do modelo. Em R-OntoUML, a Seta de Condição é um relacionamento saindo do elemento que representa a Regra até o conceito derivado a partir da Regra e deve ser representada com o estereótipo <<conclusionArrow>>.

A Conclusão da R-OntoUML pode ser classificada como os tipos encontrados na URML: Conclusão de Classificação (*Classification Atom*), Conclusão de Atribuição (*Attribution Atom*) e Conclusão de Associação Binária (*Binary Association Atom*). O tipo Conclusão de Papel (*Role Type Atom*) não deve ser considerado em R-OntoUML, pois a ideia de Papel possui um significado diferente devido ao conceito da UFO-A Papel (*Role*). Quando uma Regra de Fundamentação de Derivação tem uma Conclusão de Classificação, a construção do modelo deve levar em consideração o conceito a ser derivado e a sua classificação dentro da UFO. Como esse conceito pode ser classificado como um Papel da UFO, não é necessário o tipo Conclusão de Papel.

Regras de Fundamentação de Derivação (*Remodeled Derivation Rule*) são representadas graficamente como círculos, com o estereótipo <<rDerivationRule>> escrito internamente, junto com texto identificando a regra. Setas chegando ao círculo representam condições (com o estereótipo <<*conditionArrow*>>), enquanto setas saindo representam conclusões (com o estereótipo <<*conclusionArrow*>>). A Figura 61 apresenta a notação proposta para a Regra de Fundamentação de Derivação e seus construtos relacionados.

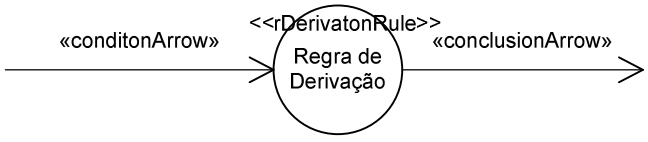


Figura 61 - Estereótipos de Regra de Fundamentação de Derivação.

Como exemplo de uso da Regra de Fundamentação de Derivação em um domínio real, a Figura 62 representa a identificação quando um *Kind* Pessoa passa para

a Fase Maior de Idade. A Seta de Condição indica o teste a ser feito com o atributo Idade de Pessoa: quando maior ou igual a 18 (dezoito), o conceito Maior de Idade é derivado, o que indicado pela Seta de Conclusão. Com essa representação, é possível identificar como uma Pessoa atinge a maior idade.

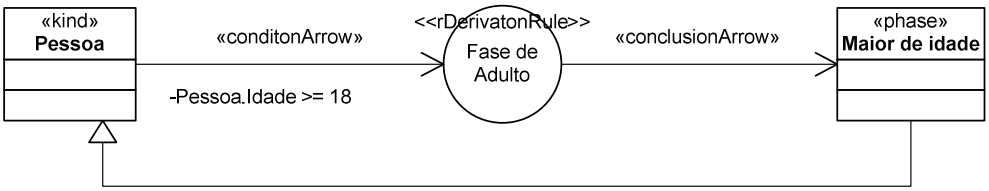


Figura 62 – Exemplo do uso de Regra de Fundamentação de Derivação para a derivação de um conceito.

4.7.2 Descrição da Regra de Fundamentação de Reação

Conforme descrito na Seção 4.3, uma Regra de Fundamentação de Reação representa uma restrição no comportamento do domínio/negócio. Quando um acontecimento no domínio ocorre, caso uma condição pré-definida da Regra de Reação seja satisfeita, um evento de reação a essa condição é disparado, indicando o que deve ser feito no domínio para manter a restrição do comportamento íntegra. Em R-OntoUML, o nome usado para o construto será Regra de Fundamentação de Reação.

Nas seções 4.5 e 4.6, foram discutidos os conceitos da UFO a serem usados para enriquecer semanticamente a representação de Regras de Fundamentação de Reação. Uma Regra de Fundamentação de Reação é composta por Eventos, sejam esses Eventos os que disparam o início da Regra ou a sua reação. Em URML, é obrigatória a existência do Evento que dispara a Regra (na Figura 35, é representada pelo relacionamento "*triggeringEventExpr*", com cardinalidade nas duas pontas da associação de no mínimo 1 e no máximo 1) e o Evento resultado dessa Regra (na Figura 35, é representada pelo relacionamento "*resultingEventExpr*", com cardinalidade nas duas pontas da associação de no mínimo 1 e no máximo 1). Em R-OntoUML, não será obrigatória a existência do Evento que dispara a Regra, pois toda Regra já é composta por Condições, representadas pelas Setas de Condição. A Regra de Reação de Fundamentação tem que ter no mínimo uma Seta de Evento (com um Evento ou Ação da R-OntoUML relacionada a essa Regra) ou uma Seta de Condição (usando o

estereótipo <<conditionArrow>>), não podendo deixar de ter no mínimo uma das duas Setas. O Evento resultante da Regra ainda é necessário, e pode ser representado como um Evento ou Ação da R-OntoUML.

Para a representação de Evento ou Ação da R-OntoUML como condição da Regra, temos a Seta de Evento (*EventArrow*) que será usada em conjunto com os conceitos classificados como Evento ou Ação, dentro da representação de uma Regra de Fundamentação de Reação. Essa Seta de Evento é representada saindo de um Evento ou Ação, terminando no elemento que representa a Regra de Fundamentação de Reação. Deve ser representada com o estereótipo <<eventArrow>>.

A Seta de Condição é representada e definida da mesma forma que foi apresentada na Seção 4.7.1.

A Regra de Fundamentação de Reação tem dois tipos de setas saindo do elemento que representa a regra: a Seta de Evento e Seta de Pós-condição. Toda representação de Regra de Fundamentação de Reação deve ter um Evento ou Ação como resultado. A Seta de Evento é usada para ligar a Regra ao elemento que representa o acontecimento resultante da Regra modelada. Possui o mesmo estereótipo já apresentado: <<eventArrow>>. A Figura 63 apresenta a notação da Seta de Evento como condição e reação de uma Regra de Fundamentação de Reação, assim como a Seta de Condição.

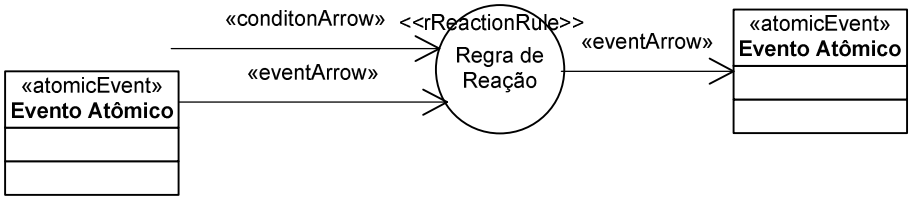


Figura 63 - Notação de Setas de Evento e Setas de Condição em uma Regra de Fundamentação de Reação.

A Seta de Pós-condição é semelhante à Seta de Condição, mas é usada para representar um resultado da Regra. Essa Seta possui uma expressão em OCL interligada, identificando alguma alteração no domínio. Em R-OntoUML deve ser representada saindo do elemento que representa a regra e chegando ao conceito no domínio que a Pós-condição afeta, com o estereótipo <<conditionArrow>>. A Figura 64

estende a Figura 63, com a notação da Seta de Condição como Pós-Condição da Regra de Reação.

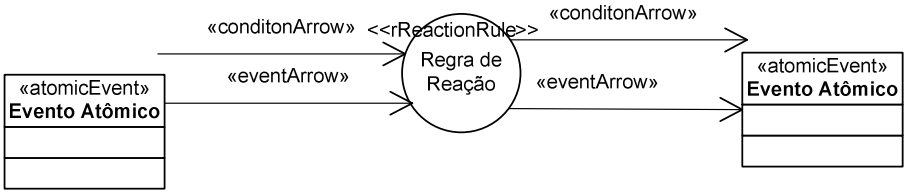


Figura 64 - Notação da Seta de Condição como Pós-Condição de uma Regra de Fundamentação de Reação.

Em R-OntoUML, não será usada a Seta de Ação da URML, pois a Seta de Evento representa toda a semântica em R-OntoUML. Como Ação é uma especialização de Evento em UFO, a Seta de Evento pode ser usada para construção da Regra de Fundamentação de Reação.

Regras de Fundamentação de Reação são representadas graficamente como um círculo com o estereótipo <<rReactionRule>> escrito internamente, junto com texto identificando a regra. Existem dois tipos de setas chegando (setas de condição ou setas de evento) e dois tipos de setas saindo (setas de evento ou setas de pós-condição). A Figura 65 mostra um exemplo da Regra de Fundamentação de Reação em um domínio real. Quando ocorre o Evento Complexo Acidente, com a participação do *Kind* Carro, a Ação Complexa Socorro é executada pelo Papel Bombeiro.

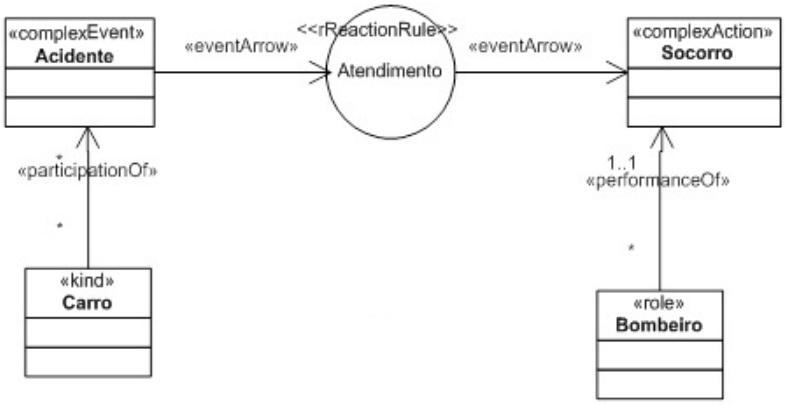


Figura 65 - Exemplo do uso da Regra de Fundamentação de Reação, indicando a reação a um acidente.

4.8 A representação de Regras de Fundamentação de Integridade usando a OCL

Os construtos da UFO utilizados para construir o perfil R-OntoUML foram apresentados nas seções 4.5 e 4.6, e os construtos da URML, na Seção 4.7. Entretanto, um dos tipos de regra a ser representado, descrito na Seção 4.1, não possui representação em OntoUML e URML: Regra de Fundamentação de Integridade. Em R-OntoUML, o nome desse construto será Regra de Fundamentação de Integridade.

Esse tipo de regra indica alguma restrição em algum elemento do domínio, seja a propriedade de um conceito ou um relacionamento. Restringe os valores de um atributo, por exemplo. Em OntoUML e URML não existe uma conceitualização ou construto que permite esse tipo de regra. Levando isso em consideração, a R-OntoUML utiliza a Invariante em OCL (escrito com "inv"), elemento descrito na Seção 3.6. A OCL foi escolhida no lugar da R2ML devido à sua maior interoperabilidade com a UML e proximidade com a URML, inclusive sendo usada para representar Filtros OCL.

O tipo Invariante aplica a regra para qualquer instância do contexto a qualquer momento de existência do sistema. Uma restrição Invariante é do tipo booleano. Costuma manipular dados do Contexto (*Context*) descrito em uma regra, aplicando alguma restrição a esse elemento ou elementos relacionados. Como é uma regra que não pode ser quebrada, se aproxima a Regra de Fundamentação de Integridade. Costuma ter a seguinte escrita, seguindo a navegabilidade que a OCL permite:

context elemento **inv:**
elemento.atributo -- Terminando com uma comparação

O conjunto de elementos usados para a comparação (por exemplo, "maior que" ou "igual a") em regras OCL é feita com os operados Infix apresentados na Seção 3.6.

Como o conjunto de regras em OCL é criado separado do modelo, em formato textual, a sua forma de uso foi alterada para que possa se enquadrar na R-OntoUML. A proposta já inclui OCL entre seus construtos, pois faz parte do Filtro OCL (*OCLFilter*). Com isso, incluir a OCL no modelo para representar Regras de Fundamentação de Integridade não afeta a R-OntoUML.

4.8.1 Descrição da Regra de Fundamentação de Integridade

Conforme descrito na Seção 4.1, Regra de Fundamentação de Integridade representa elementos no domínio que restringem algo do mesmo. Esse tipo de regra não altera o domínio, não cria novo evento ou ação que altere o mesmo, somente restringe algo já existente entre os conceitos.

Regra de Fundamentação de Integridade restringe um elemento do modelo. Regras desse tipo são representadas por retângulos com o estereótipo <<rIntegrityRule>>, contendo a sentença em OCL para representação da regra.

O retângulo que permite que a regra seja escrita deve estar ligado ao conceito ao qual a regra se refere. Esse relacionamento indica qual o Contexto que a regra referencia. Segue exemplo, na Figura 66, para facilitar o entendimento, incluindo representação em R-OntoUML e como seria a sua tradução para OCL.

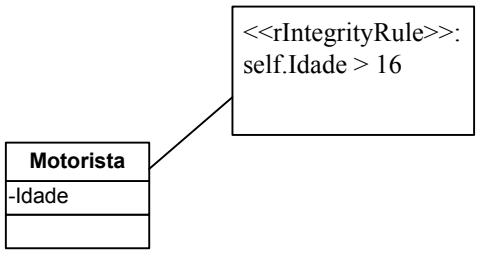


Figura 66 - Exemplo de uso da Regra de Fundamentação de Integridade.

Nesse exemplo, está indicado que o atributo Idade da classe Motorista deve ser maior que 16. A tradução dessa Regra para OCL seria:

context Motorista **inv:**
self.Idade > 16

O significado da regra em OCL é mantido nas duas representações. Altera-se somente a definição do Contexto da Regra, usando um recurso visual: o relacionamento entre o conceito e o retângulo que identifica a Regra de Fundamentação de Integridade, com o estereótipo <<rIntegrityRule>>. O elemento do modelo relacionado ao retângulo que representa a Regra é o Contexto.

4.9 Meta-modelo de R-OntoUML

Com o objetivo de resumir a definição apresentada, a Figura 67 apresenta o meta-modelo da R-OntoUML, com os construtos já discutidos e destacando os

construtos principais: *Foundational Condition*, *Foundational Conclusion*, *Foundational Logical Statement*, *Foundational Post-Condition* e *Foundation Event*.

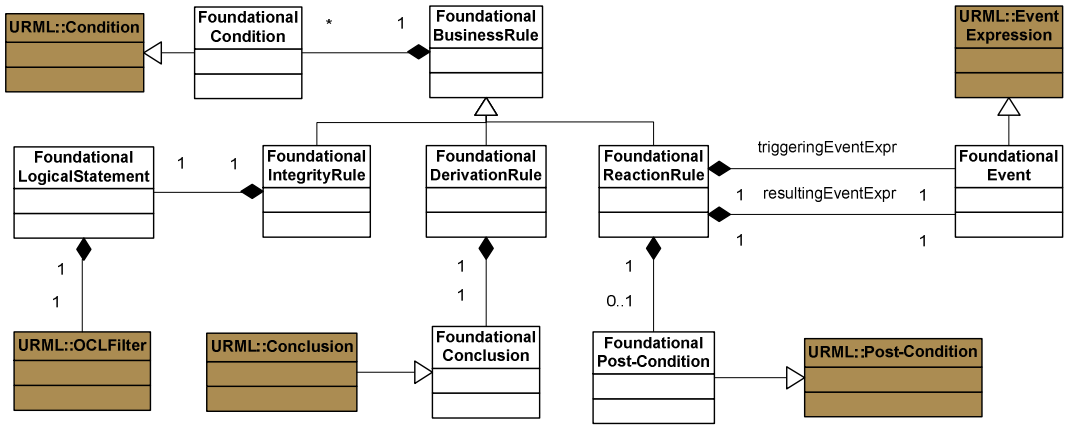


Figura 67 - Meta-modelo URML com os principais construtos.

4.10Trabalhos relacionados

Trabalhos que usam ontologias de fundamentação e a relacionam com modelagem de processos de negócio foram encontrados. Em (BAIÃO *et al*., 2008), é proposta uma forma de extrair, de modelos de processos de negócios, conceitos para uma ontologia de domínio. SANTOS JÚNIOR (2008) relaciona modelos de processos de negócio e modelos conceituais baseados em ontologia de fundamentação, em OntoUML, com o objetivo de obter uma visão única da estrutura e aspectos comportamentais de um domínio. Entretanto, esses trabalhos não tratam de regras de negócio. Exploram a relação do encadeamento das atividades pertencentes a um processo de negócio, sem explorar a representação de regras de negócio.

O trabalho (MARTINS *et al*., 2011) apresenta um perfil OntoUML, baseado na ontologia de fundamentação UFO, para apoiar a representação na modelagem de processos de negócio. O perfil, chamado E-OntoUML, estende o diagrama de atividades, acrescentando meta-propriedades da UFO a construtos desse tipo de diagrama. MARTINS *et al*., (2011) focam na representação do encadeamento das atividades, enquanto a proposta deste trabalho apresenta uma forma de modelar as regras de negócio que permeiam o encadeamento dessas atividades, em conjunto com a estrutura do domínio tratado. Os dois trabalhos são complementares: enquanto o perfil E-OntoUML descreve os processos de negócio, a proposta deste trabalho acrescenta a representação das regras de negócio que regem os processos descritos em E-OntoUML.

Outro trabalho relacionado é (BENEVIDES e GUIZZARDI, 2009). A ferramenta de edição em OntoUML é apresentada, identificando sua estrutura base e método de construção, permitindo a construção de modelos conceituais em OntoUML. A verificação do modelo é realizada pelo editor, impedindo erros de modelagem, ao não seguir as meta-propriedades dos construtos OntoUML. Como o perfil proposto neste trabalho deve ser usado em conjunto da OntoUML, parte da modelagem pode ter apoio computacional, fazendo com que a modelagem seja correta em relação à linguagem.

4.11 Conclusão

Neste capítulo, foi apresentada a estrutura do perfil R-OntoUML, que se baseia na Ontologia de Fundamentação UFO, na linguagem de representação visual de regras URML, e em parte da OCL. Com essa estrutura, usando a OntoUML para modelar o domínio além das Regras de Negócio, parte essencial da proposta, foi elaborado um Estudo de Caso para testar a hipótese, apresentado no Capítulo 5.

5 Estudo de caso para avaliação da Hipótese

No capítulo 4, a proposta de solução foi apresentada. O presente capítulo relata a avaliação da solução a partir de um estudo de caso, com uma análise qualitativa dos resultados de um questionário e dos modelos resultantes. Para o estudo de caso, considerou-se um caso de uso, retirado de (OMG, 2008), que descreve um cenário de aluguel de carros, contendo a descrição textual de seis regras de negócio a serem representadas.

Durante a realização do estudo de caso, cada participante foi instruído a elaborar dois modelos de regras de negócio. No primeiro modelo, os participantes deveriam utilizar a linguagem UML (OMG, 2009) em conjunto com a URML, enquanto que no segundo modelo, deveriam utilizar a OntoUML em conjunto com o perfil proposto desse trabalho, a R-OntoUML. O objetivo foi comparar os dois modelos quanto à completeza, validade e precisão da representação em relação à descrição original das regras de negócio.

A escolha dos participantes utilizou como critério pessoas que já conhecessem a UFO e OntoUML, visando diminuir a influência do conhecimento sobre a linguagem (utilizada na representação da parte estrutural do domínio) no resultado do estudo de caso. Foram selecionados três participantes, um deles especialista em UFO e OntoUML, inclusive com dissertação de mestrado com base no trabalho de Guizzardi (2005), e dois participantes que iniciaram seus estudos de UFO e OntoUML.

Um questionário avaliando se o modelo é completo e válido foi aplicado depois de cada modelagem, avaliando o modelo resultante de cada linguagem, para em seguida, comparar os dois modelos segundo generalidade e precisão. Em uma segunda etapa, os modelos gerados foram enviados para pessoas diferentes, e cada um deveria avaliar os modelos, respondendo o mesmo questionário. Finalizando, os participantes

responderam um questionário de percepção de uso, e compartilharam a sua experiência de uso da OntoUML e do perfil proposto neste trabalho.

Os dados coletados devem caracterizar a percepção dos participantes referentes aos modelos gerados em UML junto com URML e OntoUML junto com o perfil proposto neste trabalho, comparando a generalidade e precisão do resultado. A análise do estudo de caso teve como fonte de dados os modelos gerados e questionários respondidos.

Sessões foram realizadas com os participantes, com um treinamento inicial em URML, em OntoUML e sobre o perfil OntoUML proposto nesse trabalho. Em seguida, os participantes construíram os modelos e responderam os questionários. A segunda etapa foi realizada por correspondências eletrônicas, com o envio dos modelos de seus pares digitalizados, para que respondessem o questionário avaliando a modelagem de outro participante do estudo de caso.

As respostas aos questionários foram analisadas, vislumbrando a avaliação dos participantes da precisão e generalidade dos modelos gerados.

Na Seção 5.1, a preparação do estudo de caso é detalhada para, em seguida, o estudo de caso ser abordado e o resultado de cada participante ser analisado qualitativamente.

5.1 Preparação para o Estudo de Caso

O estudo de caso detalha um domínio de uma empresa de aluguel de carros, retirado da (OMG, 2008). Segue o resumo do domínio, retirado do Apêndice A:

“EU-Aluguel aluga carros para seus clientes. Clientes podem ser indivíduos ou empresas. Diferentes modelos de carros são oferecidos, organizados em grupos. Todos os carros em um grupo são cobrados com as mesmas taxas. Um carro pode ser alugado através de reserva feita com antecedência ou por um cliente avulso no dia do aluguel. Uma reserva de aluguel especifica o grupo de carro requisitado, as datas e horários de início e final de aluguel e a filial da EU-Aluguel de onde o aluguel iniciará. Opcionalmente, a reserva pode especificar um aluguel em um só sentido (no qual o carro é devolvido a uma filial diferente da filial onde foi retirado) e pode requisitar um modelo de carro específico dentro do mesmo grupo.

EU-Aluguel tem um clube de fidelidade. Clientes que se afiliam acumulam pontos que podem ser usados para pagar por aluguéis.

EU-Aluguel, de tempos em tempos, oferece descontos e aprimoramento do modelo do carro de graça, dependendo de condições.

EU-Aluguel registra “más experiências” com clientes (como retorno atrasado não autorizado de um aluguel, ou dano no carro durante o aluguel) e pode recusar seguidas reservas de aluguel desses clientes.”

Um subconjunto do domínio foi escolhido para fazer parte do estudo de caso. Duas regras de negócio de cada tipo foram escolhidas. As regras foram classificadas antes da entrega do documento para os participantes, segundo os tipos definidos: Regra de Integridade, Regra de Derivação e Regra de Reação.

Foram escolhidas as seguintes Regras de integridade:

- É obrigatório que a duração do aluguel de cada aluguel seja de no máximo 90 dias.
- É obrigatório que o nível de combustível do carro alugado esteja cheio.

Foram escolhidas as seguintes Regras de Derivação:

- Se a data/hora efetiva de devolução do aluguel for maior que a data/hora de devolução, então ocorre uma má experiência no aluguel.
- Se a filial de retorno é diferente da filial de retirada, então o aluguel é do tipo aluguel em um só sentido (*One Way Rental*).

Foram escolhidas as seguintes Regras de Reação:

- Se o local de devolução do aluguel não é a filial de retorno do aluguel, então é obrigatório que o aluguel sofra uma multa local.
- Se a cobrança estimada do aluguel for provisoriamente cobrada no cartão de crédito do alugador que é responsável pelo aluguel, então é obrigatório que o aluguel esteja aberto. (Caso a reserva esteja feita, mas ainda não confirmada e paga – com a cobrança estimada provisoriamente no cartão de crédito –, o aluguel fica com o estado aberto)

Complementando as regras, um conjunto de sentenças foi disponibilizado. Essas sentenças possuem os conceitos referenciados nas regras de negócio, com um formato semelhante às Sentenças Estruturais da (BRG, 2000): apresentam os conceitos estáticos do domínio, sem nenhuma restrição de comportamento. O documento da OMG (2008) apresenta sentenças atreladas a cada regra, definindo os conceitos que fazem parte da regra sendo descrita. O resumo inicial não deveria ser considerado pelos participantes

para a construção inicial do modelo, somente para solucionar alguma dúvida relativa ao domínio depois da modelagem das sentenças e regras de negócio. O Apêndice A contém a descrição do cenário escolhido para o estudo de caso.

5.1.1 Elaboração do questionário

Os questionários aplicados em cada etapa constam do Apêndice B e do Apêndice C, e apresentaram a seguinte composição:

- Perguntas para avaliar a completeza e validade dos modelos, critérios de qualidade defendidos por MOODY (2003);
- Uma pergunta para perceber se havia alguma informação ausente nas regras de negócio que dificultava o entendimento do domínio. Essa pergunta tinha o objetivo de avaliar se as meta-propriedades dos elementos da UFO traziam mais questões relacionadas às propriedades implícitas aos conceitos do domínio, quando houvesse a troca entre as linguagens. Com isso, esperava-se perceber se a linguagem levantaria questionamentos sobre o domínio que inicialmente não estariam explícitos.
- Perguntas abertas para comparar os dois modelos quanto à completude e a validade.
- Duas perguntas, baseadas no trabalho de PATIG (2004), avaliando comparativamente a generalidade e a precisão dos modelos baseadas nas linguagens.
- Um conjunto de perguntas visando identificar a percepção de uso da OntoUML e R-OntoUML pelos participantes. Esse conjunto de perguntas, com escala numérica, foi retirado diretamente de (BATRA *et al.*, 1990).
- Uma pergunta aberta final para que o participante descrevesse a experiência do uso da linguagem e do perfil proposto neste trabalho.

A seguinte pergunta foi elaborada para avaliar a completeza do modelo, em conjunto com uma explicação entre parênteses:

- “O modelo contém todas as afirmativas das regras de negócio que são corretas e relevantes? (Para esta avaliação, verifique se é possível identificar no modelo todas as informações que estão mencionadas nas regras)”

Completeza representa a capacidade de uma linguagem representar toda a conceitualização de um domínio. Essa pergunta tinha uma escala de 1 (um) a 4 (quatro), onde a opção 1 indicava que não havia no modelo as afirmativas corretas e relevantes descritas nas regras de negócio, enquanto a opção 4 indicava que o modelo continha todas as afirmativas corretas e relevantes descritas nas regras de negócio. Caso o participante respondesse os valores intermediários, teria que descrever com mais detalhes quais informações estariam faltando no modelo.

Para avaliar a validade do modelo, uma segunda pergunta foi elaborada, em conjunto com uma explicação entre parênteses:

- “Todas as afirmações do modelo são corretas e relevantes ao domínio descrito pelas regras de negócio? (Para esta avaliação, verifique se existem informações no modelo que não estejam descritas nas regras)”

Validade representa a capacidade de uma linguagem representar os conceitos corretos e relevantes ao domínio, sem representar nada além do definido pelo domínio. De forma semelhante à primeira pergunta, a escala da segunda pergunta era de 1 (um) a 4 (quatro). A opção 1 indicava a existência no modelo de afirmativas que não eram corretas e relevantes ao domínio descritas nas regras de negócio, enquanto a opção 4 indicava que não havia no modelo nenhuma afirmativa incorreta ou irrelevante descrita nas regras de negócio. Caso o participante respondesse os valores intermediários, teria que descrever com mais detalhes quais informações incorretas e irrelevantes estavam contidas no modelo.

Visando à comparação dos modelos, as perguntas foram refeitas, desconsiderando a escala de pontuação e permitindo a escrita de um texto livre como resposta. A pergunta sobre a completeza foi alterada da seguinte forma:

- “Comparando os modelos, quais as diferenças entre eles em conter todas as afirmativas das regras de negócio que são corretas e relevantes? (Para esta avaliação, verifique se é possível identificar nos modelos todas as informações que estão mencionadas nas regras, e quais as diferenças entre eles)”

A pergunta sobre validade foi alterada da seguinte forma:

- “Comparando os modelos, todas as afirmações corretas e relevantes ao domínio descrito pelas regras de negócio aparecem nos modelos? (Para esta avaliação, verifique se existem informações nos modelos que não estejam descritas nas regras, e o que pode ter ocasionado isso)”

Ainda para a comparação entre os modelos, duas perguntas específicas, de texto livre, foram elaboradas usando como base o trabalho de PATIG (2004). Esse trabalho divide em duas características a qualidade de modelos conceituais: generalização e precisão, comparando a expressividade extensional (*extension*) desses modelos.

Generalização representa a capacidade do modelo de expressar mais objetos da realidade, com maior liberdade para a instanciação desses objetos em cada classe do modelo. Com isso, é possível usar o modelo para representar um conjunto maior de elementos da realidade. A pergunta se traduziu da seguinte forma, incluindo uma explicação entre parênteses:

- “Qual modelo dá maior liberdade para instanciação dos objetos do mundo real em cada classe/relacionamento/regra, permitindo com que cada conceito tenha um conjunto maior de elementos do mundo para instanciação? (As meta-propriedades dos construtos permitem que cada construto represente mais conceitos do mundo real)”

Precisão corresponde ao decréscimo de extensão de um modelo, onde existe um número menor de elementos cujas fórmulas de descrição tem valor “verdadeiro” (PATIG, 2004), ou seja, menor quantidade de objetos da domínio instanciáveis em conceitos do modelo. Com isso, é possível usar o modelo para representar de forma mais delimitada os elementos da realidade. A pergunta se traduziu da seguinte forma, incluindo uma explicação entre parênteses:

- “Qual modelo dá maior especificidade/precisão na instanciação dos objetos do mundo real em cada classe/relacionamento/regra? (As meta-propriedades dos construtos delimitam com maior precisão a quantidade de conceitos do mundo real que cada construto pode representar)”

Após as perguntas voltadas à avaliação, um conjunto de questões voltadas para a percepção do usuário foi incluído, baseado no trabalho de BATRA *et al*. (1990). Segue a lista de questões explorando características do uso da OntoUML e do perfil proposto neste trabalho:

- Acredito que a técnica de modelagem é desconfortável de usar.

- Usar a técnica de modelagem foi frustrante.
- Usar a técnica de modelagem exigiu muito esforço mental.
- A técnica de modelagem é clara e compreensível para mim.
- Considerando tudo, acredito que a técnica de modelagem é fácil de usar.

Cada pergunta deveria ser respondida em uma escala numérica, aonde 1 (um) indica que “Concordo fortemente” com a afirmação, enquanto 7 (sete) representa “Discordo fortemente”. Esse questionário está reproduzido no Apêndice C.

Finalizando o questionário, o participante tinha a liberdade de descrever em texto livre como foi a sua percepção de uso da OntoUML e R-OntoUML na modelagem. A pergunta tem seguinte texto:

- Por favor, explique como foi a experiência de uso da OntoUML e R-OntoUML.

Com a descrição do cenário escolhido e o questionário prontos, o estudo de caso com os participantes tornou-se viável.

5.1.2 Procedimento adotado para o estudo de caso

Os estudos de caso foram realizados em sua maioria, separadamente. Aconteceram três encontros separados com o primeiro participante. Antes da primeira etapa da modelagem, a URML e OCL foram apresentadas, incluindo exemplos de um domínio diferente do abordado no caso de uso. Todos os tipos de regras foram explicados e exemplificados, junto com seus construtos.

Antes de segunda etapa de modelagem, os construtos da OntoUML foram revisados, novamente, com exemplos fora do domínio de aluguel de carros. Em seguida, os construtos do perfil R-OntoUML foram apresentados, de forma semelhante às explicações anteriores. Foi explicitado que o uso dos construtos especializados da URML e OCL do perfil R-OntoUML seria semelhante à forma usada na primeira modelagem. Entretanto, durante a construção do modelo, deveriam ser consideradas as meta-propriedades dos construtos da OntoUML e dos construtos do perfil R-OntoUML especializados dos conceitos da UFO-B e UFO-C. O Apêndice D contém o documento usado para a explicação das linguagens.

O encontro com o primeiro participante ocorreu em três momentos distintos, devido a restrições de horário. No primeiro encontro, foi explicado OntoUML, URML e OCL, levando a construção do primeiro modelo. No encontro seguinte, o participante

avaliou o primeiro modelo, para em seguida, receber a explicação da proposta desse trabalho e construir o segundo modelo. No último encontro, avaliou seu último modelo e comparou os seus dois modelos. Os encontros com os outros dois participantes foram executados em um período único, sem separação. Como os participantes aprenderam a URML antes da execução da segunda etapa, ao fazer o modelo em OntoUML e R-OntoUML havia a tendência do resultado ser melhor, devido à experiência da construção do primeiro modelo. Essa situação é uma possível ameaça à validade dos resultados da pesquisa, porém o modelo resultante não estava sendo avaliado. A percepção dos participantes do uso e características da UML com URML e OntoUML com R-OntoUML estava em avaliação, e a percepção de uso poderia ter sido favorável, indicando facilidade no uso da OntoUML e R-OntoUML. Caso alguma dúvida sobre a linguagem surgisse, os construtos eram explicados novamente, usando os mesmos exemplos anteriores. Como a maioria dos casos de dúvidas surgia para esclarecer dúvidas sobre a UFO, dada a quantidade grande de construtos e meta-propriedades, a repetição da explicação do construto ou exemplo usado solucionava as dúvidas. Foi verificada dificuldade de uso da OntoUML, o que pode ser confirmada nas respostas dos participantes para as perguntas de percepção de uso. Em casos em que a repetição não resolvesse, o participante explicava quais opções pensava aplicar no modelo, tendo como resposta qual resultado cada uma das opções trazia, a partir dos construtos considerados pelo participante e suas meta-propriedades. Dúvidas sobre o domínio eram discutidas com a preocupação de não criticar ou defender a utilização dos construtos das linguagens.

Após a conclusão dos modelos pelos participantes, estes foram revisados, com o objetivo de detectar possíveis erros de uso da linguagem, assim como interpretação errada do domínio. Com os modelos prontos, os participantes responderam seus questionários.

5.2 Resultados do estudo de caso com o primeiro participante

Os modelos construídos pelo primeiro participante se encontram no Anexo A e questionário respondido por ele no Anexo B.

5.2.1 Modelagem em UML e URML

Usando a UML, o primeiro participante modelou cinco classes: Aluguel, Carro, Alugador, Cobrança Provisória. Com quatro relacionamentos, a classe Aluguel está ligada a todas as outras classes. O conceito Aluguel concentrou quase todos os atributos do domínio. As regras de negócio foram modeladas de acordo com o domínio. A Figura 68 apresenta o modelo em UML e URML do primeiro participante.

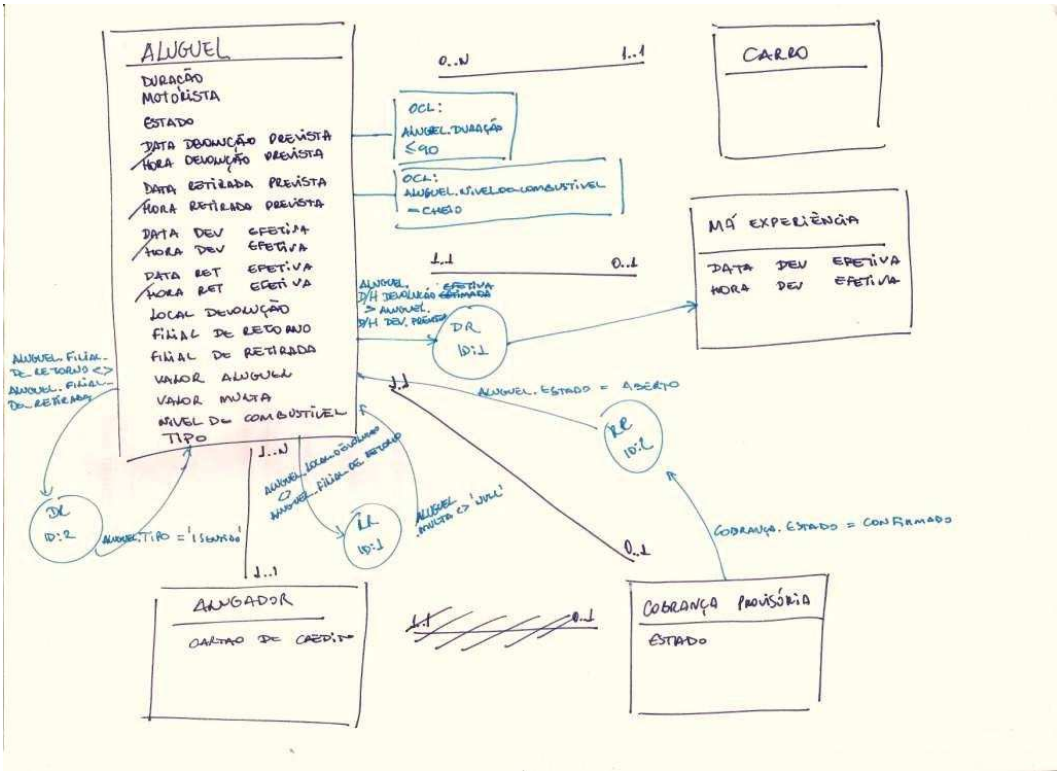


Figura 68 - Modelo em UML e URML do primeiro participante.

As Regras de Integridade foram relacionadas aos atributos de Aluguel: Duração máxima de noventa dias e Nível de Combustível como “Cheio”. A representação construída pelo participante explicita os conceitos dessas regras da forma descrita no domínio.

A representação da Regra de Derivação que demonstra como se classifica um Aluguel como uma Má Experiência usa os construtos das linguagens corretamente, assim como a Regra de Derivação que indica como um Aluguel é classificado como do tipo Aluguel de Um Só Sentido, utilizando o atributo Tipo de Aluguel. A Figura 69 apresenta esse trecho do modelo do participante.

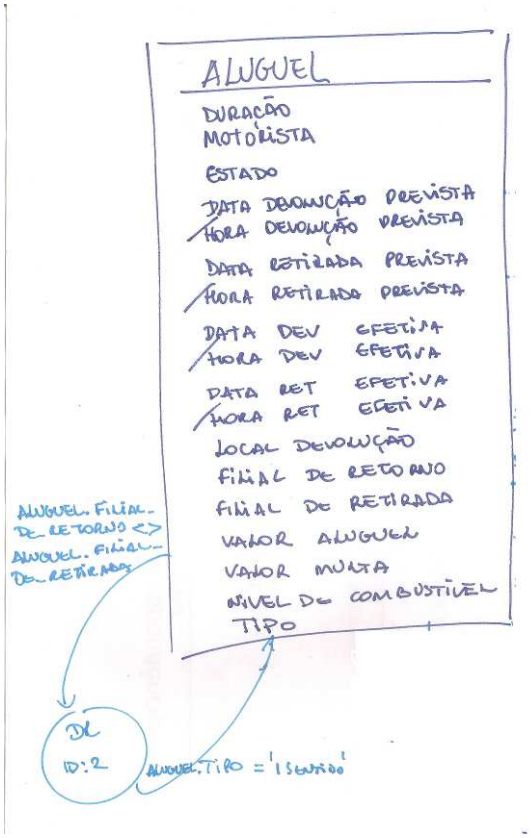


Figura 69 - Trecho do modelo, destacando a Regra de Derivação de Aluguel de Um Só Sentido.

A Regra de Reação da Cobrança Provisória, onde o Aluguel fica como “Aberto”, tem como o Estado da Cobrança “estar confirmado”, concluindo que o atributo Estado de Aluguel seja “Aberto”. A modelagem da aplicação de Multa manipula esse atributo de Aluguel, indicando que, caso o Local de Retorno seja diferente da Filial de Retorno prevista, o atributo Multa deve ser diferente de vazio.

Em seu questionário, o participante respondeu que este modelo continha todas as afirmativas corretas e relevantes, ou seja, era um modelo completo. Anotou uma observação, demonstrando a sua dificuldade para a representação da Cobrança, demorando para identificar a forma de modelar esse conceito.

Entretanto, o participante indicou que o modelo não era completamente válido, pois, segundo este participante, a regra de negócio “Se a cobrança estimada do aluguel for provisoriamente cobrada no cartão de crédito do alugador que é responsável pelo Aluguel, então é obrigatório que o Aluguel esteja aberto”. Da forma como foi modelado, obriga que o estado do aluguel seja sempre “aberto”, quando poderia (de acordo com o domínio) passar para estados seguintes (em curso, terminado). Isso se repetiu na pergunta final, onde o participante indicou a falta de informações sobre o

domínio para a correta modelagem de “cobrança provisória do aluguel e das condições de ocorrência da mesma”.

Em UML, o participante percebeu que a forma usada para modelar a segunda Regra de Reação, que identifica como o Aluguel fica em estado aberto, forçaria esse estado sempre ao Aluguel.

5.2.2 Modelagem em OntoUML e R-OntoUML

Ao elaborar o segundo modelo em OntoUML e R-OntoUML, o participante 1 identificou mais classes do que em seu primeiro modelo, além de outras diferenças em função do uso dos construtos UFO. A Figura 70 apresenta esse modelo.

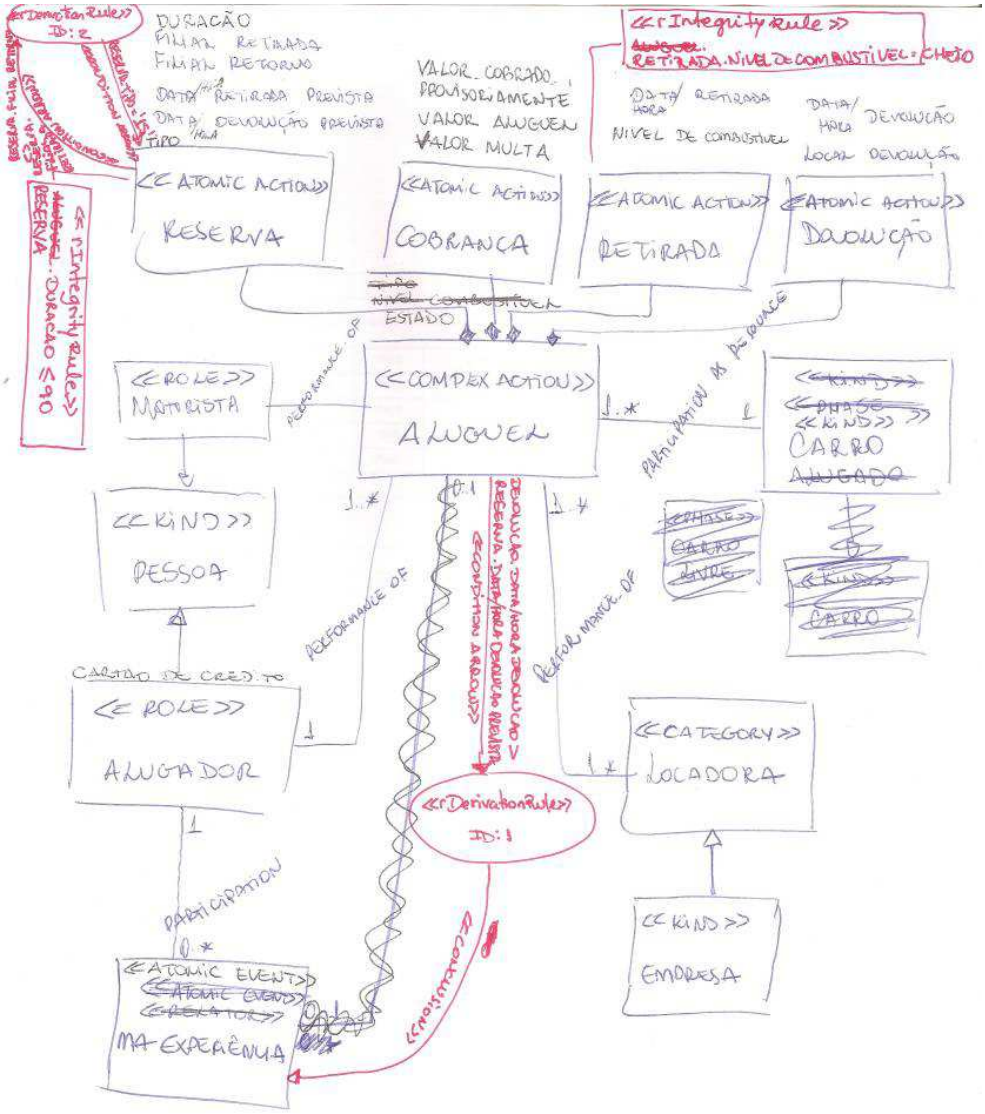


Figura 70 - Modelo em OntoUML e URML do primeiro participante.

Foram representadas doze classes. A *Ação Complexa* Aluguel é composta pelas *Ações Atômicas* Reserva, Cobrança, Retirada e Devolução, indicando que várias *Ações* são executadas durante o Aluguel. Aluguel teve a participação do recurso Carro (um *Kind*), sendo executado (relacionamentos “performanceof”, que ficaram com os símbolos “<<” e “>>” faltando, entretanto não deixam de ser estereótipos, que era a intenção de modelagem do participante) pelos *Papéis* Motorista e Alugador – ambas as

especializações do *Kind* Pessoa – , e pela *Categoria* Locadora: todos esses três conceitos fazem parte da execução do Aluguel. Os elementos que se relacionam com Aluguel da forma descrita estão destacados na Figura 71. Esse modelo apresenta modificações realizadas pelo participante durante a modelagem. O participante pensou em definir Carro Alugado como uma *Fase* do *Kind* Carro, com outra *Fase*, chamada Carro Livre, relacionada. Entretanto, o participante desistiu dessa representação do domínio, mantendo somente o *Kind* Carro.

O *Kind* Empresa é especialização da *Categoria* Locadora. A Má Experiência em um Aluguel foi modelada como um *Evento Atômico*, com a participação do Alugador.

O participante 1 poderia ter identificado quais participações (os agentes ou recursos) as *Ações Atômicas* Reserva, Cobrança, Retirada e Devolução teriam, entretanto isto não ficou no modelo final. Entende-se que as participações relacionadas ao conceito Aluguel também fazem parte das *Ações Atômicas* que compõem esse conceito. No modelo 2 resultante, porém, não fica explícito se todas as participações ocorrem em todas as *Ações*, por exemplo.

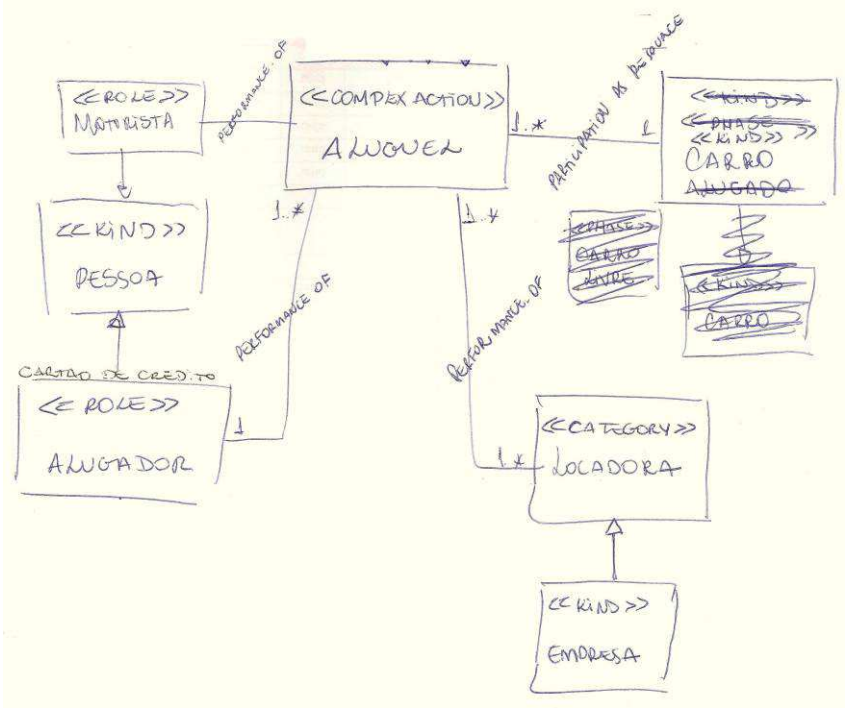


Figura 71 - Elementos relacionados com a Ação Complexa Aluguel.

A relação entre Aluguel e Má Experiência foi modelada com a Regra de Derivação de ID 1, indicando que se (na condição de) o momento de devolução for posterior à data de devolução prevista, então se chega à conclusão da existência da Má Experiência. Essa representação é interessante, pois não relaciona os conceitos por um

construto baseado na UFO, usando o recurso introduzido pela R-OntoUML para a representação da conceitualização da Regra. A Figura 72 apresenta esse trecho do modelo. A figura também explicita uma mudança na modelagem do participante. Inicialmente, a relação entre Aluguel e Má Experiência seria com um relacionamento, mas o modelo final usou a própria Regra de Fundamentação de Reação para indicar que os elementos tinham uma relação. O participante usou os construtos R-OntoUML para representar seu entendimento do domínio.

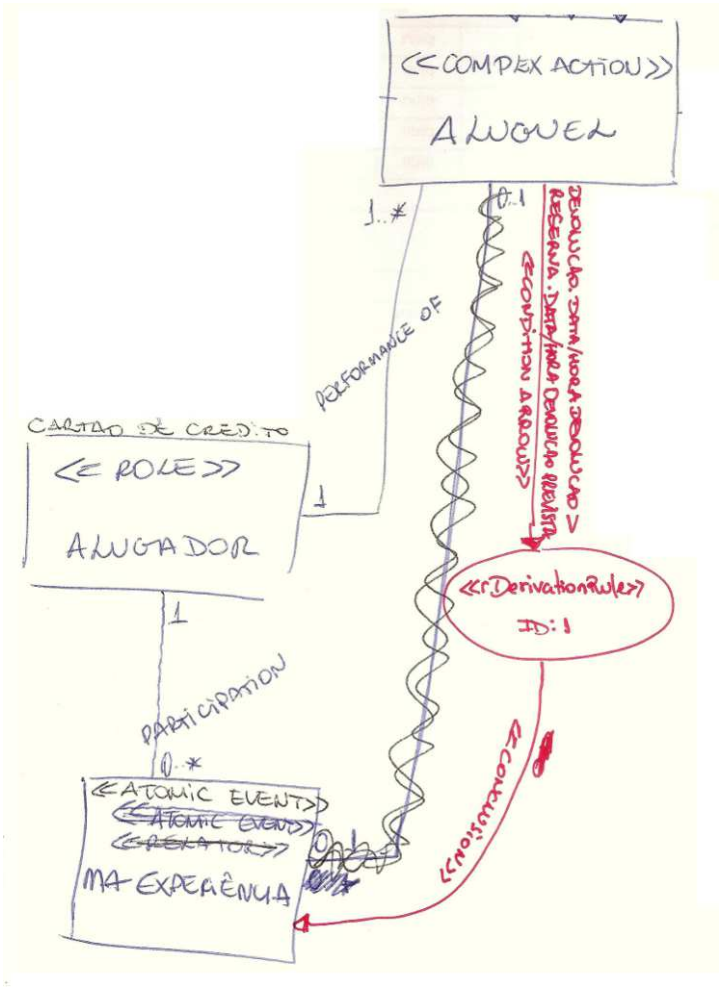


Figura 72 – Modelagem de Regra de Derivação que define Má Experiência.

A Regra de Integridade que indica que o Nível de Combustível deve estar “Cheio” está representada pela Regra de Fundamentação de Integridade relacionada diretamente com a *Ação Atômica* Retirada, classe que teve o atributo de Nível de Combustível atrelado.

A Regra de Integridade que limita da duração do Aluguel a noventa dias está relacionada ao conceito de Reserva. Essa construção demonstra que a ideia da restrição da Duração está relacionada à Reserva feita pelo Alugador, não ao Aluguel em si: caso

o Aluguel ultrapasse noventa dias, isso é permitido por essa modelagem, enquanto que o ato da Reserva impede do Alugador reservar o carro por mais tempo. Com isso, seria possível a representação de uma Regra de Reação que indicaria o que fazer, caso o Aluguel tenha Duração maior que a pré-definida na Reserva. Apesar de isso não estar descrito no domínio, a representação do participante apresenta um uso relevante dos construtos propostos, conforme descrito.

A Regra de Derivação sobre o Aluguel de Um Só Sentido foi modelada de maneira diferente em relação ao primeiro modelo UML deste participante. O atributo Tipo assume o valor que indica um só sentido. Entretanto, ele está ligado a Reserva, registrando que no momento da Reserva as informações de Filial de Retorno e Filial de Retirada são comparadas.

Inicialmente, as Regras de Reação estavam modeladas incorretamente: toda Regra de Reação obrigatoriamente termina em um *Evento* ou especialização e essa meta-propriedade foi violada na modelagem do participante 1. Em sua primeira modelagem, a Regra de Reação que especifica como a Multa é aplicada tem a sua condição ligada a Aluguel. É comparado se a Filial de Retorno é diferente do Local de Devolução efetivo, chegando a *Ação Atômica* Cobrança, que terá o valor de Multa diferente de vazio. A informação se o Aluguel está aberto, fechado, em curso ou terminado foi representada como uma propriedade de Aluguel, com o nome Estado. A regra ficou com formato semelhante, representado na Regra de Fundamentação de Reação de ID 2, onde, ligando a *Ação Atômica* Cobrança, como condição, e a *Ação Complexa* Aluguel, onde, se o Valor Cobrado Provisoriamente seja maior que zero, o Estado do Aluguel terá o valor “Aberto”. Essa representação mantém o estado do Aluguel como um atributo do conceito. A Figura 73 apresenta o trecho das Regras de Reação.

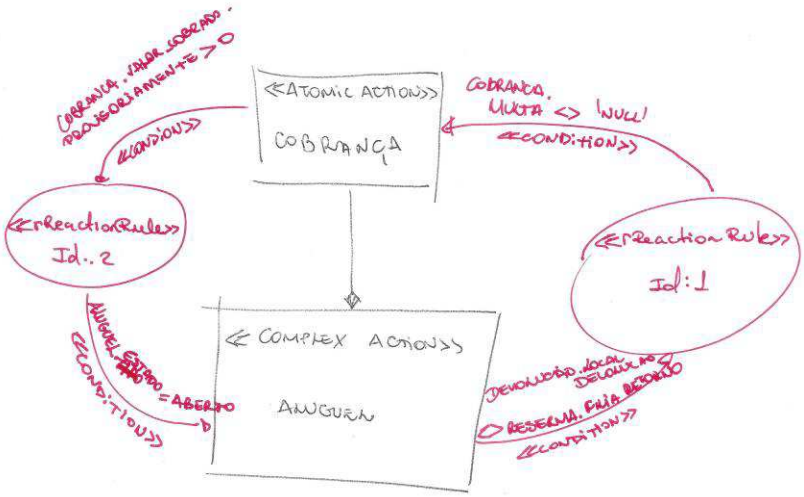


Figura 73 – Trecho das Regras de Reação que não segue as meta-propriedades da R-OntoUML.

Ao ser questionado sobre a obrigatoriedade da Regra de Reação, o participante alterou o modelo. Com relação à primeira Regra de Reação, usou uma *Seta de Evento*, diretamente ao conceito Cobrança. Indicou que a semântica da Regra ficou refletida em sua modelagem: uma Cobrança ocorre devido à diferença entre o Local Efetivo de Devolução e o Local de Devolução inicialmente combinado, com um Valor de Multa atrelado a essa cobrança.

Na representação da segunda Regra de Reação, percebeu que a mesma solução não era possível, pois não fazia sentido o Aluguel ser o *Evento* de reação da regra. Para resolver a dúvida sobre o domínio, foi indicado que ocorre uma abertura do Aluguel por parte da Empresa de Aluguel. Com essa informação, o participante acrescentou o *Evento Atômico* Abertura (que compõe o *Evento Complexo* Aluguel), ligando a Regra de Reação a partir da *Seta de Evento*. Essas alterações refletiram os conceitos do domínio e são aceitas pelos construtos da R-OntoUML.

A Figura 74 apresenta o mesmo trecho da Figura 73 revisado.

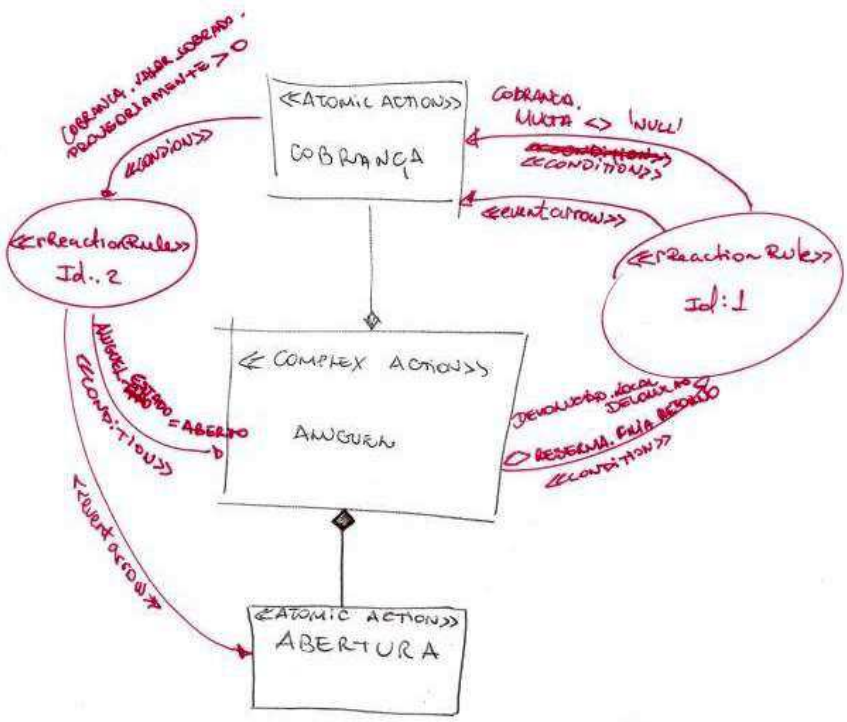


Figura 74 - Trecho das Regras de Reação revisadas.

Nas respostas do questionário, o participante respondeu que o modelo estava completo e percebeu que poucas afirmativas não são corretas ou relevantes ao domínio: “mesmo caso da primeira modelagem”, onde o participante se referiu ao estado sempre aberto de Aluguel. Com isso, continuou com a mesma interpretação do modelo em UML e URML: “Mesmo caso da primeira modelagem”, onde o estado de Aluguel será sempre o mesmo. Novamente, se repete na pergunta final, onde indica “Mesmo caso da primeira modelagem”, assim como a relação disso com o estado do Aluguel.

5.2.3 Avaliação cruzada entre participantes e percepção de uso da linguagem proposta

A segunda etapa do estudo de caso do primeiro participante está no Anexo C.

O primeiro participante avaliou o modelo do segundo participante. A avaliação do modelo UML e URML descreveu o modelo como completo e com poucas sentenças incorretas e irrelevantes ao domínio. Segundo o participante, o modelo não é completamente válido, pois “obriga que o conceito Carro esteja vinculado a apenas um Aluguel e que o nível de combustível esteja sempre cheio”. Também não havia

percebido em sua modelagem inicial que existe uma diferença entre o Carro físico e o Carro que participa do Aluguel, que foi modelado pelo segundo participante.

O participante avaliou o modelo OntoUML e R-OntoUML como completo e com poucas afirmativas que não são corretas ou relevantes ao domínio. Ele percebeu uma diferença entre o modelo UML e o modelo OntoUML do segundo participante. Enquanto no primeiro modelo, as fases (o nome usado pelo primeiro participante durante a avaliação do seu próprio modelo foi "Estado", nome alterado na avaliação do modelo do segundo participante, usando "Fase") de um Aluguel são representadas como um atributo, o modelo OntoUML possui a diferenciação entre as várias fases, usando o construto próprio (*Phase*) da OntoUML. Essa modelagem ficou diferente inclusive do modelo do primeiro participante, que manteve a ideia do atributo no modelo em OntoUML para representar as *Fases* de Aluguel. O primeiro participante questionou em sua modelagem que a representação obriga que o Aluguel esteja sempre aberto. Ao ser questionado sobre qual o significado de sua resposta, o participante respondeu que, na modelagem do segundo participante que usa a ideia das *Fases*, ficou faltando a relação dos *Eventos* representados no modelo (Má Experiência e Multa) e a relação com as *Fases* de Aluguel: em quais fases o acontecimento pode acontecer, se um *Evento* dispara uma *Fase*. Não questionou a obrigação de o Aluguel estar sempre Aberto, o que, com a representação como *Fase*, não é mais verdade. Como *Fase* ocorre em um determinado período, o Aberto dentro da representação da Regra é temporário, diferente do modelo em UML. Nas regras usadas nesse estudo de caso, há somente a regra que descreve a *fase* "Aberto"; seria necessário buscar mais informações sobre o domínio para identificar como ocorrem as outras *Fases* de Aluguel. Entretanto, a solução proposta pelo participante deveria alterar o entendimento do conceito Aluguel, pois é um *perdurant*, enquanto que, as *fases* criadas são *endurants*: como os subtipos de Aluguel são *perdurants*, Aluguel também o deveria ser.

Na comparação dos dois modelos, o participante respondeu que os dois modelos eram completos. Porém, encontrou dificuldades em avaliar a validade do modelo em OntoUML, devido à quantidade de meta-propriedades dos construtos, requerendo "uma compreensão muito profunda desses construtos e dos relacionamentos entre eles". Segundo ele, o "modelo UML contém algumas informações além das regras que são mais facilmente identificadas".

O participante identificou a representação em UML e URML como sendo de maior generalidade e em OntoUML e R-OntoUML como sendo mais precisa.

Em seguida, o questionário de percepção de uso foi respondido. O participante 1 discordou em parte que a técnica da OntoUML e R-OntoUML é desconfortável de usar. Discordou que a técnica é frustrante, concordando que exige muito esforço mental para ser usada. O participante se disse neutro com relação se a técnica é clara e compreensível, e se é fácil de usar.

5.3 Resultados do estudo de caso com o segundo participante

Os modelos construídos pelo segundo participante se encontram no Anexo D e questionário respondido por ele no Anexo E.

5.3.1 Modelagem em UML e URML

A modelagem de em UML e URML do segundo participante gerou nove classes: Aluguel, Um Só Sentido (como o aluguel de um só sentido, referenciado na segunda regra de derivação; uma especialização de Aluguel), Má Experiência, Multa, Local de Devolução, Carro Alugado, Alugador, Cobrança e Cobrança Provisória (especialização de Cobrança). A Figura 75 apresenta o modelo construído pelo segundo participante

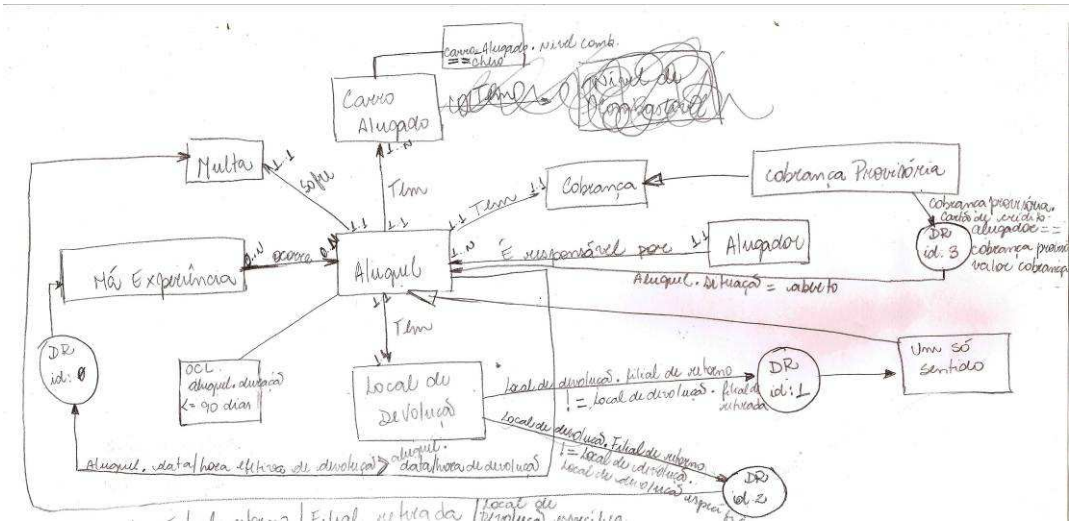


Figura 75 - Modelo em UML e URML do segundo participantes.

Aluguel tem os atributos Duração, Motorista, Situação (com o domínio fechado, aberto, em curso, terminado), Filial de Retorno, Filial de Retirada, Data/Hora de Devolução, Data/Hora Efetiva de Devolução, Data/Hora de Retirada, e Data/Hora

Efetiva de Retirada. Local Devolução tem Filial de Retorno, Filial de Retirada e Local de Devolução Específica. Multa tem o atributo Valor. Alugador tem os atributos Motorista e Cartão de Crédito. Carro Alugado tem atributo Nível de Combustível. Má Experiência tem atributo Data/Hora Efetiva de Devolução. Cobrança tem atributo Alugador. Cobrança Provisória tem Cartão de Crédito do Alugador e Valor da Cobrança.

Os seguintes relacionamentos foram representados: Aluguel tem Local de Devolução, Aluguel tem Cobrança, Aluguel sofre Multa, Aluguel tem Carro Alugado, Má Experiência ocorre em Aluguel, Alugador é responsável por Aluguel.

A primeira Regra de Integridade está relacionada ao atributo Duração de Aluguel, enquanto que a segunda Regra está relacionada ao atributo Nível Combustível de Carro Alugado.

A primeira Regra de Derivação liga Aluguel e Má Experiência, explicitando como uma Má Experiência é derivada no domínio. A segunda Regra de Derivação liga Local de Devolução e a especialização de Aluguel chamada Um Só Sentido.

A primeira Regra de Reação liga Local de Devolução a Multa, indicando que caso a Data/Hora de Devolução for maior do que a prevista, a Multa deve ser aplicada. A segunda Regra de Reação liga Cobrança Provisória e Aluguel, indicando que o Aluguel deve estar Aberto.

Respondendo o questionário de avaliação, o participante identificou que o modelo gerado é completo, entretanto, possui poucas frases que não são corretas e relevantes ao domínio. Destacou o conceito Valor da Cobrança, que teve dificuldades em modelar. Não sentiu falta de nenhuma informação nas regras para o entendimento do domínio.

5.3.2 Modelagem em OntoUML e R-OntoUML

O modelo construído em OntoUML e R-OntoUML manteve um conjunto de conceitos. Aluguel foi mantido, modelado como um *Relator*. Carro Alugado também foi mantido, modelado como um *Papel*, uma especialização de um novo conceito modelado, o *Kind* Carro. Cobrança foi mantida, modelada como um *Kind*, assim como sua especialização Cobrança Provisória, um *Mode* de Cobrança. Alugador foi modelado em OntoUML como um *Papel*, especializando um novo conceito, o *Kind* Cliente. O

conceito “Um Só Sentido”, representando o aluguel de um só sentido, foi modelado como um *Mode* de Aluguel. Multa foi modelada como um *Evento Atômico*, enquanto que a Má Experiência foi modelada como um *Evento Complexo*. Local de Devolução foi modelado como um *Kind*, enquanto que o atributo Duração de Aluguel virou uma classe classificada como uma *Quantidade*. Visando a representação da Regra de Reação que descreve a Cobrança Provisória fazendo com que o Aluguel fique em Aberto, o *Evento Complexo* Situação foi modelado (lembrando que toda Regra de Fundamentação de Reação deve terminar em um Evento).

Novos elementos modelados merecem destaque. Conforme destacado pelo primeiro participante ao avaliar esse modelo, descrito na seção 5.2.3, as fases de Aluguel foram modeladas. Essa representação acrescenta semântica ao modelo não presente na UML, principalmente, devido à dinâmica que as meta-propriedades do construto *Fase* trazem. As classes Terminado, Fechado, Aberto e Em Curso são especialização de Aluguel, representadas como *Fase*, ou seja, um Aluguel deve assumir no mínimo uma dessas fases e no máximo uma. É possível, com essa informação, interpretar que um Aluguel pode passar de uma *Fase* para outra. A Figura 76 apresenta o trecho do modelo das Fases de Aluguel.

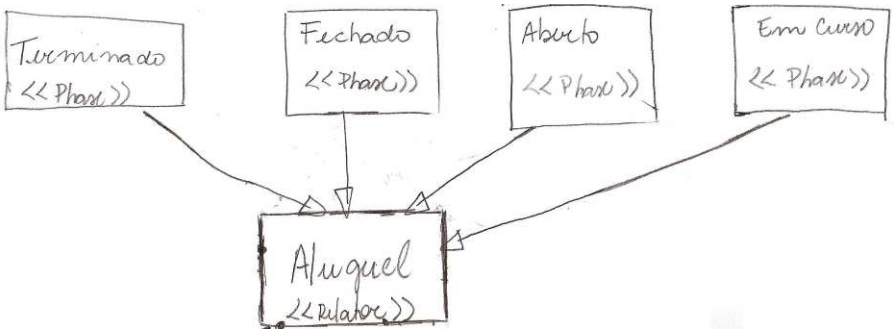


Figura 76 – Trecho que representa as diferentes fases de Aluguel.

Antes da análise dos relacionamentos, foi necessário aproveitar a discussão das fases de Aluguel para explorarmos a segunda Regra de Reação, que rege a classificação de um Aluguel na fase Aberto, a partir de uma Cobrança Provisória. Conforme também foi destacado pelo primeiro participante em sua avaliação desse modelo, somente a representação da Regra de Reação que identifica como um Aluguel assume a *Fase* “Aberto” foi modelada, pois as outras regras que regeriam o restante das classificações nas diferentes fases não estão presentes no domínio. Seria necessário questionar o especialista do domínio para obter as informações necessárias para a representação do

restante das regras. Da forma atual que está construído, o domínio não está completamente representado, conforme destacado também pelo primeiro participante.

O segundo participante encontrou dificuldades para modelar a segunda Regra de Reação do domínio. Demorou um período mais longo para decidir como construir no modelo a Regra, questionando a necessidade de uma Regra de Reação terminar em no mínimo um Evento. A solução encontrada foi a criação do *Evento Complexo* Situação, que tem a participação de um Carro, indicando em OCL que o atributo Status de Situação é igual a Aberto. A Figura 77 apresenta esse trecho do modelo.

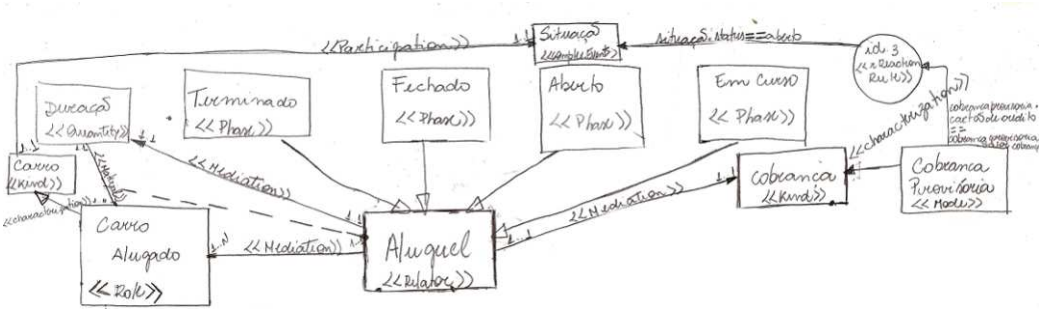


Figura 77 - Trecho do modelo que apresenta a Regra de Reação que indica que o Aluguel está Aberto.

O participante poderia ter questionado mais sobre o domínio, para tentar entendê-lo melhor, além da dificuldade do uso da linguagem. O especialista poderia explicar a ocorrência da atividade abertura de aluguel, executada pela filial responsável pelo mesmo. Com essa informação, poderia modelar a Regra de Reação terminando em um *Evento Atômico* Abertura de Aluguel, que é executado (relacionamento "performanceOf") pela Filial com a participação como recurso (relacionamento "participationAsResource") de Aluguel. Outra seta sairia dessa Regra de Reação, indo para a *Fase* Aberto, indicando que ela é uma Pós-Condição dessa Regra. Essa seria uma representação mais completa do domínio, porém, as Regras disponibilizadas aos participantes estavam limitadas a poucas informações.

Os atributos ficaram distribuídos de forma semelhante ao modelo em UML e URML. Diferente do primeiro modelo, o participante modelou um conjunto maior de relacionamentos.

Ao classificar um Aluguel como um *Relator*, a derivação de relacionamentos *Materiais* foi explicitada. Uma dessas relações foi o relacionamento *Material* entre Carro Alugado e Alugador, com os relacionamentos *Mediação* entre Aluguel com Carro Alugado e Aluguel com Alugador. As cardinalidades usadas nessa estrutura explicitam parte da semântica do domínio. O relacionamento entre Aluguel e Carro Alugado do

modelo em UML foi modelado como um relacionamento de *Mediação* entre os dois, mantendo a cardinalidade: um Aluguel pode ter um ou mais Carros Alugados, e um Carro Alugado faz parte de um e somente um Aluguel. O relacionamento entre Carro Alugado e Alugador apresenta a relação *Material* entre os dois conceitos, indicando que o Alugador pode ter um ou mais Carros Alugados ao mesmo tempo, e o Carro Alugado tem um e somente um Alugador. O relacionamento de *Mediação* entre Aluguel e Alugador demonstra que um Alugador pode ter um ou mais Aluguéis, enquanto que um Aluguel é de um e somente um Alugador. Toda essa semântica do domínio ficou implícita na modelagem em UML, sendo explicitada pelo modelo em OntoUML. As meta-propriedades dos construtos guiaram o participante na construção dessa estrutura.

Outro relacionamento *Material* derivado do *Relator* Aluguel é entre o *Kind* Local de Devolução e o Papel Carro Alugado. Não havia essa relação direta entre os dois conceitos no modelo em UML, e foi derivado do *Relator*; o relacionamento já existente era entre Aluguel e Local de Devolução, representado em OntoUML como um relacionamento de *Mediação*, com a mesma cardinalidade do primeiro modelo. O relacionamento *Material* entre Local de Devolução e Carro Alugado não existia inicialmente, explicitando que um Carro Alugado possui um e somente um Local de Devolução e o Local de Devolução pode possuir um ou vários Carros Alugados. O relacionamento de *Mediação* entre Aluguel e Carro Alugado também não estava presente em UML, e possui a mesma semântica apresentada no parágrafo anterior.

O terceiro relacionamento *Material* derivado de Aluguel é entre a *Quantidade* Duração e o *Papel* Carro Alugado. Essa estrutura não havia sido representada no primeiro modelo, pois a classe Duração havia sido representada como um atributo de Aluguel. Esse relacionamento *Material* indica que uma Duração pode ter um ou mais Carros Alugados, enquanto que um Carro Alugado tem uma e somente uma Duração. O relacionamento de *Medição* entre Aluguel e Duração indica que um Aluguel tem uma e somente uma Duração, e uma Duração possui um e somente um Aluguel. O relacionamento de *Mediação* entre Aluguel e Carro Alugado possui a mesma semântica já apresentada: um Aluguel pode ter um ou mais Carros Alugados, e um Carro Alugado faz parte de um e somente um Aluguel.

O último relacionamento *Material* derivado de Aluguel ocorre entre Cobrança e Carro Alugado (apesar do erro do relacionamento de Derivação que está ligado ao *Kind* Cobrança; como existe somente um *Relator* no modelo, o Aluguel será considerado

como o elemento que derivou o relacionamento entre os dois conceitos). No modelo em UML, o relacionamento entre Aluguel e Cobrança existia, entretanto sem a semântica da *Mediação*. As cardinalidades não foram alteradas. O relacionamento *Material* entre Cobrança e Carro Alugado traz mais informações sobre o domínio: uma Cobrança ocorre com um ou mais Carros Alugados, enquanto que um Carro Alugado se relaciona com uma e somente uma Cobrança. Completando a estrutura que acompanha um *Relator*, o relacionamento de *Mediação* entre Aluguel e Carro Alugado é igual aos relacionamentos apresentados anteriormente.

Esse conjunto de relacionamentos *Materiais* demonstra que os conceitos centrais desse domínio são o Aluguel e Carro Alugado.

Existem outros dois relacionamentos que merecem destaque: os *Modes* e seus relacionamentos de *Characterization*. Cobrança Provisória foi modelada como um *Mode* que acrescenta características (o relacionamento *Characterization* entre os dois) ao *Kind* Cobrança, sendo existencialmente dependente dele. O *Mode* Um Só Sentido acrescenta características ao *Relator* Aluguel, com a presença do relacionamento *Characterization* entre eles. A representação do elemento Um Só Sentido apresentou uma semântica diferente da usada pelos outros participantes, que usaram atributos para diferenciar os tipos de Aluguel.

As Regras de Integridade foram modeladas de forma semelhante ao modelo em UML, relacionando-se aos mesmos conceitos.

A primeira Regra de Derivação foi modelada de forma semelhante ao modelo em UML. Entretanto, um destaque que se pode fazer é a criação do relacionamento de *Participação* entre Má experiência e Aluguel. Como um Aluguel é um *Relator*, o que caracteriza Aluguel como um *Moment Universal* e não um *Subtantial*, esse relacionamento não poderia estar no modelo. A Regra de Derivação entre os dois elementos já seria o suficiente para ele estar correto.

A segunda Regra de Derivação também foi modelada de forma semelhante do primeiro modelo, com o diferencial do conceito Um Só Sentido estar modelado como um *Mode*. Com essa Regra, é possível identificar como o conceito Um Só Sentido ocorre no domínio modelado, explicitando que, além de ser existencialmente dependente de Aluguel e acrescentar propriedades a ele, o conceito Um Só Sentido é encontrado no domínio quando a Filial de Retorno é diferente da Filial de Retirada.

Diferente da já mencionada segunda Regra de Reação, a primeira Regra desse tipo não apresentou dificuldades ao participante. Durante a modelagem de Regra de Reação que descreve a Multa, o participante expressou que a construção da Regra se encaixou com os conceitos previamente modelados: a Reação se ligou com o previamente modelado Evento Atômico Multa. Dessa forma, o participante conseguiu atender o pré-requisito da Regra de Fundamentação de Reação que lhe apresentou dificuldades na representação da Regra de Reação relacionada ao estado Aberto de Aluguel. A Figura 78 apresenta esse trecho do modelo.

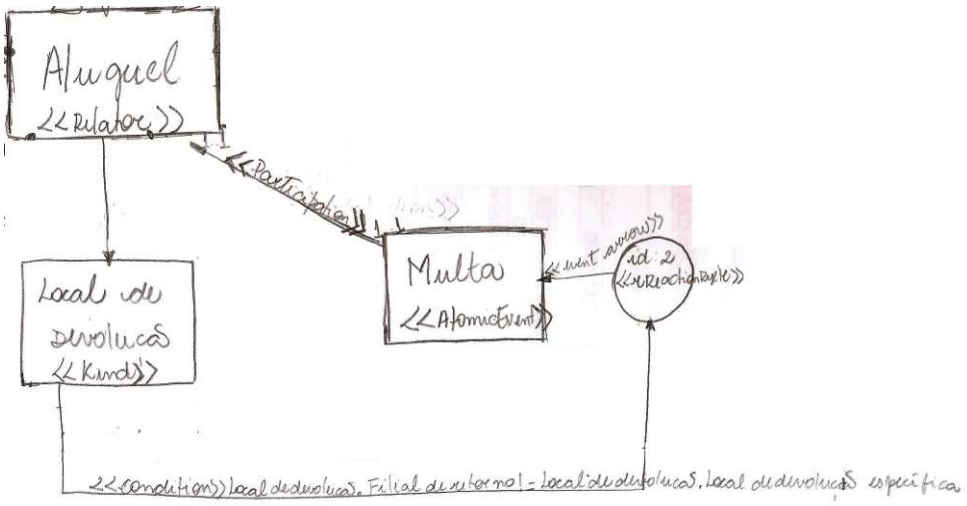


Figura 78 - Trecho do modelo que representa a Regra de Reação da Multa.

Depois da modelagem terminada, o participante respondeu o questionário. Com relação à OntoUML e R-OntoUML, respondeu que o modelo também é completo, e com afirmativas incorretas ou irrelevantes ao domínio, com a mesma dificuldade de representação do Valor da Cobrança: “Assim como em UML e URML, senti falta do valor da cobrança”.

Em seguida, a comparação entre as duas modelagens foi realizada. Com relação à Completeza, indicou que os dois modelos contêm “todas as informações relatadas nas regras”. Destacou que isso se deve a simplicidade do domínio analisado. Entretanto, ao avaliar a Validade, indicou que os modelos possuem informações não corretas ou relevantes ao domínio, destacando o conceito Valor da Cobrança. Com relação a Generalidade e Precisão, indicou que o uso da OntoUML e R-OntoUML traz mais semântica ao modelo: “As restrições de relacionamentos entre os construtos permitem retirar grande parte das ambiguidades na representação”. Essa visão se dá devido a representação de Aluguel como *Relator*, o que guiou o segundo participante a modelar

um conjunto maior de relacionamento do tipo *Material*, explicitando conceitos antes não modelados.

5.3.3 Avaliação cruzada entre participantes e percepção de uso da linguagem proposta

A segunda etapa do estudo de caso do segundo participante está no Anexo F.

O segundo participante avaliou os modelos do terceiro participante. Considerou o modelo UML e URML completo, mas com poucas afirmativas incorretas ou irrelevantes ao domínio. Destacou um atributo de Alugador, chamado Clube de Fidelidade. No entanto, esse elemento não está fora do domínio. Apesar da solicitação de não considerar o resumo, somente as Regras e o item “Informações necessárias”, o terceiro participante modelou a informação usando o resumo: “Clientes que se afiliam acumulam pontos que podem ser usado para pagar por aluguéis”. Ou seja, o modelo continua válido.

Na avaliação do modelo OntoUML e R-OntoUML, o segundo participante o descreveu como completo e não totalmente válido, marcando a terceira opção. Destacou os seguintes conceitos, inseridos “para melhor representação inserção dos conceitos da UFO”: Cliente, AlugadorInd, AlugadorEmpre, Organização e Pessoa. Novamente, o terceiro participante modelou elementos presentes no resumo do domínio, o que não está fora do escopo do mesmo. Considerando somente essas restrições apontadas, o modelo em OntoUML e R-OntoUML também é valido.

Ao comparar os dois modelos, foram considerados completos, com uma ressalva para a modelagem em OntoUML, que “exigiu definir melhor cada elemento do modelo. a fim de que fosse possível atribuir os estereótipos da UFO”. Com o uso dos conceitos da UFO, um conjunto maior de informações é explicitada, o que não ocorre ao utilizar a UML, pois não há a presença das meta-propriedades da UFO. Avaliando a validade dos modelos, o participante destacou a quantidade maior de informações no modelo construído com OntoUML e R-OntoUML. O uso dos estereótipos da UFO e necessidade de atribuí-los ao conceito do domínio, “o modelo precisou estar o menos ambíguo possível”, para respeitar as meta-propriedades dos elementos da UFO.

Quando comparando a Generalidade e Precisão, o segundo participante respondeu que a representação em UML e URML teve maior Generalidade, enquanto que o modelo construído em OntoUML e R-OntoUML foi o mais preciso.

Finalizando a sua avaliação, o segundo participante expressou a sua dificuldade no uso da OntoUML e R-OntoUML na modelagem. O participante concorda fortemente que a técnica é desconfortável de usar, mas discorda fortemente que seu uso foi frustrante. Concorda fortemente que o uso da técnica exigiu muito esforço mental, concorda em parte que a técnica é clara e compreensível, porém, considerando tudo, discorda fortemente que a técnica é fácil de usar. Em seguida, descreveu a sua experiência no uso da OntoUML e R-OntoUML, declarando que ficou “clara a qualidade semântica nos modelos onde a OntoUML e R-OntoUML foram aplicadas”. Reforçou a dificuldade de uso dos dois: seu aprendizado e uso não são triviais, “principalmente em pouco tempo”. Para o participante, com experiência na modelagem com essas ferramentas, é possível retratar “com uma carga semântica mais próxima do real”, enquanto que, com pouco conhecimento, “não é possível garantir que modelagem corresponderá às expectativas”.

5.4 Resultados do estudo de caso com o terceiro participante

Os modelos construídos pelo terceiro participante se encontram no Anexo G e questionário respondido por ele no Anexo H.

5.4.1 Modelagem em UML e URML

O terceiro participante modelou sete classes em UML e URML: Aluguel, Carro, a sua especialização Carro Alugado, Alugador, Má Experiência, Motorista e Filial. Aluguel foi modelado como uma classe associativa entre Carro Alugado e Alugador, indicando que o relacionamento entre os dois conceitos é de um para um. A Figura 79 apresenta o modelo UML e URML do terceiro participante

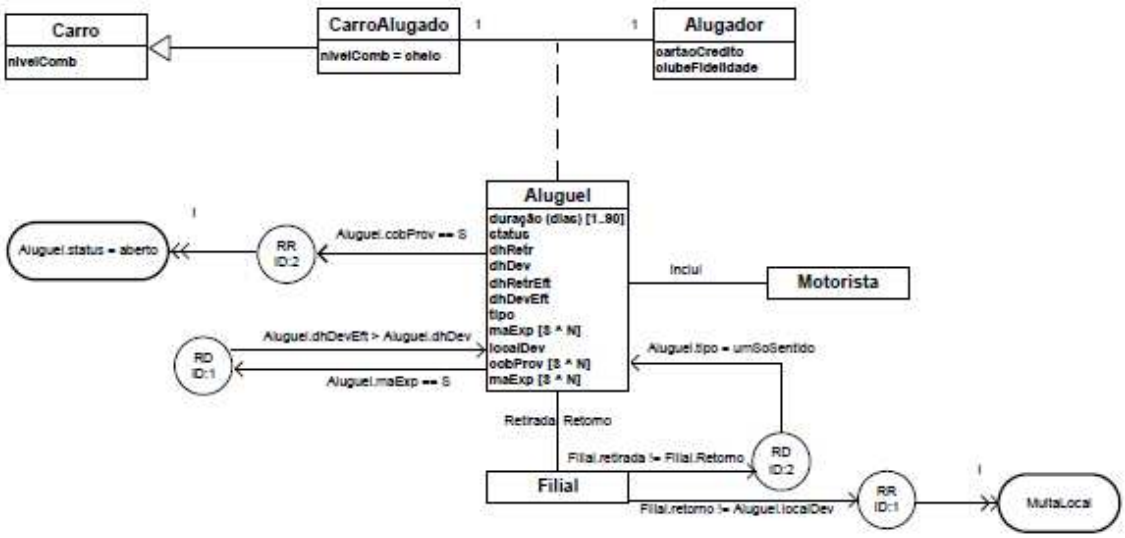


Figura 79 - Modelo UML e URML do terceiro participante.

Essa representação destacou-se dos outros participantes não só pelo uso da classe associativa, mas também da inclusão do elemento da Atividade “MultaLocal”, usando construtos do diagrama de atividades da UML, permitido pela URML. O participante usou esses elementos para modelar a primeira Regra de Reação. Essa representação mistura construtos de diagramas diferentes, acarretando em uma mistura de meta-propriedades entre elementos (desde sua figura, como sua semântica e sintática). DIJKSTRA (1982) discute o que ele chama de separação de interesses (uma tradução livre de “Separation of Concerns”), onde a mistura de diferentes visões pode dificultar o entendimento do modelo, sobrecarregando o mesmo com mistura de informações. A representação está correta, entretanto, pode trazer implicações no entendimento do domínio por uma analista que não é especialista na linguagem, ou pelo especialista do domínio, que não conhece a tecnologia.

De forma semelhante ao primeiro participante, a maioria dos atributos ficou atrelada ao conceito Aluguel, incluindo a informação da ocorrência ou não de Má Experiência (atributo “maExp”) e da Cobrança Provisória (atributo “cobProv”).

O atributo Nível de Combustível ficou relacionado a Carro e herdado pelo Carro Alugado. O participante usou esse atributo para representar a segunda Regra de Integridade, com a expressão “nivelComb=cheio”. Essa representação ficou diferente do esperado pela Regra, que indica que “é obrigatório que o nível de combustível do carro alugado esteja cheio”. O uso da expressão indica que o valor padrão (*default*) do atributo é cheio, mas não obriga que o valor mantenha esse valor sempre. Ao ser

questionado sobre a representação, o participante decidiu não alterá-lo, pois a sua interpretação foi “que o ‘carro alugado’ é uma especialização de ‘carro’ que tem o valor ‘cheio’ como default para o atributo ‘nível de combustível’”. Essa representação está mais próxima das regras do tipo Deônticas, onde as regras podem ser violadas: o valor padrão pode ser alterado em um dado momento, diferente da representação da Regra feita pelos outros participantes. A Figura 80 apresenta o conceito Carro Alugado.

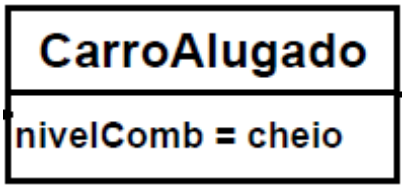


Figura 80 - Trecho da classe CarroAlugado.

A segunda Regra de Integridade também foi representada de forma diferente, usando o construto de domínio em UML: o uso de colchetes para indicar as possíveis instâncias do domínio para um atributo. Com essa estrutura, o terceiro participante escreveu “[1..90]” junto ao atributo Duração da classe Aluguel, explicitando que os possíveis valores desse atributo obedecem a Regra de Integridade sendo modelada.

As representações de Regras de Integridade feitas pelo terceiro participante demonstraram a necessidade de uma maior discussão sobre esse tipo de Regra, o que foge ao escopo desse trabalho.

A primeira Regra de Derivação foi modelada usando somente a UML. O participante criou um relacionamento entre Má Experiência e Aluguel, escrevendo no seu relacionamento a condição da regra: data/hora de devolução efetiva é maior que a data/hora de devolução planejada, com o texto “dhDevEft > dhDev”. Essa representação fere as regras da UML, pois o texto usado em relacionamentos é usado para nomear o mesmo, ou para identificar o papel de um elemento no relacionamento. Ao ser questionado do uso da regra, o participante decidiu alterar a representação, usando os construtos da URML em sua representação, resultando na Regra de Derivação de identificador 1. A Figura 81 apresenta o trecho da Regra de Derivação Má Experiência antes da alteração.

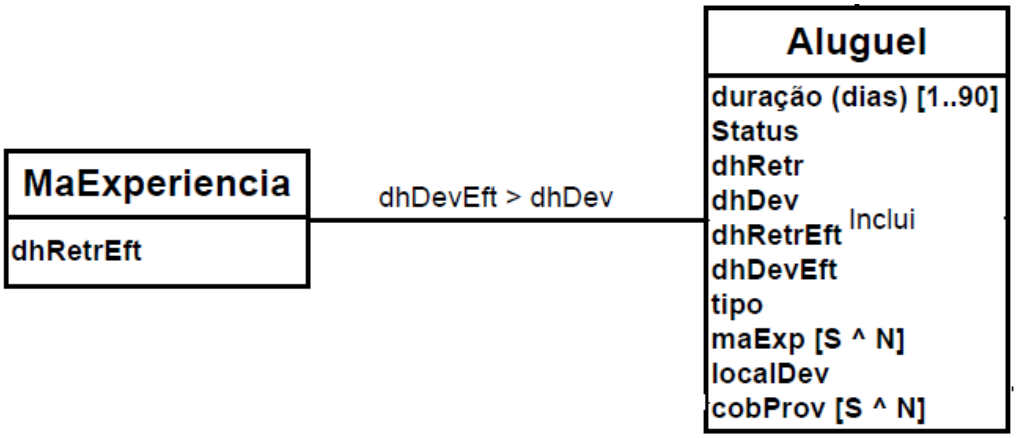


Figura 81 - Trecho da Regra de Derivação Má Experiência.

A segunda Regra de Derivação foi representada usando o atributo Tipo da classe Aluguel, seguindo a estrutura da URML.

O participante alterou a representação da segunda Regra de Reação modelada, depois de questionado sobre as outras Regras. A representação inicial utiliza os construtos da Regra de Reação, alterando o valor do atributo “maExp” de Aluguel. Na revisão das outras regras, o participante fez uma alteração, representando a Má Experiência como uma atividade. Dessa forma, usou os mesmos construtos na representação da primeira Regra de Reação. A Figura 82 apresenta o resultado da alteração feita pelo participante.

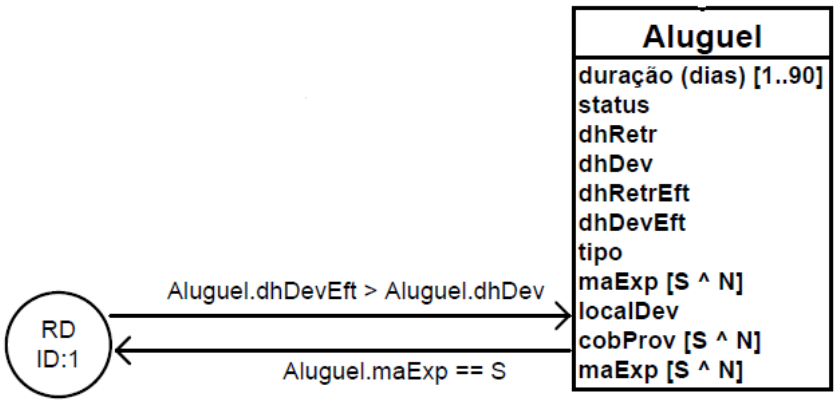


Figura 82 - Trecho da Regra de Derivação Má Experiência alterado.

A modelagem do terceiro participante explicitou características não exploradas pelos outros participantes do conjunto de Regras, especificamente as Regras de Integridade. Esse trabalho tem seu escopo definido na representação de regras do tipo *Alethic*, e uma discussão ontológica é necessária para tratar as Regras de Integridade em sua totalidade.

O participante respondeu o questionário, indicando que o modelo não é totalmente completo. Disse que a frase “Alugador é responsável pelo aluguel” não pode ser lida diretamente do modelo, assim como a leitura da regra de derivação “se a filial de retorno é diferente da filial de retirada, então o aluguel é do tipo aluguel em um só sentido”. Também sentiu falta da relação entre a Multa e o Aluguel. Com relação à validade, o participante indicou que o atributo Nível de Combustível estar ligado a classe mais genérica Carro “não está estipulado nas regras de negócio”.

5.4.2 Modelagem em OntoUML e R-OntoUML

Como o terceiro participante tem maior contato com a UFO e OntoUML, o modelo em OntoUML e R-OntoUML resultou em uma estrutura mais complexa que os outros modelos gerados pelos demais participantes.

Aluguel foi modelado como um *Relator*, de forma semelhante ao segundo participante. A diferença está no relacionamento *Material* que esse *Relator* deriva. O relacionamento ternário AlugarCarro ocorre entre o *Papel* CarroAlugado (especialização do *Kind* Carro), o *RoleMixin* Cliente e o *Papel* Motorista (especialização do *Kind* Pessoa). Essa interpretação apresenta uma característica interessante do domínio, separando dois *papéis*, o de Motorista e Cliente, tratando Cliente como um *RoleMixin*, ou seja, como um conceito abstrato do domínio que representa propriedades de dois ou mais *papéis* do domínio. Os *Papéis* Alugador Individual e Alugador Empresarial herdam as propriedades desse *RoleMixin*, especializando também os *Kinds* Pessoa e Empresa, respectivamente. Apresenta também os relacionamentos de *Mediação* entre Aluguel e os elementos que participam do relacionamento *Material*, explicitando outras informações do domínio, de forma semelhante ao do modelo do segundo participante.

O conceito Cliente executa (relacionamento *performanceOf*) as *Ações Atômicas* Retirada de Carro e Retorno do Carro. O *Papel* Carro Alugado participa dessas duas *Ações Atômicas* como recurso (relacionamento *participationAsResource*). A *Ação* Retirada de Carro é a fundação (relacionamento *foundationOf*) do *Relator* Aluguel, uma construção em R-OntoUML não usada pelos outros participantes. Esse relacionamento indica que o Aluguel somente é iniciado quando o Cliente executa a Retirada do Carro Alugado.

O conceito Organização é especializado no *Papel* Filial, que participa do relacionamento de Retirada e Devolução com Carro Alugado, e executa a *Ação Atômica* Aplica Multa.

O participante manteve as representações de Regras de Integridade do modelo em UML, com o domínio do atributo Duração de Aluguel entre um e noventa dias, e o Nível de Combustível de Carro Alugado com valor padrão igual a cheio.

A primeira Regra de Derivação compara a data/hora de devolução do Aluguel com a data/hora de devolução do Retorno do Carro, que, caso seja maior do que a data programada, traz a conclusão que houve uma Má Experiência, alterando o atributo "maExp" de Aluguel para "Sim". Essa representação foi feita pelo participante mesmo depois do uso da comparação no relacionamento em UML: o participante preferiu usar os construtos da R-OntoUML para representar essa Regra.

A segunda Regra de Derivação compara a Filial da Ação Retirada do Carro com a Filial do Retorno do Carro. Caso sejam diferentes, o aluguel de um só sentido se configura, alterando o atributo Tipo de Aluguel para "umSoSentido". As duas Regras de Derivação estão apresentadas na Figura 83.

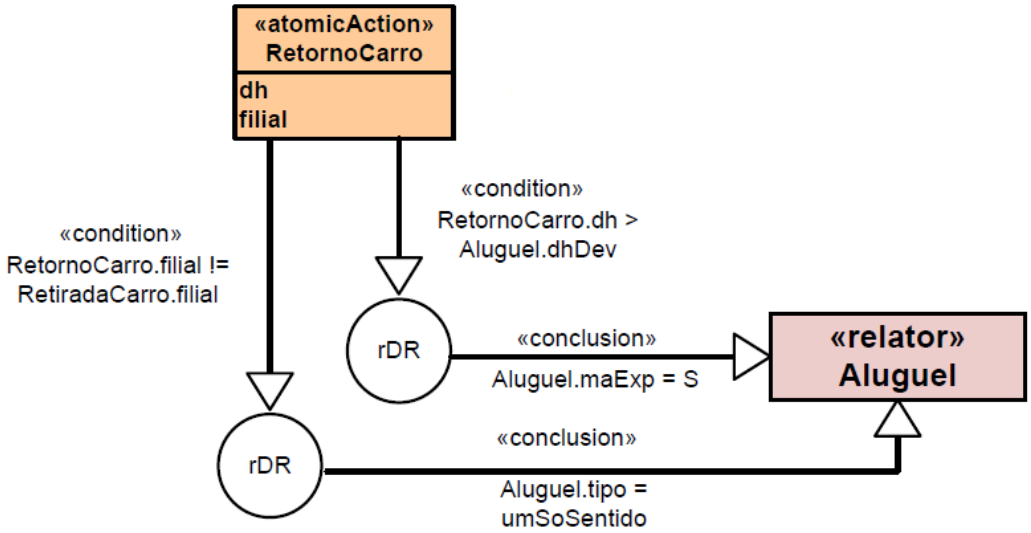


Figura 83 – Trecho da modelagem das Regras de Derivação do terceiro participante.

A primeira Regra de Reação foi modelada verificando se a Filial de Retorno do Carro é diferente do Local de Devolução planejado no momento do Aluguel. Caso ocorra, a *Ação Atômica* Aplica Multa é executada pela Filial.

Diferente da modelagem do segundo participante, o terceiro representou a segunda Regra de Reação usando a *Ação Atômica* "AbreAluguel" como evento de saída

da Regra, e a Pós-condição indicando que o atributo Status é igual a Aberto. Não houve interação, como perguntas feitas pelo primeiro participante sobre o domínio, para que a modelagem tivesse esse resultado. O terceiro participante, a partir da meta-propriedade de Regra de Fundamentação de Reação (necessidade de ter uma *Seta de Evento* saindo da Regra para um Evento ou especialização), identificou que o Aluguel estar aberto necessita de uma *Ação Atômica*, modelada como Abre Aluguel. Esse conceito foi explicitado pelo uso da R-OntoUML. Além dessa Ação, a Pós-Condição indica a alteração do atributo Status de Aluguel para "Aberto". A Figura 84 apresenta esse trecho.

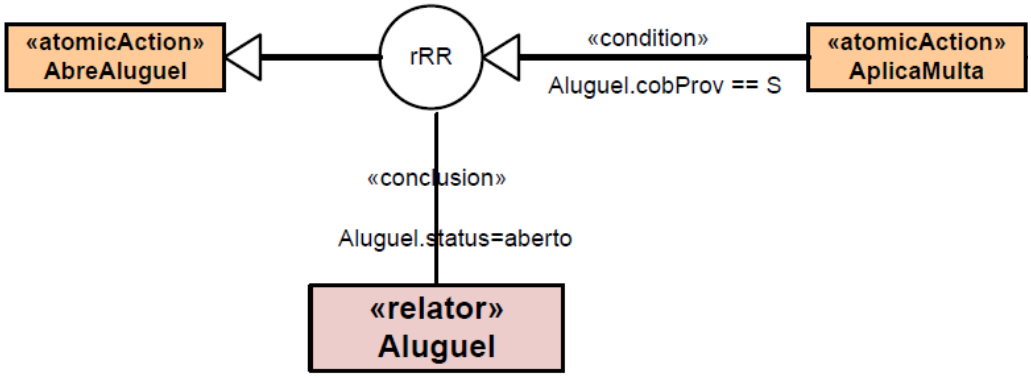


Figura 84 - Trecho da Regra de Reação do Aluguel em estado Aberto.

Depois de finalizado o modelo, o participante respondeu o restante do questionário. Ao avaliar o modelo em OntoUML e R-OntoUML, indicou que o modelo é completo e válido.

Comparando os modelos, o participante indicou que o modelo OntoUML e R-OntoUML é mais completo que o modelo UML e URML: "Os construtos da OntoUML e da R-OntoUML, e as respectivas meta-propriedades subjacentes, fornecem maior riqueza de significado, aos conceitos e eventos representados". Com relação à validade, respondeu que o modelo em OntoUML e R-OntoUML "parece ser mais válido em relação ao domínio", pois os construtos oferecem maior fidedignidade com o domínio.

A avaliação da generalidade do participante tratou UML como o mais genérico, já que as meta-propriedades das classes, quase que completamente, idênticas. Citou os conceito Aluguel e Carro que, apesar de representados com o mesmo construto, possuem propriedades distintas, que se perdem com o uso da UML. Para o participante, o modelo em OntoUML e R-OntoUML é mais preciso: "já que semanticamente mais rico, é mais preciso em termos de representação do domínio".

5.4.3 Avaliação cruzada entre participantes e percepção de uso da linguagem proposta

A segunda etapa do estudo de caso do terceiro participante está no Anexo I.

O terceiro participante avaliou o modelo do primeiro participante. Avaliou o primeiro modelo como não completo, faltando algumas afirmativas corretas e relevantes ao domínio. Com relação ao domínio, não encontrou a presença do conceito Motorista, representado como um atributo. Questiona relação entre Alugador e Aluguel, que não indica que o primeiro conceito é responsável pelo segundo, e o fato do Nível de Combustível não estar ligado a Carro, mas a Aluguel. Com relação à validade, respondeu que o modelo não é valido, indicando que o atributo Estado de Cobrança não está presente nas regras. Destaca novamente o atributo Nível de Combustível estar representado como atributo de Aluguel, conforme descrito na segunda Regra de Integridade. O participante identificou algumas discrepâncias do modelo com o domínio. Entretanto, são diferenças em pontos específicos da modelagem, a principal delas a modelagem de Nível de Combustível como atributo de Aluguel. O entendimento do primeiro participante o levou a modelar esse atributo como uma propriedade de Aluguel, retirando-o do conceito Carro.

Ao validar o modelo OntoUML e R-OntoUML, respondeu que o modelo é completo. Com relação à validade, assim como no primeiro modelo, avaliou o modelo como não válido. Questionou a presença de um conjunto de atributos relacionados ao conceito Reserva, que compõe a *Ação Complexa* Aluguel: “isso, conceitualmente, não é correto visto que, além de não estar descrito nas descrições e regras, o “minimundo” afirma que não é necessário haver reserva para que o aluguel ocorra”. Caso não haja Reserva, esses atributos seriam perdidos, pois Aluguel não os contém. O primeiro participante, durante sua modelagem, comentou que todas as *Ações Atômicas* fazem parte do tempo de Aluguel. Outro ponto destaca é o atributo Nível de Combustível, novamente modelado como uma propriedade de outro conceito diferente de Carro, no caso a *Ação Atômica* Retirada. Finaliza seus comentários indicando que o conceito Má Experiência tem a participação do Alugador, porém, não tem relação com a *Ação Atômica* Devolução, mesmo que “a má experiência ocorra em decorrência de um atraso na devolução”.

Ao comparar os dois modelos, questionou a possibilidade de comparação entre os modelos, pois os conceitos foram alterados. Os construtos UFO levam ao questionamento das meta-propriedades dos conceitos do domínio, e o primeiro participante alterou a sua interpretação inicial do domínio. No entanto, ressalta que o modelo em OntoUML e R-OntoUML “permite uma melhor clareza na leitura”. Devido às diferenças entre os modelos, o terceiro participante não comparou a validade entre eles. Com relação à generalidade, para o participante, o modelo em UML e URML é mais genérico, dificultando o entendimento do domínio “devido a oferecer construtos mais genéricos”. Finaliza a avaliação respondendo que a modelagem em OntoUML e R-OntoUML é mais precisa “em termos de conceitos e de relacionamentos”.

Terminando sua participação, o terceiro participante compartilhou a sua percepção de uso da OntoUML e R-OntoUML. Discorda que a modelagem é desconfortável de usar, e discorda fortemente que o uso da técnica foi frustrante. Entretanto, concorda que a técnica exige muito esforço mental. O participante discorda em parte da frase “a técnica de modelagem é clara, compreensível para mim”. Considerando o todo, discorda que a técnica é fácil de usar.

Em sua explicação, o participante destacou que OntoUML e R-OntoUML “exige um conhecimento maior dos construtos da linguagem, conhecimento esse que não se adquire de forma intuitiva”, retorquindo que a linguagem é nova, exigindo um tempo maior de aprendizado. Por fim, destaca que o trabalho relaciona-se com trabalhos anteriores.

5.5 Conclusão

A avaliação dos participantes do estudo de caso indicou que duas das três das modelagens ficou completa. Os dois primeiros participantes concordaram que os modelos em UML com URML e OntoUML com R-OntoUML são completos, ao avaliarem seus modelos e de seus pares. O terceiro participante responde que seu modelo UML não era completo, destacando que algumas definições do domínio não foram explicitamente entendidas a partir do modelo, mas considerou seu modelo em OntoUML com R-OntoUML completo. Na avaliação dos modelos de seu par, também questionou a completeza do modelo (respondendo a terceira opção), especificamente a

interpretação do outro participante do domínio, que trouxe diferenças entre o modelo e o domínio, entretanto, avaliou o modelo como completo, apesar de ser um pouco confuso.

Essas avaliações indicam que o modelo em OntoUML e R-OntoUML tende a ser mais completo e livre de algumas inconsistências do que a representação em UML e URML, como vislumbradas por um dos participantes. A maioria dos questionamentos advém da interpretação do domínio de cada um, que acaba ficando mais coesa quando os construtos UFO são utilizados, pois as meta-propriedades desses elementos para a construção do modelo guiam as perguntas que o participante se questiona, visando à classificação correta dos conceitos do domínio.

A avaliação da validade dos modelos destaca ainda mais as diferenças entre as interpretações do domínio e a dificuldade do uso da OntoUML e R-OntoUML, questionando o resultado de alguns pontos da modelagem. O primeiro participante indicou que a forma como está representado em UML e em OntoUML obriga que o Aluguel esteja sempre no estado “Aberto”. Esse conceito poderia ser classificado como uma fase de Aluguel, conforme modelado pelo segundo participante, o que levaria a interpretação de como o Aluguel assume a fase “Aberto”. Porém, como uma fase é passageira, essa obrigação não restringe o Aluguel a estar sempre na fase Aberto, ou seja, o modelo continuaria válido. Esse tipo de interpretação não é possível em uma modelagem em UML, pois os construtos que definem classes não possuem distinções entre si. Ao avaliar o modelo de seu par, o primeiro participante questionou novamente a validade, mas a interpretação que o modelo do segundo participante foi diferente da interpretação do primeiro. Indicou a sua dificuldade de avaliar a validade em OntoUML, devido à dificuldade de leitura do modelo gerado, com um conjunto grande de meta-propriedades a serem consideradas durante a sua leitura.

O segundo participante também respondeu que os modelos não são completamente válidos, mas se atentou a outro conceito do domínio, o Valor da Cobrança. A dificuldade de representação desse conceito foi levantada durante as sessões, com questionamentos sobre o que queria dizer o Valor da Cobrança e a sua relação com a segunda Regra de Reação. Ao comparar os dois modelos, explicitou que talvez essa dificuldade seja justificada pelo seu entendimento do domínio. Avaliando o modelo de seu par, o segundo participante questionou conceitos que não estavam nas Regras e Sentenças, mas estavam descritas no resumo inicial do domínio. O terceiro participante sentiu necessidade de acrescentar esses conceitos, tirando dúvidas sobre

como modelar o domínio usando o resumo inicial. Ou seja, o modelo se mantém válido, pois os participantes tinham liberdade de questionar sobre o domínio e usar o próprio resumo para tentar sanar suas dúvidas.

O terceiro participante questionou um ponto de seu modelo UML e URML, destacando a presença de um atributo na especialização de uma classe, quando esse atributo também deveria estar na classe mais genérica. Quando avaliou o modelo OntoUML e R-OntoUML, respondeu que é completamente válido. Sua comparação entre os modelos destacou o uso da OntoUML e R-OntoUML, pois devido à riqueza semântica que seus construtos trouxeram, o resultado possui maior fidedignidade, em relação ao domínio. O modelo de seu par não foi avaliado como completamente válido. O terceiro participante mencionou escolhas de modelagem que seu par tomou para modelar o domínio, indicando que certos atributos estavam em classes diferentes das descritas nas regras. Respondeu que o modelo em OntoUML e R-OntoUML de seu par não é valido, listando um conjunto de conceitos modelados de forma diferente da esperada, seguindo informações do resumo inicial (o que está correto), relacionando atributos a classes de forma diferente da descrita e deixando de relacionar conceitos que participavam de algumas regras. A interpretação do domínio entre os participantes foi diferente e, como cada um tinha uma experiência diferente com a OntoUML e R-OntoUML, o modelo resultante explorou construtos diferentes do que o terceiro participante esperava. Comparou os dois modelos respondendo que “há diferenças de modelagem de um modelo para o outro”. Essa diferença dificulta a comparação entre esses modelos.

Os participantes concordaram que o modelo em UML e URML é mais genérico, permitindo a instanciação de mais objetos do mundo real em suas classes.

Todos os participantes também afirmaram que o modelo em OntoUML e R-OntoUML é mais preciso, especificando com mais detalhes o domínio e a instanciação dos objetos.

O primeiro participante destacou que o uso de UML “não impõe tantas meta-propriedades” quanto o uso da OntoUML, mas a sua modelagem é mais simples, enquanto OntoUML e R-OntoUML ajudam na construção do modelo, guiando as escolhas de classificação dos conceitos entre seus construtos. O segundo participante explicitou que a UML permitiu a instanciação de mais elementos reais, devido às “suas menores restrições”, ou seja, menos meta-propriedades. O uso da OntoUML e R-

OntoUML agregou maior semântica ao modelo e permitiu “retirar dos modelos grande parte da ambiguidade na representação”. O terceiro participante respondeu a questão de generalidade destacando que as classes em UML são, quase totalmente, “definidas pelas mesmas meta-propriedades”, sendo mais genérico, enquanto que o modelo em OntoUML e R-OntoUML é mais preciso para representação do domínio, “já que semanticamente mais rico”.

De forma geral, as avaliações dos 3 participantes deste estudo de caso apontaram indícios de que a hipótese desta pesquisa (que a linguagem de regras baseada em ontologia de fundamentação produzirá modelos mais completos, mais válidos e mais precisos) foi confirmada. Os participantes também dividiram sua percepção do uso da OntoUML e R-OntoUML. A Tabela 1 apresenta a agregação das respostas dos participantes ao questionário.

Tabela 1 - Agregação das respostas dos participantes em relação à percepção de uso.

Escala	Concordo fortemente	Concordo	Concordo em parte	Neutro	Discordo em parte	Discordo	Discordo fortemente
1. Acredito que a técnica de modelagem é desconfortável de usar.	P2				P1	P3	
2. Usar a técnica de modelagem foi frustrante.						P1	P2 P3
3. Usar a técnica de modelagem exigiu muito esforço mental.	P2	P1 P3					
4. A técnica de modelagem é clara compreensível para mim.			P2	P1	P3		
5. Considerando tudo, acredito que a técnica de modelagem é fácil de usar.				P1		P3	P2

Para facilitar a leitura da Tabela 1, o primeiro participante foi identificado com o texto “P1”, o segundo participante com “P2” e o terceiro participante com “P3”. Em sua maioria, as respostas da percepção de uso foram semelhantes, com exceção da primeira pergunta. O segundo participante concorda fortemente que o uso da linguagem é desconfortável, enquanto que os outros participantes discordam dessa frase. Retratou sua dificuldade quando descreveu sua experiência: “Entender e por em prática como cada conceito e relacionamento se comportam não é tranquilo, principalmente em pouco tempo”. Essa dificuldade mostra a necessidade de mais tempo (o que também foi destacado pelos demais participantes) para entender os construtos usados, acostumando-se com o uso da linguagem OntoUML e do perfil R-OntoUML. O participante mais experiente destacou a dificuldade de uso da R-OntoUML, por ser uma linguagem nova.

Há o custo do aprendizado, que demanda tempo, caso seja escolhido o uso da OntoUML e R-OntoUML na modelagem de um domínio ou negócio e as Regras de Negócio que o regem. Todos discordaram que a construção do modelo foi frustrante e concordaram que o modelo exige muito esforço mental. Com relação à clareza da modelagem, a resposta foi neutra, mas todos concordam que a proposta é difícil de usar. Considerando a resposta da primeira e da última perguntas, o treinamento mais intenso e período de adaptação da OntoUML e R-OntoUML é necessário para que a modelagem seja mais confortável para a construção do modelo.

A curva de aprendizado da linguagem foi mencionada por todos os participantes. Inclusive, o participante mais experiente, apesar de conhecer a OntoUML, destacou a dificuldade de sua modelagem, assim como o pouco tempo disponível para aprendizado da R-OntoUML, acarretando em uma modelagem ainda mais complexa. Os outros participantes encontraram dificuldades de uso da OntoUML e R-OntoUML também, porém, todos indicaram que seu uso agrega semântica ao modelo resultante, tornando-o mais preciso (todos os participantes tiveram essa percepção). Essas considerações evidenciam ainda mais a dificuldade de execução do estudo de caso, pois a necessidade de participantes experientes em UFO e OntoUML traria uma avaliação mais rica. Conforme mencionado, não foi possível o acesso a um contingente maior de especialistas nessa ontologia de fundamentação e linguagem no período disponível para a realização do estudo de caso.

A modelagem dos participantes foi trabalhosa e extensa, durando em média seis horas. Além das dificuldades expressadas pelos participantes em início de aprendizado (o primeiro e o segundo participantes, em especial), o tempo gasto nessa avaliação é mais um indicativo do custo do uso da OntoUML e R-OntoUML. Com isso, seu uso deveria ser direcionado para casos em que a ambiguidade deve ser mínima, como em um cenário de integração de dados, aonde a precisão do trabalho é importante.

Apesar da dificuldade na modelagem, todos os participantes concordaram que a representação em OntoUML e R-OntoUML é mais precisa. Outro consenso é a característica de generalidade do modelo construído com UML e URML.

Como o objetivo do trabalho era uma modelagem precisa, o estudo do caso trouxe um resultado satisfatório, fornecendo indícios que confirmam a hipótese do trabalho. A representação do terceiro participante das Regras de Integridade foi feita de forma diferente da esperada, levantando questões fora do escopo desse trabalho: a

discussão da representação ontológica de Regras de Integridade, visando resolver a diferente forma de modelagem usada por esse participante.

O cenário usado no estudo de caso foi considerado simples pelos participantes. Essa simplicidade não influenciou na avaliação. Caso fosse mais complexo, poderia dificultar ainda mais o uso da OntoUML e R-OntoUML. Um exemplo mais complexo traria mais elementos de discussão, mas foi possível chegar a conclusões com o uso do domínio atual.

6 Conclusão

Regras de Negócio são fontes importantes de conhecimento sobre a conceitualização de um domínio. Entretanto, as linguagens de representação de regras de negócio atuais possuem uma representação pouco precisa. Com isso, as regras podem gerar mais de uma interpretação de um mesmo domínio. A má interpretação do domínio pode acarretar em implementações incorretas de Sistemas de Informação, na definição de estruturas inadequadas de armazenamento dos dados, em falhas na interoperabilidade entre diferentes sistemas, e em problemas de comunicação em geral.

O presente trabalho propôs uma representação de regras de negócio baseada em ontologias de fundamentação, visando uma modelagem do domínio mais completa, mais válida e mais precisa. Usando conceitos da UFO e a linguagem de representação visual de regras URML, o perfil R-OntoUML foi criado, para ser usado em conjunto com a linguagem OntoUML.

Comparando com modelos construídos com UML e URML com a proposta de modelagem em OntoUML e R-OntoUML, um estudo de caso foi realizado, abordando três tipos de regras de negócio dentro de um domínio de aluguel de carros: Regras de Integridade, Regras de Derivação e Regras de Reação. Três participantes criaram os modelos, avaliando se os resultados estavam completos, válidos e precisos.

O questionamento da validade dos modelos foi uma constante entre os participantes. Isso se deve principalmente à interpretação de cada um sobre o domínio e da necessidade de mais informações sobre ele, pois as afirmações e regras disponibilizadas não foram suficientes. Outro ponto ressaltado pelos participantes foi quanto ao tempo de aprendizado da OntoUML que, embora esteja fora do escopo do presente trabalho, levou à dificuldade na modelagem, fazendo com que os participantes usassem recursos da linguagem que, em certas situações, não seriam os mais indicados. Como exemplo, temos a avaliação do terceiro participante de seu par, mencionando que

o modelo, apesar de correto, ficou confuso. Os participantes usaram o bom senso (devido à simplicidade do domínio usado no estudo de caso), o que acarretou em diferentes modelagens sobre os conceitos do domínio. Com isso, certas afirmações do modelo fugiam da descrição do domínio. Isso poderia ser solucionado caso os participantes tirassem as suas dúvidas durante a modelagem. O caso da representação da segunda Regra de Reação é um exemplo dessa afirmação: dois participantes pediram mais informações sobre essa parte do domínio, o que gerou um modelo semelhante para os dois participantes, com a criação de uma ação que identifica a abertura do aluguel.

Todos os participantes concordaram que a modelagem em OntoUML e R-OntoUML gera modelos mais precisos. As meta-propriedades presentes nos construtos usados acrescentam mais semântica ao domínio. Durante a construção do modelo, os participantes precisaram de mais informações do domínio para classificar os conceitos da forma mais correta possível, algo que ocorre durante a ponderação de como um elemento do domínio se encaixa dentro dos conceitos da UFO.

Dentre as limitações do trabalho, pode-se ressaltar que nem todos os tipos de regras foram tratados pela proposta. A modelagem das Regras de Integridade gerou discussões que fogem ao escopo deste trabalho, fugindo a ideia da necessidade da regra não poder ser violada (a relação entre regras do tipo *Alethic* e *Deontic*). O estudo ontológico desse tipo de regra é necessário.

A proposta se mostrou uma possibilidade de representação das regras de negócio de um domínio de forma mais precisa.

6.1 Contribuições desse trabalho

A contribuição principal do presente trabalho foi o desenvolvimento de um perfil de representação – denominado R-OntoUML – para a modelagem de regras de negócio, especificamente, regras do tipo Derivação e Reação. O perfil R-OntoUML deve ser usado em conjunto com a OntoUML. R-OntoUML foi criado visando ser compatível com os construtos usados por OntoUML, que se baseia na ontologia de fundamentação UFO-A. Para tal, aproveitou-se dos conceitos da UFO-B e da UFO-C, acrescentando construtos modificados da URML e OCL para tratar os elementos da representação de Regras de Negócio que a UFO-A não considera. Outra contribuição é o uso dos conceitos da UFO-B e UFO-C em um perfil UML, o que apoia uma futura extensão da

OntoUML visando cobrir tais conceitos. Outra contribuição é a execução de uma avaliação da aplicabilidade da R-OntoUML e da própria OntoUML, para a qual faltam experimentos de avaliação.

A revisão da literatura sobre Regras de Negócio e Linguagens de Modelagem de Regras de Negócio é outra contribuição desse trabalho.

A discussão sobre os modelos gerados a partir das regras também apoia interessados no uso da R-OntoUML na utilização da linguagem, fornecendo exemplos de uso e podendo servir de base para definição de boas práticas de modelagem de regras de negócio usando R-OntoUML.

6.2 Trabalhos Futuros

O escopo deste trabalho não vislumbra a discussão ontológica das regras de negócio do tipo Restrição de Integridade. Um trabalho futuro seria a exploração dessa visão, estendendo o perfil R-OntoUML, tratando especificamente os operadores aléticos e deônticos da lógica modal (HALPIN e MORGAN, 2008) e os construtos da representação das regras: suas condições, conclusões, eventos resultantes e o restante da estrutura baseada na URML. A discussão ontológica diferencia-se muito do realizado neste trabalho: traria a fundamentação filosófica e a conceitualização do que são regras, incluindo a discussão de regras do tipo *Alethic* e *Deontic*, além de fundamentações que não foram pesquisadas neste trabalho.

Um destaque na avaliação dos participantes do estudo de caso foi a dificuldade do uso da R-OntoUML para a modelagem do domínio. Essa dificuldade torna a tarefa de construção dos modelos mais complexa. A criação de uma metodologia de representação de Regras de Negócio usando a R-OntoUML amenizaria essa complexidade. O apoio sistemático e possivelmente computacional ao método tornaria a modelagem mais confortável, ágil e menos complexa.

Considerando o trabalho de BENEVIDES e GUIZZARDI (2009), outro possível caminho é a extensão de uma ferramenta de edição de modelos de regras de negócio, englobando o perfil R-OntoUML. O apoio computacional garante que o modelo atenda às meta-propriedades do perfil. Essa ferramenta poderia trabalhar com os vários níveis de representação do domínio, conforme a ideia de volume de informação proposta em LOPES *et al.* (2010). Um modelo em OntoUML traz um conjunto de informações

grande para uma pessoa não experiente na linguagem, e o acréscimo de mais conhecimento sobre o domínio com a representação de regras de negócio pode dificultar ainda mais a leitura do modelo por uma pessoa não especialista. A ferramenta poderia alterar os construtos que aparecessem no modelo, escondendo informações para deixar a leitura do modelo mais simples, até chegar ao modelo completo, mais complexo. Isso reduziria a dificuldade dos modeladores apontadas no estudo de caso. Uma primeira visão traria o modelo em UML sem os estereótipos em OntoUML. Esse modelo é mais comum em sistemas de informação, assim como o seu entendimento. A segunda visão, com maior complexidade, mostraria a OntoUML completa, com todos os estereótipos e meta-propriedades dos conceitos. Apesar da dificuldade da leitura, o modelo fica mais completo e preciso, mantendo a sua validade, enriquecendo a discussão sobre os conceitos do domínio. Em paralelo a essas visões, a ferramenta poderia acrescentar a visão das regras de negócio. Em UML, os construtos da URML apareceriam, mostrando as regras que regem o domínio. No nível OntoUML, o perfil R-OntoUML seria apresentado, mostrando essas mesmas regras, em conjunto da linguagem baseada em uma ontologia de fundamentação.

O modelo resultante do uso da OntoUML e R-OntoUML está no nível conceitual. Esse modelo pode ser fonte para a construção de modelos lógicos e físicos de banco de dados e para modelos usando linguagens ontológicas voltadas para a Web Semântica (OWL, por exemplo) e mecanismos de inferência. Espera-se que a riqueza de informações do domínio e precisão desse modelo conceitual gere diagramas , ou mesmo ontologias em linguagens de codificação como OWL, mais fiéis ao domínio. Tais ontologias podem, inclusive, usar linguagens de representação de regras, como o SWRL (W3C, 2004c), usada em conjunto com a OWL.

A UFO-B e UFO-C não foram formalizadas em uma linguagem, diferente da UFO-A, formalizada na OntoUML. Esse trabalho deve ser revisto, caso essas duas ontologias de fundamentação tenham uma linguagem que as utilizem como base para as meta-propriedades de seus construtos, especificamente a relação dos conceitos de Evento e Ação, e a relação deles com o restante dos construtos UFO, incluindo elementos não explorados neste trabalho, como as relações de ALLEN (1983).

7 Bibliografia

ALLEN, J.F., “Maintaining Knowledge about Temporal Intervals”. *Communications of the ACM*, v. 26, n. 11, pp. 832-843, Nov. 1983.

BAIÃO, F., LOPES, M., GUIZZARDI, G., FALBO, R., *Reverse engineering a domain ontology to uncover fundamental ontological distinctions: An Industrial Case Study in the Domain of Oil and Gas Production and Exploration*. In: International Conference on Enterprise Information Systems (ICEIS), Itália, 2009.

BAIÃO, F.; SANTORO, F. M.; IENDRIKE, H.; CAPPELLI, C.; LOPES, M.; NUNES, V. T., *Towards a Data Integration Approach based on Business Process Models and Domain Ontologies*. In: International Conference on Enterprise Information Systems (ICEIS), Barcelona, 2008.

BENEVIDES, A., GUIZZARDI, G., *A Model-Based Tool for Conceptual Modeling and Domain Ontology Engineering in OntoUML*. In: Int´l Conf Enterprise Information Systems (ICEIS), Milan. LNBIP, Springer-Verlag. 2009.

BERNERS-LEE T., HENDLER, J., LASSILA,, O. “The Semantic Web”. *Scientific American*, Maio, 2001.

BORST, W.N., 1997, *Construction of Engineering Ontologies*. PhD thesis, Centre for Telematica and Information Technology, University of Twenty, Enschede, HL.

BRG. Business Rules Group, *Defining Business Rules – What are they really?*. 2000. Disponível em: http://www.businessrulesgroup.org/first_paper/br01c0.htm. Último acesso em Setembro de 2011.

BRICKLEY, D., GUHA,, R.V. *RDF Vocabulary Description Language 1.0: RDF Schema*. Relatório, Recomendação W3C Recommendation, 2004.

CARDOSO, E.C., SANTOS JUNIOR, P.S., ALMEIDA, J.P.A., GUIZZARDI, R.S.S., GUIZZARDI, G., *Semantic Integration of Goal and Business Process Modeling*, In:

IFIP International Conference on Research and Practical Issues of Enterprise Information Systems (CONFENIS), Rio Grande do Norte, Brazil, 2010.

CARDOSO, J., "The Semantic Web Vision: Where are we?". *IEEE Intelligent Systems*: 22-26, Setembro/ Outubro de 2007.

CHEN, P.P., "The Entity-Relationship Model-Toward a Unified View of Data". *ACM Transactions on Database Systems*, v. 1, n. 1 (1976), pp. 9-36, 1976.

CORCHO, O., FERNÁNDEZ-LÓPEZ, M., GÓMEZ-PÉREZ, "A. Methodologies, tools and languages for building ontologies. Where is their meeting point?". *Data & Knowledge Engineering*, v. 46, n. 1, pp. 41-64, 2003a.

CORCHO, O., GÓMEZ-PÉREZ, A., GUERRERO-RODRÍGUEZ, D.J., et al., *Evaluation experiment of ontology tools' interoperability with the WebODE ontology engineering workbench*. In: ISWC2003 Workshop on Evaluation of Ontology Tools (EON2003), Sanibel Island, Florida, Outubro de 2003b.

BADICA, C., GIURCA, A., WAGNER, G.M *Using Rules and R2ML for Modeling Negotiation Mechanisms in E-Commerce Agent Systems*. In Dirk Draheim and Gerald Weber, editors, Proceedings of the 2nd International Conference on Trends in Enterprise Application Architecture, TEAA2006, volume 4473 of Lecture Notes in Computer Science, pages 84–99. Springer, November 2006.

CRUZ, T., 2003, *Sistemas, métodos & processos: administrando organizações por meio de processos de negócios*. Editora Atlas, 2003, São Paulo.

BATRA, D., HOFFER, J.A., BOSTROM, R.P., "A Comparison of User Performance Between the Relational and the Extended Entity Relationship Models in the Discovery Phase of Database Design", *Communications of the ACM*, (33, 2), pp 126-139, Fevereiro, 1990.

HIRTLE, D., BOLEY, H., GROSOF, B., KIFER, M., SINTEK, M., TABET, S., WAGNER, G.. *Schema Specification of RuleML 0.91*. Disponível em http://www.ruleml.org/0.91/. Último acesso em Setembro de 2011, Agosto de 2006.

DEAN, M.S., G.; BECHHOFER, S.; HARMELEN, F. V.; ENDLER, J.; HORROCKS, I.; MCGUINNES, D. L.; PATEL-SCHNEIDER, P. F.; STEIN, L. A. *OWL: Web Ontology Language Reference. 2006*. Disponível em <http://www.w3.org/TR/owl-ref/>, Último acesso em Setembro de 2011.

DIJKSTRA, E.W., 1982, *Selected Writings on Computing: A Personal Perspective*. ISBN 0–387–90652–5. pp. 60-66, Springer-Verlag.

ELMASRI, R., NAVATHE, S., 2002, Sistemas de Banco de Dados - Fundamentos e Aplicações. Terceira edição. Editora LTC.

GENESERETH, M.R., FIKES, 1992, R.E. *Knowledge Interchange Format, Version 3.0 Reference Manual*. In: Logic-92-1, Computer Science Department, Stanford University.

GIURCA, A., LUKICHEV, S., WAGNER, G., 2006, "Modeling Web Services with URML", *In; proceedings of Semantics for Business Process Management Workshop*, Budva, Montenegro, June.

GONÇALVES, B., GUIZZARDI, G., FILHO, J. G. P., "An Electrocardiogram (ECG) Domain Ontology". In: *II Workshop on Ontologies and Metamodeling in Software and Data Engineering*, João Pessoa, PB, pp. 68-81, 2007.

GRUBER, T.R., 1993, "A translation approach to portable ontology specifications". *Knowledge Acquisition*.

GRUBER, T.R., 1992, "Ontolingua: A Mechanism to Support Portable Ontologies". *Knowledge Systems Laboratory*, Stanford University.

GRUBER, T., 2009, "Ontology". *Encyclopedia of Database Systems*, Liu. L., Özsu. M. (eds.), Springer-Verlag

GUIZZARDI, G., 2005, *Ontological Foundations for Structural Conceptual Models*. Ph.D. Thesis, University of Twente, The Netherlands.

GUIZZARDI, G., 2006, "The Role of Foundational Ontology for Conceptual Modeling and Domain Ontology Representation". In: *7th International Baltic Conf. on Databases and Information Systems*, Vilnius, Lithuania.

GUIZZARDI, G., FALBO, R.A., GUIZZARDI, R.S.S., 2008, "Grounding Software Domain Ontologies in the Unified Foundational Ontology (UFO): The case of the ODE Software Process Ontology". In: *Anais do XI Worshop Iberoamericano de Ambientes de Software e Engenharia de Requisitos*, Recife, Brazil.

GUIZZARDI, G., WAGNER, G., 2005, "Towards Ontological Foundations for Agent Modeling Concepts using UFO". *Lecture Notes in Computer Science*, v. 3508.

GUIZZARDI, G.; LOPES, M.; BAIÃO, F.; FALBO, R., 2010., "On the importance of truly ontological representation languages". In: *International Journal of Information Systems Modeling and Design* (IJISMD), Information Resources Management Association (IRMA), Editor-in-Chief: Remigijus Gustas, IGI Publishing, Hershey-New York, USA, Volume 1, Issue 2, April-June.

GUIZZARDI, R. S. S., GUIZZARDI, G., 2010, "Applying the UFO Ontology to Design an Agent-oriented Engineering Language". In: *14th East-European Conference on Advances in Databases and Information Systems* (ADBIS 2010), Novi Sad, Serbia.

HALLE, B. V., 1994, "Back to Business Rule Basics". *Database Programming & Design*, October, pp. 15-18.

HALLE, B. V. *Business Rules Applied: Builing Better Systems Using the Business Rules Approach*. 1. ed. New York: John Wiley & Sons, 2002.

HALPIN, T., 1995, *Conceptual Schema & Relational Database Design*. Second Edition, Sydney:Prentice Hall, Australia.

HALPIN, T., 2006, "Business Rule Modality", In: *CAiSE'06 Workshops*, eds T, Latour & M. Petit, Namur University Press, pp. 383-94.

HALPIN, T.A. 1995, *Conceptual Schema and Relational Database Design*, 2nd edn, Prentice Hall Australia, Sydney.

HALPIN, T.A. 1997, *Object-Role Modeling: an overview*. Disponível em www.orm.net. Último acesso em Setembro de 2011.

HALPIN, T.A., "Business Rules and Object-Role Modeling". *Database Programming & Design*, vol. 9, no. 10, 1996.

HALPIN, T.A., MORGAN, T., *Information Modeling and Relational Databases*. 2nd edition, Morgan-Kaufmann-Elsevier, 2008.

HEUSER C. A., PERES E. M., RICHTER G., "Towards a complete conceptual model: Petri nets and entity-relationship diagrams". *Information Systems*, v. 18, n. 5; pp. 275-298, 1993.

KAMADA, A. , 2006, *Execução de serviços baseada em regras de negócio*. Tese de Doutorado. Universidade Estadual de Campinas, Brasil.

LASSILA, O., SWICK, R. , 1999, *Resource description framework (RDF): model and syntax specification*. Relatório, W3C recommendation.

LOPES, M., SOUZA, J., BAIÃO, F., NUNES, V., CAPPELLI, C., *Um estudo para representação da semântica de diagramas entidade-relacionamento em OWL*. In: Relatórios Técnicos do Departamento de Informática Aplicada da UNIRIO n° 0004/2009, v. 3, n. 1, 2009.

LOPES, M., BAIAO, F., SIQUEIRA, S., 2010, "Expressing Action Assertions in Foundational-based Domain Ontologies". In: *International Conference on Information*

Integration and Web-based Applications & Services (iiWAS2010), Paris. Proceedings, 2010. v. 1. p. 1-4.

LUKICHEV S., JARRAR, M. . 2009, "Graphical Notations for Rule Modeling". *Handbook of Research on Emerging Rule-Based Languages and Technologies*. Idea Group Inc.

LUKICHEV, S., WAGNER, G., 2007, "Visual rules modeling", In: *PSI'06 Proceedings of the 6th international Andrei Ershov memorial conference on Perspectives of systems informatics*, Springer-Verlag Berlin, Heidelberg

MACGREGOR, R.M. , 1991, "Inside the LOOM description classifier". *ACM SIGART Bulletin*, v. 2, n. 3, pp. 88-92.

MARTINS, A.F., FALBO, R.A; GUIZZARDI, G., ALMEIDA, J.P.A., 2011, "Uso de uma Ontologia de Fundamentação para Dirimir Ambiguidades na Modelagem de Processos de Negócio", In: *VII Simpósio Brasileiro de Sistemas de Informação* (SBSI 2011), Salvador, Brazil.

MARTINS, A., FALBO, R. A., Guizzardi, G., Almeida, J.P.A. , 2011, "Uso de uma Ontologia de Fundamentação para Dirimir Ambiguidades na Modelagem de Processos de Negócio". In: *VII Simpósio Brasileiro de Sistemas de Informação* (SBSI 2011), 2011, Salvador.

MARTINS, A.F., 2009, *Construção de Ontologias de Tarefa e sua Reutilização na Engenharia de Requisitos*. Dissertação (Mestrado em Informática) - Universidade Federal do Espírito Santo.

MOODY, D., 2003, "Measuring the Quality of Data Models: An Empirical Evaluation of the Use of Quality Metrics in Practice". In: *Proceedings of the 11th European Conference on Information Systems*, Naples, Jun.

MOTTA, E., 1999, *Reusable Components for Knowledge Modelling: Case Studies in Parametric Design Problem Solving*. Amsterdam, IOS Press.

NIST., 1993, *Integration Definition for Information Modeling (IDEF1X)*. National Institute of Standards and Technology. FIPS PUB 184.

OMG, 2003, *MDA Guide V1.0.1*, Disponível em http://www.omg.org/cgi-bin/doc?mda-guide, Último acesso em Setembro de 2011.

OMG, 2009, *OMG Unified Modeling Language™ (OMG UML), Infrastructure – Version 2.2*, http://www.omg.org/spec/UML/2.2/, Último acesso em Setembro de 2011.

OMG., 2006a, *Object Constraint Language: OMG Available Specification, Version 2.0*. Disponível em: http://www.omg.org/spec/OCL/2.0/, Maio, Último acesso em Setembro 2011.

OMG., 2008, *Semantics of Business Vocabulary and Business Rules (SBVR), v1.0.*. Disponível em: http://www.omg.org/spec/SBVR/1.0/PDF, Janeiro. Último acesso em Setembro de 2011.

PARK, J., RAM, S., 2004, "Information Systems Interoperability: What lies beneath". *ACM Transactions on Information Systems*, Vol. 22, No. 4, pp. 595–632.

PATIG, S., 2004, "Measuring Expressiveness in Conceptual Modeling", In: *CAiSE 2004*, LNCS 3084, pp. 127–141, 2004. Springer-Verlag Berlin Heidelberg.

REWERSE, *URML-Metamodel*, Disponível em: https://oxygen.informatik.tu-cottbus.de/strelka/URML-Metamodel.htm, Último acesso em Setembro de 2011.

SANTOS JR., P. S. ; ALMEIDA, J. P. A. ; GUIZZARDI, G., 2009, "Uma interpretação para os elementos de EPCs com base em uma ontologia de fundamentação". In: *III Workshop Brasileiro em Gestão de Processos de Negócio*, 2009, Fortaleza. Simpósio Brasileiro de Sistemas Multimídia e Web (Webmedia) 2009.

SMITH, M. K., WELTY, C., MCGUINNESS, D.L., 2004, *OWL Web Ontology Language Guide*. Disponível em http://www.w3.org/TR/owl-guide/, Último acesso em Setembro de 2011. Fevereiro.

USCHOLD, M., JASPER, R., 1999, "A Framework for Understanding and Classifying Ontology Applications". In: *IJCAI-99 Workshop on Ontologies and Problem-Solving* Methods, Stockholm, Sweden.

W3C., 2004a, *Web Ontology Language (OWL)*. Disponível em: http://www.w3.org/2004/OWL/. Último acesso em Setembro de 2011.

W3C., 2004b, *Resource Description Framework (RDF): Concepts and Abstract Syntax*. Disponível em http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/. Último acesso em Setembro de 2011.

W3C., 2004c, *SWRL: A Semantic Web Rule Language – Combining OWL e RuleML*. Disponível em http://www.w3.org/Submission/SWRL/. Último acesso em Setembro de 2011.

WAGNER, G., 2004, "The Abstract Syntax of RuleML - Towards a General Web Rule Language Framework", In: *WI '04 Proceedings of the 2004 IEEE/WIC/ACM*

International Conference on Web Intelligence, IEEE Computer Society Washington, DC, USA.

WAGNER, G., GIURCA, A., Lukichev, S., 2006, "A Usable Interchange Format for Rich Syntax Rules. Integrating OCL, RuleML and SWRL". In: *Reasoning on the Web Workshop at WWW2006*, Maio

ZAMBORLINI, V., GONÇALVES, B.N., GUIZZARDI, G., 2008, "Codification and Application of a Well-Founded Heart-ECG Ontology". In: *3rd Workshop on Ontologies and Metamodels in Software and Data Engineering* (WOMSDE 2008), 2008, Campinas. *23rd Brazilian Symposium on Databases* (SBBD)/*22nd Brazilian Symposium on Software Engineering* (SBES).

ZIEGLER, P., DITTRICH, K., 2007, *Data Integration — Problems, Approaches, and Perspectives. Conceptual Modelling in Information Systems Engineering*, Springer, Berlin, Heidelberg, pp. 39-58.

Apêndice A. DOMÍNIO DO ESTUDO DE CASO

Domínio de Validação

Empresa de aluguel de carros, com filiais em vários países, chamada EU-Aluguel.

Resumo do domínio

EU-Aluguel aluga carros para seus clientes. Clientes podem ser indivíduos ou empresas. Diferentes modelos de carros são oferecidos, organizados em grupos. Todos os carros em um grupo são cobrados com as mesmas taxas. Um carro pode ser alugado através de reserva feita com antecedência ou por um cliente avulso no dia do aluguel. Uma reserva de aluguel especifica o grupo de carro requisitado, as datas e horários de início e final de aluguel e a filial da EU-Aluguel de onde o aluguel iniciará. Opcionalmente, a reserva pode especificar um aluguel em um só sentido (no qual o carro é devolvido a uma filial diferente da filial onde foi retirado) e pode requisitar um modelo de carro específico dentro do mesmo grupo.

EU-Aluguel tem um clube de fidelidade. Clientes que se afiliam acumulam pontos que podem ser usados para pagar por aluguéis.

EU-Aluguel, de tempos em tempos, oferece descontos e aprimoramento do modelo do carro de graça, dependendo de condições.

EU-Aluguel registra “más experiências” com clientes (como retorno atrasado não autorizado de um aluguel, ou dano no carro durante o aluguel) e pode recusar seguidas reservas de aluguel desses clientes.

Informações necessárias:

Aluguel tem Duração.

Aluguel tem Carro Alugado.

Aluguel tem Alugador

Aluguel tem Motorista.

Aluguel está fechado.

Aluguel está Aberto.

Aluguel está Em Curso.

Aluguel está Terminado.

Aluguel tem Data/Horário de Devolução.

Aluguel tem Data/Horário de Retirada.

Aluguel tem Data/Horário Efetiva de Devolução.

Aluguel tem Data/Horário Efetiva de Retirada.

Má experiência ocorre durante aluguel.

Má Experiência tem Data/Horário Efetiva de Devolução.

Aluguel tem Local de Devolução.

Aluguel tem Filial de Retorno.

Aluguel tem Filial de Retirada.

Aluguel sofre uma multa local.

Aluguel tem Cobrança de Aluguel.

A Cobrança Provisória do Aluguel é provisoriamente cobrada do Cartão de Crédito.

Alugador tem Cartão de Crédito.

Alugador é responsável pelo Aluguel.

Carro Alugado tem Nível de Combustível.

Regras de integridade

É obrigatório que a duração do aluguel de cada aluguel seja de no máximo 90 dias.

É obrigatório que o nível de combustível do carro alugado esteja cheio.

Regras de Derivação

Se a data/hora efetiva de devolução do aluguel for maior que a data/hora de devolução, então ocorre uma má experiência no aluguel.

Se a filial de retorno é diferente da filial de retirada, então o aluguel é do tipo aluguel em um só sentido.

Regras de Reação

Se o local de devolução do aluguel não é a filial de retorno do aluguel, então é obrigatório que o aluguel sofra uma multa local.

Se a cobrança estimada do aluguel for provisoriamente cobrada no cartão de crédito do alugador que é responsável pelo aluguel, então é obrigatório que o aluguel esteja aberto.

Apêndice B. QUESTIONÁRIO USADO NO ESTUDO DE CASO

Levando em consideração o modelo criado usando as linguagens UML e URML, responda as perguntas a seguir.

- (Completeza) O modelo contém todas as afirmativas das regras de negócio que são corretas e relevantes? (Para esta avaliação, verifique se é possível identificar no modelo todas as informações que estão mencionadas nas regras)

  Responda de 1 a 4, e justifique a resposta caso tenha marcado 2 ou 3:

  1) Não contém as afirmativas corretas e relevantes descritas nas regras de negócio.
  2) Contém poucas afirmativas corretas e relevantes descritas nas regras de negócio.
  3) Contém muitas afirmativas corretas e relevantes descritas nas regras de negócio.
  4) Contém as afirmativas corretas e relevantes descritas nas regras de negócio.

  Quais informações estão faltando no modelo?

_____

_____

_____

_____

_____

- (Validade) Todas as afirmações do modelo são corretas e relevantes ao domínio descrito pelas regras de negócio? (Para esta avaliação, verifique se existem informações no modelo que não estejam descritas nas regras)

  Responda de 1 a 4, e justifique a resposta caso tenha marcado 2 ou 3:

  1) Contém afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  2) Contém muitas afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  3) Contém poucas afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  4) Contém nenhuma afirmativa que não seja correta e relevante ao domínio descrito nas regras de negócio.

Quais informações que não são corretas e relevantes ao domínio que estão modeladas?

Levando em consideração o modelo criado usando as linguagens OntoUML e R-OntoUML, responda as perguntas a seguir.

- (Completeza) O modelo contém todas as afirmativas das regras de negócio que são corretas e relevantes? (Para esta avaliação, verifique se é possível identificar no modelo todas as informações que estão mencionadas nas regras)

  Responda de 1 a 4, e justifique a resposta caso tenha marcado 2 ou 3:

  1) Não contém as afirmativas corretas e relevantes descritas nas regras de negócio.
  2) Contém poucas afirmativas corretas e relevantes descritas nas regras de negócio.
  3) Contém muitas afirmativas corretas e relevantes descritas nas regras de negócio.
  4) Contém as afirmativas corretas e relevantes descritas nas regras de negócio.

  Quais informações estão faltando no modelo?

- (Validade) Todas as afirmações do modelo são corretas e relevantes ao domínio descrito pelas regras de negócio? (Para esta avaliação, verifique se existem informações no modelo que não estejam descritas nas regras)

  Responda de 1 a 4, e justifique a resposta caso tenha marcado 2 ou 3:

  1) Contém afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  2) Contém muitas afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  3) Contém poucas afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  4) Contém nenhuma afirmativa que não seja correta e relevante ao domínio descrito nas regras de negócio.

  Quais informações que não são corretas e relevantes ao domínio que estão modeladas?

Comparando os dois modelos:

- (Completeza) Comparando os modelos, quais as diferenças entre eles em conter todas as afirmativas das regras de negócio que são corretas e relevantes? (Para esta avaliação, verifique se é possível identificar nos modelos todas as informações que estão mencionadas nas regras, e quais as diferenças entre eles)

- (Validade) Comparando os modelos, todas as afirmações corretas e relevantes ao domínio descrito pelas regras de negócio aparecem nos modelos? (Para esta avaliação, verifique se existem informações nos modelos que não estejam descritas nas regras, e o que pode ter ocasionado isso)

- (Generalidade) Qual modelo dá maior liberdade para instanciação dos objetos do mundo real em cada classe/relacionamento/regra, permitindo com que cada conceito tenha um conjunto maior de elementos do mundo para instanciação? (As meta-propriedades dos construtos permitem que cada construto represente mais conceitos do mundo real)

- (Precisão) Qual modelo dá maior especificidade/precisão na instanciação dos objetos do mundo real em cada classe/relacionamento/regra? (As meta-propriedades dos construtos delimitam com maior precisão a quantidade de conceitos do mundo real que cada construto pode representar)

Apêndice C. QUESTIONÁRIO DE PERCEPÇÃO DE USO

Percepção de uso da OntoUML e R-OntoUML

Escala	Concordo fortemente	Concordo	Concordo em parte	Neutro	Discordo em parte	Discordo	Discordo fortemente
1. Acredito que a técnica de modelagem é desconfortável de usar.	1	2	3	4	5	6	7
2. Usar a técnica de modelagem foi frustrante.	1	2	3	4	5	6	7
3. Usar a técnica de modelagem exigiu muito esforço mental.	1	2	3	4	5	6	7
4. A técnica de modelagem é clara compreensível para mim.	1	2	3	4	5	6	7
5. Considerando tudo, acredito que a técnica de modelagem é fácil de usar.	1	2	3	4	5	6	7

Por favor, explique como foi a experiência de uso da OntoUML e R-OntoUML.

Apêndice D. DOCUMENTO USADO PARA EXPLICAÇÃO DA PROPOSTA

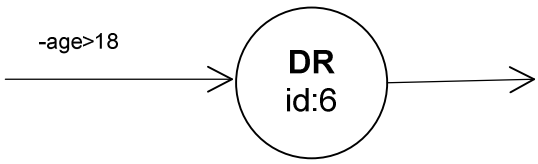
URML

Condition arrows (setas de condição) referem a um *elemento de condição do modelo*, que é um *classifier* como uma classe ou associação. Pode possuir uma *expressão de filtro* que seleciona instâncias do *classifier*, escrita em OCL.

Negated condition arrows (setas de condição negada) são cruzadas na origem da seta. Elas denotam a negação da condição que deve ser conjunta com uma ou mais setas de condição positivas para que suas variáveis sejam cobertas por elas.

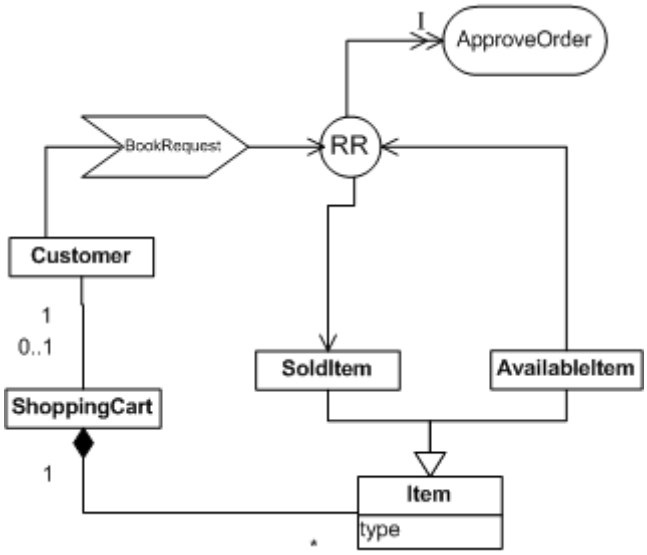
Derivation Rule (Regra de Derivação) são representadas graficamente como círculos, com o texto "DR" escritos internamente e um identificador de regra. Setas chegando ao círculo representam condições, enquanto que setas saindo representam conclusões.

- **Conclusion arrows** (setas de conclusão) significa que o predicado representado pelo *classifier* da conclusão é aplicado para qualquer instância que satisfaz todas as condições da regra



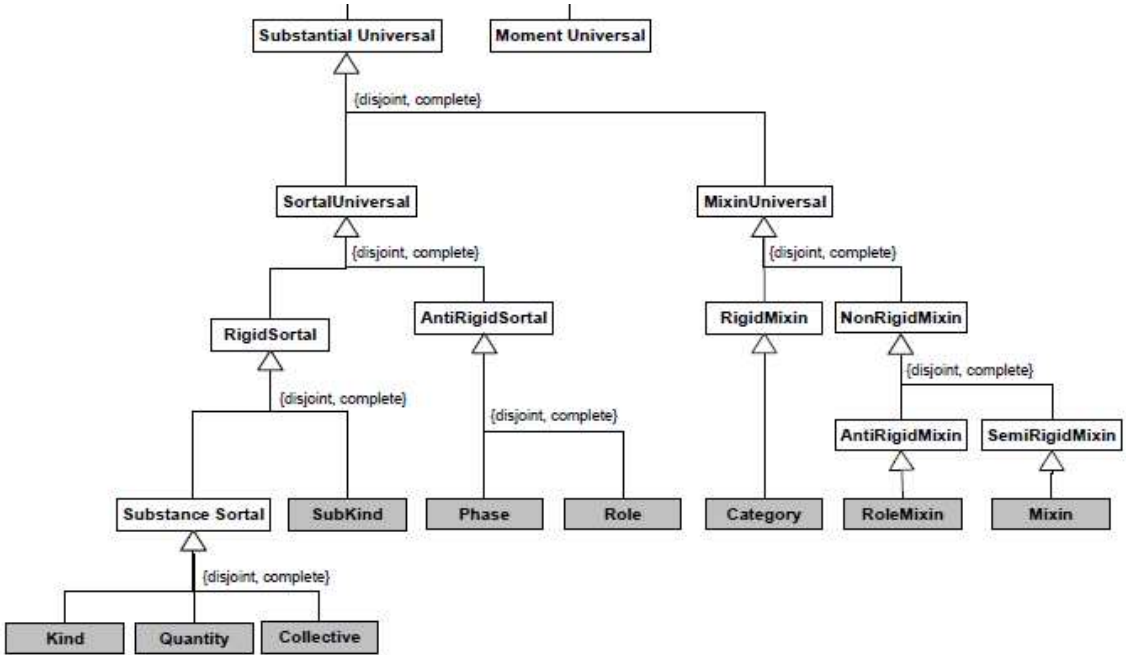
Reaction rules (Regras de Reação) são representadas graficamente como um círculo com o texto "RR" e um identificador de regra. Existem 2 tipos de setas chegando (setas de condição ou setas de evento) e 2 tipos de setas saindo (setas de ação ou setas de pós-condição, ambas com ponta de seta dupla).

- **Event arrows** (setas de evento) referem-se a uma regra ou a uma classe.
- **Action arrows** (setas de ação) referem-se a uma classe (em caso de criação, deleção, atribuição ou invocação de uma ação) ou a uma atividade.


1
ApproveOrder
BookRequest
RR
Customer
1
0..1
SoldItem
AvailableItem
ShoppingCart
1
Item
type
*

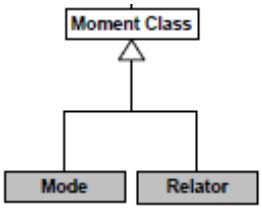
UFO / OntoUML

Construtos da UFO-A



Construto UFO	Descrição
Kind	Objetos do mundo que são definidos por suas propriedades e que não são alterados.
Collective	Conjunto de entidades iguais, por exemplo, um baralho, que é composto de cartas.
Phase	Instancia objetos durante um determinado período da sua existência, mas sendo disjuntos e completos. Por exemplo, as fases de adolescência e fase adulta de uma pessoa: a pessoa não é adolescente e adulto ao mesmo tempo, assumindo essas fases durante períodos da sua vida.
Role	Instancia objetos em um contexto específico, durante um evento ou quando participa de um determinado relacionamento, podendo ter até dois ou

	mais *Roles*. Por exemplo, o papel de uma pessoa: uma pessoa não é estudante durante sua vida toda, podendo deixar de ser um e voltar posteriormente, ou até assumir dois papéis, como estudante e professor.
Mixin	Define propriedades que são essenciais para algumas de suas instâncias e acidentais para outras. Por exemplo, a característica de ser "passível de se sentar em" algo é essencial pra uma cadeira, mas acidental para um engradado sólido
RoleMixin	Define propriedades que são essenciais para algumas instâncias de *Role* e acidentais para outras. Por exemplo, um cliente, que pode ser uma organização ou uma pessoa.
Categoria	Classifica entidades que pertencem a tipos diferentes, mas que dividem uma propriedade essencial em comum. Por exemplo, temos os mamíferos.
Quantity	Fornece princípios de identidade para conceitos representados por quantidades, que são definidos por sua massa ou espaço que ocupam. Por exemplo, água, ar ou petróleo.



Construto UFO	Descrição
Mode	Elemento que é definido por um momento específico do domínio sendo retratado. Ele provê mais características a um conceito em um dado momento sendo existencialmente dependente dele. Por exemplo, uma doença de um paciente.
Relator	Representa um evento que ocorre entre duas entidades, identificando a existência de um relacionamento entre essas duas entidades, sendo existencialmente dependente dessas entidades. O relacionamento derivado de um *Relator* é do tipo *material*.

Relacionamentos da UFO-A

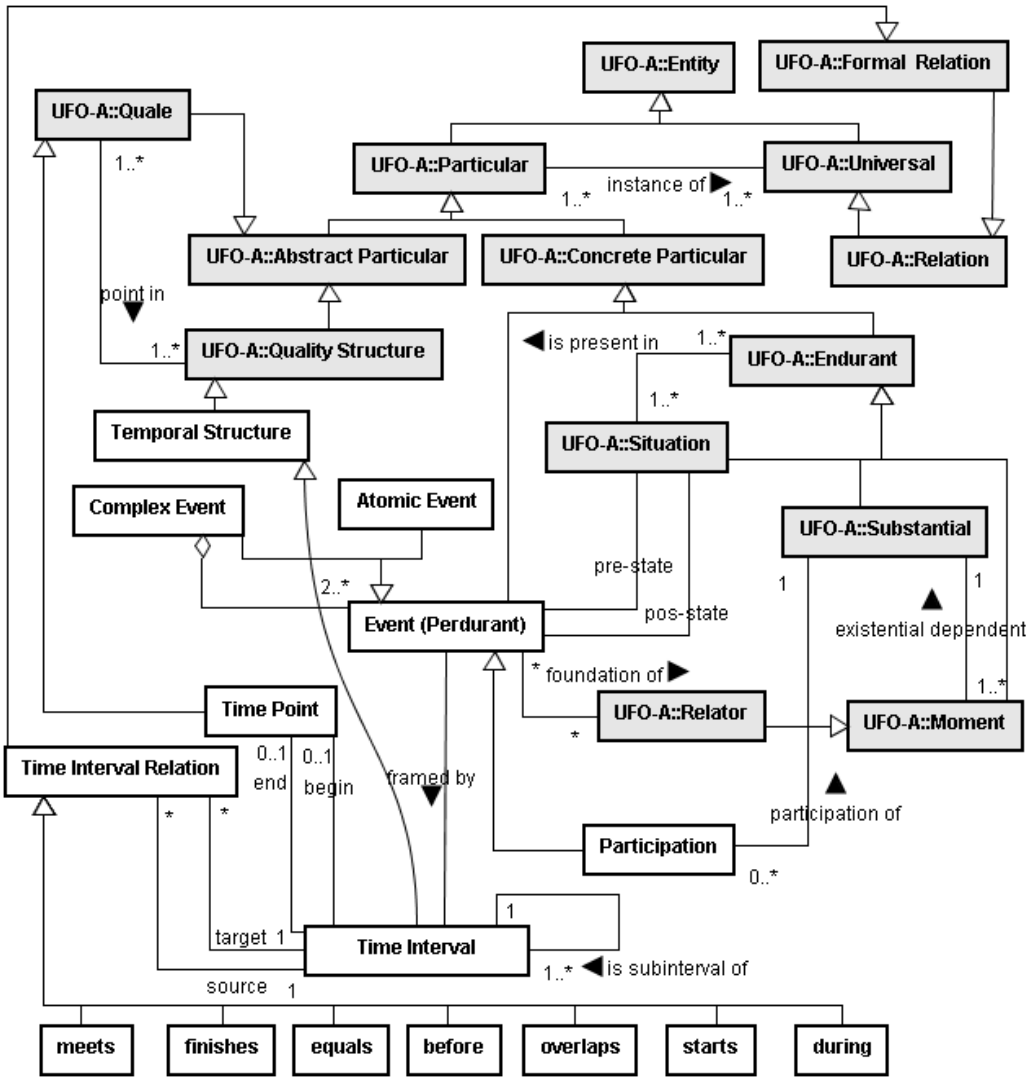
Construto UFO	Descrição
Properties	Os atributos presentes no meta modelo UML, representando propriedades dos conceitos modelados.
Characterization	Uma relação que identifica que um *mode* está acrescentando características que identificam um conceito, em um dado momento, durante certo período.
Material	Representa um relacionamento material,

	entre duas entidades ligadas a um *Relator* (esta ligação é representada pelo relacionamento *mediation*), que é derivado a partir da existência deste *Relator*
Derivation	Identifica que o relacionamento *material* é derivado de um *relator*, e é identificado por uma linha pontilhada com um círculo preto, ligando o *relator* e o relacionamento *material* derivado.
Mediation	Liga um *relator* e cada entidade que faz parte do evento que o *relator* representa.
Formal	Representa relações formais, sejam uma relação de comparação ou uma relação interna.

Relacionamentos “parte-todo” da UFO-A

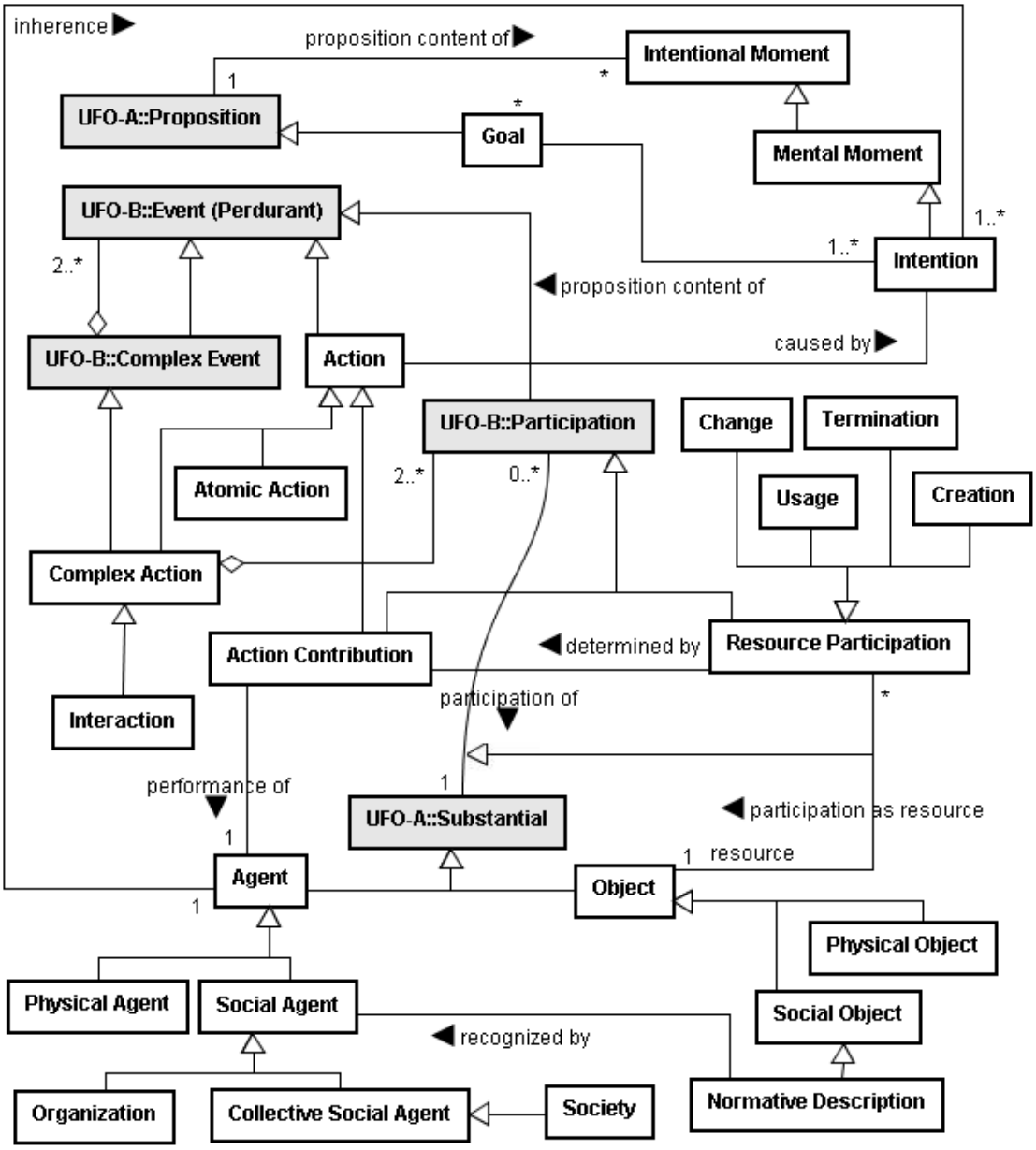
Construto UFO	Descrição
componentOf	Representa um relacionamento de parte-todo entre conceitos complexos, como *kind* e *role*.
subQuantityOf	Representa um relacionamento parte-todo entre quantidades, sempre não compartilhável e limitado a cardinalidade de no máximo um no fim da associação
subCollectionOf	Representa um relacionamento parte-todo entre *collectives*, limitado a cardinalidade de no máximo um no fim da associação.
memberOf	Representa um relacionamento parte-todo entre um conceito complexo (como um *kind* ou *role*) ou um *collective* (como uma parte) e outro *collective* (como um todo).

Construtos da UFO-B



Construto UFO	Descrição
AtomicEvent	Algo ocorrido que possui um início e fim baseado em intervalo de tempo, sendo indivisível em partes menores.
ComplexEvent	Algo ocorrido que possui um início e fim baseado em intervalo de tempo, que é dividido em partes menores.
Participation	Relacionamento entre *Event* e os *Substantials* que participaram do evento representado.

Construtos da UFO-C



Construto UFO	Descrição
AtomicAction	Algo ocorrido que possui um início e fim baseado em intervalo de tempo, causado pela intenção de um *Agent*. Não pode ser dividido em ações menores.

ComplexAction	Algo ocorrido que possui um início e fim baseado em intervalo de tempo, causado pela intenção de um ou mais *Agent*. Pode ser dividido em ações menores.
ActionContribution	Relacionamento que demonstra a participação do agente na ação representada pela *Action*.
ResourceParticipation	Relacionamento que demonstra a participação do objetos manipulados na ação representada pela *Action*.

Condition arrows (setas de condição) referem a um *elemento de condição do modelo*, que é um *classifier* como uma classe ou associação. Pode possuir uma *expressão de filtro* que seleciona instâncias do *classifier*, escrita em OCL. Usam o estereótipo <<condition>>.

Negated condition arrows (setas de condição negada) são cruzadas na origem da seta. Elas denotam a negação da condição que deve ser conjunta com uma ou mais setas de condição positivas para que suas variáveis sejam cobertas por elas. Usam o estereótipo <<condition>>.

Remodeled Integrity Rule (Regra de Integridade Remodelada) restringem um *classifier* do modelo. São regras que não pode ser quebradas: devem ser sempre verdade. São representados por retângulos com o estereótipo <<rIntegrityRule>>, contendo a sentença em OCL para representação da regra.

Remodeled Derivation Rule (Regra de Derivação Remodelada) são representadas graficamente como círculos, com o estereótipo <<rDerivationRule>> escrito internamente, junto com texto identificando a regra. Setas chegando ao círculo representam condições, enquanto que setas saindo representam conclusões.

- **Conclusion arrows** (setas de conclusão) significa que o predicado representado pelo *classifier* da conclusão é aplicado para qualquer instância que satisfaz todas as condições da regra. Usam o estereótipo <<conclusion>>.

Remodeled Reaction Rule (Regras de Reação Remodelada) são representadas graficamente como um círculo com o estereótipo <<rReactionRule>> escrito internamente, junto com texto identificando a regra. Existem 2 tipos de setas chegando (setas de condição ou setas de evento) e 2 tipos de setas saindo (setas de evento ou setas de pós-condição, ambas com ponta de seta dupla).

- **Event arrows** (setas de evento) referem-se a uma regra ou a uma classe. Quando usadas na conclusão da Remodeled Reaction Rule deve estar ligada a um *Event* (podendo ser uma *Action*, que é subtipo de *Event*). Usam o estereótipo <<eventArrow>>.

Anexo A. MODELO DO PRIMEIRO PARTICIPANTE

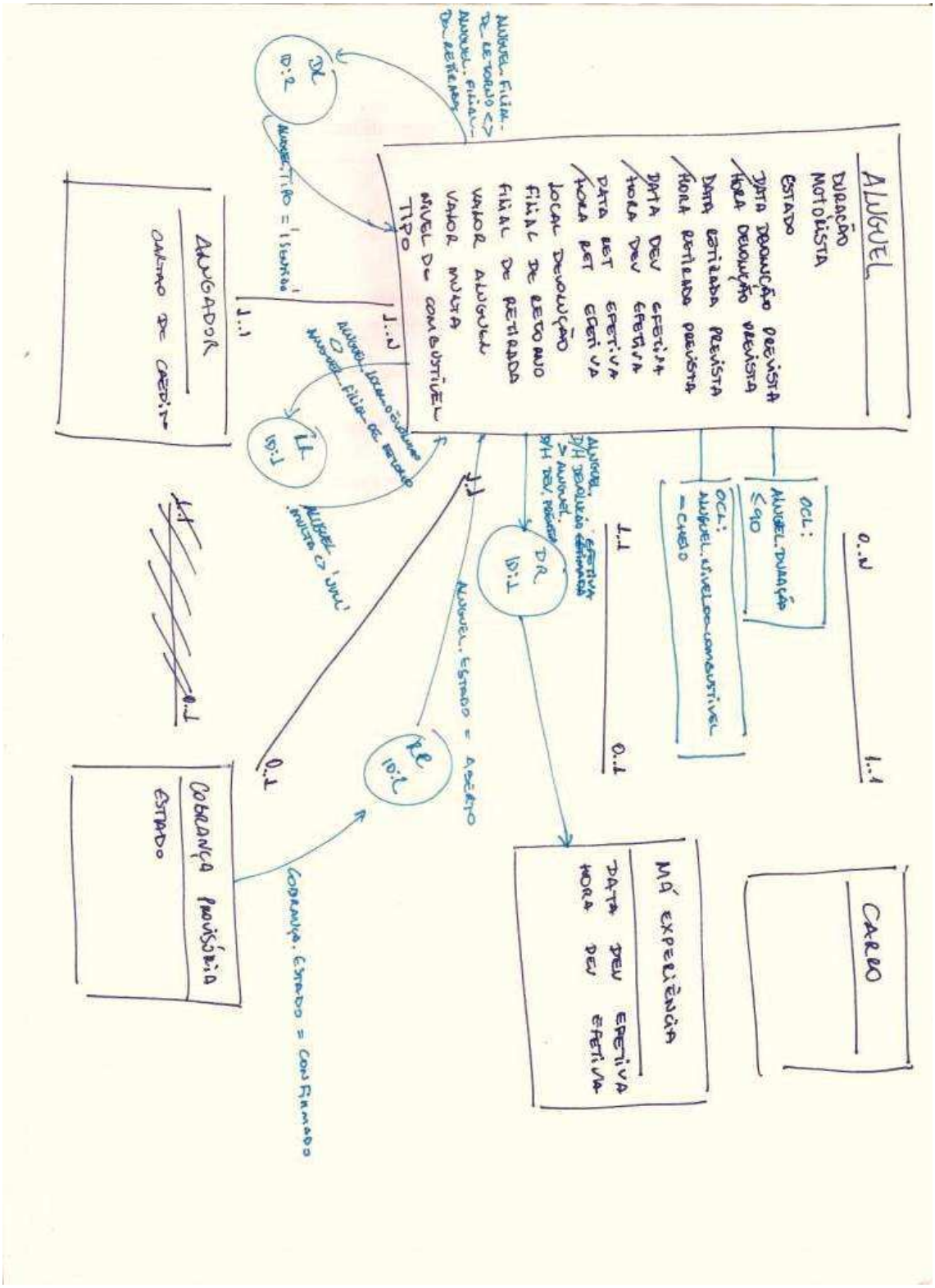


Figura 85 – Modelagem UML e URML do primeiro participante.

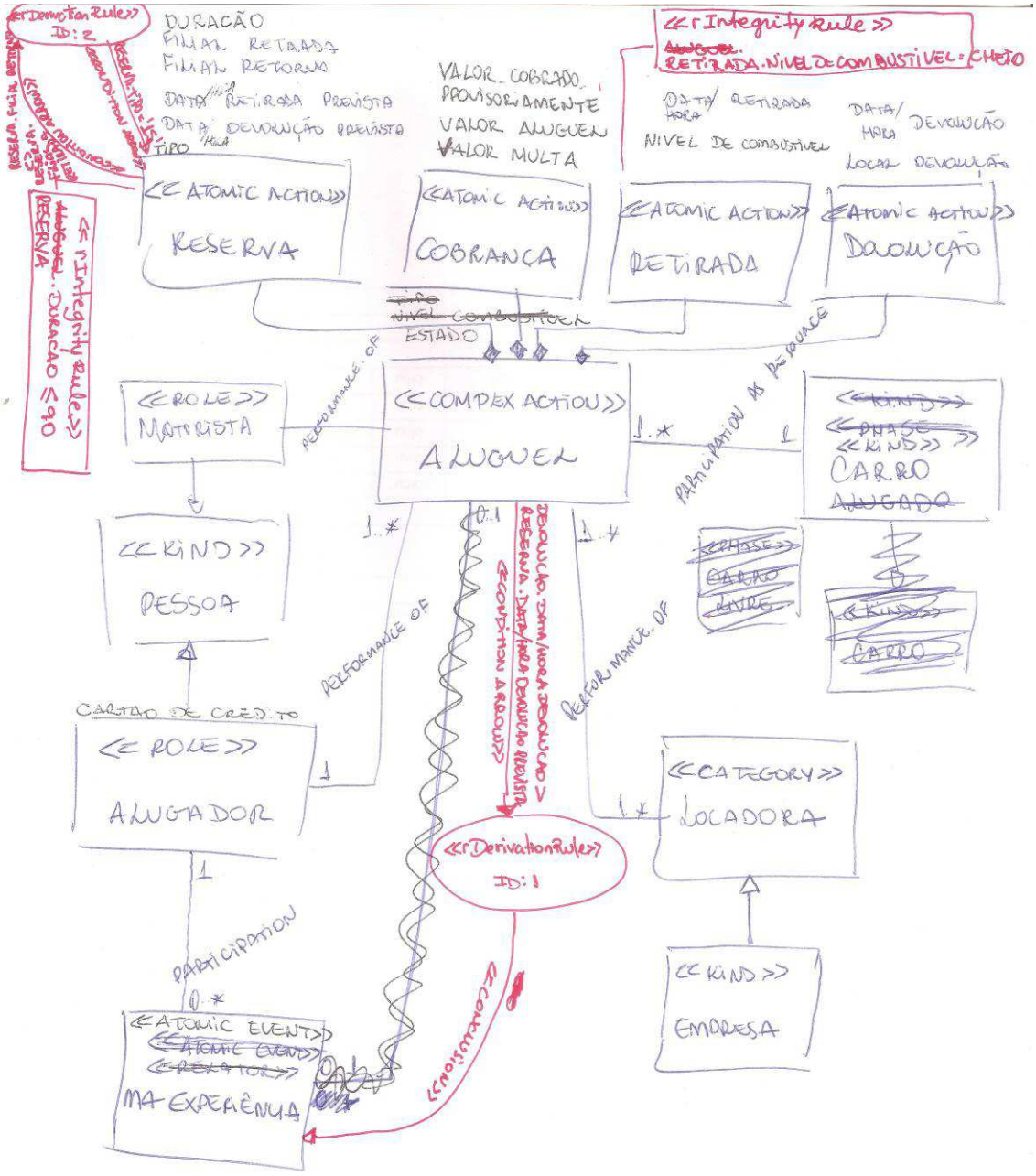


Figura 86 – Primeira parte da modelagem OntoUML e R-OntoUML do primeiro participante.

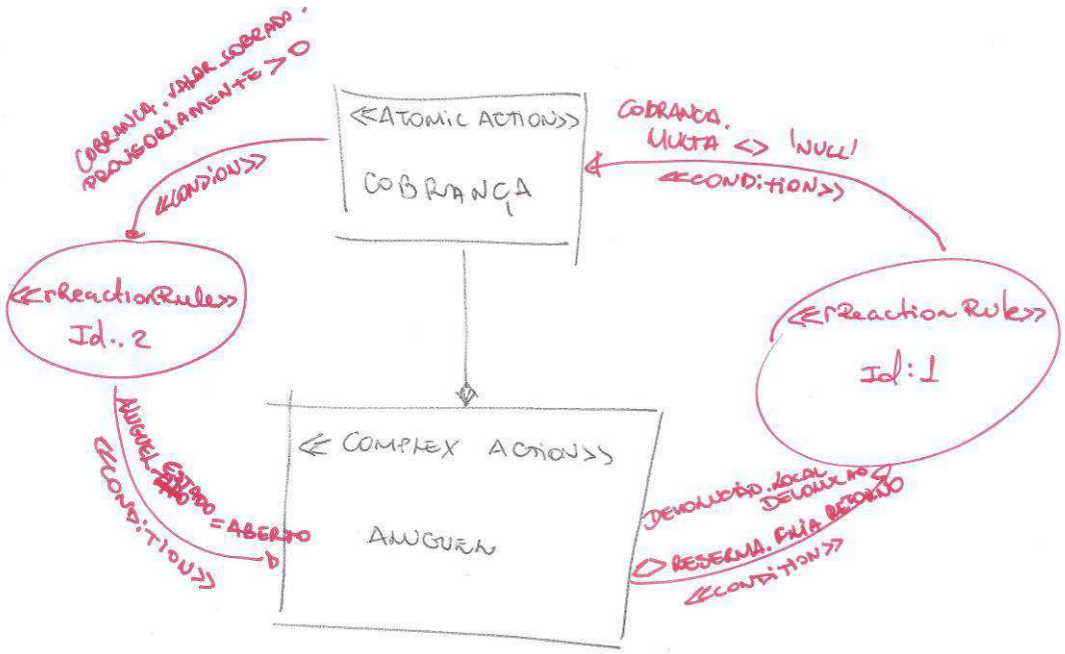


Figura 87 - Segunda parte da modelagem OntoUML e R-OntoUML do primeiro participante.

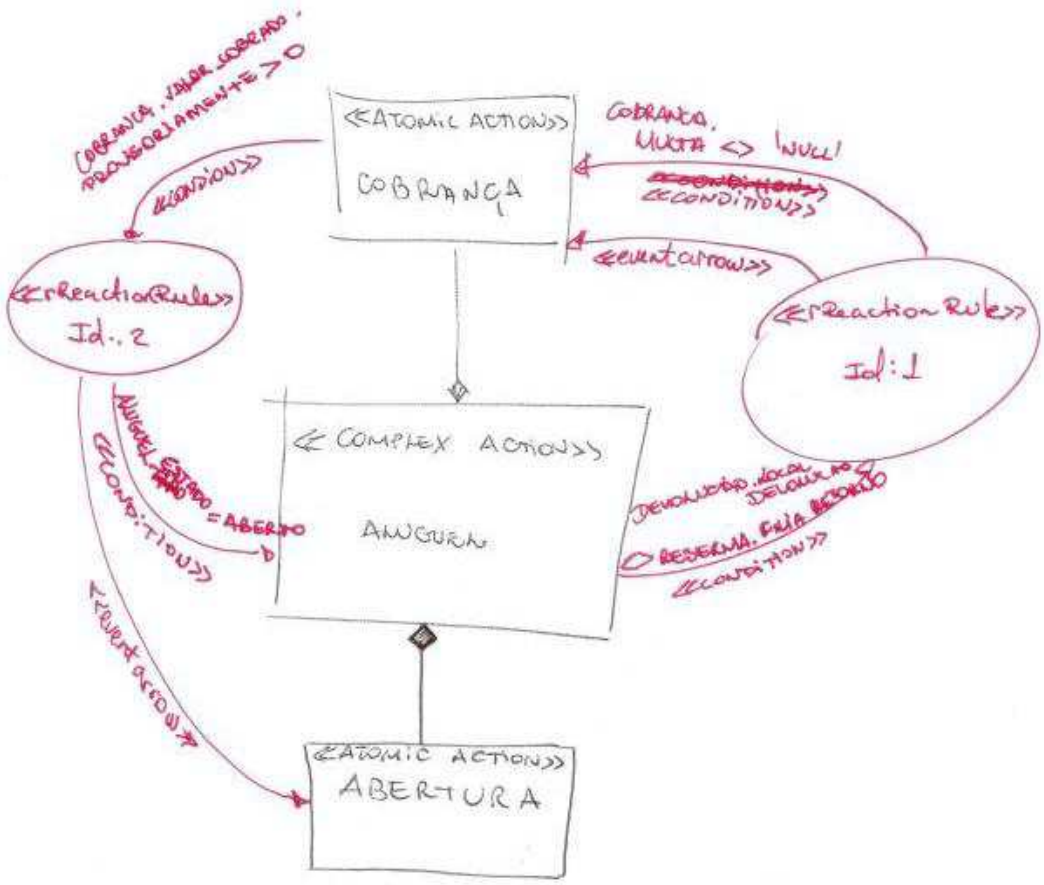


Figura 88 - Segunda parte revisada da modelagem OntoUML e R-OntoUML do primeiro participante.

Anexo B. RESPOSTA DO QUESTIONÁRIO DO PRIMEIRO PARTICIPANTE

Validação:

Depois da modelagem usando UML e URML.

- (Completeza) O modelo contém todas as afirmativas das regras de negócio que são corretas e relevantes? (Para esta avaliação, verifique se é possível identificar no modelo todas as informações que estão mencionadas nas regras)

  Responda de 1 a 4, e justifique a resposta caso tenha marcado 2 ou 3:

  1) Não contém as afirmativas corretas e relevantes descritas nas regras de negócio.
  2) Contém poucas afirmativas corretas e relevantes descritas nas regras de negócio.
  3) Contém muitas afirmativas corretas e relevantes descritas nas regras de negócio.
  4) Contém as afirmativas corretas e relevantes descritas nas regras de negócio.

  Quais informações estão faltando no modelo?

OBS:

DIFICULDADE REPRESENTAÇÃO DA COBRANÇA

- (Validade) Todas as afirmações do modelo são corretas e relevantes ao domínio descrito pelas regras de negócio? (Para esta avaliação, verifique se existem informações no modelo que não estejam descritas nas regras)

  Responda de 1 a 4, e justifique a resposta caso tenha marcado 2 ou 3:

  1) Contém afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  2) Contém muitas afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  3) Contém poucas afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  4) Contém nenhuma afirmativa que não seja correta e relevante ao domínio descrito nas regras de negócio.

  Quais informações que não são corretas e relevantes ao domínio que estão modeladas?

Ex: RR 10:2 ESSA REGRA OBRIGA QUE O ESTADO

DO ALUGUEL SEJA SEMPRE "ABERTO", QUANDO PODERIA (DE ACORDO COM O DOMÍNIO) PASSAR PARA ESTADOS SEGUINTES (EM CURSO, TERMINADO).

- IMAGINANDO QUE VOCÊ FOSSE O ESPECIALISTA DO DOMÍNIO, QUAIS INFORMAÇÕES QUE VOCÊ SENTIU FALTA NAS REGRAS QUE FAZEM FALTA PARA O ENTENDIMENTO DO DOMÍNIO?

INFORMAÇÕES SOBRE A COBRANÇA PROVISÓRIA DO ALUGUEL E DAS CONDIÇÕES PARA OCORRÊNCIA DA MESMA.

Depois da modelagem usando OntoUML e R-OntoML.

- (Completeza) O modelo contém todas as afirmativas das regras de negócio que são corretas e relevantes? (Para esta avaliação, verifique se é possível identificar no modelo todas as informações que estão mencionadas nas regras)

  Responda de 1 a 4, e justifique a resposta caso tenha marcado 2 ou 3:

  1) Não contém as afirmativas corretas e relevantes descritas nas regras de negócio.
  2) Contém poucas afirmativas corretas e relevantes descritas nas regras de negócio.
  3) Contém muitas afirmativas corretas e relevantes descritas nas regras de negócio.
  (4) Contém as afirmativas corretas e relevantes descritas nas regras de negócio.

  Quais informações estão faltando no modelo?

  _____

  _____

  _____

  _____

  _____

- (Validade) Todas as afirmações do modelo são corretas e relevantes ao domínio descrito pelas regras de negócio? (Para esta avaliação, verifique se existem informações no modelo que não estejam descritas nas regras)

  Responda de 1 a 4, e justifique a resposta caso tenha marcado 2 ou 3:

  1) Contém afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  2) Contém muitas afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  (3) Contém poucas afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  4) Contém nenhuma afirmativa que não seja correta e relevante ao domínio descrito nas regras de negócio.

  Quais informações que não são corretas e relevantes ao domínio que estão modeladas?

  MESMO CASO DA PRIMEIRA MODELAGEM

  _____

3ª QUESTÃO:

- MESMO CASO DA PRIMEIRA MODELAGEM
- SENTI FALTA DE INFORMAÇÕES SOBRE AS TRANSIÇÕES ENTRE OS ESTADOS DO ANGUEL (ABERTO, EM CURSO, TERMINADO)

Comparando os dois modelos:

- (Completeza) Comparando os modelos, quais as diferenças entre eles em conter todas as afirmativas das regras de negócio que são corretas e relevantes? (Para esta avaliação, verifique se é possível identificar nos modelos todas as informações que estão mencionadas nas regras, e quais as diferenças entre eles)

AVALIO QUE NOS DOIS MODELOS CONTÉM AS AFIRMATIVAS DAS REGRAS DE NEGÓCIO CORRETAS E RELEVANTES

- (Validade) Comparando os modelos, todas as afirmações corretas e relevantes ao domínio descrito pelas regras de negócio aparecem nos modelos? (Para esta avaliação, verifique se existem informações nos modelos que não estejam descritas nas regras, e o que pode ter ocasionado isso)

NOS DOIS MODELOS IDENTIFIQUEI UMA INCONSISTÊNCIA NA SEGUNDA REGRA DE REAÇÃO.

- (Generalidade) Qual modelo dá maior liberdade para instanciação dos objetos do mundo real em cada classe/relacionamento/regra, permitindo com que cada conceito tenha um conjunto maior de elementos do mundo para instanciação? (As meta-propriedades dos construtos permitem que cada construto represente mais conceitos do mundo real)

A MODELAGEM UML e URML. OS CONSTRUTOS NÃO IMPÕEM TANTAS META-PROPRIEDADES, CONTUDO A MODELAGEM É MAIS SIMPLES NO SENTIDO DE NÃO EXIGIR O ENTENDIMENTO DOS CONSTRUTOS DA UFO

- (Precisão) Qual modelo dá maior especificidade/precisão na instanciação dos objetos do mundo real em cada classe/relacionamento/regra? (As meta-propriedades dos construtos delimitam com maior precisão a quantidade de conceitos do mundo real que cada construto pode representar)

A MODELAGEM R-ONTO UML. AS META-PROPRIEDADES, QUANDO BEM COMPREENDIDAS, GUIAM A MODELAGEM, DELIMITANDO OS AVANÇOS DO "MODELADOR" E TRAZENDO PARA OS OBJETOS SEMÂNTICA ADICIONAL, O QUE É POSITIVO NO PROCESSO.

Anexo C. SEGUNDA ETAPA DA VALIDAÇÃO DO PRIMEIRO PARTICIPANTE

Validação:

Levando em consideração o modelo criado usando as linguagens UML e URML, responda as perguntas a seguir.

- (Completeza) O modelo contém todas as afirmativas das regras de negócio que são corretas e relevantes? (Para esta avaliação, verifique se é possível identificar no modelo todas as informações que estão mencionadas nas regras)

  Responda de 1 a 4, e justifique a resposta caso tenha marcado 2 ou 3:

  1) Não contém as afirmativas corretas e relevantes descritas nas regras de negócio.
  2) Contém poucas afirmativas corretas e relevantes descritas nas regras de negócio.
  3) Contém muitas afirmativas corretas e relevantes descritas nas regras de negócio.
  4) Contém as afirmativas corretas e relevantes descritas nas regras de negócio.

  Quais informações estão faltando no modelo?

- (Validade) Todas as afirmações do modelo são corretas e relevantes ao domínio descrito pelas regras de negócio? (Para esta avaliação, verifique se existem informações no modelo que não estejam descritas nas regras)

  Responda de 1 a 4, e justifique a resposta caso tenha marcado 2 ou 3:

  1) Contém afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  2) Contém muitas afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.

3) Contém poucas afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
4) Contém nenhuma afirmativa que não seja correta e relevante ao domínio descrito nas regras de negócio.

Quais informações que não são corretas e relevantes ao domínio que estão modeladas?

O modelo obriga que um carro esteja vinculado a apenas um aluguel e que o nível de combustível desse carro esteja sempre cheio (não percebi a diferença entre o carro (físico) e o carro alugado (carro como recurso de um aluguel). O fato de haver uma cobrança provisória também implica que o estado do aluguel seja sempre aberto.

Levando em consideração o modelo criado usando as linguagens OntoUML e R-OntoUML, responda as perguntas a seguir.

- (Completeza) O modelo contém todas as afirmativas das regras de negócio que são corretas e relevantes? (Para esta avaliação, verifique se é possível identificar no modelo todas as informações que estão mencionadas nas regras)

  Responda de 1 a 4, e justifique a resposta caso tenha marcado 2 ou 3:

  1) Não contém as afirmativas corretas e relevantes descritas nas regras de negócio.
  2) Contém poucas afirmativas corretas e relevantes descritas nas regras de negócio.
  3) Contém muitas afirmativas corretas e relevantes descritas nas regras de negócio.
  4) Contém as afirmativas corretas e relevantes descritas nas regras de negócio.

  Quais informações estão faltando no modelo?

- (Validade) Todas as afirmações do modelo são corretas e relevantes ao domínio descrito pelas regras de negócio? (Para esta avaliação, verifique se existem informações no modelo que não estejam descritas nas regras)

  Responda de 1 a 4, e justifique a resposta caso tenha marcado 2 ou 3:

  1) Contém afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  2) Contém muitas afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  3) Contém poucas afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  4) Contém nenhuma afirmativa que não seja correta e relevante ao domínio descrito nas regras de negócio.

  Quais informações que não são corretas e relevantes ao domínio que estão modeladas?

Não consegui perceber a partir da modelagem o relacionamento dos eventos com as fases do aluguel (na primeira modelagem essa informação é tratada como um atributo

do aluguel e esses relacionamentos ficam implícitos para o modelador). Ao tratar essas fases como momentos distintos e disjuntos do processo esperaria que os eventos possíveis estivessem relacionados a essas fases distintas.

Comparando os dois modelos:

- (Completeza) Comparando os modelos, quais as diferenças entre eles em conter todas as afirmativas das regras de negócio que são corretas e relevantes? (Para esta avaliação, verifique se é possível identificar nos modelos todas as informações que estão mencionadas nas regras, e quais as diferenças entre eles)

Acredito que os dois modelos estão completos.

- (Validade) Comparando os modelos, todas as afirmações corretas e relevantes ao domínio descrito pelas regras de negócio aparecem nos modelos? (Para esta avaliação, verifique se existem informações nos modelos que não estejam descritas nas regras, e o que pode ter ocasionado isso)

Achei difícil avaliar a validade do modelo em OntoUML pois como os construtos implicam um volume considerável de meta-propriedades, perceber informações adicionais no modelo que não estão explícitas nas regras requer uma compreensão muito profunda desses construtos e dos relacionamentos entre eles. O modelo UML contém algumas informações além das regras que são mais facilmente identificadas. Embora não tenha conseguido precisar todos os pontos, sinto que o modelo em OntoUML traz mais dificuldades na garantia da validade do modelo.

- (Generalidade) Qual modelo dá maior liberdade para instanciação dos objetos do mundo real em cada classe/relacionamento/regra, permitindo com que cada conceito tenha um conjunto maior de elementos do mundo para instanciação? (As meta-propriedades dos construtos permitem que cada construto represente mais conceitos do mundo real)

UML e RUML.

- (Precisão) Qual modelo dá maior especificidade/precisão na instanciação dos objetos do mundo real em cada classe/relacionamento/regra? (As meta-propriedades dos construtos delimitam com maior precisão a quantidade de conceitos do mundo real que cada construto pode representar)

OntoUML e R-OntoUML.

Percepção de uso da OntoUML e R-OntoUML

Escala	Concordo fortemente	Concordo	Concordo em parte	Neutro	Discordo em parte	Discordo	Discordo fortemente
1. Acredito que a técnica de modelagem é desconfortável de usar.	1	2	3	4	5	6	7
2. Usar a técnica de modelagem foi frustrante.	1	2	3	4	5	6	7
3. Usar a técnica de modelagem exigiu muito esforço mental.	1	2	3	4	5	6	7
4. A técnica de modelagem é clara compreensível para mim.	1	2	3	4	5	6	7
5. Considerando tudo, acredito que a técnica de modelagem é fácil de usar.	1	2	3	4	5	6	7

Por favor, explique como foi a experiência de uso da OntoUML e R-OntoUML.

A complexidade introduzida pela utilização dos construtos da UFO torna um pouco mais difícil a tarefa de modelagem mas ao mesmo tempo as meta-propriedades funcionam como um “guia”, limitando os caminhos do modelador e garantindo uma maior consistência na modelagem (escolha de entidades e relacionamentos). Senti um pouco de dificuldade de compreender todos os construtos, mas nos momentos em que identifiquei uma “saída”, a continuidade da modelagem foi mais fácil. Contudo, perceber que outra pessoa utilizou os construtos de forma muito diferente me faz perceber ainda mais a importância de entendimento desses construtos para que a OntoUML funcione de fato como um guia da modelagem e as meta-propriedades

impliquem mais em ganho semântico do que dificuldade de garantir a validade do modelo.

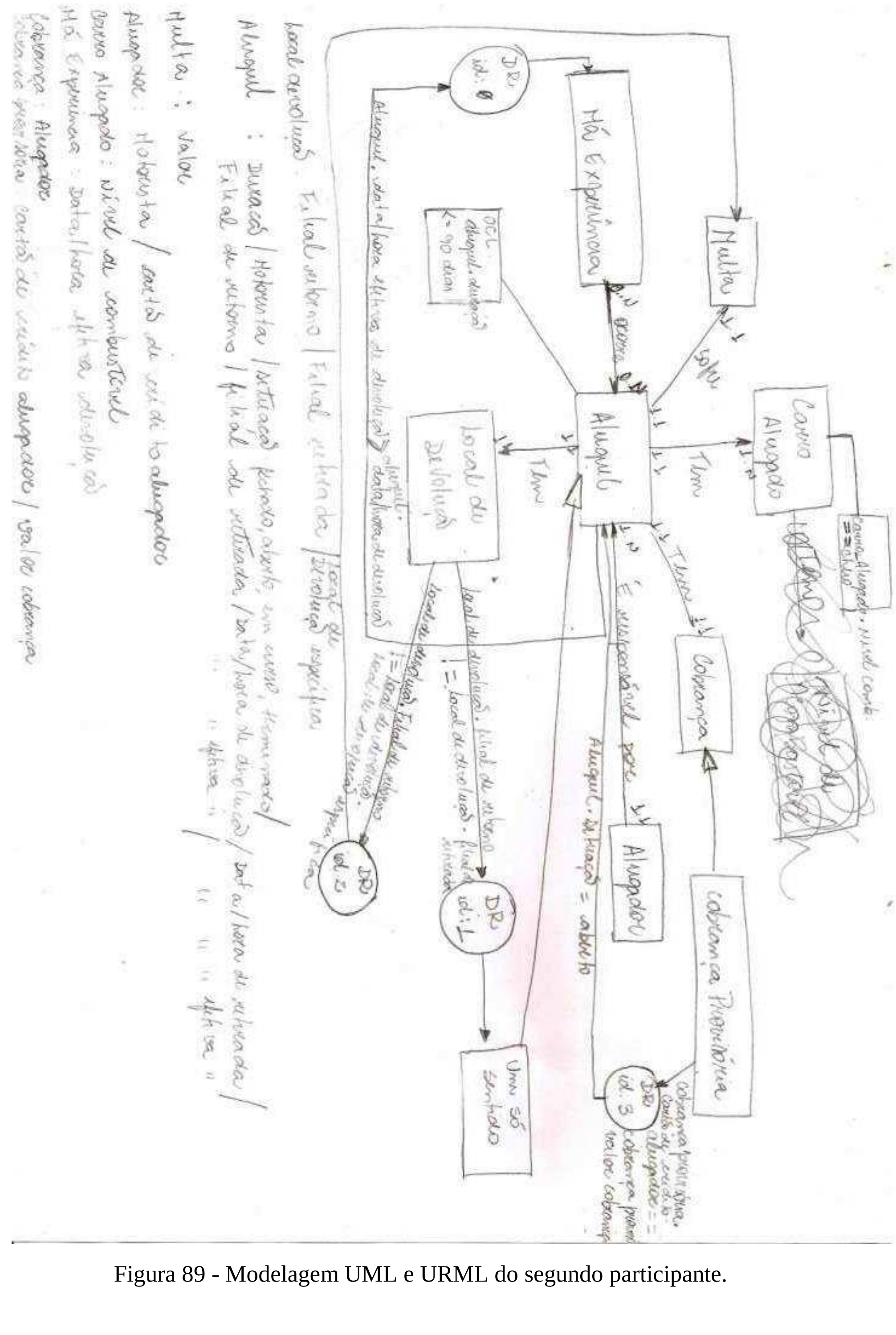


Figura 89 - Modelagem UML e URML do segundo participante.

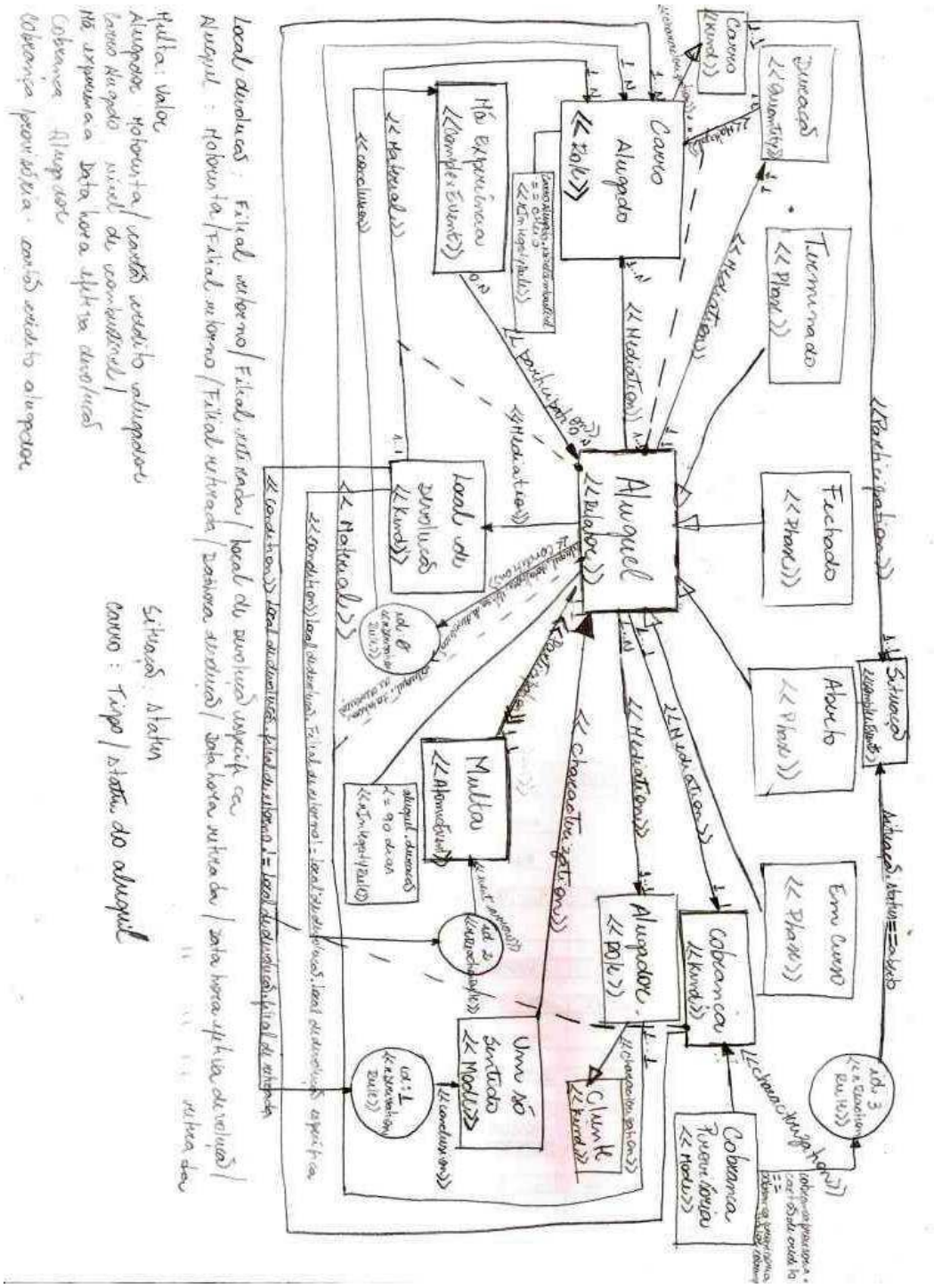
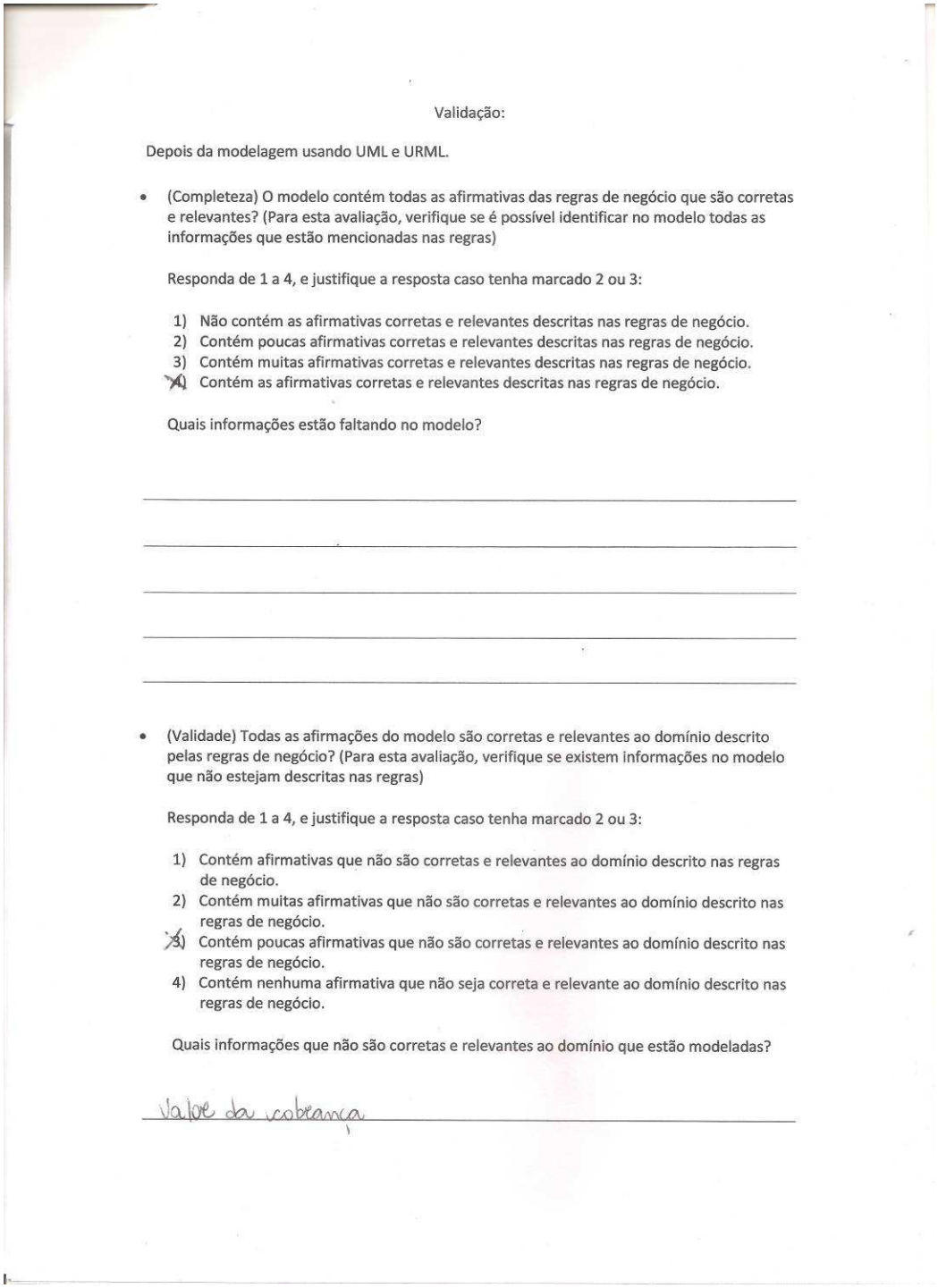


Figura 90 - Modelagem OntoUML e R-OntoUML do segundo participante.

Anexo E. RESPOSTA DO QUESTIONÁRIO DO SEGUNDO PARTICIPANTE



Validação:

Depois da modelagem usando UML e URML.

- (Completeza) O modelo contém todas as afirmativas das regras de negócio que são corretas e relevantes? (Para esta avaliação, verifique se é possível identificar no modelo todas as informações que estão mencionadas nas regras)

  Responda de 1 a 4, e justifique a resposta caso tenha marcado 2 ou 3:

  1) Não contém as afirmativas corretas e relevantes descritas nas regras de negócio.
  2) Contém poucas afirmativas corretas e relevantes descritas nas regras de negócio.
  3) Contém muitas afirmativas corretas e relevantes descritas nas regras de negócio.
  4) Contém as afirmativas corretas e relevantes descritas nas regras de negócio.

  Quais informações estão faltando no modelo?

- (Validade) Todas as afirmações do modelo são corretas e relevantes ao domínio descrito pelas regras de negócio? (Para esta avaliação, verifique se existem informações no modelo que não estejam descritas nas regras)

  Responda de 1 a 4, e justifique a resposta caso tenha marcado 2 ou 3:

  1) Contém afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  2) Contém muitas afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  3) Contém poucas afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  4) Contém nenhuma afirmativa que não seja correta e relevante ao domínio descrito nas regras de negócio.

  Quais informações que não são corretas e relevantes ao domínio que estão modeladas?

  Valor da cobrança

- IMAGINANDO QUE VOCÊ FOSSE O ESPECIALISTA DO DOMÍNIO, QUAIS INFORMAÇÕES QUE VOCÊ SENTIU FALTA NAS REGRAS QUE FAZEM FALTA PARA O ENTENDIMENTO DO DOMÍNIO?

R: Não senti falta de informações do domínio para seu entendimento.

Depois da modelagem usando OntoUML e R-OntoML.

- (Completeza) O modelo contém todas as afirmativas das regras de negócio que são corretas e relevantes? (Para esta avaliação, verifique se é possível identificar no modelo todas as informações que estão mencionadas nas regras)

  Responda de 1 a 4, e justifique a resposta caso tenha marcado 2 ou 3:

  1) Não contém as afirmativas corretas e relevantes descritas nas regras de negócio.
  2) Contém poucas afirmativas corretas e relevantes descritas nas regras de negócio.
  3) Contém muitas afirmativas corretas e relevantes descritas nas regras de negócio.
  4) X Contém as afirmativas corretas e relevantes descritas nas regras de negócio.

  Quais informações estão faltando no modelo?

- (Validade) Todas as afirmações do modelo são corretas e relevantes ao domínio descrito pelas regras de negócio? (Para esta avaliação, verifique se existem informações no modelo que não estejam descritas nas regras)

  Responda de 1 a 4, e justifique a resposta caso tenha marcado 2 ou 3:

  1) Contém afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  2) Contém muitas afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  3) X Contém poucas afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  4) Contém nenhuma afirmativa que não seja correta e relevante ao domínio descrito nas regras de negócio.

  Quais informações que não são corretas e relevantes ao domínio que estão modeladas?

  Assim como na UML e URML, senti falta do valor da cobrança

Comparando os dois modelos:

- (Completeza) Comparando os modelos, quais as diferenças entre eles em conter todas as afirmativas das regras de negócio que são corretas e relevantes? (Para esta avaliação, verifique se é possível identificar nos modelos todas as informações que estão mencionadas nas regras, e quais as diferenças entre eles)

É possível verificar nos modelos todas as informações relatadas nas regras. Como o domínio analisado não apresentou muita complexidade, acredito que isto justifique a minha afirmação

- (Validade) Comparando os modelos, todas as afirmações corretas e relevantes ao domínio descrito pelas regras de negócio aparecem nos modelos? (Para esta avaliação, verifique se existem informações nos modelos que não estejam descritas nas regras, e o que pode ter ocasionado isso)

Não. Nem todas aparecem. Exemplo: Valor da cobrança. O motivo desta ocorrência pode ser justificado pelo meu entendimento pessoal do domínio.

- (Generalidade) Qual modelo dá maior liberdade para instanciação dos objetos do mundo real em cada classe/relacionamento/regra, permitindo com que cada conceito tenha um conjunto maior de elementos do mundo para instanciação? (As meta-propriedades dos construtos permitem que cada construto represente mais conceitos do mundo real)

A aplicação do OntoUML e R-OntoUML agrega maior semântica ao modelo construído. As restrições de relacionamentos entre os construtos permitem retirar dos modelos grande parte da ambiguidade na representação. Portanto acredito que a UML permite que cada conceito representado possua um maior conjunto de elementos reais instanciados, visto que seus construtos são apresentados com menores restrições, assim como a UFO.

- (Precisão) Qual modelo dá maior especificidade/precisão na instanciação dos objetos do mundo real em cada classe/relacionamento/regra? (As meta-propriedades dos construtos delimitam com maior precisão a quantidade de conceitos do mundo real que cada construto pode representar)

Conforme respondido na pergunta anterior, a ONTOUML e R-ONTOUML permite a criação de modelos menos ambíguos.

Anexo F. SEGUNDA ETAPA DA VALIDAÇÃO DO SEGUNDO PARTICIPANTE

Validação:

Levando em consideração o modelo criado usando as linguagens UML e URML, responda as perguntas a seguir.

- (Completeza) O modelo contém todas as afirmativas das regras de negócio que são corretas e relevantes? (Para esta avaliação, verifique se é possível identificar no modelo todas as informações que estão mencionadas nas regras)

  Responda de 1 a 4, e justifique a resposta caso tenha marcado 2 ou 3:

  1) Não contém as afirmativas corretas e relevantes descritas nas regras de negócio.
  2) Contém poucas afirmativas corretas e relevantes descritas nas regras de negócio.
  3) Contém muitas afirmativas corretas e relevantes descritas nas regras de negócio.
  4) Contém as afirmativas corretas e relevantes descritas nas regras de negócio.

  Quais informações estão faltando no modelo?

- (Validade) Todas as afirmações do modelo são corretas e relevantes ao domínio descrito pelas regras de negócio? (Para esta avaliação, verifique se existem informações no modelo que não estejam descritas nas regras)

  Responda de 1 a 4, e justifique a resposta caso tenha marcado 2 ou 3:

  1) Contém afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  2) Contém muitas afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  3) Contém poucas afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  4) Contém nenhuma afirmativa que não seja correta e relevante ao domínio descrito nas regras de negócio.

  Quais informações que não são corretas e relevantes ao domínio que estão modeladas?

Clube fidelidade que foi apresentado como um atributo da classe Alugador

Levando em consideração o modelo criado usando as linguagens OntoUML e R-OntoUML, responda as perguntas a seguir.

- (Completeza) O modelo contém todas as afirmativas das regras de negócio que são corretas e relevantes? (Para esta avaliação, verifique se é possível identificar no modelo todas as informações que estão mencionadas nas regras)

  Responda de 1 a 4, e justifique a resposta caso tenha marcado 2 ou 3:

  1) Não contém as afirmativas corretas e relevantes descritas nas regras de negócio.
  2) Contém poucas afirmativas corretas e relevantes descritas nas regras de negócio.
  3) Contém muitas afirmativas corretas e relevantes descritas nas regras de negócio.
  4) Contém as afirmativas corretas e relevantes descritas nas regras de negócio.

  Quais informações estão faltando no modelo?

- (Validade) Todas as afirmações do modelo são corretas e relevantes ao domínio descrito pelas regras de negócio? (Para esta avaliação, verifique se existem informações no modelo que não estejam descritas nas regras)

  Responda de 1 a 4, e justifique a resposta caso tenha marcado 2 ou 3:

  1) Contém afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  2) Contém muitas afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  3) Contém poucas afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  4) Contém nenhuma afirmativa que não seja correta e relevante ao domínio descrito nas regras de negócio.

  Quais informações que não são corretas e relevantes ao domínio que estão modeladas?

Cliente, AlugadorIndiv, AlugadorEmpr, Organização, Pessoa. Essas informações foram inseridas para melhor representação inserção dos conceitos da UFO. Elas não foram apresentadas nas regras no nível de detalhe apresentado na modelagem.

Comparando os dois modelos:

- (Completeza) Comparando os modelos, quais as diferenças entre eles em conter todas as afirmativas das regras de negócio que são corretas e relevantes? (Para esta avaliação, verifique se é possível identificar nos modelos todas as informações que estão mencionadas nas regras, e quais as diferenças entre eles)

Ambos os modelos apresentaram de forma geral as informações relevantes. Como principal diferença entre eles, o modelo usando OntoUML e R-OntoUML exigiu definir melhor cada elemento do modelo. a fim de que fosse possível atribuir os estereótipos da UFO. Isso se dá devido as regras inerentes à abordagem de representação proposta pela UFO.

- (Validade) Comparando os modelos, todas as afirmações corretas e relevantes ao domínio descrito pelas regras de negócio aparecem nos modelos? (Para esta avaliação, verifique se existem informações nos modelos que não estejam descritas nas regras, e o que pode ter ocasionado isso)

De fato, existem informações a mais no modelo que usou OntoUML e R-OntoUML . Isto ocorreu porque para fosse possível atribuir estereótipos da UFO na modelagem conceitual, o modelo precisou estar o menos ambíguo possível, a fim de que as regras inerentes à abordagem de representação proposta pela UFO fosse aplicada.

- (Generalidade) Qual modelo dá maior liberdade para instanciação dos objetos do mundo real em cada classe/relacionamento/regra, permitindo com que cada conceito tenha um conjunto maior de elementos do mundo para instanciação? (As meta-propriedades dos construtos permitem que cada construto represente mais conceitos do mundo real)

O modelo UML e URML.

- (Precisão) Qual modelo dá maior especificidade/precisão na instanciação dos objetos do mundo real em cada classe/relacionamento/regra? (As meta-propriedades dos construtos delimitam com maior precisão a quantidade de conceitos do mundo real que cada construto pode representar)

Percepção de uso da OntoUML e R-OntoUML

Escala	Concordo fortemente	Concordo	Concordo em parte	Neutro	Discordo em parte	Discordo	Discordo fortemente
1. Acredito que a técnica de modelagem é desconfortável de usar.	1	2	3	4	5	6	7
2. Usar a técnica de modelagem foi frustrante.	1	2	3	4	5	6	7
3. Usar a técnica de modelagem exigiu muito esforço mental.	1	2	3	4	5	6	7
4. A técnica de modelagem é clara compreensível para mim.	1	2	3	4	5	6	7
5. Considerando tudo, acredito que a técnica de modelagem é fácil de usar.	1	2	3	4	5	6	7

Por favor, explique como foi a experiência de uso da OntoUML e R-OntoUML.

Ficou clara a qualidade semântica nos modelos onde a OntoUML e R-OntoUML foram aplicadas. O principal problema que eu identifiquei é quanto ao uso de OntoUML e R-OntoUML. Os conhecimentos necessários para esta aplicação não são triviais. Entender e por em prática como cada conceito e relacionamento se comportam não é tranqüilo, principalmente em pouco tempo. Com isso, identificar a relação entre conceitos e relacionamentos do cenário proposto com os definidos na estrutura da OntoUML e R-OntoUML foi trabalhoso .

Acredito que com a estrutura de OntoUML e R-OntoUML bem fundamentada nos conhecimento do modelador, a construção dos modelos será mais rápida e confiável, retratando um cenário com uma carga semântica mais próxima do real. Em situação contrária, ou seja, com o modelador tendo pouco ou nenhum conhecimento sobre

OntoUML e R-OntoUML, não é possível garantir que modelagem corresponderá as expectativas.

Anexo G. MODELO DO TERCEIRO PARTICIPANTE

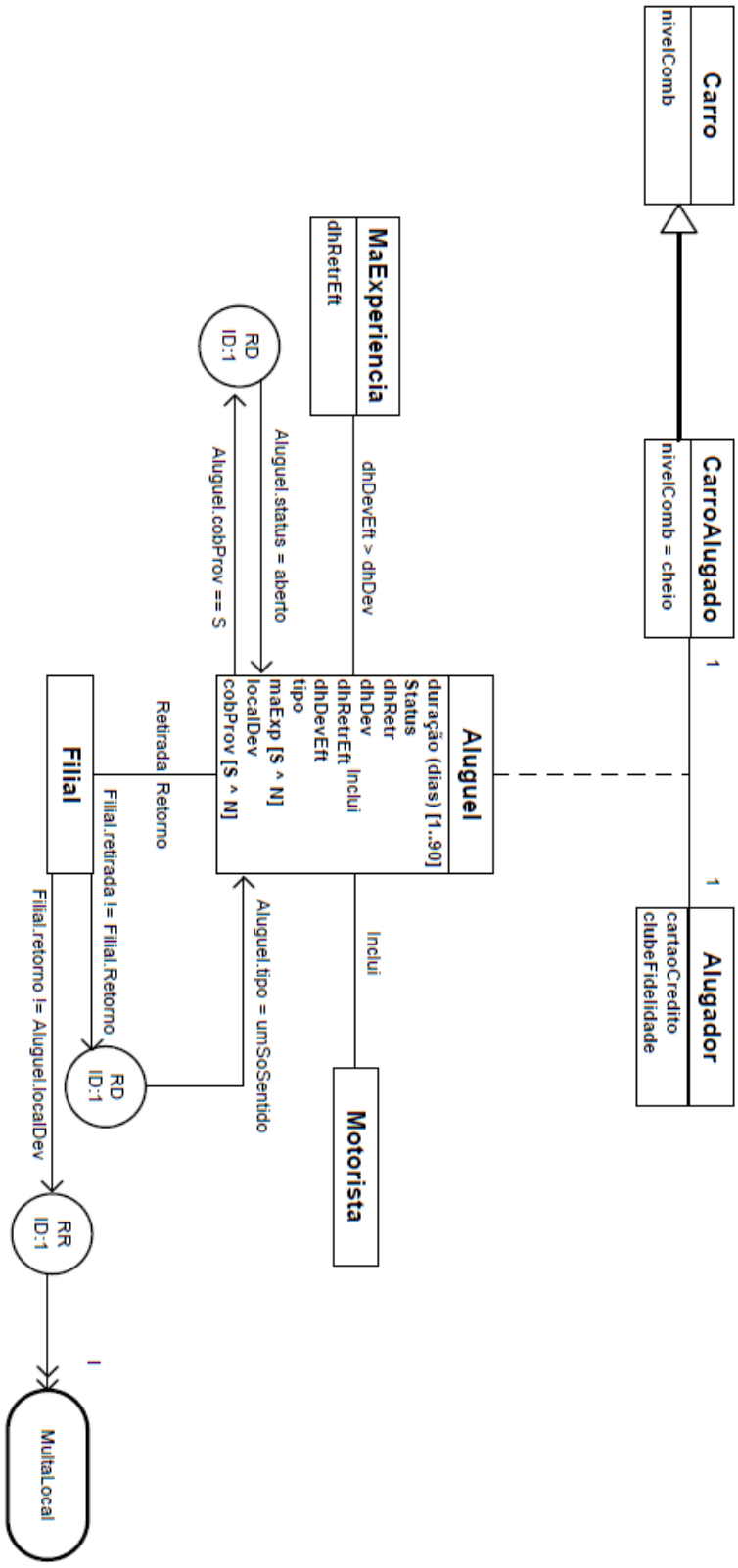


Figura 91 - - Modelagem UML e URML do terceiro participante inicial.

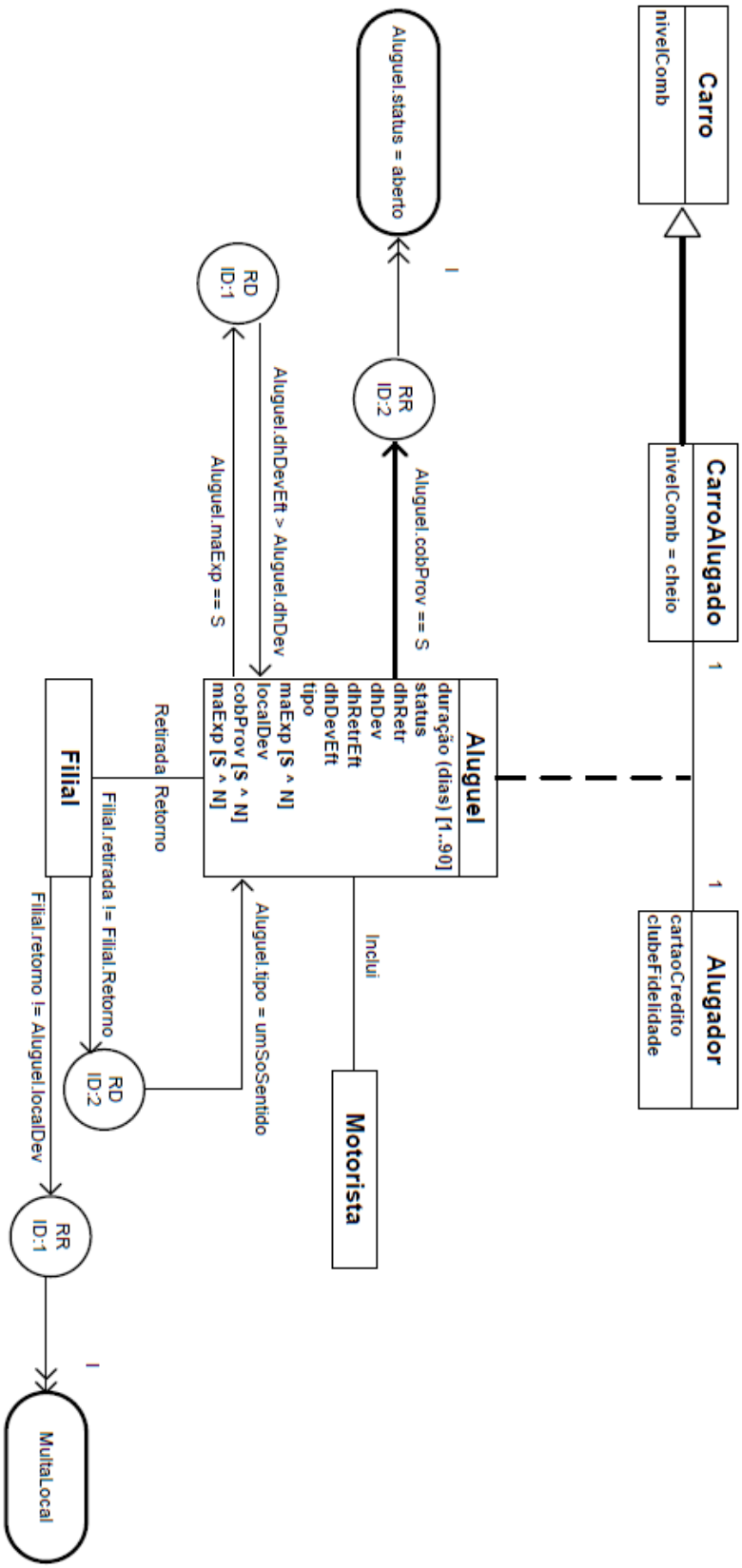


Figura 92 - Modelagem UML e URML do terceiro participante corrigido.

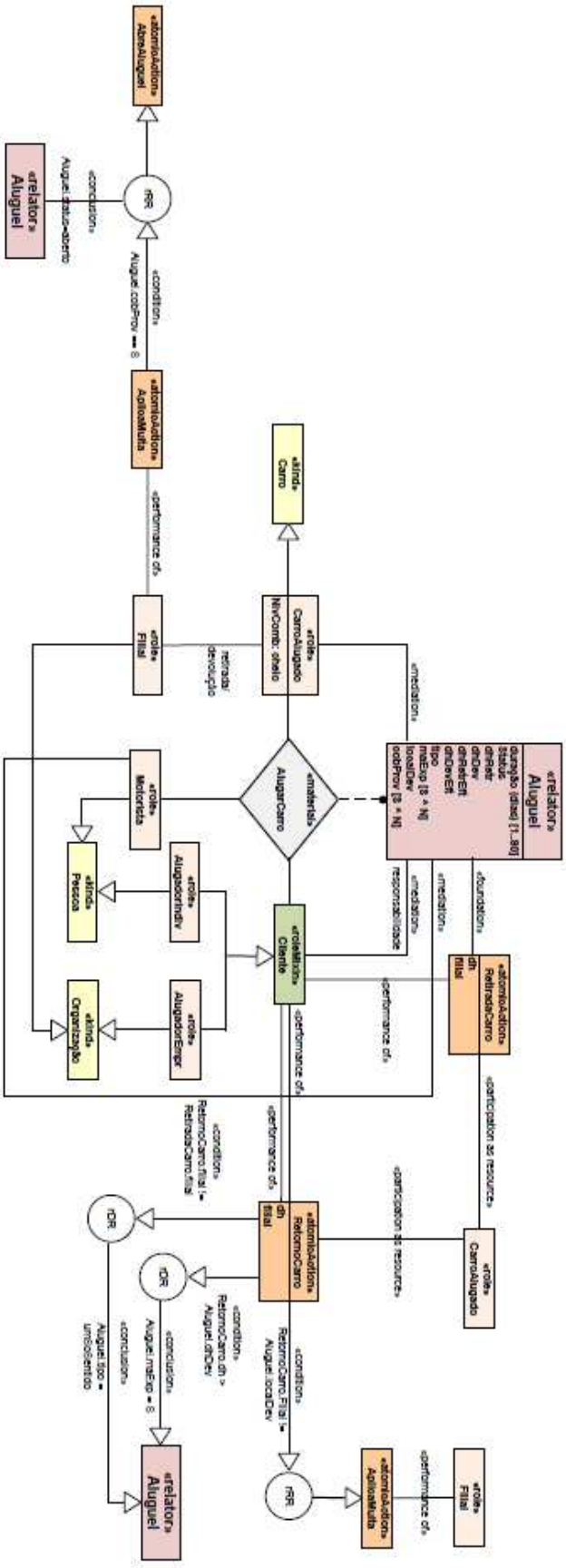


Figura 93 - Modelagem OntoUML e R-OntoUML do terceiro participante.

Anexo H. RESPOSTA DO QUESTIONÁRIO DO TERCEIRO PARTICIPANTE

Validação:

Depois da modelagem usando UML e URML.

- (Completeza) O modelo contém todas as afirmativas das regras de negócio que são corretas e relevantes? (Para esta avaliação, verifique se é possível identificar no modelo todas as informações que estão mencionadas nas regras)

  Responda de 1 a 4, e justifique a resposta caso tenha marcado 2 ou 3:

  1) Não contém as afirmativas corretas e relevantes descritas nas regras de negócio.
  2) Contém poucas afirmativas corretas e relevantes descritas nas regras de negócio.
  3) Contém muitas afirmativas corretas e relevantes descritas nas regras de negócio.
  4) Contém as afirmativas corretas e relevantes descritas nas regras de negócio.

  Quais informações estão faltando no modelo?

3: "Alugador é responsável pelo aluguel" não pode ser lida explicitamente no modelo; a partir do modelo não se vê que a multa local está relacionada ao aluguel; da mesma forma fica confusa a leitura da regra de derivação "se a filial de retorno é diferente da filial de retirada, então o aluguel é do tipo aluguel em um só sentido". Se se levar em conta o "minimundo", falta ainda a modelagem do cliente e de seus tipos.

- (Validade) Todas as afirmações do modelo são corretas e relevantes ao domínio descrito pelas regras de negócio? (Para esta avaliação, verifique se existem informações no modelo que não estejam descritas nas regras)

  Responda de 1 a 4, e justifique a resposta caso tenha marcado 2 ou 3:

  1) Contém afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  2) Contém muitas afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  3) Contém poucas afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  4) Contém nenhuma afirmativa que não seja correta e relevante ao domínio descrito nas regras de negócio.

  Quais informações que não são corretas e relevantes ao domínio que estão modeladas?

3: a especialização carro ⟵ carro alugado inclui um atributo nível de combustível na classe mais genérica que não está estipulado nas regras de negócio.

- Imaginando que você fosse o especialista do domínio, quais informações de que você sentiu falta nas regras que fazem falta para o entendimento do domínio?

Aluguel tem motorista – diante dessa afirmativa, considerei que todo aluguel de carro inclui serviço de motorista. Isso é verdadeiro?

Aluguel pode estar fechado, aberto, em curso e terminado. O que leva à mudança de estado? Qual a diferença entre fechado e terminado?

Local de devolução e filial de retorno são a mesma coisa? Se não, qual a diferença?

Todo alugador tem que ter cartão de crédito? Se não, em que casos são necessárias as cobranças provisórias?

O que quer dizer "alugador é responsável pelo aluguel"?

Carro alugado tem nível de combustível; carro devolvido não tem? Nem carro para alugar?

Qual o efeito de uma má experiência do aluguel? Quem é afetado por essa má experiência?

Se a filial de retorno é igual à de retirada, qual o tipo de aluguel?

Depois da modelagem usando OntoUML e R-OntoML

- (Completeza) O modelo contém todas as afirmativas das regras de negócio que são corretas e relevantes? (Para esta avaliação, verifique se é possível identificar no modelo todas as informações que estão mencionadas nas regras)

  Responda de 1 a 4, e justifique a resposta caso tenha marcado 2 ou 3:

  1) Não contém as afirmativas corretas e relevantes descritas nas regras de negócio.
  2) Contém poucas afirmativas corretas e relevantes descritas nas regras de negócio.
  3) Contém muitas afirmativas corretas e relevantes descritas nas regras de negócio.
  4) Contém as afirmativas corretas e relevantes descritas nas regras de negócio.

  Quais informações estão faltando no modelo?

4

- (Validade) Todas as afirmações do modelo são corretas e relevantes ao domínio descrito pelas regras de negócio? (Para esta avaliação, verifique se existem informações no modelo que não estejam descritas nas regras)

  Responda de 1 a 4, e justifique a resposta caso tenha marcado 2 ou 3:

  1) Contém afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  2) Contém muitas afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  3) Contém poucas afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  4) Contém nenhuma afirmativa que não seja correta e relevante ao domínio descrito nas regras de negócio.

  Quais informações que não são corretas e relevantes ao domínio que estão modeladas?

4

Comparando os dois modelos:

- (Completeza) Comparando os modelos, quais as diferenças entre eles em conter todas as afirmativas das regras de negócio que são corretas e relevantes? (Para esta avaliação, verifique se é possível identificar nos modelos todas as informações que estão mencionadas nas regras, e quais as diferenças entre eles)

O modelo em OntoUML e R-OntoUML é mais completo em relação ao modelo em UML. Os construtos da OntoUML e da R-OntoUML, e as respectivas metapropriedades subjacentes, fornecem maior riqueza de significado, aos conceitos e eventos representados.

- (Validade) Comparando os modelos, todas as afirmações corretas e relevantes ao domínio descrito pelas regras de negócio aparecem nos modelos? (Para esta avaliação, verifique se existem informações nos modelos que não estejam descritas nas regras, e o que pode ter ocasionado isso)

O modelo em OntoUML e R-OntoUML parece ser mais válido em relação ao domínio modelado, visto que seus construtos fornecem maior riqueza semântica e, conseqüentemente, maior fidedignidade.

- (Generalidade) Qual modelo dá maior liberdade para instanciação dos objetos do mundo real em cada classe/relacionamento/regra, permitindo com que cada conceito tenha um conjunto maior de elementos do mundo para instanciação? (As meta-propriedades dos construtos permitem que cada construto represente mais conceitos do mundo real)

O modelo de classes UML é mais genérico – visto que as classes são, quase em sua totalidade, definidas pelas mesmas metapropriedades – é mais genérico e, conseqüentemente, menos preciso. Os conceitos Alugador e Carro, por exemplo, são representados pelo mesmo tipo de classe, ainda que seus significados, bem como as metapropriedades lingüísticas, percam-se em grande parte ma modelagem.

- (Precisão) Qual modelo dá maior especificidade/precisão na instanciação dos objetos do mundo real em cada classe/relacionamento/regra? (As meta-propriedades dos construtos delimitam com maior precisão a quantidade de conceitos do mundo real que cada construto pode representar)

O modelo em OntoUML e R-OntoUML, já que semanticamente mais rico, é mais preciso em termos de representação do domínio.

Anexo I. SEGUNDA ETAPA DA VALIDAÇÃO DO TERCEIRO PARTICIPANTE

Validação:

Levando em consideração o modelo criado usando as linguagens UML e URML, responda as perguntas a seguir.

- (Completeza) O modelo contém todas as afirmativas das regras de negócio que são corretas e relevantes? (Para esta avaliação, verifique se é possível identificar no modelo todas as informações que estão mencionadas nas regras)

  Responda de 1 a 4, e justifique a resposta caso tenha marcado 2 ou 3:

  1) Não contém as afirmativas corretas e relevantes descritas nas regras de negócio.
  2) Contém poucas afirmativas corretas e relevantes descritas nas regras de negócio.
  3) Contém muitas afirmativas corretas e relevantes descritas nas regras de negócio.
  4) Contém as afirmativas corretas e relevantes descritas nas regras de negócio.

  Quais informações estão faltando no modelo?

3. O que falta: Não há nenhuma indicação de “motorista” nem de sua ligação com “aluguel”; não há indicação de “status” de “aluguel” (aberto, em curso, ou terminado); a modelagem não indica que o “alugador” é responsável pelo “aluguel”; a modelagem não indica que “carro” tem “nível de combustível”.

- (Validade) Todas as afirmações do modelo são corretas e relevantes ao domínio descrito pelas regras de negócio? (Para esta avaliação, verifique se existem informações no modelo que não estejam descritas nas regras)

  Responda de 1 a 4, e justifique a resposta caso tenha marcado 2 ou 3:

  1) Contém afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  2) Contém muitas afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  3) Contém poucas afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  4) Contém nenhuma afirmativa que não seja correta e relevante ao domínio descrito nas regras de negócio.

  Quais informações que não são corretas e relevantes ao domínio que estão modeladas?

1. Há uma indicação de “estado” para cobrança provisória, que não é mencionada nas regras; o “nível de combustível” aparece em uma OCL como atributo de “aluguel” e não de “carro”, conforme descrito na regra de integridade; a RR ID:2 implica em “aluguel.estado=aberto” mas não há atributo “estado” modelado em “aluguel”.

Levando em consideração o modelo criado usando as linguagens OntoUML e R-OntoUML, responda as perguntas a seguir.

- (Completeza) O modelo contém todas as afirmativas das regras de negócio que são corretas e relevantes? (Para esta avaliação, verifique se é possível identificar no modelo todas as informações que estão mencionadas nas regras)

  Responda de 1 a 4, e justifique a resposta caso tenha marcado 2 ou 3:

  1) Não contém as afirmativas corretas e relevantes descritas nas regras de negócio.
  2) Contém poucas afirmativas corretas e relevantes descritas nas regras de negócio.
  3) Contém muitas afirmativas corretas e relevantes descritas nas regras de negócio.
  4) Contém as afirmativas corretas e relevantes descritas nas regras de negócio.

  Quais informações estão faltando no modelo?

4. O modelo contém afirmativas corretas e relevantes em relação às regras de negócio; entretanto, a modelagem retrata os conceitos de maneira um tanto confusa.

- (Validade) Todas as afirmações do modelo são corretas e relevantes ao domínio descrito pelas regras de negócio? (Para esta avaliação, verifique se existem informações no modelo que não estejam descritas nas regras)

  Responda de 1 a 4, e justifique a resposta caso tenha marcado 2 ou 3:

  1) Contém afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  2) Contém muitas afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  3) Contém poucas afirmativas que não são corretas e relevantes ao domínio descrito nas regras de negócio.
  4) Contém nenhuma afirmativa que não seja correta e relevante ao domínio descrito nas regras de negócio.

  Quais informações que não são corretas e relevantes ao domínio que estão modeladas?

1. A informação relativa à “reserva”, apesar de existir menção a esse conceito no “minimundo”, não consta das regras nem das restrições; o modelo determina que “filial de retirada”, “filial de retorno”, “data/hora de retirada prevista”, “data/hora de devolução prevista” e “tipo” seriam atributos dessa ação (atomic action) componente da ação compllexa “aluguel”; isso, conceitualmente, não é correto visto que, além de não estar descrito nas descrições e regras, o “minimundo” afirma que não é necessário haver reserva para que o aluguel ocorra – se esses atributos forem de “reserva” e, a ação “aluguel” não incluir um componente reserva, esses dados serão perdidos? Da mesma forma, uma das “Integrity rule” indica que “nivel de combustivel” é um atributo da ação “retirada”, quando a regra estipula que “nivel de combustivel” é relacionado a “carro”. O modelo apresenta ainda um construto «category», identificado como “locadora”,

como supertipo do «kind» “empresa” que não constam das regras e cujos conceitos não ficam claros. O evento atômico “má experiência” tem o «role» “alugador” como participante, sem nenhuma ligação com a ação atômica “devolução”, embora a má experiência ocorra em decorrência de um atraso na devolução.

Comparando os dois modelos:

- (Completeza) Comparando os modelos, quais as diferenças entre eles em conter todas as afirmativas das regras de negócio que são corretas e relevantes? (Para esta avaliação, verifique se é possível identificar nos modelos todas as informações que estão mencionadas nas regras, e quais as diferenças entre eles)

A comparação entre os dois modelos fica comprometida na medida em que os conceitos foram alterados de um modelo para o outro. Entretanto, cabe ressaltar que o modelo construído em OntoUML e R-OntoUML permite uma melhor clareza na leitura.

- (Validade) Comparando os modelos, todas as afirmações corretas e relevantes ao domínio descrito pelas regras de negócio aparecem nos modelos? (Para esta avaliação, verifique se existem informações nos modelos que não estejam descritas nas regras, e o que pode ter ocasionado isso)

Conforme dito na pergunta anterior, há diferenças de modelagem de um modelo para o outro.

- (Generalidade) Qual modelo dá maior liberdade para instanciação dos objetos do mundo real em cada classe/relacionamento/regra, permitindo com que cada conceito tenha um conjunto maior de elementos do mundo para instanciação? (As meta-propriedades dos construtos permitem que cada construto represente mais conceitos do mundo real)

O modelo em UML tende a ser mais genérico, oferecendo mais dificuldade de compreensão dos conceitos, devido a oferecer construtos mais genéricos.

- (Precisão) Qual modelo dá maior especificidade/precisão na instanciação dos objetos do mundo real em cada classe/relacionamento/regra? (As meta-propriedades dos construtos delimitam com maior precisão a quantidade de conceitos do mundo real que cada construto pode representar)

O modelo em OntoUML e R-OntoUML é mais preciso em termos de conceitos e de relacionamentos.

Percepção de uso da OntoUML e R-OntoUML

Escala	Concordo fortemente	Concordo	Concordo em parte	Neutro	Discordo em parte	Discordo	Discordo fortemente
1. Acredito que a técnica de modelagem é desconfortável de usar.	1	2	3	4	5	6	7
2. Usar a técnica de modelagem foi frustrante.	1	2	3	4	5	6	7
3. Usar a técnica de modelagem exigiu muito esforço mental.	1	2	3	4	5	6	7
4. A técnica de modelagem é clara compreensível para mim.	1	2	3	4	5	6	7
5. Considerando tudo, acredito que a técnica de modelagem é fácil de usar.	1	2	3	4	5	6	7

Por favor, explique como foi a experiência de uso da OntoUML e R-OntoUML.

Construir modelos em OntoUML e R-OntoUML exige um conhecimento maior dos construtos da linguagem, conhecimento esse que não se adquire de forma intuitiva. Sendo a linguagem relativamente, ou completamente, nova, a construção de modelos exige um tempo maior, visto que o conhecimento necessário a essa tarefa ainda não foi assimilado a ponto de permitir um raciocínio mais rápido. De uma maneira geral, achei interessante esse trabalho, visto que ele relaciona-se com trabalhos anteriores.