Pedro Pignaton Negri

GORO - Uma ontologia sobre requisitos baseados em objetivos

Vitória, ES

2017

Pedro Pignaton Negri

GORO - Uma ontologia sobre requisitos baseados em objetivos

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Informática da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do Grau de Mestre em Informática.

Universidade Federal do Espírito Santo – UFES

Centro Tecnológico

Programa de Pós-Graduação em Informática

Orientador: Prof. Dr. Vítor E. Silva Souza

Vitória, ES

2017

Ficha catalográfica disponibilizada pelo Sistema Integrado de Bibliotecas - SIBI/UFES e elaborada pelo autor

---

Negri, Pedro Pignaton, 1987-
N386g GORO - Uma ontologia sobre requisitos baseados em objetivos / Pedro Pignaton Negri. - 2017.
79 f. : il.

Orientador: Vítor E. Silva Souza.
Dissertação (Mestrado em Informática) - Universidade Federal do Espírito Santo, Centro Tecnológico.

1. requisitos. 2. ontologia. 3. orientado por objetivos. 4. GORE. 5. UFO. 6. GORO. I. Souza, Vítor E. Silva. II. Universidade Federal do Espírito Santo. Centro Tecnológico. III. Título.

CDU: 004

Aos meus pais,
por me ensinarem a importância e responsabilidade de ***ser****.*

Agradecimentos

Realizar este mestrado foi um grande desafio. Muito aprendizado sobre o que está aqui apresentado neste trabalho e também muito aprendizado sobre mim mesmo. É com muita alegria que chego ao final desta importante etapa em minha vida: pelas contribuições feitas academicamente (e com vislumbre de mais contribuições futuras) e também pelo crescimento pessoal que tive nesses anos. Muitos participaram desse processo, foi fundamental todo apoio que tive.

Começo agradecendo a Flávia, minha parceira evolutiva, a quem muito sou grato pelo companheirismo, paciência e carinho. Você é a melhor companhia que eu poderia ter nesses últimos anos e em todo tempo que temos pela frente.

Meu orientador, Vitor, muito obrigado pela confiança, apoio e compreensão. Aprendi muito com você não só sobre Engenharia de Requisitos mas, principalmente, sobre postura com os desafios da vida, você é um grande exemplo.

Meus pais e irmã, minha base para poder seguir caminhando, minha referência de humanidade, obrigado por estarem sempre ao meu lado me apoiando e ampliando meus horizontes.

Os colegas do Nemo, sempre dispostos a ajudar, colegas que tornaram mais leve as difíceis tarefas ao longo desses anos, os excelentes professores que tive nessa jornada, os amigos que fiz durante a pesquisa, em especial a professora Renata Guizzardi, muito obrigado por compartilharem o tempo e conhecimento de vocês.

Aos muitos que contribuíram das mais diversas formas, as vezes nem mesmo sabendo que estavam ajudando, gratidão!

"Se você não sabe aonde está indo, qualquer lugar servirá"

(adapatado de Gato de Cheshire, Alice no País das Maravilhas - Lewis Carrol)

Resumo

A Engenharia de Requisitos Baseada em Objetivos (GORE) tornou-se uma importante área de pesquisa nas últimas décadas. Ainda assim, algumas questões permanecem incertas, uma vez que as diferentes abordagens GORE ainda não proveem uma conceituação bem fundamentada do domínio e não são consensuais, podendo levar, assim, a uma compreensão ambígua ou frágil dos conceitos relacionados. Neste trabalho é introduzida a Ontologia de Requisitos Baseados em Objeivos (*Goal-Oriented Requirements Ontology* — GORO), uma ontologia de referência de domínio fundamentada em UFO (*Unified Foundational Ontology*) que pretende representar a natureza e as relações dos conceitos relacionados ao domínio GORE. GORO é utilizada para explorar e esclarecer a semântica usada, muitas vezes implicitamente, por algumas abordagens GORE populares.

Palavras-chaves: requisitos, ontologia, orientado por objetivos, GORE, UFO, GORO

Abstract

Goal-Oriented Requirements Engineering (GORE) has grown into an important area of research in the past decades. Still, some of its corners remain dark, since different GORE approaches do not provide a well-founded conceptualization of the domain and are not consensual. This may lead to ambiguous or weak understanding of GORE concepts. In this work, we introduce the Goal-Oriented Requirements Ontology (GORO), a domain ontology founded on the Unified Foundational Ontology (UFO) that intends to represent the nature and relations of concepts surrounding the GORE domain. GORO is used to explore and clarify the semantics used, sometimes implicitly, by well-known GORE approaches.

Keywords: requirements, ontology, goal-oriented, GORE, UFO, GORO

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

FR	Functional Requirement
GORE	Goal-oriented Requirements Engineering
GORO	Goal-oriented Requirements Ontology
NFR	Non-Functional Requirement
OSA	Ontology of Software Artifacts
RE	Requirements Engineering
UFO	Unified Foundational Ontology

Sumário

1 Introdução

Estamos em constante movimento para que o mundo ao nosso redor se comporte de uma maneira específica desejada. No ambiente de trabalho, quando os colaboradores envolvidos em uma questão precisam executar tarefas de uma forma determinada, por exemplo, ou, ainda, quando o processo de agendamento de reuniões em uma empresa pode ser melhorado para gerenciar recursos de forma mais otimizada e as salas de reunião tenham controle automático da temperatura para os dias ensolarados de verão.

O cruzamento de duas importantes vias de trânsito de uma cidade seria caótico se não houvesse ali um sistema para controle do tráfego. A Figura 1 mostra o que aconteceu no cruzamento das avenidas Faria Lima e Juscelino Kubitschek, em São Paulo, em um dia em que tal sistema para controle parou de funcionar. Neste sistema, onde o intuito é gerenciar tal cruzamento para que seja seguro e otimizado, há uma série de questões envolvidas que precisam ser levadas em consideração. Não pode ser permitido o cruzamento de veículos em vias ortogonais ao mesmo tempo, por exemplo. Assim, os diferentes sentidos de fluxo devem ser liberados para cruzar as pistas de forma alternada. Há também uma restrição importante de que pedestres só podem atravessar a rua quando não houver perigo. Em uma situação específica, pode ser necessário que uma ambulância tenha prioridade de passagem sobre todos os outros veículos.

É possível que um sistema de *software* seja utilizado como solução para controle deste cruzamento. É necessário que haja semáforos em todas as vias, inclusive sinalizando a travessia de pedestres. É também importante que haja uma área de manobra para dar passagem a possíveis ambulâncias ou então uma via dedicada a veículos em situação de prioridade. Pode também ser desejável ou necessária a intervenção direta de agentes humanos de trânsito no local, contribuindo para o comportamento desejado deste fragmento do mundo. É também importante o uso de um padrão de cores nos semáforos indicando se os motoristas podem prosseguir na direção especificada ou não, em que, se assume, os motoristas conhecem o significado do padrão e, além disso, irão respeitá-lo.

A Engenharia de Requisitos (RE - *Requirements Engineering*) é a área que tem interesse em entender essas restrições, esses requisitos, levantar a descrição de *o quê* o sistema fará para transformar aquela realidade de acordo com os interesses dos envolvidos. Especificamente, a Engenharia de Requisitos Baseados em Objetivos (GORE - *Goal-oriented Requirements Engineering*) é uma abordagem que explora os objetivos dos envolvidos, suas intenções e desejos, suas motivações, para entender o problema como um todo e, então, se chegar aos requisitos de uma solução pretendida (LAMSWEERDE, 2001; LAMSWEERDE; LETIER, 2002).

Figura 1 – Congestionamento em cruzamento de avenidas em São Paulo.
Foto extraída do Twitter da BandNews FM, dia 2 de Fevereiro de 2017 -
https://twitter.com/radiobandnewsfm/status/827117740917940224

1.1 Motivação

A Engenharia de Requisitos é uma das etapas iniciais no processo de criação de soluções envolvendo sistemas de *software*. Busca a caracterização do problema formalmente, definindo o escopo da solução e descrevendo qual o comportamento desejado no ambiente (independente de máquina) (JACKSON; ZAVE, 1995; ZAVE; JACKSON, 1997). Assim, é uma etapa crítica influenciando diretamente todo processo de desenvolvimento da solução desejada (SOMMERVILLE, 2010).

Também assume grande importância pelo impacto financeiro e de esforço que pode gerar. Uma pesquisa conduzida em 3800 organizações de 17 países na Europa mostra que a maior parte dos problemas nos projetos de software está na área de especificação de requisitos (European Software Institute, 1996 apud AURUM; WOHLIN, 2005). Uma outra pesquisa realizada com doze empresas do Reino Unido aponta que os problemas relacionados a requisitos representam 48% de todos os problemas de software (BEECHAM; HALL; RAINER, 2003 apud AURUM; WOHLIN, 2005). Ainda, o esforço para corrigir erros ainda na fase de requisitos é menor que o esforço para corrigi-lo quando o sistema está na fase de testes ou já foi finalizado. Um estudo revelou que o custo de correção ou retrabalho nas últimas fases do projeto podem ser de 50 a 200 vezes maior do que nas fases iniciais (BOEHM; PAPACCIO, 1988). Alguns desses trabalhos são relativamente

antigos e foram conduzidos em uma realidade que não é a atual. Hoje há também outros desafios em novas áreas. Porém, ainda assim, essas pesquisas revelam objetivamente a importância e o impacto que processos de Engenharia de Requisitos têm no sucesso de um projeto de *software*.

Abordagens GORE lidam com requisitos, mas buscando uma visão mais ampla do problema, explorando o propósito dos interesses envolvidos. Existem uma série de iniciativas GORE, como KAOS (DARDENNE; LAMSWEERDE; FICKAS, 1993), $i^\star$ (YU, 1995) e Techne (BORGIDA et al., 2009), que exploram requisitos baseados em objetivos nas práticas de Engenharia de Requisitos.

Tais abordagens, porém, não apresentam aprofundamento na semântica dos conceitos do domínio de GORE e utilizam, por vezes, metamodelos como forma de caracterização desse domínio. Como observado por Fayouni, Kavakli e Loucopoulos (2015), a maioria das abordagens não define nem mesmo a sintaxe abstrata da linguagem e, quando fornece o metamodelo, estes são expressos por meio de constructos não padronizados, omitindo informações relacionadas a cardinalidade, restrições de especializações e outros. Por esta deficiência, trabalhos buscam entender e explicitar o significado dos constructos utilizados nas linguagens GORE, tal como (LÓPEZ; FRANCH; MARCO, 2011). Outros trabalhos na literatura apresentam diferentes propostas de metamodelos para $i^\star$ e KAOS (LUCENA et al., 2008; CARES et al., 2010; MATULEVICIUS; HEYMANS, 2005; HEAVEN; FINKELSTEIN, 2004; NWOKEJI; CLARK; BARN, 2013; FAYOUNI; KAVAKLI; LOUCOPOULOS, 2015). Ainda nesses casos, metamodelos descrevem os aspectos de notação, a sintaxe de uma linguagem e não têm o propósito de esclarecer a semântica dos elementos (HAREL; RUMPE, 2000).

Como visto acima, mesmo GORE sendo área de grande relevância, ainda falta aprofundamento da conceituação presente no domínio. Mylopoulos (1992) define modelagem conceitual como "*a atividade de descrever formalmente aspectos físicos e sociais do mundo a nossa volta, com propósito de entendimento e comunicação*" (tradução nossa). Desta forma, uma descrição formal de requisitos baseados em objetivos através de um modelo conceitual bem fundamentado é de suma importância para esclarecer o significado, a natureza profunda dos conceitos e suas relações, servindo, então, como referência semântica sobre GORE. Impacta, assim, promovendo melhor entendimento e comunicação mais eficaz entre os envolvidos já que uma descrição mais precisa das entidades do domínio GORE é explicitamente descrita e compartilhada.

1.2 Objetivos

Este trabalho visa definir uma representação formal do domínio de Engenharia de Requisitos Baseados em Objetivos, com uma descrição precisa e bem fundamentada das

entidades que compõem o domínio e suas relações, fornecendo, assim, embasamento para melhor entendimento e uso dos conceitos no escopo de abordagens GORE.

Para tal, têm-se os seguintes objetivos específicos:

1. Estabelecimento de uma concepção compartilhada do domínio GORE e seus limites, através do estudo de diferentes abordagens GORE existentes;
2. Definição de uma ontologia de referência sobre requisitos baseados em objetivos;
3. Esclarecimento dos conceitos adotados pelas abordagens GORE estudadas através de uma análise fundamentada na ontologia proposta.

1.3 Método

Entendimento do domínio GORE. Foi realizado estudo de abordagens distintas de GORE, a fim de entender como tais abordagens lidam com este domínio (conceitos relevantes, suas relações, seus objetivos). A escolha das abordagens foi feita de acordo com a relevância e foram lidas publicações referentes a tais abordagens. Assim, com as diferentes contribuições e enfoques de cada abordagem para a área, foi obtida uma caracterização e escopo do domínio GORE.

Construção e Formalização da Ontologia. Após a caracterização do domínio GORE através do estudo de abordagens relacionadas, foi desenvolvida a ontologia GORO (*Goal-Oriented Requirements Ontology*) — uma ontologia de referência de domínio. Tal ontologia foi construída seguindo uma metodologia de engenharia de ontologias, fundamentada em uma ontologia de fundamentação e reutilizzou ontologias já existentes relacionadas ao domínio.

Uso da Ontologia para explorar as abordagens GORE. Ontologia construída foi utilizada como embasamento conceitual para analisar como diferentes abordagens GORE disponíveis na literatura abordam este domínio. Assim, os elementos utilizados pelas diferentes abordagens foram explicados à luz de GORO.

Uso da Ontologia para explorar as abordagens GORE. Ontologia construída foi utilizada como embasamento conceitual para analisar os elementos e conceitos utilizados pelas diferentes abordagens GORE.

1.4 Organização da Dissertação

Neste capítulo inicial foram apresentados o contexto e a motivação deste trabalho, além dos objetivos e o método para alcançá-los nesta pesquisa. Além disso, esta dissertação é organizada como descrito a seguir:

Capítulo 2 - Referencial Teórico. Neste capítulo é apresentado todo embasamento teórico utilizado como referência neste trabalho. É abordada GORE e são apresentadas algumas de suas metodologias ($i^\star$, KAOS e Techne), que formam o embasamento teórico para entendimento do domínio. É também explorada a ontologia de fundamentação UFO (*Unified Foundatiounal Ontology*) utilizada para fundamentar os conceitos do domínio de GORE. Também são apresentadas outras ontologias já existentes que tratam de temas relevantes ao domínio de GORE e, desta forma, foram reutilizadas na construção de GORO. Ainda, descreve o método para construção de ontologias SABiO que foi utilizado no desenvolvimento da ontologia proposta.

Capítulo 3 - Goal-Oriented Requirements Ontology (GORO). Neste capítulo é detalhado o processo de construção da ontologia proposta e a mesma é devidamente apresentada.

Capítulo 4 - Avaliação da Ontologia. Uma interpretação de algumas abordagens existentes de GORE é feita à luz de GORO. Desta forma, a ontologia proposta neste trabalho ajuda a entender o significado dos conceitos e relações que as abordagens adotam.

Capítulo 5 - Conclusões. Por fim, no último capítulo, são apresentadas as conclusões deste trabalho e suas principais contribuições, bem como a verificação de satisfação dos objetivos levantados, as limitações do trabalho e as perspectivas futuras para pesquisa.

2 Referencial Teórico

Inicialmente, na Seção 2.1 é apresentada a área de Engenharia de Requisitos. Em seguida, a Seção 2.2 discute um estudo sobre as *suposições* no contexto de Engenharia de Requisitos. Na Seção 2.3 é feita uma apresentação sobre Engenharia de Requisitos Baseada em Objetivos (GORE) e uma introdução de diferentes abordagens GORE. Em seguida nas seções 2.4 e 2.5, é introduzido o assunto Ontologias e a *Unified Foundational Ontology*, ontologia de fundamentação utilizada como embasamento da GORO, é apresentada. Na Seção 2.6 é apresentada uma ontologia reutilizada neste trabalho. A Seção 2.7 aborda o método de desenvolvimento de ontologias utilizada na construção da GORO: o método SABiO e, por fim, a Seção 2.8 apresenta uma série de trabalhos correlatos ao desenvolvido nesta dissertação.

2.1 Engenharia de Requisitos

O desenvolvimento de um sistema pode ser classificado como bem sucedido quando o sistema atinge o propósito para o qual ele foi projetado. Porém, saber exatamente o propósito de um sistema, os motivos que levaram ao desejo de que ele fosse desenvolvido, as intenções dos envolvidos para com ele, não é tarefa trivial. Brooks (1987) cita em seu trabalho seminal que a "*a parte mais difícil de construir um sistema de software é decidir exatamente o que construir*". Para dar suporte na solução desse problema, a Engenharia de Requisitos surge como o processo que visa elicitar, registrar, analisar e negociar os requisitos que um sistema deve satisfazer para atingir seu propósito. Os requisitos descrevem, então, *o que* um sistema deve fazer para solucionar um problema específico.

Já em 1985, Roman (1985) considerava a especificação de requisitos como o processo no qual se busca entender o problema em questão antes de elaborar qualquer tipo de solução. Ainda, Greenspan (1984 apud MYLOPOULOS, 1992) propõe, associado à fase de definição de requisitos, representar conhecimento em uma chamada "*modelagem do mundo*". Neste contexto, o levantamento dos requisitos é uma das fases mais iniciais no processo de desenvolvimento de um sistema-solução, uma vez que aborda diretamente o problema ajudando a entender o escopo do mesmo e o fragmento do mundo no qual tal problema está inserido, servindo, assim, como referência para as próximas etapas de desenvolvimento de uma solução. É comum, inclusive, que os maiores interessados nesta futura solução não sejam os que irão desenvolvê-la e, assim, a Engenharia de Requisitos se torna uma etapa ainda mais importante, criando uma ponte entre os envolvidos nos processos de desenvolvimento da solução e aqueles conhecem o problema e demandaram tal solução. Há, neste ponto, uma questão comunicativa de suma importância que guia

todas as etapas subsequentes no desenvolvimento.

Requisitos influenciam todo o processo de desenvolvimento e aqueles elicitados erroneamente são levados adiante no processo impactando diretamente o comportamento do produto final, não na forma de erros, de *bugs*, de falhas, mas fazendo com que o sistema se comporte de forma não desejada, não alinhada ao problema, não resolvendo o problema. Assim, esta é uma área que impacta diretamente na validação do sistema — que avalia se o que foi desenvolvido atende às necessidades e expectativas iniciais para esta solução. É claro o quanto esta etapa é importante, uma vez que ela visa, justamente, entender e caracterizar (formalmente) o escopo do problema com que se está lidando. Assim, é um processo que influencia diretamente todas as fases seguintes no processo de desenvolvimento da solução. Como visto na Seção 1.1, pesquisas mostram o grande impacto que essa fase pode gerar na alocação de recursos de um projeto (AURUM; WOHLIN, 2005).

Zave e Jackson (JACKSON; ZAVE, 1995; ZAVE; JACKSON, 1997) fazem contribuições seminais sobre os fundamentos de Engenharia de Requisitos e definem requisitos como propriedades desejadas no ambiente (uma porção de interesse do mundo real). Considerando que sistemas de *software* podem interagir com o ambiente a fim de promoverem tais propriedades, requisitos, então, são sentenças que falam também sobre o que pode ser observável numa interface entre o sistema de *software* e o ambiente, e nada mais sobre o *software*. Desta forma, o escopo dos requisitos é definido: requisitos tratam do domínio do problema, levantando *o que* um sistema deve fazer para atuar neste problema (e não *como* o sistema atuará). Assim, Zave e Jackson propõem uma fórmula para o problema de Engenharia de Requisitos: $A, S \models R$. Isso significa que, considerando uma série de suposições sobre o meio (***A****ssumptions*), uma especificação (***S****pecification*) de sistema é estabelecida a fim de satisfazer os requisitos (***R****equirements*).

2.2 *Assumptions*

Sendo muito utilizado em áreas diversas, o termo *assumption* (suposição) tem uma sobrecarga de significados que pode dificultar o entendimento de sua real semântica. Wang et al. (2016) esclarecem o conceito de *assumption* em Engenharia de Software, explorando diferentes tipos de suposições neste contexto — alguns já abordados na literatura e outros novos propostos.

Na Figura 2, o trabalho apresenta um recorte originalmente proposto por Zave e Jackson (1997) das fronteiras entre os fenômenos da máquina e do mundo, estendendo-o e enfatizando as *suposições* relacionadas. Do lado esquerdo, o mundo (*world*) e um recorte de seus fenômenos de interesse (*world phenomena*). No outro lado, a máquina (*machine*) e o recorte de seus fenômenos de interesse (*machine phenomena*). Na interseção, a interface (*interface*) entre esses dois recortes com os fenômenos (*interface phenomena*) do mundo e

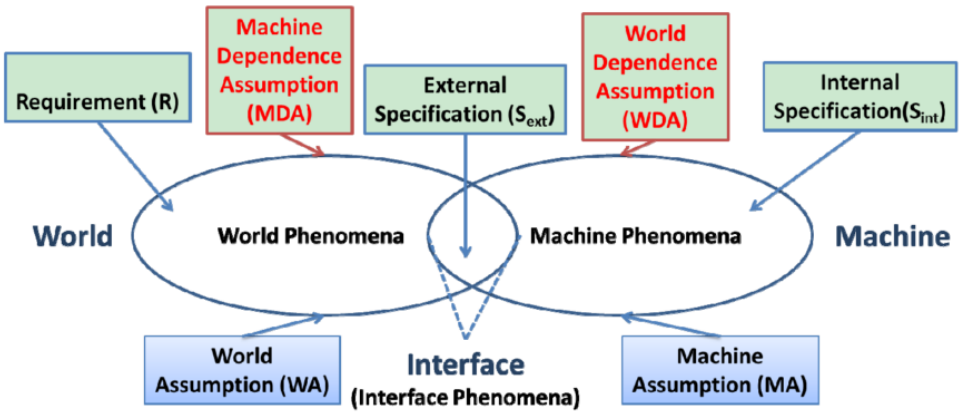

Figura 2 – Interação mundo e máquina e suposições (*assumptions*) relacionadas (WANG et al., 2016)

da máquina que podem ser vistos pelos dois, ou seja, é como a máquina o mundo interagem.

Ainda na Figura 2, requisitos (*requirements*) falam de estados intencionados das coisas no mundo. Especificação externa (*external specification*) é o conjunto de ações que deve acontecer na interface para satisfação dos requisitos. E especificação interna (*internal specifications*) é o conjunto de ações internas na máquina para apoiar a ocorrência dos fenômenos descritos na especificação externa.

Um *world assumption* é uma suposição sobre um fenômeno no mundo/ambiente, não visível pela máquina. Em um sistema que controla a temperatura de uma sala, por exemplo, uma *world assumption* poderia ser que o sistema de ar-condicionado tem capacidade de resfriamento suficiente para diminuir a temperatura desta sala. Ou então, em um sistema de agendamento de reuniões, que exista salas disponíveis para todos os pedidos de reunião. Tal suposição significa que o sistema solução não funcionará bem quando não houver sala disponível para um pedido de reunião (no caso de um período com muitos pedidos de reunião, por exemplo).

Por outro lado, *machine assumptions* são suposições sobre fenômenos internos às máquinas e visíveis somente por elas próprias. A ação de registrar uma reunião adicionando-a em uma tabela, por exemplo, requer a suposição de que sempre há espaço disponível na tabela. Tal suposição assegura a disponibilidade de recursos na máquina. Em outro exemplo, assume-se que existe energia para a máquina funcionar. Isto significa que se esta suposição falhar, o agendamento de reuniões talvez não funcione.

As suposições acima abordam o mundo e a máquina de forma independente, não descrevendo conexões casuais entre os estados da máquina e do mundo ao seu redor. Esta conexão acontece por outros dois tipos de suposição. *Machine dependence assumptions* (ou suposições dependentes de máquina) lidam com fenômenos do mundo externo que são dependentes de algum fenômeno da máquina. Pode-se assumir, por exemplo, que uma determinada reunião foi agendada no mundo real já que foi adicionada uma nova entrada

na tabela de reuniões com os dados da reunião e com uma sala sendo reservada no horário especificado (o que significa que ela não poderá ser usada para nenhum outro propósito neste horário).

Já *world dependence assumptions* (suposição dependente do mundo) tratam de fenômenos da máquina que dependem de algum fenômeno no mundo externo. Pode se assumir, por exemplo, que uma sala que aparece com o *status* de livre no sistema da máquina é por que a sala está realmente livre no mundo. Também se pode assumir que quando uma reunião aparece como requisitada na máquina, é por que alguém realmente tem a intenção de agendar tal reunião. Uma suposição de dependência de mundo é uma suposição sobre a correspondência entre estados internos da máquina e fenômenos no mundo que esses estados se propõem a representar. Em outras palavras, as suposições de dependência implicam que fenômenos ocorridos na máquina e no mundo devem ser simultaneamente refletidos no mundo e na máquina para garantir que os estados do mundo estejam corretamente representados na máquina e vice-versa.

Ainda, o trabalho aborda outros dois tipos de *assumptions* independentes das quatro definições anteriores: *assumption-used* e *assumption-needed*. *Assumptions-used* são proposições usadas *a priori* ao construir um novo argumento — falam sobre propriedades que fazem parte do contexto da solução. Precisam ser consideradas durante a criação do sistema, pois estarão naturalmente presentes na situação em que a solução existir. As leis da física, por exemplo. Ou, ainda, no sistema de refrigeração, o fabricante poderia considerar que o fornecimento de energia elétrica para o aparelho será com uma diferença de tensão de 110V com variação máxima de 5%.

Assumptions-needed são proposições necessárias para sustentar uma *conclusão prévia*, explicando a situação na qual a solução projetada funciona. Tratam de propriedades que precisam ser mantidas de alguma forma para o sistema funcionar corretamente, ou seja, existe uma ação direta para tal suposição acontecer no contexto do sistema. O aparelho de ar-condicionado é instalado com o condensador do lado de fora da sala e o evaporador do lado de dentro da sala. Ou então que o ar-condicionado será instalado em uma sala com determinada dimensão máxima. Assim, nessas condições, o fabricante garante o bom funcionamento do aparelho.

Wang et al. (2016) trazem uma rica contribuição sobre suposições que é aproveitada neste trabalho.

2.3 Engenharia de Requisitos Baseada em Objetivos

São reconhecidas limitações nas práticas tradicionais de Engenharia de Requisitos (LAMSWEERDE; LETIER, 2002). A grande maioria das técnicas tem escopo limitado, não oferecendo suporte para raciocinar sobre o sistema completo, composto pelo software

e o ambiente em que ele estará inserido e, assim, focando na modelagem do software isoladamente. Suposições inadequadas sobre o ambiente em que o *software* atuará são responsáveis por muitos erros na especificação dos requisitos (JACKSON, 1995; LEVESON, 1995). Ainda, requisitos consistem basicamente de dados e processos e não capturam os propósitos que os originaram.

Desta forma, práticas tradicionais de RE são deficientes em prover recursos para uma análise mais completa do sistema intencionado. Como resposta, novas abordagens foram propostas introduzindo o conceito de objetivo (*goal*). Nascia a (sub)área de pesquisa de Engenharia de Requisitos Baseados em Objetivos (GORE - *Goal-Oriented Requirements Engineering*). Objetivos são declarações de intenção a serem alcançadas. Podem expressar intenções em diferentes níveis de abstração: de objetivos estratégicos de alto nível de abstração (por exemplo, *ser reconhecido como uma banda de rock* ou *estar entre os primeiros da lista da Rolling Stone*) até objetivos mais técnicos, de baixo nível de abstração (como *ter escolhido o repertório para gravar um CD*). Satisfazer os objetivos pode ser considerado um requisito de um sistema desejado (LAMSWEERDE, 2001).

GORE trouxe uma série de benefícios à prática de RE como (LAMSWEERDE, 2001; LAMSWEERDE; LETIER, 2002):

Visão holística. GORE possibilita uma visão mais ampla sobre o sistema, explicitando seu entorno e os interessados com suas motivações.

Motivação dos requisitos. Inserir o conceito de *objetivo* no modelo significa explicitar a *intencionalidade* dos envolvidos e, assim, fica descrito no próprio modelo qual o propósito de cada requisito (o que eles pretendem satisfazer). Desta forma, objetivos ajudam na compreensão do propósito de requisitos que não são necessariamente compreensíveis para todos envolvidos.

Verificação de completude. Provê um método mais claro para avaliar se os requisitos elicitados são suficientes para atender ao propósito do sistema desejado (verificando se os objetivos levantados são alcançados pelos requisitos especificados, levando em consideração as propriedades do domínio).

Requisitos complexos. Provê uma estrutura mais legível para interpretação de requisitos complexos.

Resolução de conflitos. Por trazer também os propósitos dos requisitos através do conceito objetivo, facilita a detecção e resolução de conflitos entre requisitos.

Tais benefícios têm estimulado a proposta de diferentes abordagens GORE. KAOS (DARDENNE; LAMSWEERDE; FICKAS, 1993), Techne (BORGIDA et al., 2009) e $i^\star$ (YU, 1995) foram estudadas no escopo deste trabalho para obtenção de melhor entendimento do domínio GORE e são, assim, expostas nas subseções a seguir.

2.3.1 KAOS

KAOS (DARDENNE; LAMSWEERDE; FICKAS, 1993) é uma abordagem sistemática para descobrir e estruturar requisitos. Em KAOS, um objetivo (*goal*) é uma declaração prescritiva de uma intenção que um sistema (*software* + ambiente externo) deve satisfazer (LAMSWEERDE, 2009). Objetivos podem expressar objetivos de alto nível (mais estratégicos, relacionados ao escopo do sistema e às intenções dos envolvidos), e objetivos de baixo nível (mais técnicos). Os objetivos de alto nível são consecutivamente refinados em objetivos de níveis mais baixos, mais operacionais, para expressar *como* esses mais altos podem ser alcançados. Em KAOS não são modelados os agentes, os envolvidos, dos quais se tenha extraído os objetivos/intenções.

É possível abordar KAOS por 4 submodelos que focam em algumas temáticas (complementares entre si) da modelagem GORE (LAMSWEERDE, 2001). Tais submodelos são aqui apresentados separadamente para fins didáticos, mas serão tratados de forma unificada no decorrer deste trabalho.

Modelo de objetivos. Elaborado elicitando os objetivos dos interessados no sistema e expandindo a árvore de objetivos através de perguntas *como?*, para detalhar um objetivo, e *por quê?* para embasar a motivação de um objetivo.

Modelo de objetos. Descreve objetos (suas relações e atributos) derivados das especificações de objetivos.

Modelo de responsabilidade. Identifica os agentes responsáveis a satisfazerem os objetivos.

Modelo operacional. Descreve as operações e comportamentos dos agentes para cumprir requisitos.

Objetivos podem ser funcionais (associados a serviços a serem providos) e não-funcionais (associados à qualidade dos serviços oferecidos). Ainda, é feita uma distinção entre *softgoals*, cuja satisfação não pode ser estabelecida de forma clara, e *hardgoals* que têm sua satisfabilidade verificada por técnicas de verificação objetivas. Assim, KAOS traz que *softgoals* são especialmente úteis para comparar objetivos alternativos em um refinamento, ajudando a escolher qual dos subobjetivos alternativos melhor contribui para satisfação do objetivo pai.

Objetivos podem ser refinados por duas possíveis relações. O refinamento-E (*AND-refinement*) relaciona um objetivo a um conjunto de subobjetivos no qual satisfazer todos os subobjetivos é necessário para satisfazer o objetivo pai. O refinamento-OU (*OR-refinement*) relaciona um objetivo com um conjunto de alternativas no qual satisfazer alguma das alternativas é suficiente para satisfação do objetivo pai. A intenção nesses modelos é refinar um objetivo até que ele se torne pequeno o suficiente para ser satisfazível por apenas um

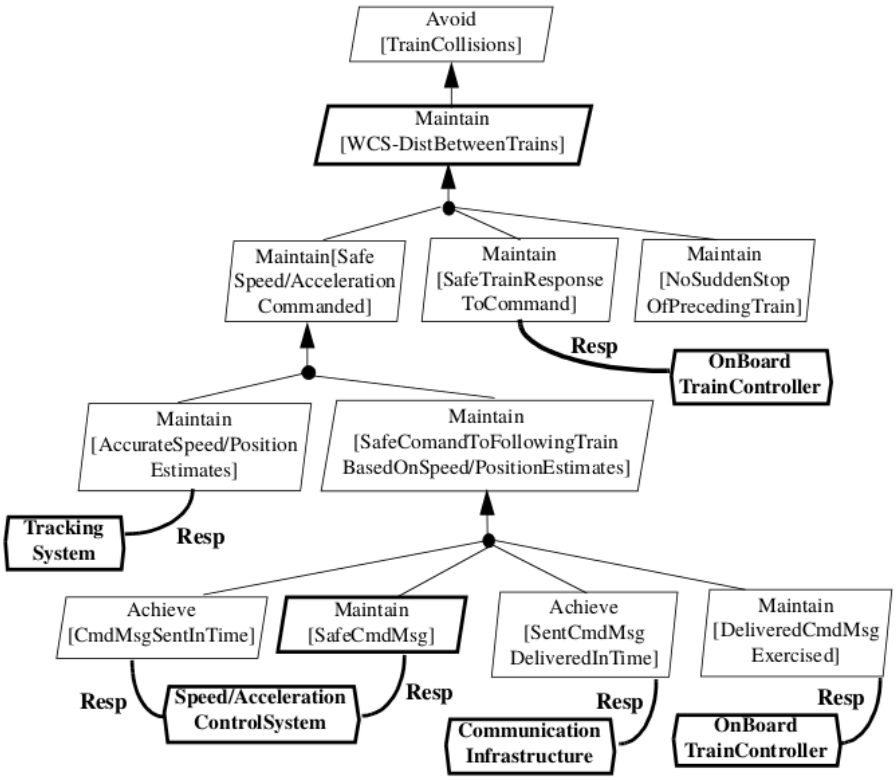

Figura 3 – Exemplo de um modelo de objetivos e responsabilidade em KAOS (LAMSWEERDE, 2001)

agente (humano ou não) — ou seja, um objetivo pequeno o suficiente para ser satisfeito, alcançado, por apenas um agente. Em KAOS, é feita uma diferenciação explícita de objetivos que irão ser satisfeitos por agentes humanos ou por *software*: quando um objetivo realizável está sob responsabilidade de um agente de *software*, ele é chamado requisito; quando ele está sob responsabilidade de um agente no ambiente, ele é considerado uma expectativa.

Na Figura 3 tem-se um exemplo de um modelo de requisitos baseados em objetivos feito em KAOS. No exemplo, os losangos são os objetivos e são decompostos desde o o objetivo de mais alto nível (Evitar colisão de trens, *Avoid Train Collisions*), até objetivos mais operacionais como 'Manter estimativas de posição e velocidades precisas' (*Maintain Accurate Speed/Position Estimates*). Este último objetivo, por exemplo, tem modelado um agente específico responsável por sua satisfação: o sistema de rastreamento, inscrito em um hexágono em negrito (*TrackingSystem*). Outros agentes responsáveis pela satisfação de outros objetivos também estão modelados como o Controlador de Trem a Bordo (*OnBoardTrainController*).

2.3.2 *i**

O Framework *i** (YU, 1995) preza por modelar aspectos sociais da Engenharia de Requisitos com enfoque no conceito de *objetivo*. Assim, diferente de KAOS, os atores

envolvidos, que trazem suas intenções, seus objetivos, são mostrados no modelo como agentes, como o papel que exercem ou como sua posição. Desta forma, o objetivo *ter reunião de efetivação de compra agendada* poderia ser de uma pessoa específica (agente), de um cliente (papel) ou do gerente de vendas da empresa (posição).

São propostos diferentes modelos para diferentes níveis de abstração. No nível mais alto, modelos de Dependência Estratégica (SD - *Strategic Dependency*) visam descrever relações de dependências entre atores: um ator depende de algum outro para satisfazer alguma intenção e, assim, delega para este último a responsabilidade de satisfação do mesmo. Um elemento intencional pode ser um objetivo, uma tarefa, um recurso ou um *softgoal*. No nível mais baixo de abstração, modelos de Racionalidade Estratégica (SR - *Strategic Rationale*) descrevem e refinam os elementos intencionais dentro do escopo de cada ator.

A abordagem $i^\star$ original foca na fase de requisitos iniciais (*early-requirements*), identificando os envolvidos (atores, *stakeholders*) e trazendo seus objetivos estratégicos para o modelo. O propósito é fornecer ferramentas para modelar aspectos sociais na Engenharia de Requisitos. Tropos (BRESCIANI et al., 2004), uma variação de $i^\star$, inclui o conceito de decomposição de objetivo para diminuir o nível de abstração dos objetivos rumo a uma especificação de requisitos tardios (*late-requirements*). Uma recente versão de $i^\star$, chamada iStar 2.0 (DALPIAZ; FRANCH; HORKOFF, 2016), inclui refinamento de objetivos, que pode ser utilizado para derivar requisitos de sistemas de objetivos de alto nível de abstração.

$i^\star$ conta também com o conceito de meio-para (*means-end*) que sinaliza qual o meio para um objetivo ser satisfeito. Em sua proposta inicial, Yu (1995) traz o conceito de *means-end* como uma relação entre um fim — que pode ser um objetivo a ser atingido, uma tarefa a ser cumprida, um recurso a ser produzido ou um *softgoal* a ser satisfeito — e um meio para alcançá-lo. Ainda, Yu afirma que o *meio* para o *fim* é, geralmente, expresso em forma de *tarefa*.

Nota-se, porém, que o adotado em variantes de $i^\star$, como Tropos, são elementos como *recurso* ou outro *objetivo* sendo utilizados como *meios* para atingir um *fim* (*objetivo*) (GUIZZARDI; FRANCH; GUIZZARDI, 2012). Ainda, na versão 2.0 da linguagem, *means-end* deixa de existir e um único elemento denominado *refinamento* (*refinement*) representa de forma genérica os conceitos de *means-end* e decomposição de tarefas.

Nas Figuras 4 e 5 temos exemplos de modelos de objetivos construídos em $i^\star$. A Figura 4 traz o modelo de Dependência Estratégica, onde são evidenciados os agentes envolvidos (círculos), como Estudante (*Student*), Universidade (*Univ. of Wonderland*) e Agência de Viagem (*Travel agency*), e as relações de dependências entre eles: o Estudante depende da Agência de Viagens para comprar os bilhetes de vôo (*Buy flight tickets.*)

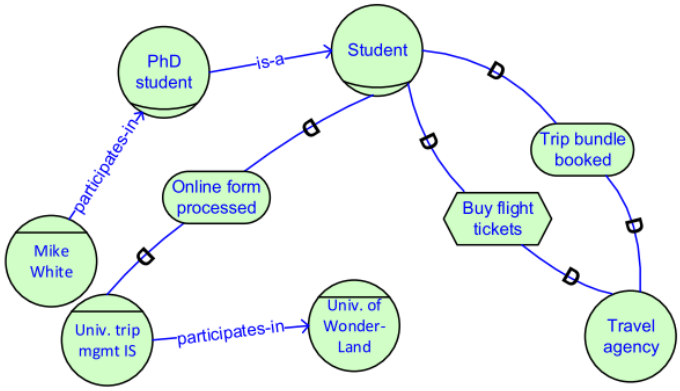

Figura 4 – Exemplo de um modelo de Dependência Estratégica (SD) em $i^\star$ (DALPIAZ; FRANCH; HORKOFF, 2016)

Já na Figura 5 temos um modelo centrado na figura do Estudante (*Student*). O modelo mostra as relações de dependência deste agente (como no modelo SD) com, por exemplo, a Agência de Viagem (*Travel Agency*) e também aprofunda os elementos intencionais no escopo do ator *Student* dentro da área pontilhada. Os elementos no formato oval são os objetivos, os elementos no formato hexagonal são tarefas, e os elementos em formato de nuvem são *softgoals*. Neste exemplo pode-se observar o Estudante com o objetivo de Ter Viagem Organizada (*Travel organized*). Este objetivo é refinado em outros dois pelo refinamento tipo *AND*: *Authorization obtained* e *Trip booked*. Assim, o objetivo principal, ter a viagem organizada, vai sendo refinado até tarefas como no objetivo Bilhete reservado (*Ticket booked*) que pode ser satisfeito ao se realizar a tarefa Agência compra o bilhete (*Agency buys tickets*) ou Estudante compre o bilhete (*Self-booked ticket*).

2.3.3 Techne

Techne (BORGIDA et al., 2009) é uma linguagem de modelagem de requisitos fundamentada na ontologia CORE (*Core Ontology for RE*) (JURETA; MYLOPOULOS; FAULKNER, 2009) que é, por sua vez, baseada na ontologia DOLCE (*Descriptive Ontology for Linguistic and Cognitive Engineering*) (MASOLO et al., 2003), uma ontologia de fundamentação que visa capturar categorias ontológicas subjacentes à linguagem natural, ao senso comum humano.

Techne distingue diferentes estados mentais dos envolvidos a fim de capturar os elementos do modelo: desejos, crenças, intenções e atitudes (JURETA; MYLOPOULOS; FAULKNER, 2008a). Desejos de agentes são requisitos que o sistema deve satisfazer e são capturados por meio de de instâncias do conceito *objetivo* quando expressam uma condição funcional verificável. Caso contrário, caso não expressem uma condição funcional verificável, os desejos de agentes são capturados pelo conceito de *softgoal*. Crenças implicam em suposições do domínio (*domain assumptions*), que são supostos estados das coisas que estão (estariam) presentes no sistema de interesse ou no ambiente. Intenções indicam

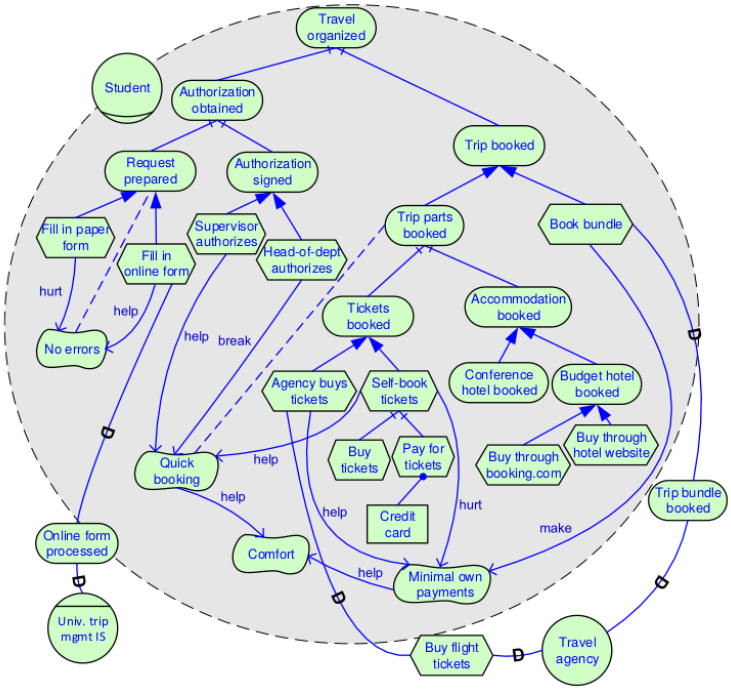

Figura 5 – Exemplo de um modelo de Dependência Estratégica (SD) e Racionalidade Estratégica (SR) em $i^\star$ (DALPIAZ; FRANCH; HORKOFF, 2016)

compromissos para agir em direção à satisfação de requisitos e, assim, são capturados pelo conceito de tarefa (*task*).

Em Techne tem-se os seguintes relacionamentos (*relationships*):

Inferência (*inference*) : expressa que a satisfação de instâncias de algum elemento (como objetivo, *softgoal*, tarefas ou relacionamentos) resulta na satisfação de uma outra instância de outro elemento. Como pode ser aplicado em elementos diversos, é o mesmo que o refinamento de objetivo (*goal refinement*) do KAOS (Seção 2.3.1), e o *means-end* e decomposição de tarefa (*task decomposition*) do $i^\star$ (Seção 2.3.3).

Conflito (*conflict*) : relaciona dois requisitos que não podem ser satisfeitos simultaneamente.

Opcionalidade (*optionality*) : mesmo sendo definido como um relacionamento, este é um atributo que indica se a satisfação de um requisito do modelo é opcional ou compulsória.

Preferência (*preference*) : compara instâncias de qualquer conceito ou relacionamento, indicando a preferência em satisfazer um desses elementos ou outro.

Borgida et al. (2009) dizem que esses quatro relacionamentos se originam de suposições de domínio (*domain assumptions*). No caso da *inferência*, acredita-se, supõe-se, que a satisfação de dois subobjetivos satisfaz um objetivo maior, por exemplo. Para o *conflito*, também há uma suposição de que os elementos relacionados não podem ser

satisfeitos em conjunto. Não fica claro como suposições originam *opcionalidade* e *preferência*, mas o trabalho traz esses dois últimos como captura do estado mental *atitude* (emoções, sentimentos e humor) e que definem o nível de satisfação de um agente com determinada coisa (JURETA; MYLOPOULOS; FAULKNER, 2008a).

Techne não define uma sintaxe concreta, somente uma sintaxe abstrata descrita nesta seção.

2.4 Ontologias

Pela Filosofia, a Ontologia é um dos ramos fundamentais da Metafísica — disciplina que explora a natureza da realidade, suas estruturas básicas e relações. Aristóteles descreveu a Ontologia como *a ciência do ser enquanto ser*. Com o intuito de estudar as características mais fundamentais e diversas dos elementos que compõem a realidade, de todos modos de *ser*, a Ontologia lida com relações que transcendem categorias, disciplinas, temas específicos, tratando de relações que podem existir entre entidades pertencentes a domínios distintos das ciências e também entidades reconhecidas, comuns, estabelecidas pelo senso comum (GUIZZARDI, 2005; GUIZZARDI, 2007). Assim, diferentemente de outras ciências que buscam entender e modelar a realidade sob uma perspectiva bem delimitada, a Ontologia busca a natureza e estrutura das coisas em sua essência, independente de qualquer outra consideração, resultando em um entendimento interdisciplinar sobre a natureza dos elementos de interesse (GUARINO; OBERLE; STAAB, 2009).

Em Computação, ontologias (com "o" minúsculo) são artefatos usados para investigar domínios de interesse, modelando formalmente as entidades relevantes e suas relações. Gruber (1993) e Borst (1997) definem ontologia de forma complementar: especificações explícitas e formais de conceituações compartilhadas. Assim, em Sistemas de Informação e Engenharia de Software, especificamente, ontologias são utilizadas para uma caracterização formal e compartilhada (em consenso) do domínio de interesse, descrevendo as entidades e sua relações, a fim de servir de referência conceitual sobre o domínio. Tornando os conceitos explicitamente descritos, servem de suporte na negociação semântica de elementos, melhorando a comunicação e entendimento sobre o domínio.

Ontologias podem ser classificadas de diferentes formas. Guarino (1998) sugere uma classificação relacionada ao nível de especificidade da ontologia (organizado na Figura 6 e explicado a seguir):

Ontologias de Fundamentação descrevem os conceitos mais fundamentais — como espaço, tempo, matéria, objeto, evento — e são construídas baseando-se em trabalhos bem estabelecidos em áreas como Psicologia Cognitiva, Ontologia Filosófica, Linguística e outros. São independentes de um domínio particular, sendo aplicáveis

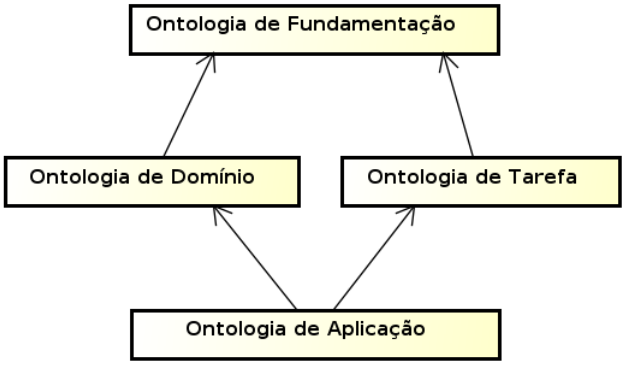

Figura 6 – Classificação de diferentes tipos de ontologia pelo seu nível de especificidade (GUARINO, 1998).

a diferentes domínios e problemas;

Ontologias de Domínio utilizam conceitos de ontologias de fundamentação, especializando-os para a realidade de um determinado domínio específico como Medicina Molecular, Automóveis ou Engenharia de Software;

Ontologias de Tarefa utilizam conceitos de ontologias de fundamentação, especializando-os para representar tarefas ou atividades genéricas como compra/venda e diagnóstico;

Ontologias de Aplicação descrevem conceitos que dependem tanto de um domínio particular quanto de uma tarefa, sendo normalmente especialização das duas ontologias relacionadas.

Outra importante classificação refere-se às intenções de uso da ontologia e difere ontologias de referência e ontologias operacionais. **Ontologias de Referência** focam em uma representação formal e fidedigna do domínio de interesse. Com intuito de serem utilizadas por seres humanos, os auxiliam em tarefas comunicativas como negociação de significado e estabelecimento de consenso sobre os conceitos do domínio. Servem como modelo conceitual a ser usado como referência sobre o conhecimento disponível do domínio em questão (GUARINO, 1998; GUIZZARDI, 2007; FALBO; BERTOLLO, 2009). Por outro lado, **Ontologias Operacionais** são baseadas em ontologias de referência, mas com intuito de serem lidas por máquinas. Assim, focam não em uma representação precisa da realidade mas em prover propriedades computacionais que as tornem processáveis por máquinas (FALBO, 2014; GUIZZARDI, 2007).

Neste trabalho, é proposta uma Ontologia de Referência de Domínio estabelecida com base em uma Ontologia de Fundamentação, a qual é apresentada a seguir.

2.5 UFO

UFO (*Unified Foundational Ontology*) (GUIZZARDI, 2005) é uma ontologia de fundamentação que se embasa em uma série de teorias de áreas diversas como Linguística, Psicologia Cognitiva, Ontologias Formais e Lógica Filosófica, provendo uma representação sólida da conceituação de entidades e suas relações existentes no mundo real. Assim, vem sendo utilizada com sucesso para proporcionar embasamento semântico no desenvolvimento de ontologias de domínio, integração de linguagens de modelagem e esclarecendo conceitos em diferentes domínios como Serviços, Estruturas Organizacionais, Leis, Qualidade de Software, Engenharia de Software, Engenharia de Requisitos, dentro outros (GUIZZARDI et al., 2015).

É divida em três fragmentos detalhados nas subseções a seguir:

UFO-A lida com aspectos estruturais da modelagem conceitual e, assim, é o núcleo de UFO. Também chamada de ontologia de *endurants* (objetos), trata de objetos, seus tipos, suas partes, os papéis que podem desempenhar e suas propriedades.

UFO-B trata de propriedades temporais das entidades como eventos, processos, participação de objetos em eventos e conceitos relacionados. É uma ontologia de *perdurants*, ou eventos.

UFO-C é uma ontologia de entidades intencionais e sociais, abordando conceitos como agentes, crenças, objetivos e ações. É construída sobre conceitos da UFO-A e UFO-B.

2.5.1 UFO-A

O primeiro conceito de UFO é o de entidade (entity), que representa algo que pode ser concebido ou percebido. Entidades são especializadas em duas categorias: individuals e universals.

Individual é a categoria ou tipo em geral que se refere a entidades que existem na realidade e possuem uma identidade única. Este tipo se aplica a indivíduos específicos como a guitarra do David Gilmour (tanto esta guitarra específica quanto esta pessoa específica, David Gilmour, são individuals). **Universal** se aplica a *conceitos* ou *categorias* de indíviduos como pessoa, adulto, guitarra, anfiteatro - padrões de características comuns em diferentes indivíduos. Como visto na Figura 7, individuals são instâncias de universals.

Individuals

Individuals podem ser especializados em dois diferentes conceitos como mostra a Figura 8: concrete individual e abstract individual. **Endurants** são concrete individuals que existem no tempo mantendo sua identidade. A Figura 8 mostra as três classes de endurants explicadas a seguir.

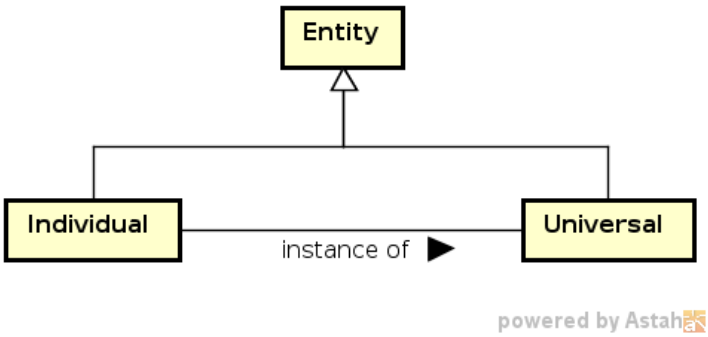

Figura 7 – Fragmento de UFO-A diferenciando Individual e Universal

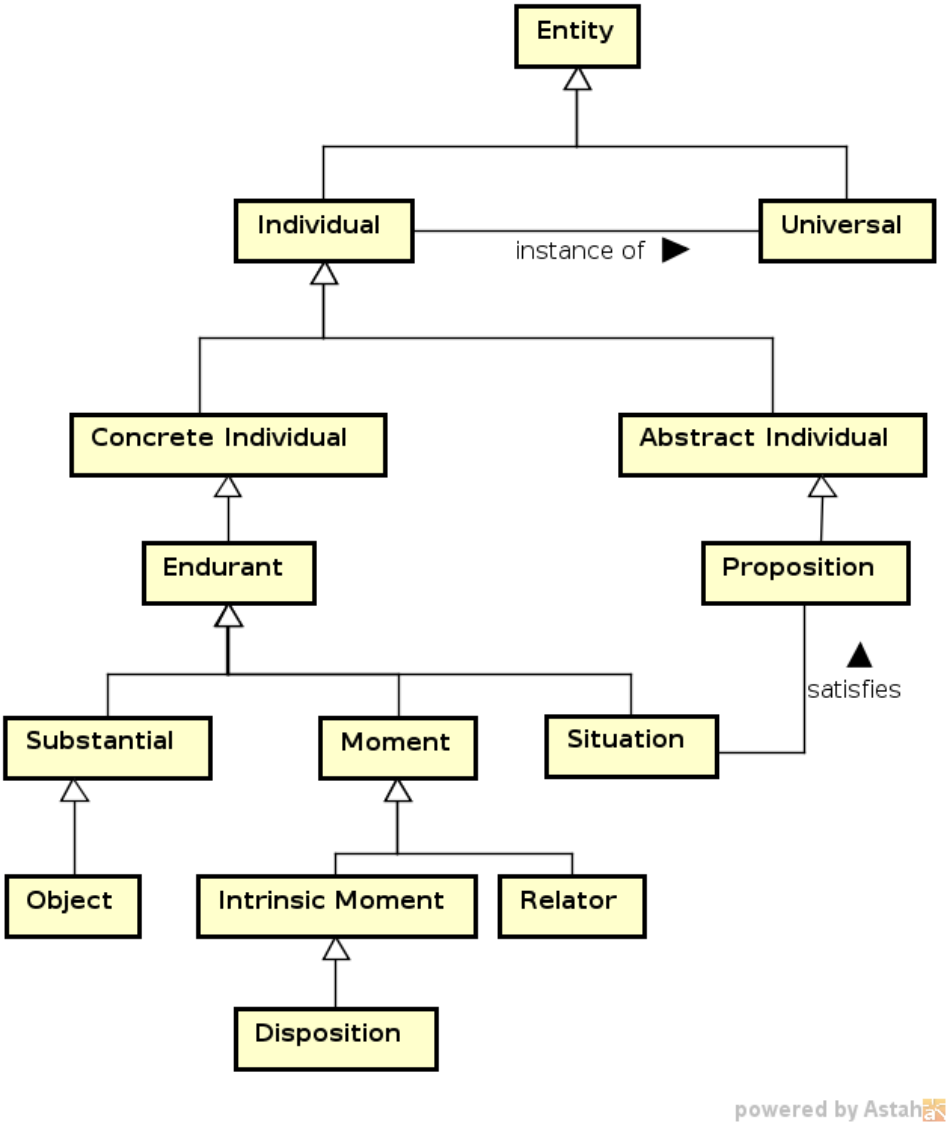

Figura 8 – Fragmento de UFO-A expandindo o elemento Individual

Substantials são existencialmente independentes, não são inerentes a outro indivíduo. Em UFO, objects são substantials com critério de identidade bem definidos e que não dependem necessariamente de todas as suas partes. Um violão com uma corda a menos, por exemplo, continua sendo um violão. Assim, a cidade de Pompéia, uma caixa de som, o próprio violão e o Roger Waters são exemplos de substantial objects.

Moments são endurants existencialmente dependentes de outro individual. Representam uma propriedade ou característica de outro individual como cor, altura ou a carga elétrica de uma bateria, e, por isso, são essencialmente inerentes a outro indivíduo. Moments podem ser intrínsecos ou relacionais. **Intrinsic moments** dependem de um único indivíduo para existir, são intrínsecos a este indivíduo como a cor ou o peso de um instrumento, a habilidade de David Gilmour de tocar guitarra e a vontade de alguém gravar um álbum de música. **Dispositions** são um tipo particular de intrinsic moments que se manifestam em situações particulares como a disposição de ondas sonoras se propagarem em um meio e a disposição de um material magnético atrair um material metálico. Diferente de intrinsic moments, **relators**, também chamados de relational moments, são individuals que conectam outros individuals como um abraço, um pedido de compras ou um casamento. Assim, são existencialmente dependentes de dois ou mais individuals.

Situations (situações) são um tipo complexo de endurant constituídos de muitos endurants (incluindo outras situações) e representam uma porção da realidade que pode ser entendida como um todo. *Roger, David, Richard e Nick estão felizes em um antigo anfiteatro romano* é um exemplo de situação. Neste caso, o subtantial *Roger* com seu intrinsic moment *feliz*, está presente (is present in) na situação assim como *David*, *Richard* e *Nick* com seus respectivos moments (*feliz*).

Proposições (**Propositions**) são uma representação abstrata (**abstract individual**) de situações. Assim, uma situação na realidade pode satisfazer (**satifies**) uma proposição.

Universals

A Figura 9 é o fragmento de UFO-A explorando o conceito universal — categoria ou tipo geral que representa os padrões de características presentes em diferentes indivíduos — e pode ser especializado em monadic universal e relation universal.

Relation Universals (Relations) são entidades que unem outras entidades e podem ser de dois tipos:

Formal relation é um tipo relações que acontece entre duas ou mais entidades diretamente, sem a interferência de qualquer outro indivíduo. Exemplos incluem *mais alto que*, *este dia é parte deste mês*, *N é subconjunto de Q*, dentre outras.

Material Relation possui uma estrutura material própria e inclui exemplos como *trabalhar*

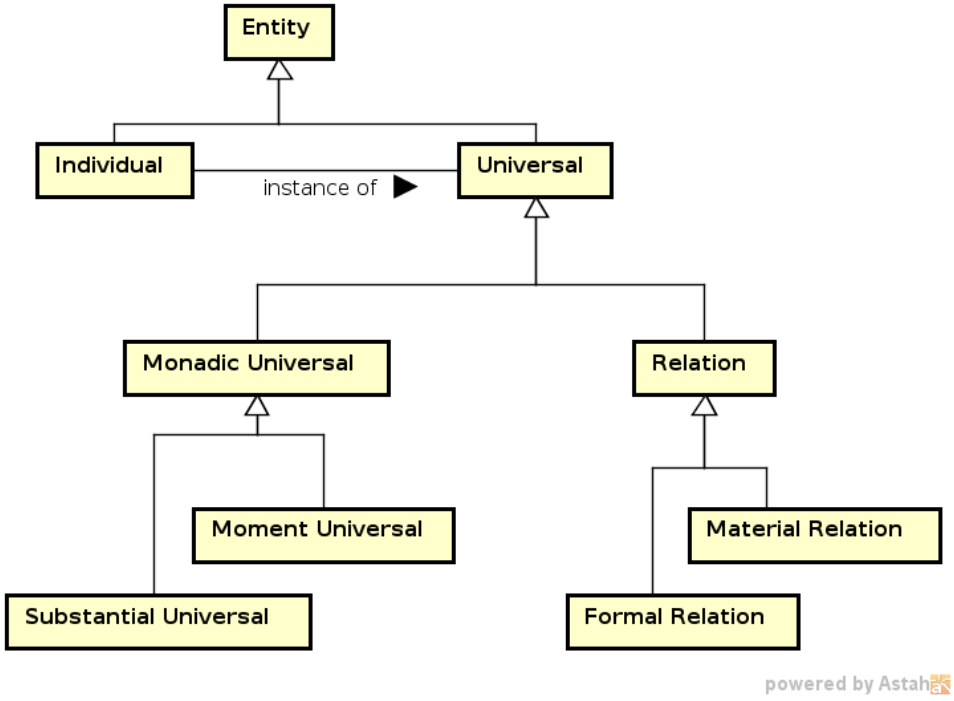

Figura 9 – Fragmento de UFO-A expandindo o elemento Universal

em e *estar casado com*. Relators (já explorados previamente) fazem a mediação das entidades relacionadas por uma material relation.

Por outro lado, ainda na Figura 9, temos **monadic univesals** como universals que não unem outras entidades. São especializados em moment universal (universal instanciado por moment individuals) ou substantial universal .

Substantial universal é especializado em sortal universal e non-sortal universal. **Sortal universals** (Figura 10) são substantial universals que carregam o princípio de identidade de seus individuals, ou seja, agregam os individuals com o mesmo princípio de identidade.

Rigid Sortals são sortal universal cujas propriedades características deste universal são aplicadas a todas suas instâncias enquanto elas existirem. Assim, são conceitos rígidos como *pessoa* e *banda*: Syd Barrett é necessariamente uma instância de *pessoa* enquanto existir. **Kinds** e **subkinds** são especializações de rigid sortals.

Anti-Rigid Sortals são Sortal Universals tal que suas propriedades não se aplicam a todas suas instâncias, ou seja, os indivíduos podem eventualmente instanciá-los enquanto existirem. Por exemplo, Syd Barrett pode enventualmente ser instância de *adulto* e *guitarrista*. **Phases** (fases) são sortais cuja sua instanciação em um indivíduo acontece por uma propriedade intrínseca a ele. Já **roles** (papeis) são sortais relacionalmente dependentes - sua instanciação no indivíduo é determinada por uma propriedade relacional.

Exemplificando esses últimos conceitos: Syd Barret é uma *pessoa* (kind) e *homem* (subkind do kind *pessoa*). Syd Barret, quando adulto (phase), exercia o papel (role) ora de

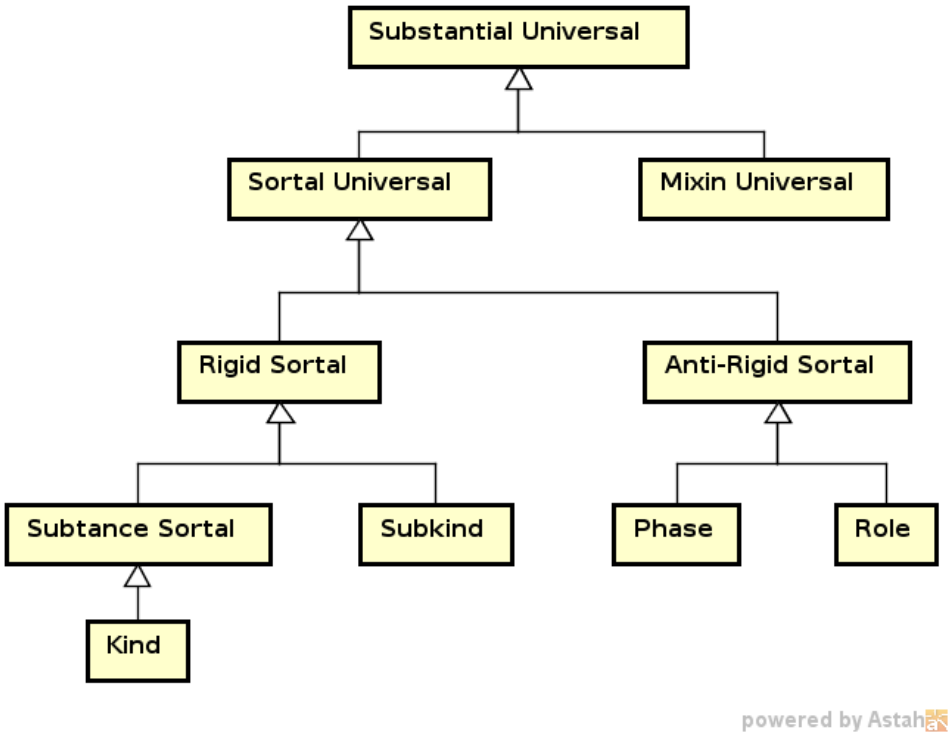

Figura 10 – Fragmento de UFO-A expandindo Sortal Universal

guitarrista, ora de *vocalista* e ora os dois na banda Pink Floyd.

Na Figura 11, tem-se o detalhamento dos **mixin universals**. Também conhecidos como Non-Sortal Universals, mixin universals não provêem uma identidade a suas instâncias e, assim, agrega indivíduos com princípios de identidade diferentes. Podem ser especializados em:

Rigid Mixins são tipos rígidos de mixin. Categoria (Category) agrega propriedades essenciais e comuns a diferentes sortal universals, generalizando conceitos rígidos com diferentes princípios de identidade. Um exemplo bastante comum é *item assegurável* que agrega conceitos rígidos com diferentes princípios de identidade como *pessoa* e *carro*. *Mamíferos* também é um exemplo de categoria.

Non-Rigid Mixins são tipos não rígidos de mixin. **RoleMixin** generalizam conceitos anti-rígidos com diferentes princípios de identidade. Representam papéis (roles) que podem ser desempenhados por tipos diferentes. *Clientes*, por exemplo, podem ser do tipo pessoa física ou pessoa jurídica. Há ainda os **mixins** que lidam com conceitos semi-rígidos como *sentável* — uma propriedade rígida, essencial, para uma *cadeira* ou *sofá*, mas uma propriedade acidental, não rígida, para outros objetos como um *caixote*.

2.5.2 UFO-B

UFO-B (GUIZZARDI; FALBO; GUIZZARDI, 2008; GUIZZARDI et al., 2013) é o fragmento de UFO com o conceito evento (event ou perdurant) como tema central

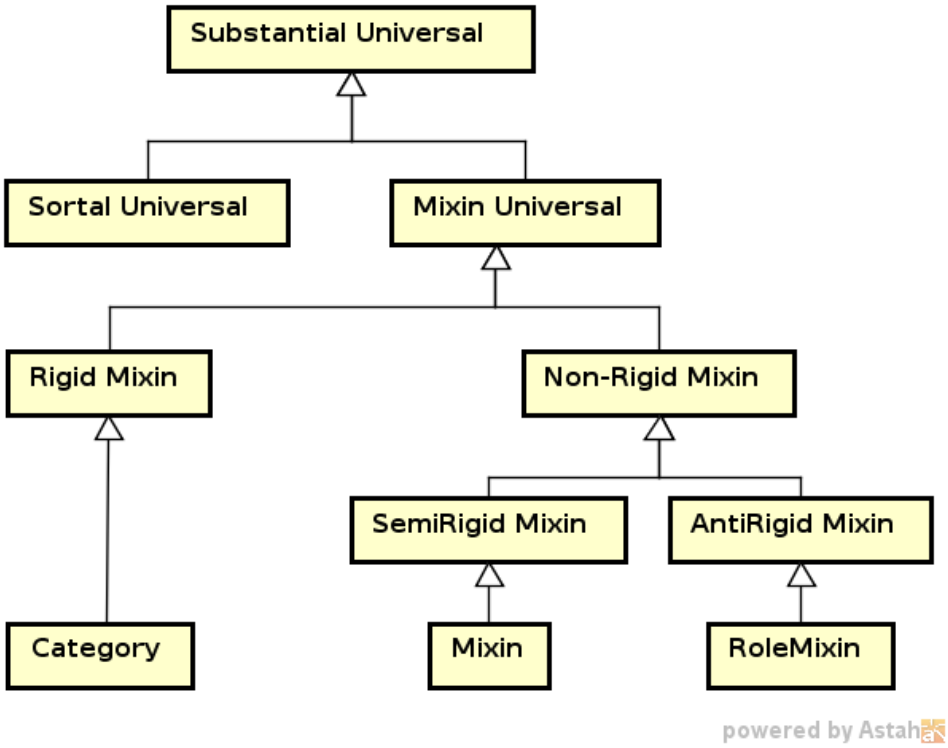

Figura 11 – Fragmento de UFO-A expandindo Mixin Universal

(Figura 12). UFO-A::Endurants são especializações de UFO-A::concrete individual que *existem* através do tempo, mudando mas sem perder sua identidade. Perdurants (events), são também UFO-A::Concrete Individuals que *acontecem no tempo*, se estendem no tempo acumulando partes temporais. Exemplos de eventos são um *jogo de futebol* e um *show de uma banda*. Quando um endurant está presente, nem toda suas partes temporais necessariamente estão presentes. No show memorável da banda Pink Floyd, *The Wall*, em diferentes instantes de tempo, diferentes partes do evento se fazem presente. O muro construído em frente ao palco não era parte do evento presente no início do *show*.

Event universals são padrões de características de eventos que podem ser realizados em várias instâncias diferentes de eventos. Como visto na Figura 12, eventos podem ser classificados de acordo com sua estrutura mereológica (relações entre partes e todo) em eventos atômicos (atomic events), quando não possuem mais de uma parte própria, ou complexos (complex events) quando compostos por dois ou mais events.

Events são entidades existencialmente dependentes no sentido que dependem de participantes (participations) para existir. Neste mesmo show *The Wall*, tem-se a participação dos integrantes da banda, dos instrumentos, do público assistindo, das caixas de som e amplificadores, etc. O evento principal é composto pelas participações individuais de cada entidade — e cada participação é um evento, complexo ou atômico, mas existencialmente dependente de um único participante (UFO-A::substantial).

Eventos impactam o mundo real transformando uma porção da realidade. São

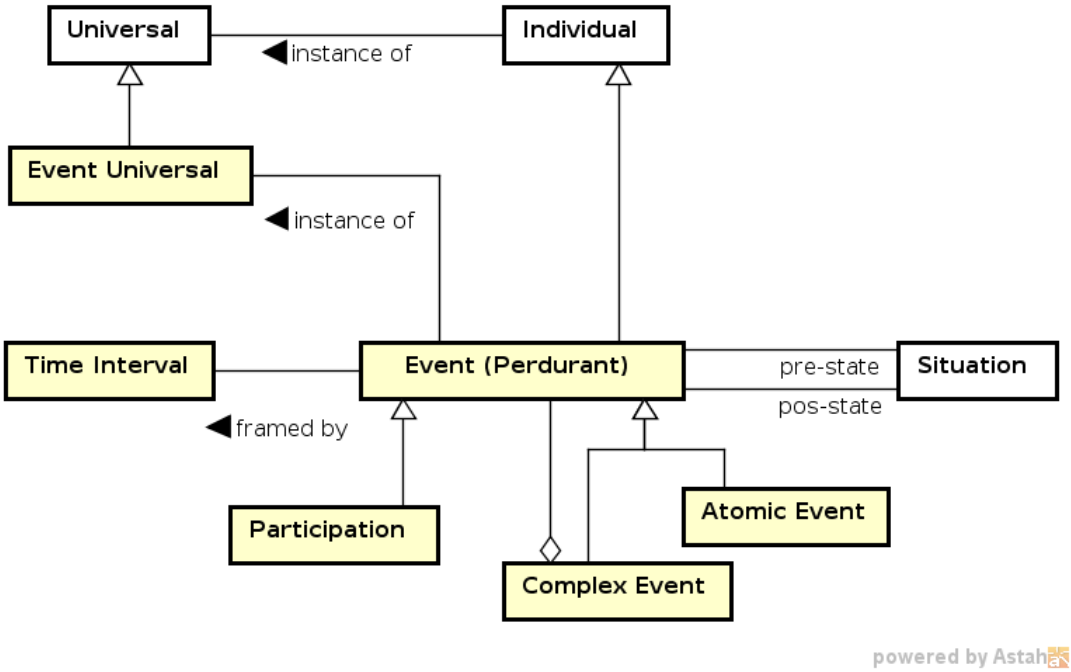

Figura 12 – Fragmento de UFO-B

capazes de alterar o estado de uma situação inicial (pre-state UFO-A:: situation) em uma situação posterior (pos-state).

2.5.3 UFO-C

UFO-C é uma ontologia de entidades sociais. A Figura 13 mostra um fragmento de UFO-C com uma distinção de dois tipos de UFO-A::substantials (UFO-A::endurants que não dependem existencialmente de outro indivíduo para existir): agentes (agents) e objetos (objects).

Agentes realizam ações. Podem ser físicos (physical agent), como uma pessoa, ou sociais (social agent) como uma organização. Por outro lado, objetos são incapazes de realizar ações, mas também podem ser classificados como físicos (physical objects), como um livro, uma árvore, uma bateria ou um microfone, ou como objetos sociais (social objects) como dinheiro e linguagem.

Agents podem ter um tipo especial de UFO-A::intrinsic moment chamado intentional moment (Figura 13). A noção de intencionalidade adotada em UFO é como a proposta por Searle (1998) como a capacidade de um indivíduo de se referir a possíveis situações na realidade. *Intentional moments*, então, são inerentes a agentes (inheres in agents) e podem ser do tipo mental moment ou social moment.

Intenções (intentions), crenças (beliefs) e desejos (desires) são tipos de mental moments. Beliefs podem ser justificados por situações na realidade, é a crença de que

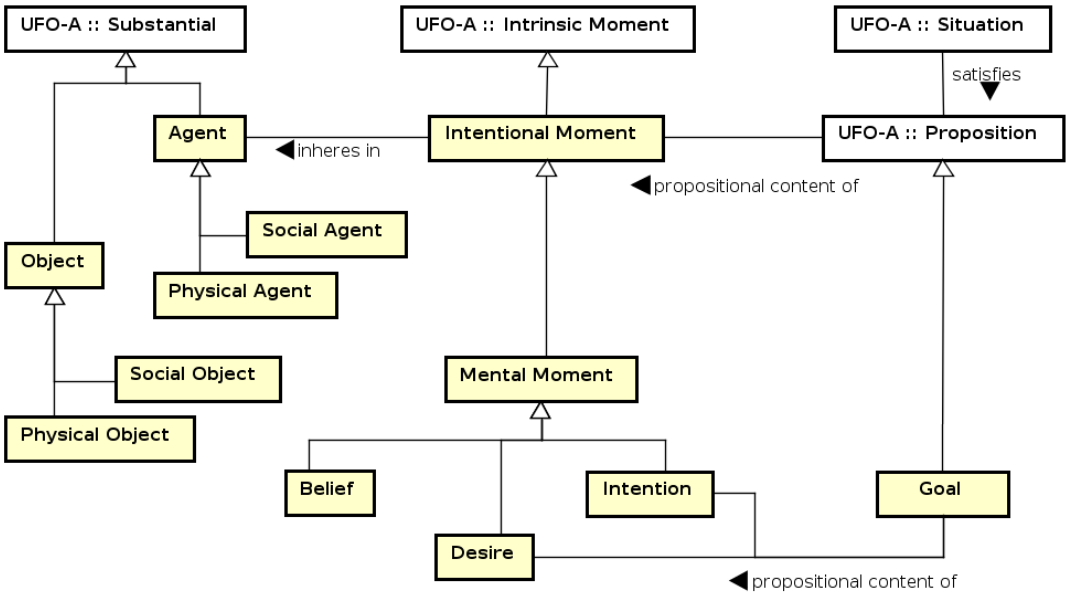

Figura 13 – Fragmento de UFO-C com *Intentional Moments*

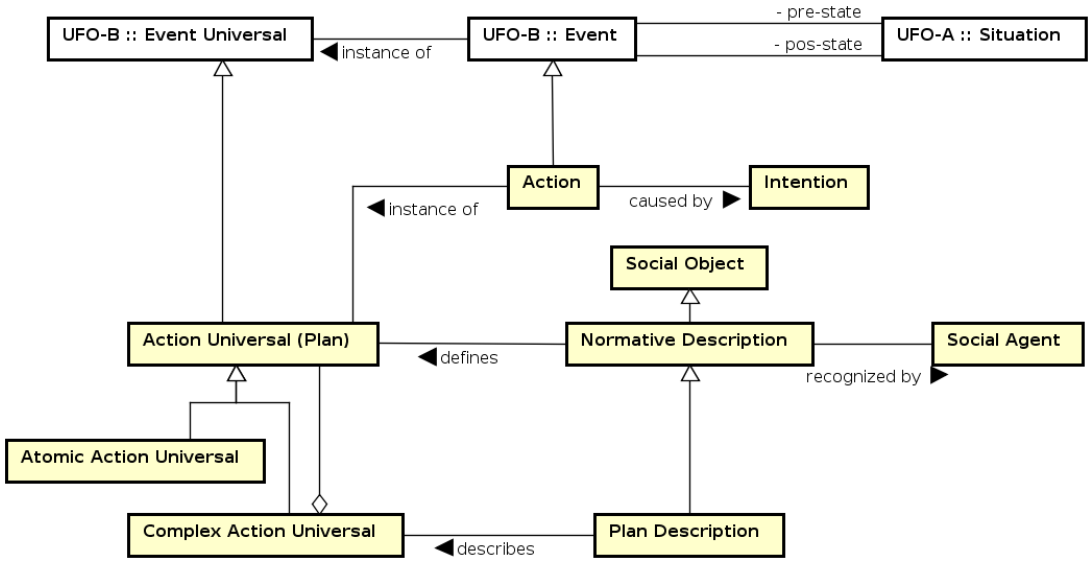

Figura 14 – Fragmento de UFO-C com action

determinada situação existe na realidade (a crença de que *a Lua orbita a Terra e existe um lado escuro na Lua*). Já desires e intentions expressam algum tipo de pretensão de um agente em direção a um estado específico da realidade e, assim, podem ser realizados ou frustrados.

Desires expressam simplesmente a vontade de um agente em determinado estado das coisas, determinada situação, na realidade, como *o desejo de viver em um país sem corrupção*. Intentions expressam também a vontade em determinada situação mas que o agente se compromete a realizar. Assim, intenção é o compromisso interno de um agente a perseguir um estado de mundo desejado levando-o, então, a realizar ações (actions) para que tal estado se realize (Figura 14).

O conteúdo proposicional (UFO-A::proposition) de um intentional moment é a representação abstrata da classe de situações referenciadas por ele e, assim, uma UFO-A::situation é capaz de satisfazer uma UFO-A::proposition. O conteúdo proposicional de uma intention é um objetivo (goal).

Na Figura 14, actions são eventos intencionais, causados pela intention de algum agente. Ações são instâncias de action universal (plan) que, são padrões de características de ações que podem acontecer em diferentes instâncias de action.

Normative descriptions, descrições normativas, são objetos sociais que definem regras/normas reconhecidas por algum agente social como contratos em geral, a constituição de um país ou um conjunto de diretivas sobre como executar ações em uma organização (NARDI et al., 2013). Plan description é um tipo especial de descrição normativa que descreve complex action universals.

2.6 Uma Interpretação Ontológica de Requisitos não-funcionais

Requisitos são comumente classificados como funcionais ou não-funcionais. Basicamente, requisitos funcionais falam sobre *o que* um sistema deve fazer enquanto requisitos não-funcionais descrevem, por exemplo, *o quão bem* o sistema deve executar suas funcionalidades (PAECH; KERKOW, 2004 apud GUIZZARDI et al., 2014). Há, porém, um certo embaraço no uso desses conceitos nas abordagens tradicionais. Em um trabalho de grande importância, Guizzardi et al. (2014) fazem uma interpretação ontológica, fundamentada em UFO, de requisitos com foco em esclarecer o conceito de requisito não-funcional e elementos relacionados.

A ontologia, apresentada na Figura 15, propõe requisitos funcionais (*functional requirement* - FRs) e requisitos não-funcionais (*non-functional requirements* - NFRs) como especializações de objetivo (UFO-C::goal). **Functional requirements** são definidos como aqueles que se referem a funções (UFO-C::functions). Em UFO, *function* é a UFO-C::disposition de uma entidade, ou seja, sua capacidade, aptidão intrínseca (UFO-A::intrinsic moment), com potencial de, em situações específicas, manifestar determinado comportamento através da execução de um evento.

Non-functional requirements referem-se a qualidades (UFO::quality). Qualidades são propriedades inerentes a um indivíduo (UFO-A::intrinsic moment) que se manifestam sempre que elas existem (no indivíduo). Assim, requisitos não-funcionais especificam que determinada qualidade de um indivíduo tenha valores em uma região de qualidade particular. Tal região de qualidade pode ser bem delineada ($0 \sim 5$ *segundos*) ou vaga (*rápida*).

Em (GUIZZARDI et al., 2014), é falado também de *gradable* NFRs: aqueles que

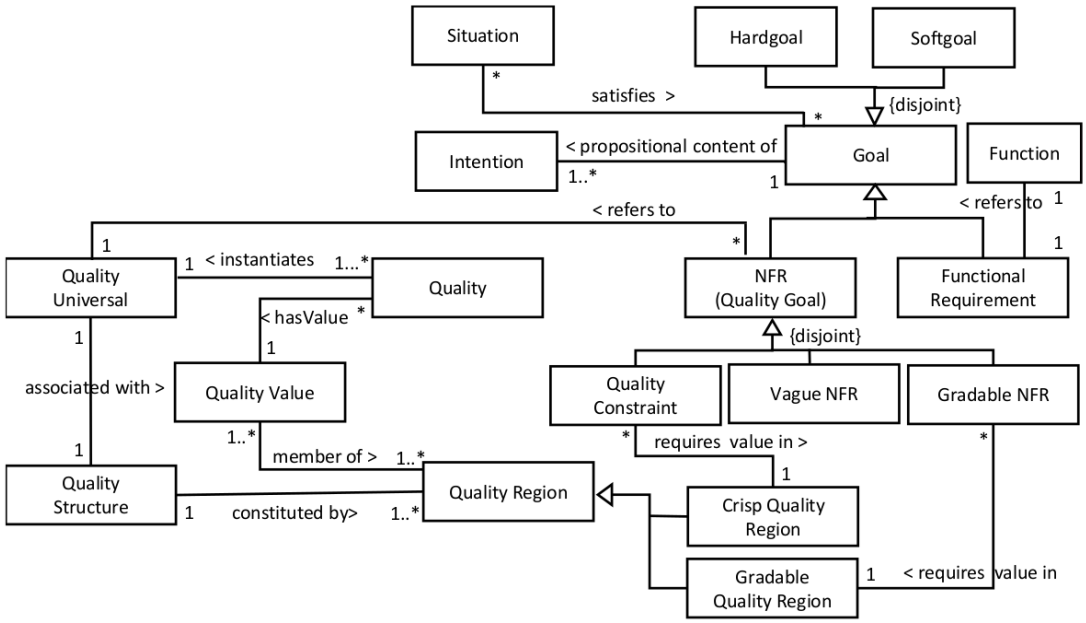

Figura 15 – Ontologia de requisitos não-funcionais e conceitos relacionados (GUIZZARDI et al., 2014)

podem ser satisfeitos em graus diferentes, de forma *boa o suficiente*, por exemplo. Tem caracteríticas relacionadas à lógica *fuzzy* (BARESI; PASQUALE; SPOLETINI, 2010 apud LI et al., 2014). Ainda, o trabalho explora com profundidade o conceito de qualidade e outras noções afins como *quality structure* e *quality region*. Tal nível de detalhamento não é aqui exposto.

De forma ortogonal, a especificidade do requisito o classifica como *hardgoal* ou *softgoals*. Independentemente de o requisito ser funcional ou não-funcional, um ***hardgoal*** é uma proposição que pode ser objetivamente satisfeita por um conjunto conhecido de situações. ***Softgoals*** referem-se a objetivos que ainda não foram devidamente explorados, sendo expressões ainda iniciais e vagas de intencionalidade. Assim, *softgoals* referem-se a critérios imprecisos de satisfabilidade que se sabe existir mas não se sabe exatamente seus limites.

Esta visão ontológica sobre requisitos e objetivos é alinhada com o escopo deste trabalho e é, assim, utilizada de forma a complementar às propostas aqui feitas.

2.7 Engenharia de Ontologias: o método SABiO

SABiO (*Systematic Approach for Building Ontologies*) (FALBO, 2014; FALBO, 2004) é um método para desenvolvimento de ontologias baseado em metodologias da Engenharia de Software já bem estabelecidas na comunidade. É composto por um processo de desenvolvimento contendo cinco fases e cinco processos de apoio como mostra a Figura 16.

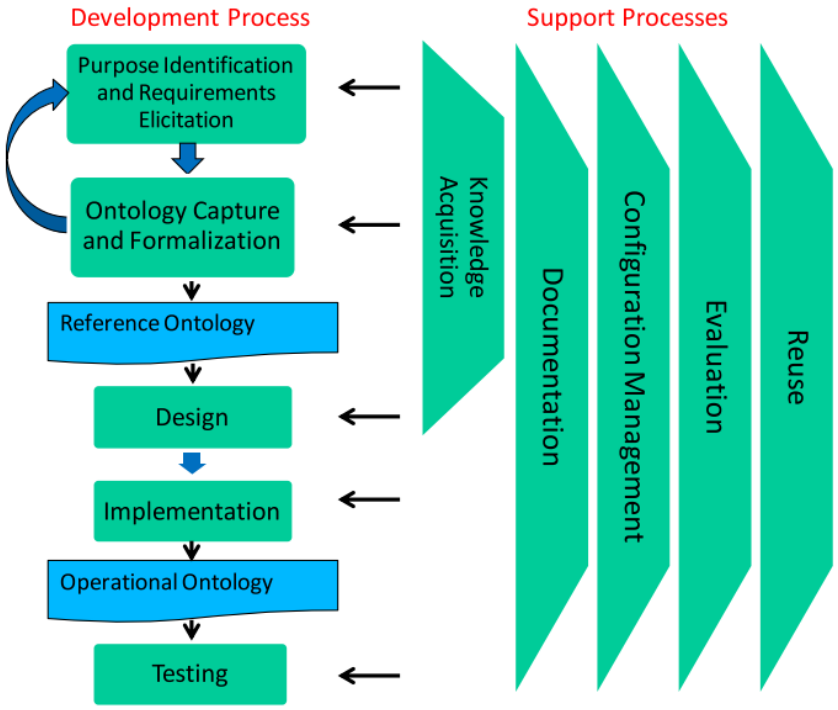

Figura 16 – Processos do Método SABiO (FALBO, 2014)

Identificação de Propósito e Elicitação de Requisitos Na primeira fase do processo é identificado o propósito da ontologia e suas intenções de uso. Em seguida, assim como requisitos de *software*, são elicitados os requisitos funcionais e não-funcionais da ontologia. Requisitos funcionais referem-se ao conhecimento (conteúdo) representado pela ontologia e são estabelecidos como questões de competência (CQs - *Competency Questions*) (GRÜNINGER; FOX, 1995) que a mesma deve responder. Requisitos não-funcionais tratam de qualidades, características e outros aspectos gerais da ontologia não relacionados diretamente ao conteúdo.

Captura e Formalização da Ontologia Guiando-se pelas questões de competências levantadas na fase anterior, esta fase consiste na captura da conceituação do domínio de interesse por meio de uma ontologia de referência de domínio. O conhecimento para construção de tal ontologia pode ser obtido de especialistas do domínio, livros, padrões internacionais, modelos de referência e outras fontes consolidadas sobre o assunto. Com foco na qualidade da representação da realidade e com intuito de ser utilizada por humanos como suporte na comunicação e negociação semântica, uma linguagem gráfica de alta expressividade deve ser utilizada para representação do domínio. Ainda, para que seja bem embasada, é importante que os conceitos e relações da ontologia de referência de domínio sejam analisados à luz de uma ontologia de fundamentação.

Design Se o intuito é obter uma ontologia operacional, para ser utilizada por máquinas, então segue-se a fase de *design*. Uma mesma ontologia de referência pode ser utili-

zada para produzir diferentes ontologias operacionais dependendo, por exemplo, de diferenças arquiteturais e tecnológicas que envolvem o uso da ontologia operacional intencionada. Desta forma, nesta fase, a especificação conceitual da ontologia de referência é transformada em uma especificação de *design* levando em consideração os aspectos particulares do ambiente alvo de uso desta ontologia operacional.

Implementação Na fase de implementação a ontologia operacional é implementada em uma linguagem de ontologias operacionais legível por máquina.

Teste A fase de testes é guiada pelas questões de competência e refere-se à verificação e validação da ontologia operacional utilizando um conjunto de casos de teste.

O intuito deste trabalho é produzir uma ontologia de referência de domínio e, assim, somente as duas primeiras fases do processo de desenvolvimento de SABiO são utilizadas.

Ainda, cinco processos de suporte são executados em paralelo às cinco fases de desenvolvimento, como mostrado na Figura 16: Aquisição de Conhecimento (*Knowledge Acquisition*), Documentação (*Documentation*), Gerência de Configuração (Configuration Management), Avaliação (*Evaluation*) e Reúso (*Reuse*).

No contexto deste trabalho, os processos de Aquisição de Conhecimento, Reúso e Avaliação foram os mais ativos. A Aquisição de Conhecimento ocorre principalmente nas fases inicias do desenvolvimento da ontologia e consiste em capturar o conhecimento sobre o domínio. Tal captura pode ocorrer de diferentes formas como consultas a especialistas do domínio e fontes confiáveis como livros, artigos e outros materiais bibliográficos consolidados na área. As principais fontes utilizadas para adquirir conhecimento sobre o domínio GORE no processo de construção da GORO foram artigos e livros publicados sobre o domínio, como os referenciados nas seções 2.1 e 2.3.

Reúso diz respeito a reutilizar, totalmente ou parcialmente, conceituações já estabelecidas para o domínio. As maiores fontes para reúso são ontologias de domínio já existentes, ontologias *core*, ontologias de fundamentação e padrões (*pattern*) ontológicos. A ontologia sobre requisitos não-funcionais (GUIZZARDI et al., 2014) explorada na Seção 2.6 foi reutilizada na construção da GORO.

Avaliação abrange verificar se a ontologia está sendo construída corretamente, ou seja, se ela cumpre as especificações previamente estabelecidas (processo de verificação) e validar se a ontologia correta está sendo construída, ou seja, se ela atende ao propósito (processo de validação). Para verificação da GORO, as questões de competências levantadas foram respondidas utilizando os elementos (conceitos e relacionamentos) da ontologia proposta. Já na validação, utilizou-se a GORO para interpretar o domínio GORE descrito por diferentes abordagens GORE.

2.8 Trabalhos Correlatos

Cares et al. (2010) fala sobre a necessidade da definição e uso de um metamodelo da abordagem GORE $i^\star$ já que, devido a grande variedade de dialetos e variações da abordagem, ao se ler um trabalho sobre $i^\star$ é necessário entender primeiro qual 'versão' $i^\star$ está sendo usada. Ainda, Cares (2012) argumenta que a definição da linguagem $i^\star$ não é sólida já que diferentes grupos optam por leves alterações das definições originais. Assim, propõe um metamodelo unificado de $i^\star$ e um *framework* para entender as variações de $i^\star$, e suportar a interoperabilidade e integração de tais variações. O trabalho traz um estudo sobre as variantes da abordagem e propõe um metamodelo de referência.

Lucena et al. (2008) buscam a integração de metamodelos da abordagem $i^\star$. Matulevicius e Heymans (2005), por outro lado, propõem uma análise precisa dos elementos de KAOS mas faz tal análise utilizando somente o metamodelo das linguagens estudadas. Nwokeji, Clark e Barn (2013) verificam que não existe um metamodelo completo de KAOS e analisa trabalhos sobre a definição da *linguagem* para consolidar um único metamodelo. Fayouni, Kavakli e Loucopoulos (2015) buscam uma visão unificada das diferentes abordagens GORE afirmando que cada linguagem traz particularidades sintáticas e semânticas. Propõem um metamodelo unificado GORE, porém fazem isto avaliando os metamodelos das diferentes abordagens GORE. Patrício et al. (2011) também propõem um metamodelo GORE, que incorpora os conceitos de abordagens GORE já estabelecidas.

Todavia, esses trabalhos utilizam os metamodelos das abordagens já estabelecidas como embasamento de suas propostas. Metamodelos, porém, descrevem os aspectos de notação, a sintaxe de uma linguagem e não têm o propósito de esclarecer a semântica dos elementos (HAREL; RUMPE, 2000). Assim, tais trabalhos não estabelecem uma conceituação bem fundamentada do domínio GORE.

2.8.1 *Core Ontology for Requirements* (CORE)

DOLCE -*Descriptive Ontology for Linguistic and Cognitive Engineering* - é uma ontologia de fundamentação que visa capturar categorias ontológicas subjacentes a linguagem natural e ao senso comum - são categorias concebidas como artefatos cognitivos dependentes da percepção humana, impressões culturais e convenções sociais. Em DOLCE um *particular* pode ser um *endurant*, um *perdurant*, uma *quality* ou um *abstract* (MASOLO et al., 2003).

Jureta, Mylopoulos e Faulkner (2008b) propõem uma ontologia para requisitos baseada em DOLCE, *Core Ontology for Requirements* (CORE), que define conceitos primitivos para o domínio de requisitos como objetivo, tarefas/planos, softgoals e outros. A definição de tais conceitos, parte de uma diferenciação dos tipos de atos de fala (*speech acts*) a seguir:

Assertivo Quando o locutor acredita na veracidade do conteúdo comunicado. Independente de ser verdadeiro ou não.

Diretivo O conteúdo do ato de fala diretiva descreve as condições que o locutor deseja ver se tornarem verdadeiras.

Comissivo O discurso comissivo indica que o locutar pretende realizar ações descritas no conteúdo da fala.

Expressivo Um ato de fala expressivo transmite a atitude, a emoção ou o sentimento do locutor sobre uma condição possível na realidade.

Declarativo Um ato de fala declarativo origina as condições especificadas em seu conteúdo, desde que o papel do locutor permita a realização de tais condições.

Declarativo representativo Um ato de fala declarativo representativo é utilizado para confirmar que as condições descritas em seu conteúdo são verdadeiras.

E, assim, caracteriza cada *speech act* como um tipo de conceito relacionado ao domínio GORE:

- Falas assertivas, declarativos ou representativas declarativas revelam um conteúdo em que o locutor acredita e são classificadas como *suposições de domínio*.
- Falas diretivas revelam um conteúdo que é desejado pelo locutor e são classificadas como *objetivos*.
- Falas expressivas dizem respeitos a avaliação do locutor e são classificadas como *avaliação*.
- Falas comissivas revelam a intenção do locutor e são classificadas como *planos*.

DOLCE, porém, não é apropriada para explicar todos fenômenos ontológicos envolvidos para definir e lidar com requisitos não-funcionais (GUIZZARDI et al., 2014), por exemplo.

3 *Goal-Oriented Requirements Ontology*

Este capítulo apresenta a ontologia proposta neste trabalho. A Seção 3.1 apresenta os requisitos de GORO. A Seção 3.2 apresenta os modelos conceituais de GORO. A Seção 3.3 aborda uma perspectiva de engenharia de requisitos evidenciando a separação do escopo do problema e escopo da solução.

3.1 Requisitos de GORO

A Ontologia de Requisitos Baseados em Objetivos (GORO) foi construída utilizando o método SABiO (FALBO, 2004). Na primeira fase do processo são identificados o propósito e os requisitos da ontologia. Inicialmente, como explicado na Seção 1.1, a área de Engenharia de Requisitos Baseados em Objetivos tem grande importância no processo de desenvolvimento de *software* mas não conta, ainda, com um modelo conceitual ontologicamente bem fundamentado do domínio. Assim, este trabalho visa preencher esta lacuna, propondo uma ontologia de referência de domínio de GORE.

Em seguida foram elicitados os requisitos funcionais e não-funcionais da ontologia. Os requisitos funcionais, que delimitam o escopo da ontologia, são estabelecidos na forma de questões de competência (CQ) e listados a seguir.

QC1. O que é um objetivo?

QC2. O que é um requisito?

QC3. De onde vem os objetivos e requisitos?

QC4. Como objetivos se relacionam entre si?

QC5. Como objetivos podem ser satisfeitos?

QC6. O que é uma especificação?

QC7. O que é uma suposição no contexto de GORE?

QC8. Que tipos de requisitos existem?

QC9. Como tarefas se relacionam com requisitos?

Como requisitos não-funcionais (NFR), os seguintes foram elicitados:

NFR1. Ser construída utilizando a caracterização do domínio absorvida de diferentes abordagens GORE.

NFR2. Ser fundamentada em uma ontologia de fundamentação conhecida.

NFR3. Reutilizar ontologias já existentes de temas relacionados ao domínio.

Para satisfazer NFR1, foram exploradas as conceituações dadas por três diferentes abordagens GORE apresentadas na Seção 2.3: $i^\star$, KAOS e Techne. Inicialmente, KAOS e $i^\star$ foram escolhidas por conta de sua popularidade. Em um recente mapeamento sistemático da literatura (HORKOFF et al., 2016) conduzindo no final do ano de 2016, 246 publicações sobre GORE em *journals* e conferências internacionais foram encontradas e analisadas. Deste montante, 21% dos trabalhos falavam de modelos de objetivos sem utilizar uma abordagem específica. KAOS e $i^\star$ (e abordagens baseadas em $i^\star$) aparecem como sendo as abordagens mais utilizadas por mais de um terço do total de publicações mostrando a grande visibilidade de tais abordagens. Techne, por outro lado, foi incluída por já ser baseada em uma ontologia e, assim, poderia contribuir na caracterização do problema com uma visão bem embasada sobre o domínio. Techne é baseada na ontologia CORE (JURETA; MYLOPOULOS; FAULKNER, 2009) que por sua vez é fundamentada em DOLCE (MASOLO et al., 2003), como já apresentado na Seção 2.8.1.

NFR2 é satisfeito embasando a GORO em UFO (GUIZZARDI, 2005; GUIZZARDI; FALBO; GUIZZARDI, 2008). Como explicado na Seção 2.5, UFO é uma ontologia de fundamentação que tem se demonstrado eficiente no embasamento semântico de ontologias em domínios diversos (GUIZZARDI et al., 2015). Há, inclusive, análises de pequenos fragmentos do domínio GORE feitas à luz de UFO como o trabalho desenvolvido por Guizzardi et al. (2014) que, como será visto, é reutilizado em GORO. UFO foi construída com o objetivo principal de desenvolver bases para modelagem conceitual, explorando os conceitos mais fundamentais — o que não é suficientemente explorado por outras ontologias de fundamentação (GUIZZARDI, 2005). DOLCE (MASOLO et al., 2003), por exemplo, foca somente em propriedades intrínsecas enquanto UFO traz noções de relações materiais e propriedades relacionais e apresenta, também, um sistema de categorias mais completo (GUIZZARDI et al., 2014).

Em relação ao NFR3, são reutilizadas conceituações bem fundamentadas de conceitos referentes ao domínio de interesse como a interpretação de requisitos não-funcionais de Guizzardi et al. (2014) — referenciada na GORO pela sigla *NFR* — e a contribuição de Wang et al. (2016) na área de suposições de domínio (*assumptions*, *domain assumptions*) — referenciada em GORO pela sigla *ASMP*. Também são utilizados, e explicados neste capítulo, outros trabalhos como referências na conceituação de elementos de interesse como a contribuição de Wang et al. (2014) sobre os trabalhos de Zave & Jackson sobre o significado e escopo de requisitos. Tais conceituações isoladas foram reutilizadas, conectando tais fragmentos para compor GORO.

3.2 A Ontologia

GORO é apresentada nas Figuras 17 e 18 em diagrama de classes UML (*Unified Modeling Language*) (OMG, 2011). Os conceitos introduzidos por GORO são representados em amarelo e conceitos das ontologias reutilizadas são prefixados com suas respectivas siglas. Sendo fundamentada em UFO, GORO utiliza os conceitos de UFO especializando-os em conceitos específicos do domínio GORE.

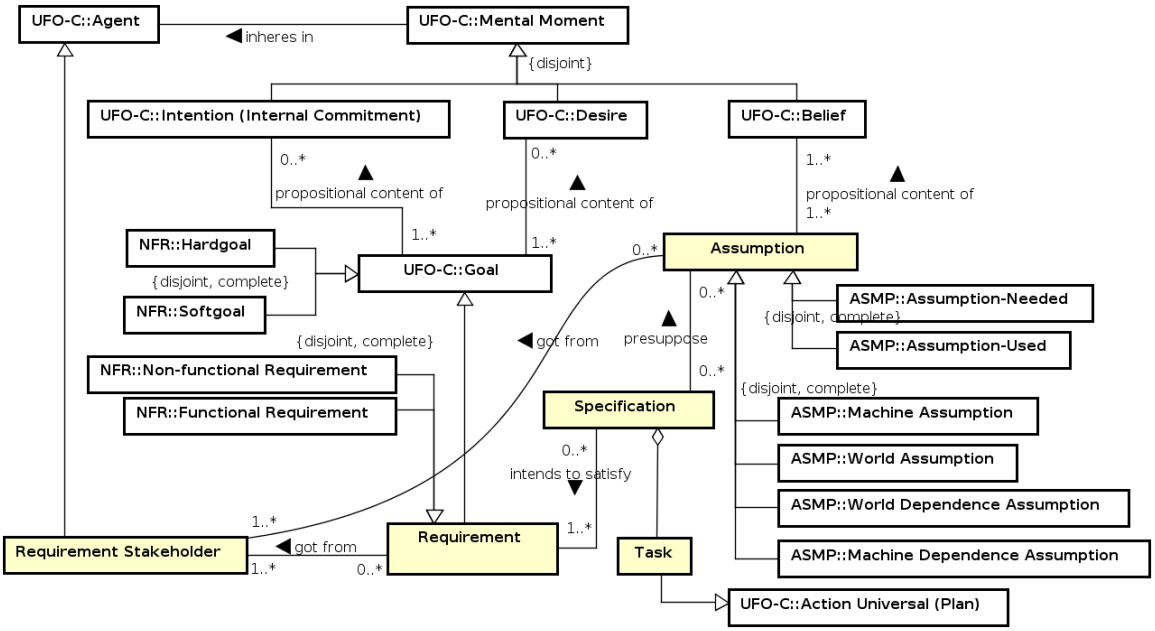

Figura 17 – Fragmento central de GORO

Intenções, Desejos e Objetivos

Em GORE, a caracterização do problema a ser solucionado inicia-se entendendo e modelando os objetivos dos envolvidos no problema para, assim, identificar os requisitos do projeto e dar continuidade aos demais processos de Engenharia de Software.

Em UFO (Seção 2.5.3), um UFO-C::agent (agente) é um substantial, entidade existencialmente independente, que cria ações (actions), percebe eventos e possuem propriedades intencionais, UFO-C::mental moments, como desejos (desire), crenças (belief) e intenções (intention). UFO-C::Mental moments são existencialmente dependentes: são inerentes a agentes. Goal é o conteúdo proposicional de dois possíveis mental moments de um agente: intention ou desire. A Figura 17 mostra o fragmento de GORO focado no conceito de momento mental e seus tipos.

Desires expressam a vontade de um agente relativa a um estado particular de coisas na realidade. É o desejo de que a temperatura na cidade esteja mais amena ou o desejo de que já esteja instalado o novo aparelho de ar-condicionado comprado na última semana pela empresa. Por outro lado, intentions representam não somente uma vontade, mas

também um compromisso interno do agente de agir em direção ao estado desejado da realidade. É o desejo de ter o novo ar-condicionado funcionando e, para tal, convocar a equipe responsável para executar a tarefa de instalá-lo ou até mesmo subir na escada e instalar o mesmo, alcançando seu objetivo. Desta forma, tanto desejos quanto intenções se relacionam com os objetivos do agente, mas um objetivo como intenção é sempre associado a um plano para realizá-lo (GUIZZARDI; GUIZZARDI, 2010).

Considerando que modelos de objetivos podem ou não ser extensivamente detalhados, normalmente não se pode determinar o tipo de mental moment de um goal expresso no modelo, já que não é comum expressar se o objetivo é originário de um desejo ou uma intenção. Porém, com o modelo concluído, goals associados a plans para realizá-los são notoriamente conteúdo proposicional de intentions. Tal associação, porém, não garante que o objetivo em questão seja de fato o conteúdo proposicional de uma intenção. Assim, não existe uma restrição formal sobre representada no modelo.

Efetivamente, quando em um processo de Engenharia de Requisitos determinados objetivos de agentes são identificados e levados adiante na modelagem, então possivelmente tais objetivos serão considerados na solução do problema e haverá um esforço para alcança-los: há intenção de satisfazê-los. Desejos originam goals sem um comprometimento para persegui-lo ativamente. Assim, é possível que tais objetivos oriundos de desires não apareçam no modelo final de objetivos.

Requisitos e Especificações

Na Figura 17, um agent se torna um requirement stakeholder quando ele é parte de um processo de Engenharia de Requisitos de forma que seus goals se tornam requirements de uma solução intencionada. Como descrito por Zave e Jackson (1997) em seu trabalho seminal, e refinado por Wang et al. (2014), um requisito é um comportamento desejado no ambiente, independente da máquina. Mesmo em processos de Engenharia de Software, onde o produto final é um sistema de *software*, requisitos não se referem ao comportamento da máquina (do *software*) e, sim, ao comportamento desejado no ambiente.

Isto porque quando falamos de requisitos estamos lidando, inicialmente, com o escopo do problema, e não da solução. Abstrair a solução (*software*) e concentrar no problema, nos objetivos dos envolvidos, é o que propicia as vantagens no uso de GORE sobre outras práticas de Engenharia de Requisitos (Seção 2.3). Assim, em GORE, ***requisito* é um *objetivo* no escopo de um problema específico, que descreve condições no ambiente a serem atingidas *por meio de uma solução desejada*, resultando na satisfação dos objetivos estratégicos subjacentes**. Assim, goals se tornam requirements quando fazem parte do escopo de um esforço que pretende satisfazê-los.

Requisitos dizem respeito a *o que* deve acontecer no mundo, qual o comportamento

desejado no ambiente — levantando o *problema*. Especificações, por outro lado, descrevem *como* tal comportamento desejado no ambiente será alcançado — lidando com a solução de tal problema. Não são artefatos ou documentos, embora normalmente estejam descritos em tais. Mas podem existir *somente* na mente de um agente (WANG et al., 2014). Na GORO, specifications essencialmente se referem ao comportamento desejado de algo, como um sistema de *software*, no intuito de cumprir *os* requisitos. Assim, em GORO, specification é uma composição de planos, tarefas (tasks), que tem a pretensão de satisfazer os requisitos, como detalhado posteriormente.

Wang et al. (2014) tratam de especificação como o comportamento desejado especificamente em uma máquina (ou em sua interface com o mundo). Em Engenharia de Requisitos, nem todos requisitos elicitados serão atendidos por uma máquina: a solução dos requisitos pode envolver ação humana através de especificação de modelos de processo, sistemas organizacionais e mais. A solução, então, pode vir por meio de sistemas sociotécnicos — envolvendo máquinas e pessoas.

GORO não modela os artefatos, os documentos, onde estariam redigidos os requisitos, as especificações ou as suposições de domínio, por exemplo. O interesse deste trabalho é entender, primeiro, o que são esses elementos, e não como eles são registrados.

A grande vantagem de processos GORE é a busca pelo entendimento do problema levantando primeiro o propósito dos envolvidos, seus objetivos, para depois elicitar os requisitos e, em seguida, partir para a solução já com uma visão ampla do escopo do problema. É inegável, porém, que requirement stakeholders requisitem soluções específicas para satisfazer determinados problemas levantados. Pode ser que a solução intencionada precise ser um *software* de determinada tecnologia e compatível com algum sistema legado, por exemplo. Apesar de GORE lidar essencialmente com o escopo do problema, é fundamental que questões relacionadas a solução também sejam abrangidas.

A Engenharia de Requisitos Baseados em Objetivos tem foco nos requisitos iniciais (*early requirements*), isto é, detalhando o domínio do problema — ao invés da solução. Entretanto, é comum utilizar GORE também descrevendo requisitos mais voltados à solução (*late requirements*) ou mesmo tratando de projeto de arquitetura da solução como, por exemplo, em (PIMENTEL et al., 2014).

Lidar com goals em diferentes níveis de abstração é uma questão importante: eles podem ser mais estratégicos, se referindo a questões sobre o domínio do problema, questões que justificam e motivam uma possível solução; ou mais técnicos, próximos ao domínio da solução, definindo como objetivos estratégicos serão satisfeitos. Porém, o limite entre objetivos estratégicos e objetivos técnicos pode ser impreciso e relativo. Desta forma, uma comparação entre os níveis de abstração de objetivos pode ser feita. Em GORO, os diferentes tipos de relações apresentados na Figura 18 mudam o nível de abstração dos objetivos de diferentes maneiras e serão tratadas adiante.

Hardgoals, Softgoals, Requisitos Funcionais e Requisitos Não-Funcionais

GORO adota a conceituação dada por Guizzardi et al. (2014) que esclarece a discussão sobre *hardgoals*, *softgoals*, requisitos funcionais e não-funcionais. GORO adere a esta conceituação porém explicita que um objetivo (goal) pode ser um requisito requirement e, este sim, pode ser do tipo funcional ou não-funcional.

Hardgoals são definidos como proposições que são objetivamente satisfeitas por um determinado conjunto de situações, como *Sala de reuniões com temperatura entre 20 e* $24°C$. Por outro lado, softgoals se referem a regiões vagas de qualidade nas quais as fronteiras exatas são desconhecidas como *Sala de reuniões com temperatura agradável*. Não sendo possível determinar *a priori* o conjunto de situações que os satisfazem, sua satisfação é normalmente relacionada com o julgamento de um agente. Assim, podem ser expressões iniciais e vagas de um objetivo que não foi suficientemente definido ainda.

De forma ortogonal, requisitos derivados de objetivos podem ser classificados como funcionais ou não-funcionais. Requisitos Funcionais se referem a funções que têm o potencial de manifestar determinado comportamento em situações particulares como *verificar temperatura ambiente de tempo em tempo* ou *contar quantas pessoas estão na sala quando o aparelho for ligado*. Já Requisitos não-funcionais se referem a qualidades. Por exemplo, *sistema com um dispositivo de controle remoto intuitivo*.

Suposições

Em GORE, suposições não expressam a vontade de um agente em direção a uma situação na realidade (como goals fazem), mas a crença de um agente de que determinada situação existe na realidade. Assim, descreve o estado das coisas no ambiente de interesse que o requirement stakeholder acredita ser verdadeiro — é o conteúdo proposicional de belief.

Pode ser necessário considerar tais situações em uma determinada solução. Como visto na Seção 2.3, erros na especificação dos requisitos podem ter origem nas suposições inadequadas do ambiente em que o sistema solução atuará (JACKSON, 1995; LEVESON, 1995). Assim, GORO utiliza a conceituação dada por Wang et al. (2016) que explora a fundo a natureza das suposições classificando-as em diferentes tipos, como mostrado na Figura 17 e explicada a seguir.

Uma world assumption é uma suposição sobre um fenômeno no ambiente, não visível pela máquina. Machine assumption é uma suposição sobre um fenômeno interno da máquina. Machine dependence assumptions e world dependence assumptions conectam os estados do mundo e da máquina. Isto implica que fenômenos ocorridos na máquina e no mundo devem estar simultaneamente refletidos no mundo e na máquina para garantir que estados no mundo estão corretamente representados na máquina e vice-versa.

Expressar todas suposições relacionadas em um projeto é incomum em GORE já que algumas vezes elas podem ser óbvias. Em um sistema de controle de temperatura, por exemplo, assume-se que os dados do sistema deveriam refletir a temperatura real da sala. Neste caso, a world dependence assumption é normalmente suprimida do modelo.

Entretanto, representar algumas suposições se torna necessário em situações em que elas não são tão óbvias ou mesmo quando é de suma importância que elas sejam levadas em consideração no projeto de solução e, assim, deseja-se enfatizá-las.

No mesmo sistema, poderia haver o objetivo "supervisor informado quando a temperatura do *data-center* passar dos 25°C". Para tal, uma possível especificação seria "Informar o supervisor via mensagem de texto". Uma suposição importante neste cenário é a machine dependence assumption de que "quando a mensagem é enviada pelo sistema, o supervisor a receberá". Revelar esta suposição, óbvia, pode ser importante já que expõe uma nuance bastante relevante que poderia não ter sido pensada *a priori*: e se a mensagem não for entregue? Talvez por uma falha no sistema de envio, queda de energia, ou uma falha em sistemas de terceiros, como a operadora de telefonia.

Tais suposições expõem pontos de confiança que temos, muitas vezes inconscientemente, que as coisas funcionarão como o esperado. Assim, explicitar tais suposições implica possibilidade de prever possíveis falhas, quando o ambiente não se comporta como se acredita, e criar alternativas de solução.

Suposições ainda se diferem em dois tipos (ortogonais aos ditos acima): assumptions-used e assumptions-needed (WANG et al., 2016). As primeiras são proposições usadas *a priori* na construção de um novo argumento. Em um modelo, então, assumptions-used descrevem o ambiente em que o sistema estará inserido e, assim, o que deverá ser levado em consideração no desenvolvimento do mesmo. Já assumptions-needed são proposições necessárias posteriormente para sustentar uma conclusão anterior, evidenciando a situação na qual a solução projetada funcionará. Ou seja, assumptions-needed descrevem como o ambiente deverá estar para o correto funcionamento da solução.

O controle do ar-condicionado, por exemplo, eventualmente será levemente molhado pelas mãos não secas de alguém que o manipula — assumption-used — mas funcionará somente em ambiente com humidade abaixo de 100% (não está submerso ou debaixo de chuva) — assumption-needed.

GORO incorpora tal conceitualização. Estes dois tipos de assumptions tem implicações muito diferentes. Se o mal funcionamento de um sistema, por exemplo, é causado por uma assumption-used errada, então o problema é de projeto já que foram assumidas situações no ambiente da solução que, algumas vezes, não acontecem. Se o problema é por conta de assumption-needed, então o usuário utilizou o sistema fora do escopo intencionado para uso do mesmo.

Expressar ou não esses tipos de elementos é uma decisão do modelador. O escopo do problema pode tornar uma suposição importante ou não de ser representada. Evidentemente, explicitar esses diferentes tipos de suposição em modelos GORE pode ser útil para um entendimento mais completo do problema. Ainda, entender os diferentes tipos existentes de suposição como os usados e GORO ajudam simplesmente por guiar o modelador a pensar nessas possibilidades.

Decomposição e Alternativas

A Figura 18 mostra o framento de GORO focado nas relações de goal. Primeiramente, o conceito de decomposition refina um **goal requirement** em seus sub-objetivos. Sendo um goal uma proposição, podemos separá-lo em diferentes partes que, juntas, compõem o goal original ($G \iff G_1 \wedge G_2 \wedge G_3 \wedge ... \wedge G_n$). Assim, G é satisfeito exatamente pelas situações que satisfazem $G_1 \ldots G_n$ conjuntamente e, então, satisfazer todos subgoals $G_1, G_2, \ldots, G_n$ significa satisfazer o supergoal G (GUIZZARDI; FRANCH; GUIZZARDI, 2012).

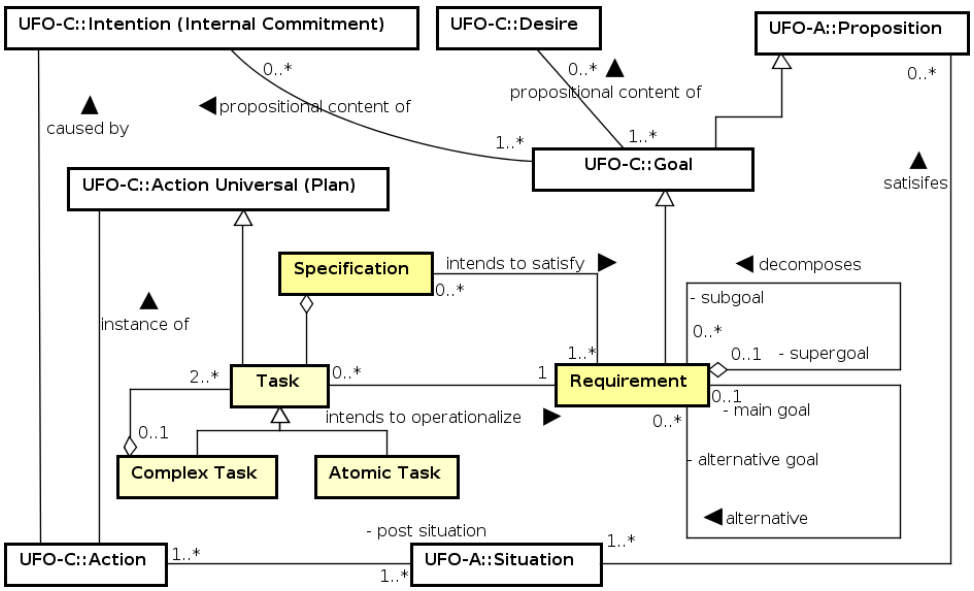

Figura 18 – Fragmento de GORO focado nas relações

A relação de decomposição normalmente detalha um objetivo, não o expandindo, mas ampliando suas partes divididas. Na Figura 19, do lado esquerdo (*decompose*), o supergoal G é decomposto especificando suas subpartes G_1 e G_2 que talvez não estivessem suficientemente explícitas inicialmente. As nuances do goal principal são então descritas, expandindo o entendimento de cada fragmento do goal e mostrando como ele pode ser satisfeito satisfazendo suas subpartes, tornando-o menos abstrato. A decomposição pode também ser utilizada para separar as partes de um objetivo para facilitar a elaboração de estratégias para cumpri-lo. Assim, são especificadas estratégias específicas para cada subobjetivo modelado.

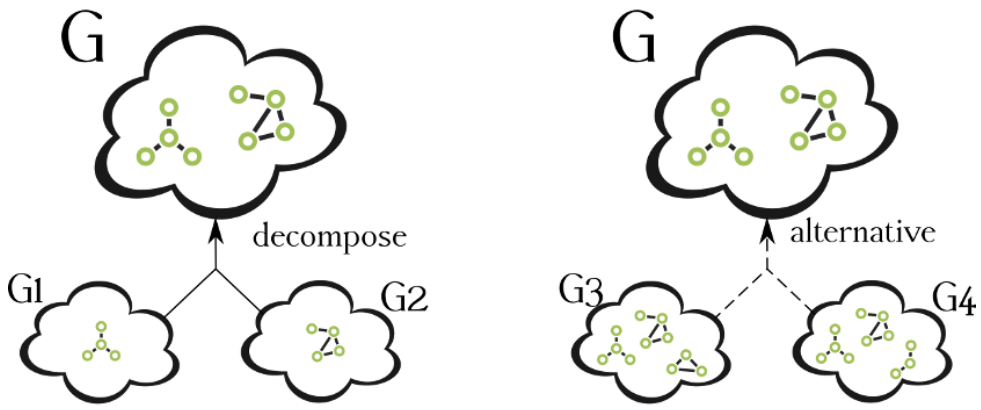

Figura 19 – Conjunto de situações de um objetivo decomposto (à esquerda - *decompose*) e de objetivos alternativos (à direita - *alternative*)

Na relação de alternative (também na Figura 18), o goal não é refinado em suas sub-partes, mas em objetivos alternativos (alternative goals) em que uma situação que satisfaça (satisfies) cada um deles (separadamente), também satisfaz o objetivo principal (main goal). Assim, considerando S como satisfazendo $S(G) \iff S(G_1) \vee S(G_2) \vee \ldots \vee S(G_n)$ (GUIZZARDI et al., 2013). Ou seja, satisfazer G_1 ($S(G_1)$), *ou* satisfazer G_2 ($S(G_2)$), ..., *ou* satisfazer G_n ($S(G_n)$), significa satisfazer G ($S(G)$).

Relações de alternativa especificam variadas formas de se alcançar um objetivo. Os objetivos alternativos descrevem situações que também comportam a situação do objetivo principal. Na Figura 19, por exemplo, o objetivo G se refere a um conjunto de situações na realidade. No lado direito da figura, com a relação de alternativa (*alternative*), os objetivos alternativos G_3 e G_4 descrevem situações na realidade que incluem a situação referente ao objetivo G mas que não são exatamente iguais. Se o conjunto de situações a que se refere um objetivo for exatamente igual ao conjunto de situações a que se refere outro objetivo, então esses dois objetivos são o mesmo.

Então, G_3 e/ou G_4 não podem descrever exatamente o mesmo conjunto de situações que G, pois, neste caso, seriam o mesmo que G. Por outro lado, nem G_3 e nem G_4 poderiam descrever uma situação menor, que não englobe todas situações de G pois, neste caso, não seriam alternativas viáveis para G. Um objetivo alternativo detalha mais a situação alternativa para satisfazer o principal. Isto por que G_3 e G_4, então, descrevem um conjunto de situações que engloba o conjunto de situações de G, mas que são maiores do que este último. G_1 e G_2, então, detalham o objetivo principal através de objetivos alternativos que se referem a situações mais específicas (inserindo mais elementos à situação). Relações de alternativa diminuem o nível de abstração de um objetivo provendo diferentes formas (*como*) de satisfazê-lo.

Cabe aos envolvidos, requirement stakeholders, determinarem quais as alternativas

de um objetivo ou de que forma ele será decomposto. Esta é uma escolha que depende da intencionalidade dos envolvidos. Deve ser observado, porém, se os objetivos decompostos ou alternativos referem-se a situações que respeitam as condições de satisfabilidade do respectivo objetivo principal, como explicado previamente.

Operacionalização

Requirement stakeholders talvez tenham a vontade de satisfazer um requisito agindo de uma forma específica ou, até mesmo, modeladores podem querer descrever no modelo de objetivos *como* satisfazer algum requisito, saindo do domínio do problema e explorando o domínio da solução.

As tarefas que compõem uma especificação são descritas com a intenção de operacionalizar um requisito. Assim, a relação intends to operationalize conecta uma task a um requirement, expressando de que maneira a desejada situação na realidade deve ser alcançada ou, em outras palavras, como satisfazer o requisito. Um UFO::plan é simplesmente uma maneira específica de fazer algo. No contexto GORE, task revela a intenção do requirement stakeholder de perseguir um objetivo como requisito de uma forma específica, ou seja, a intenção de que aquele plano seja realizado para que se alcance tal objetivo. Isto significa que uma ou mais execuções desta UFO::action (como instâncias de task) produzem, supostamente, uma pós-situação que satisfaz o conteúdo proposicional do objetivo (GUIZZARDI; FRANCH; GUIZZARDI, 2012).

Goal como conteúdo proposicional de intenções são sempre associados com tasks como meio para atingir o objetivo já que existe um compromisso interno do agente de perseguir aquele goal. Porém, tasks são modeladas com a *intenção* de satisfazer o objetivo — não se pode afirmar convictamente que a execução dela o satisfará. Podem existir questões que influenciam na satisfação do requisito que não tenham sido pensadas ou consideradas durante a modelagem.

Tasks podem ser atômicas ou complexas. Complex tasks são decompostas em duas ou mais tarefas (atômicas ou complexas) enquanto atômicas não são decompostas.

A relação intends to operationalize, em essência, é utilizada para descrever a forma de satisfazer um objetivo. Assim, representa uma grande mudança no nível de abstração explicando explicita e diretamente *como* satisfazer um goal.

3.3 Comparação com Engenharia de Requisitos de *Software*

Wang et al. (2014) fazem uma importante contribuição esclarecendo a natureza de *software*, numa perspectiva de Engenharia de Requisitos, explicando a natureza de *programa*, *sistema de software* e *produto de software* e propondo uma ontologia de artefato

de *software* (OSA) apresentada na Figura 20 e explicada a seguir.

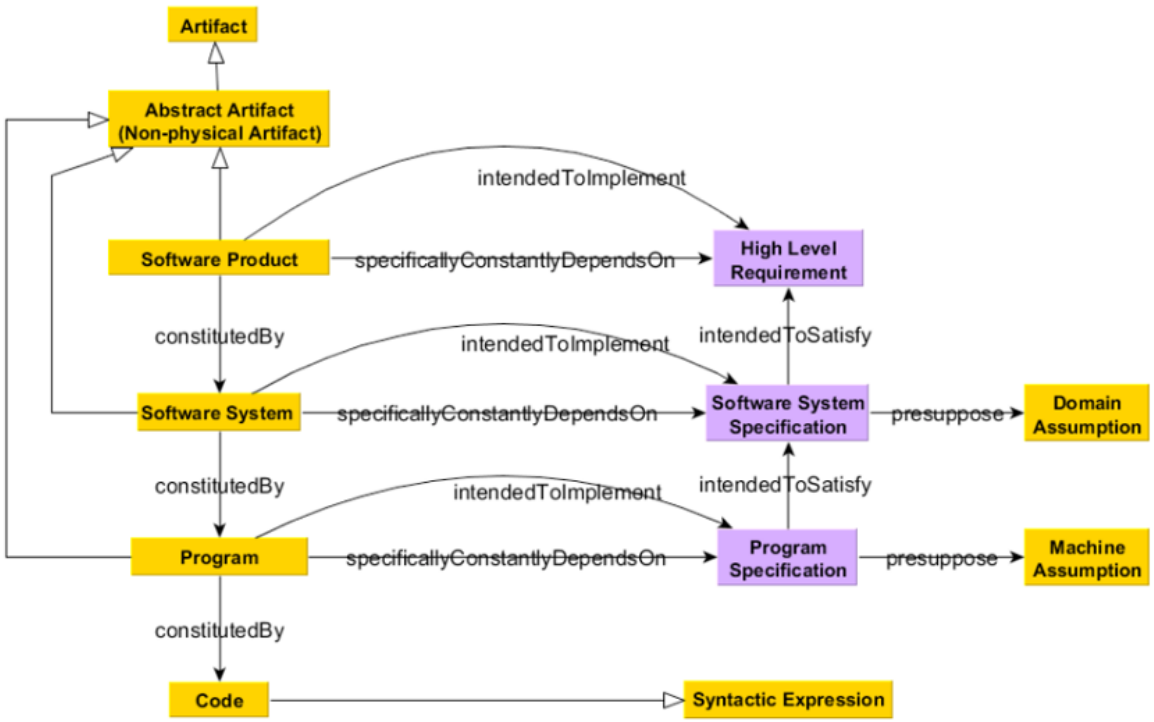

Figura 20 – Ontologia de *Software* (WANG et al., 2014)

Programa é constituído por código-fonte, uma sequência de instruções lógicas em uma linguagem de programação, e criado com o propósito de produzir um comportamento interno na máquina. O programa tem intenção de implementar, de se comportar conforme especificado em um *program specification* (que leva em consideração *machine assumptions*). Um programa tem sua identidade diretamente relacionada com sua funcionalidade. Dois programas que geram o mesmo comportamento interno na máquina, que implementam o mesmo algoritmo mesmo com diferentes códigos-fonte, são considerados como idênticos. O que não é válido para *código* já que estes podem ser considerados os mesmos somente se possuírem exatamente a mesma estrutura sintática.

Software Systems são constituídos de programas e controlam o comportamento externo da máquina. Visam implementar *software system specifications* que descrevem o comportamento esperado na interface da máquina, levando em conta *domain assumptions*. Por outro lado, *Software Products* são constituídos por *Software Systems* e dizem respeito aos efeitos do comportamento da interface da máquina no ambiente externo. São projetados com intenção de implementar *High Level Requirements* — comportamentos desejados no ambiente independentes da máquina — e dizem respeito ao ambiente, excluindo a máquina.

Buscando aprofundar e esclarecer conceitos como *sistema de software*, *produto de software*, *programa* e *código*, a perspectiva de engenharia de requisitos adotada em (WANG et al., 2014) é parcial, focando na construção de um *software* como solução — naturalmente

pelo escopo da proposta.

Jackson (2001) destaca a importância de separar o domínio do problema do domínio da solução. Ainda, *goal-modeling (requirements)* podem ser o ponto de partida para uma série de possíveis análises e aplicações que não envolvem somente *software*. Uma série de outras aplicações para modelos de requisitos baseados em objetivos podem ser listadas, como em modelagem de organizações e processos de negócio (LÓPEZ; FRANCH; MARCO, 2011; CARES, 2012).

Os próprios conceitos de *delegação* em $i^\star$ (delegando um objetivo ou tarefa para um agente humano) e *expectation* em KAOS (quando um requisito é de responsabilidade de um agente humano) revelam a possibilidade de que objetivos sejam satisfeitos não necessariamente por *software*. Em um dos trabalhos iniciais de $i^\star$, também, uma das aplicações levantadas é voltada para suporte em reengenharia de processos de negócio (YU, 1995).

Trabalhos disponíveis na literatura usam Engenharia de Requisitos ou GORE, especificamente, associado a processos de negócio (FRANCH, 2009 apud CARES et al., 2010) (KOLIADIS; GHOSE, 2006; LAPOUCHNIAN; YU; MYLOPOULOS, 2007; DECREUS; POELS, 2011), e a requisitos de serviço, ajudando a entender a natureza intencional dos envolvidos (LIU et al., 2006). Ainda, modelos orientados a objetivos podem dar suporte na construção de esquemas de banco de dados (JIANG et al., 2006) ou até auxiliar na construção de modelos de arquitetura empresarial (QUARTEL et al., 2009).

Assim, GORO não prevê, *a priori*, um *sistema de software* para satisfação dos objetivos modelados. GORO busca uma perspectiva bastante inclinada à fase de requisitos iniciais, ou seja, focada no entendimento e caracterização do problema, lidando com questões intencionais, objetivos dos envolvidos e explorando o domínio através de *assumptions*. Complementa-se, então, a contribuição e os esclarecimentos feitos por Wang et al. (2014), dando um passo atrás no processo de Engenharia de Requisitos numa visão mais voltada a requisitos iniciais. Assim, é proposta uma revisão da conceituação de Wang et al. (2014) sob uma perspectiva generalista, separando o escopo do problema do escopo da solução.

A Figura 21 apresenta essa proposta. *Goals as requirements* trazem à tona o domínio do problema descrevendo o estado/comportamento desejado do mundo. Uma *solução* para determinado problema visa controlar o ambiente de forma a atingir o estado intencionado, de forma a satisfazer os objetivos como requisitos, fazendo o mundo se comportar da maneira esperada.

Systems são entidades interagindo de forma interdependente formando uma unidade com um propósito. Expressam-se influenciando e sendo influenciados pelo ambiente ao seu redor através de sua interface, seu contato com o mundo.

Specifications descrevem como satisfazer os requisitos levando em consideração

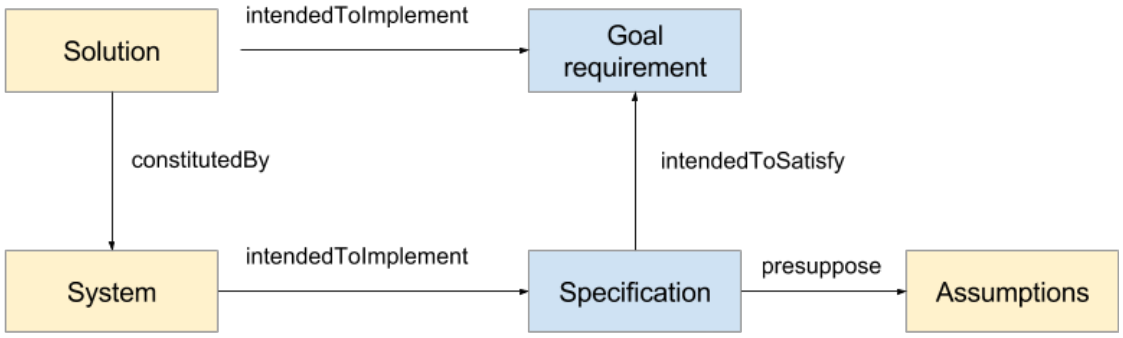

Figura 21 – Relações entre *solução*, *requisitos*, *sistemas* e *especificação*

as (supostas) propriedades do domínio (*assumptions*). Falam sobre o comportamento esperado na interface do sistema com o ambiente para se obter o estado desejado do mundo, satisfazendo os objetivos. Assim, sistemas buscam implementar as especificações.

Sistemas são constituídos de entidades capazes de promover determinados comportamentos ou estados internos para gerar o comportamento ou estado na interface externa — na interface do sistema com o mundo. Na perspectiva de Wang et al. (2014), neste ponto tem-se o sistema constituído de *programa* (implementando uma *especificação de programa*). No entanto, numa perspectiva mais ampla, outras possibilidades que não *programa* podem compor o sistema como mostrado na Tabela 1.

Composição do Sistema	**Especificação Particular**
Programa	Especificação de Programa
Estrutura Organizacional	Especificação de Estrutura Organizacional
Máquina Mecânica	Modelo do Mecanismo e Especificação de Tipo de Material
Objeto Físico	Modelo de Design do Objeto (Modelo 3D)
Ação Humana Orquestrada	Modelo de Processo de Negócio

Tabela 1 – Exemplos de composição do sistema e as especificações específicas relacionadas.

A Figura 22 detalha a Figura 21 com algumas possíveis composições para o sistema. Por exemplo, uma determinada solução para um problema pode envolver mudança na disposição dos móveis em uma sala de aula: para promover a interação entre alunos, pode-se organizar as carteiras da sala formando grupos (especificado em um modelo de arquitetura). Outro caso pode envolver a necessidade de especificar um processo (modelo de processo de negócio) a ser realizado por seres humanos.

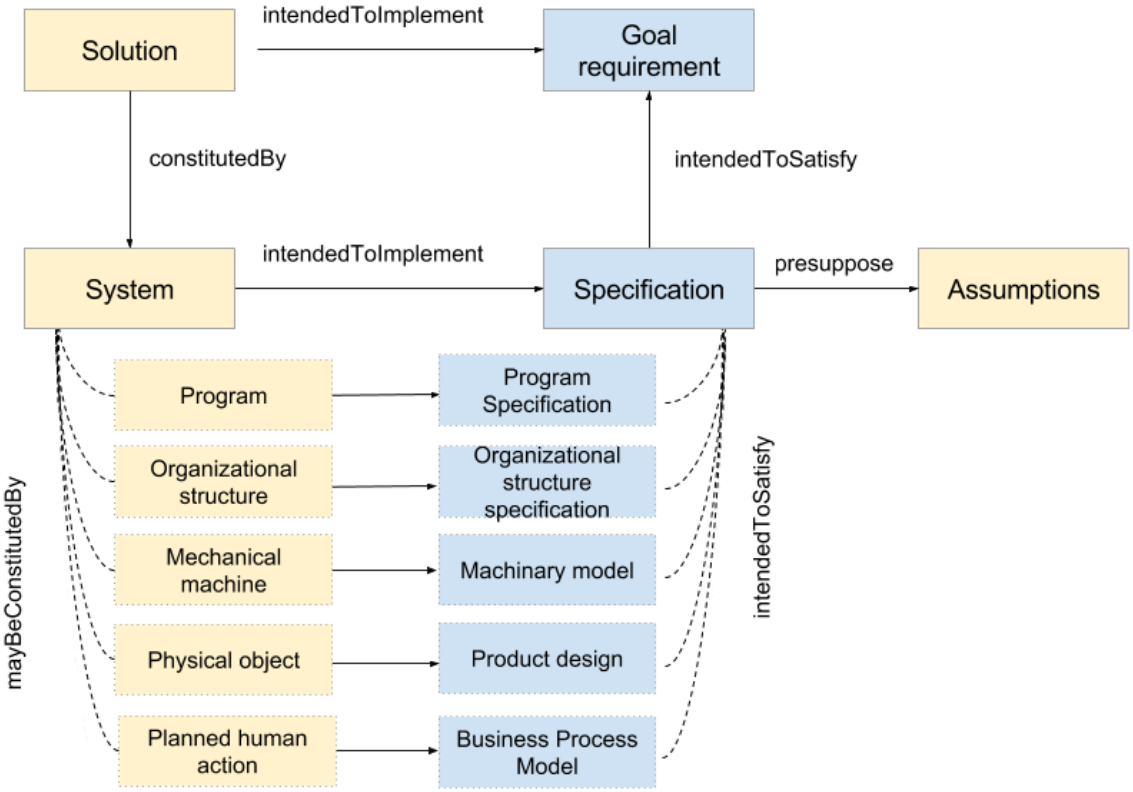

Figura 22 – Relações entre *solução*, *requisitos*, *sistemas* e *especificação* com exemplos de possíveis composições do sistema

3.4 Conclusões do Capítulo

Na Seção 3.1 foi apresentado o processo de construção da GORO pela aplicação do método SABiO. Primeiramente foram levantados os requisitos funcionais na forma de nove questões de competência. Os requisitos funcionais referem-se ao conhecimento, conteúdo, a ser capturado pela ontologia pretendida. A verificação de tais requisitos será feita no capítulo seguinte. Os requisitos não-funcionais, aqueles que se referem a qualidade e características gerais da ontologia (não diretamente relacionados ao conteúdo da mesma), foram também elicitados e a satisfação dos mesmos ocorreu da seguinte forma:

NR1: ser construída utilizando a caracterização do domínio absorvida de diferentes abordagens GORE: para tal, foram utilizados três diferentes abordagens para caracterização do domínio (*i**, KAOS e Techne);

NFR2: ser fundamentada em uma ontologia de fundamentação conhecida: GORO é embasada em UFO (*Unified Foundational Ontology*);

NFR3: reutilizar ontologias já existentes de temas relacionados ao domínio: são utilizadas conceituações disponíveis na literatura sobre objetivos e requisitos (GUIZZARDI et al., 2014), suposições (WANG et al., 2016) e outro trabalho que explora a fundo o conceito de requisito (WANG et al., 2014).

A utilização de três diferentes abordagens para caracterização do domínio implica em um maior entendimento do mesmo dada a diversidade das visões das diferentes abordagens sobre o mesmo assunto. A fundamentação em UFO permite melhor embasamento semântico dos conceitos capturados pela ontologia. Também, são conhecidos outros trabalhos com conceituações de fragmentos do domínio GORE e que, assim, são utilizados na construção da GORO.

A Seção 3.2, então, apresenta a GORO e explica os elementos da ontologia, seu embasamento em UFO e em outras conceituações importantes. Por fim, a Seção 3.3 apresenta o posicionamento da GORO como uma ontologia de requisitos independente de solução — em harmonia com o trabalho de Wang et al. (2014).

4 Avaliação de GORO

GORO foi avaliada usando técnicas de verificação e validação, como proposto por SABiO. Para verificação, é examinado se a ontologia é capaz de responder as questões de competências (CQs) levantadas na primeira etapa do método SABiO como parte da elicitação de requisitos da ontologia (Seção 3.1). A Tabela 2 ilustra o resultado desta atividade, respondendo as questões de competência elicitadas.

Além da verificação da ontologia, realizamos ainda outras duas atividades de avaliação. A Seção 4.1 apresenta uma análise embasada em GORO dos termos e conceituação adotada por diferentes abordagens GORE conhecidas disponíveis na literatura. Adiante, a Seção 4.2 traz uma interpretação, também baseada em GORO, de um modelo de objetivos de um trabalho na literatura.

4.1 Análise das abordagens GORE

A seguir é apresentada uma análise das abordagens GORE selecionadas, utilizando como referência a conceituação definida pela GORO. É utilizado fonte **San Serif** em negrito para elementos de GORO e fonte `Teletype` para elementos das abordagens referenciadas.

Stakeholder

KAOS e Techne não explicitam no modelo o **stakeholder** ao qual os estados mentais, **mental moment**, são inerentes. Sendo a Engenharia de Requisitos sobretudo uma atividade social, é observada a importância de representação desses agentes. Em GORE, objetivos não são somente o ponto de partida para se chegar aos requisitos. Objetivos são elementos que se aproximam da intenção dos agentes, de seus estados mentais intencionais, e podem ser explorados ainda no alto nível de abstração. Revelar as partes interessadas em cada objetivo pode ser útil no entendimento e comunicação sobre as intenções que o modelo busca capturar. Representar os **stakeholders** impacta diretamente em algumas das vantagens de GORE trazidas na Seção 2.3: visão holística, motivação dos requisitos e resolução de conflitos.

Em $i^\star$ o constructor `actor` pode ser especializado em agente (`agent`) ou papel (`role`). `Atores` são entidades autônomas ativas que buscam atingir seus objetivos. Já `role` é uma caracterização abstrata do comportamento social de uma categoria de `atores` dentro de um contexto específico como um estudante. Por outro lado, `agentes` são `atores` com manifestação concreta, como um indivíduo humano ou uma organização. Em $i^\star$, todos esses três elementos podem ter objetivos associados (DALPIAZ; FRANCH; HORKOFF,

CQ	Descrição	Conceitos and *Relações*
CQ1	O que é um objetivo?	Um goal é um *subtipo* de proposition e o conteúdo proposicional (*propositional content*) de uma intenção (intention) ou desejo (desire).
CQ2	O que é um requisito?	Um requisito é um *subtipo* de goal que é o *conteúdo proposicional* de um mental moment do stakeholder do qual o requisito é obitido.
CQ3	De onde vem os objetivos e requisitos?	Requisito é um *subtipo* de goal que, por sua vez, é conteúdo proposicional de intenção ou desejo. Intenções e desejos (intentions e desires) são *subtipos* de mental moments que são, por sua vez, *inerentes* a um agent.
CQ4	Como objetivos se relacionam entre si?	Uma *relação de decomposição* relaciona um objetivo como supergoal com outros objetivos como subgoals. Uma *relação de alternativa* relaciona objetivos como main goals a outros objetivos como alternative goals.
CQ5	Como objetivos podem ser satisfeitos?	A relação *intends to operationalize* é uma relação entre objetivos e tasks (*subtipos* de planos); uma action é uma *instância* de Plano que produz uma situation como post-situation, que *satisfaz* o goal. Objetivos também podem ser satisfeitos via *relações de decomposição* (a satisfação de todos subgoals implica na satisfação do supergoal) ou *relação de alternativa* (satisfazer qualquer alternative goal signfica satisfazer o main goal), conforme CQ4.
CQ6	O que é uma especificação?	Uma especificação tem *intenção de satisfazer* um ou mais requisitos de um sistema intencionado. Requisitos, por sua vez, são *subtipos* de objetivos.
CQ7	O que é uma suposição no contexto de GORE?	Uma suposição é um *subtipo* de proposição que é o *conteúdo proposicional* de uma crença, que é um mental moment de um stakeholder.
CQ8	Que tipos de requisitos existem?	Os requisitos podem ser funcionais (functional requirements) ou não-funcionais (non-functional requirements).
CQ9	Como tarefas se relacionam com requisitos?	Tarefas compõem a especificação criada afim de satisfazer os requisitos. As tarefas são, então, planejadas a fim de operacionalizar (inteds to operationalize) um requisito específico.

Tabela 2 – Verificação: respostas às questões de competências elicitadas para construção da GORO.

2016).

Em GORO, o **stakeholder** é a entidade capaz de ter estados mentais (**mental moments**) como **intenção** e **desejo**. De intenções e desejos é possível extrair o **conteúdo proposicional** e temos, assim, os objetivos. Entidades abstratas categóricas (`role` em $i^\star$)

e alguns tipos de entidades representados por `agent` como organizações e departamentos, também em $i^\star$, não tem um estado mental inerente a eles. Uma organização não tem, intrinsecamente, intenções, objetivos. Um papel, estudante ou programador, não têm um estado mental inerente. As intenções, desejos, os estados mentais, estão, sim, inerentes às pessoas que desempenham esses papéis ou compõem tal organização, por exemplo. Assim, GORO não entende que `roles` ou `agents` que representem organizações sociais em $i^\star$ tenham **intentional moments** e, assim, objetivos associados. O constructor **stakeholder** em GORO é relacionado somente com o `agent` em $i^\star$ que representa entidades humanas.

Goals

KAOS explicitamente diferencia objetivos como comportamentais (`behavioral-goals`) ou `softgoals` e funcionais/não-funcionais (LAMSWEERDE, 2009). Entretanto, o termo comportamental expressando o oposto de `softgoal` soa inadequado já que pode ser associado a propriedades funcionais. Apesar de KAOS fazer a distinção de `softgoal` e `behaviorial goals` em poucos — e mais recentes — trabalhos, na proposta inicial, quando não havia ainda sido explorada a ideia de *softgoals* (posteriormente introduzida na linguagem), `goals` já eram vistos com uma interpretação de comportamento. Dardenne, Lamsweerde e Fickas (1993) utilizam expressões como *'implementar' objetivos* no texto e, ainda, descrevem diferentes *patterns* de objetivos, todos eles descritos pela forma como influenciam o 'comportamento': alguns gerando comportamento (*achieve* e *cease*), outros restringindo comportamento (*maintain* e *avoid*) e outro comparando comportamentos afim de maximizar ou minimizar uma função, por exemplo (*optimize*). Desta forma, mesmo não sendo explícito desde o início, destaca-se a nuance 'comportamental' de objetivos em KAOS. Ainda, em (LAMSWEERDE, 2003) diz-se que `softgoals` prescrevem comportamentos preferidos, sendo usados para selecionar, qualitativamente, alternativas preferenciais. Lamsweerde (2009) diz que diferentemente de um `behavioral goals`, é difícil dizer exatamente qual comportamento específico de um sistema satisfaz um `softgoal`.

De fato, é difícil entender a diferença entre `behavioral goals` e `softgoals` e interpretar os conceitos a luz de GORO. Como apresentado na Seção 2.5.1, UFO utiliza a conceituação de Mumford (2003) de *disposition* e define função como a disposição de uma entidade de expressar determinado comportamento em uma situação específica (GUIZZARDI et al., 2013). Assim, diferenciar objetivo comportamental como oposto a *softgoal* não é uma escolha adequada. Assim, `behavioral goals` e `softgoals` talvez possam ser interpretadas em GORO como **hardgoals** e **softgoals** mas como **requisitos funcionais**.

Techne conceitua `goals`, `softgoals` e restrições de qualidade (`qualityconstraints`). Um `goal` é definido como uma condição *funcional* verificável. Em alguns trabalhos é definido como condução funcional, binária e verificável (JURETA et al., 2010). `Quality constraints` se referem a desejos relacionados a características não-binárias mensurá-

veis no sistema intencionado. `Softgoals` são equivalentes a `quality-constraints` mas se referindo a espaços de qualidade mal definidos. Não é claro, porém, o que *binário* e *não-binário* significam (infere-se que se refiram, respectivamente, ao resultado ser *sim* ou *não* ou possuir níveis de gradação). Cabe, ainda, uma observação quanto à caracterização de `goal` como "condição funcional" — enquanto em GORO, um **objetivo** por se referir a um **requisito funcional** ou **requisito não-funcional**. Uma intenção que se refere a qualidade (não-funcional) não deixa de ser um objetivo. Assim, argumenta-se que `quality constraints` em Techne, também são objetivos. `Softgoals`, além de se referirem a qualidades, ainda fazem uma referência a uma região vaga de qualidade. É, então, um constructo com sobrecarga semântica.

Em suas versões iniciais $i^\star$ definia `softgoal` assim como adotado por GORO: um estado desejado cujos critérios de satisfação não são bem definidos (YU, 1995). Em sua nova versão, iStar (DALPIAZ; FRANCH; HORKOFF, 2016) distingue somente `goals` e `qualities`, sendo o primeiro equivalente ao **hardgoal** em GORO e, o segundo, similar ao **non-functional requirement**. Como defendido explicitamente na nova versão da abordagem, esta simplificação é feita para promover a adoção da linguagem.

Todas essas diferentes conceituações das diferentes abordagens revelam alguma confusão no uso desses conceitos. Ainda, tais conceitos não são formalmente definidos nas abordagens. Como visto na Seção 3.2, GORO adota a conceituação dada por uma interpretação ontológica bem fundamentada feita por Guizzardi et al. (2014) explorada na Seção 2.6, que clareia esta discussão e define **hardgoal** e **softgoal** como conceitos ortogonais a **requisito funcional** e **requisito não-funcional**.

Techne explora o que é chamado de "modo psicológico" dos *stakeholders*. Assim, o desejo de um *stakeholder* torna-se instância de `goal` em modelos Techne enquanto sua intenção de agir de um modo específico se torna instância de `task` (JURETA et al., 2010).

Em GORO, o elemento **goal** não é uma instância do estado mental em si. Um estado mental é um **mental moment** inerente ao agente, não pode ser completamente representado em palavras por que é o que o agente 'sente'. GORO esclarece que **goal** se refere ao conteúdo de um **mental moment**, o que aquele **mental moment** expressa. Ainda, a intenção de alguém em direção a um determinado estado das coisas expressa seu desejo neste estado e o compromisso em persegui-lo. Em GORO, este compromisso pode ser expresso com um plano (**task**) para atingir tal objetivo, tal estado das coisas na realidade.

Uma `task` em Techne é interpretada como uma **task** em GORO já que esta não é, de fato, a ação ou o evento em si, mas um plano definindo uma maneira específica de agir.

Refinamentos

KAOS explora o refinamento de objetivos estratégicos, refinando-os em grãos menores em direção aos de mais baixo nível de abstração, mais realizáveis, até que eles originem requisitos (DARDENNE; LAMSWEERDE; FICKAS, 1993). Assim, um requisito em KAOS é um `goal` sob a responsabilidade de um único agente do sistema intencionado. Se o `goal` realizável for de responsabilidade de um único agente humano, então ele não é um *requirement*, mas sim uma expectativa (`expectation`). GORO argumenta que requisitos não são intrinsecamente apenas objetivos refinados, a ponto de serem satisfazíveis por um único agente. De acordo com GORO, **requirement** fala sobre o ambiente — incluindo o *software* mas não somente ele — e, assim, um requisito não será necessariamente satisfeito por *software*. Requisitos em GORO estão no escopo do problema, sua natureza não muda dependendo de quem tem a responsabilidade de satisfazê-lo.

Desta forma, a responsabilidade atribuída a agentes (de *software* ou humanos) em direção a satisfação de um objetivo não altera a identidade do mesmo e, assim, não são utilizados em GORO diferentes constructos para representar tal diferenciação como KAOS sugere. Ademais, a delegação de responsabilidade em satisfazer os objetivos não é tradicionalmente realizada durante a fase de Engenharia de Requisitos, mas sim durante as fases posteriores, mais relacionadas ao projeto de arquitetura.

$i^\star$ também não faz tal distinção, mas a ideia de expectativa pode ser observada quando o objetivo de um ator é delegado para outro. Isto não modifica a natureza do objetivo, mas expressa a relação de dependência entre esses dois agentes e uma expectativa de quem delega, para quem é delegado a satisfazer tal objetivo. Tal expectativa também expressa a suposição de que ele será satisfeito. Em GORO esta relação está associada ao conceito de **assumption-nedeed**. Mas a relação de delegação não é tratada nesta versão da ontologia e será assunto de trabalhos futuros.

KAOS adota os termos `AND/OR-Refinement` para expressar associações entre `goals`, permitindo diminuir o nível de abstração de objetivos estratégicos em busca de sub-objetivos mais realizáveis (LAMSWEERDE, 2003). Entretanto, o termo *refinamento* é genérico e tem diferentes significados quando utilizado com o prefixo *AND* ou *OR*. Quando com o prefixo *AND*, o objetivo pai será satisfeito com a satisfação de todos os sub-objetivos do refinamento. Com *OR*, são expressas múltiplas alternativas para satisfação do objetivo pai. As alternativas podem ser conjuntos de refinamentos do tipo *AND*. Relacionando com GORO, `AND-refinement` é o mesmo que **decompose** e `OR-Refinement` é o mesmo que **alternative link**.

KAOS especifica como um objetivo pode ser satisfeito através de operações (`operations`), que podem se referir a procedimentos no ambiente (fora dos limites do sistema) ou no próprio software. O conceito de `operation` proposto por KAOS é, de fato,

um plano a ser realizado para atingir um objetivo específico. Assim sendo, a conexão entre `goal` e `operation` em KAOS é interpretada em GORO como **intends to operationalize**.

Em Techne, o constructo `inference` é utilizado para relacionar um conjunto de instâncias de `goal`, por exemplo, a uma outra instância de `goal` no modelo, indicando que satisfazendo a combinação dos primeiros significa satisfação do último. Isto é interpretado em GORO como o link **decomposes**. O nó de inferência em Techne também pode ser usado para relacionar um conjunto de `tasks` a um `goal`: neste caso representa a relação GORO **intends to operationalize**. Assim, a relação `inference` em Techne abrange tanto relação `AND-refinament` em KAOS como a de `means-end` de $i^\star$- analisada adiante.

Em Techne existe ainda a relação de `preference` que é relacionada com a relação de alternativa em GORO. Na verdade, `preference` em Techne expressa não somente o conjunto de alternativas para satisfazer um objetivo, mas também a preferência entre todas essas alternativas. Neste contexto, é valido argumentar sobre a necessidade de cuidado ao utilizar um único constructo referenciando mais de um conceito.

Nas versões iniciais de $i^\star$, tais relações costumavam ser modeladas com diferentes constructos: `AND/OR-Decomposition` (como **decompose** e **alternative** em GORO) e `means-end` (GORO **intends to operationalize**) (GUIZZARDI et al., 2013). A versão mais recente de iStar (DALPIAZ; FRANCH; HORKOFF, 2016), entretanto, fundiu todas essas relações em um único constructo chamado `refinement`. A semântica do constructo é, então, dada pelos elementos relacionados a ele no modelo: quando entre objetivos, `AND-refinement` é o mesmo que GORO **decomposes**, enquanto o `OR-refinement` é o mesmo que GORO **alternative**. Se um `goal` é refinado em `tasks`, a `task` é o modo de realizar o objetivo.

Em $i^\star$, também é possível dizer que um `goal` é o caminho para um `task`. GORO não reconhece esta ideia já que um estado desejado das coisas na realidade não pode ser o meio, o caminho, para uma ação por meio de um plano. Esta relação entre objetivo e tarefa pode ser relacionada com outros conceitos GORE não discutidas neste trabalho, como a ideia de contribuição (GUIZZARDI et al., 2013).

Como explicitamente explicado na documentação de iStar 2.0 (DALPIAZ; FRANCH; HORKOFF, 2016), a escolha de um termo genérico se referenciando a diferentes tipos de relação é feita para facilitar e, então, promover a adoção da linguagem na comunidade. Esta simplificação pode de fato ser útil para tal situação. Porém, demonstra que o objetivo não é uma conceituação precisa do domínio GORE. Assim, GORO pode ser utilizada como fundamentação, conceituação bem embasada, uma ontologia de domínio, para fundamentar todas essas abordagens.

4.2 Análise de um modelo GORE

Nesta seção é analisado o modelo de objetivos presente no trabalho de Souza (2012), que representa um sistema de despacho de ambulância, usando GORO como embasamento. A Figura 23 apresenta o modelo de objetivos para o sistema de despacho de ambulâncias. O modelo da figura utiliza a sintaxe abstrata de Techne com a sintaxe concreta de $i^{\star}$.

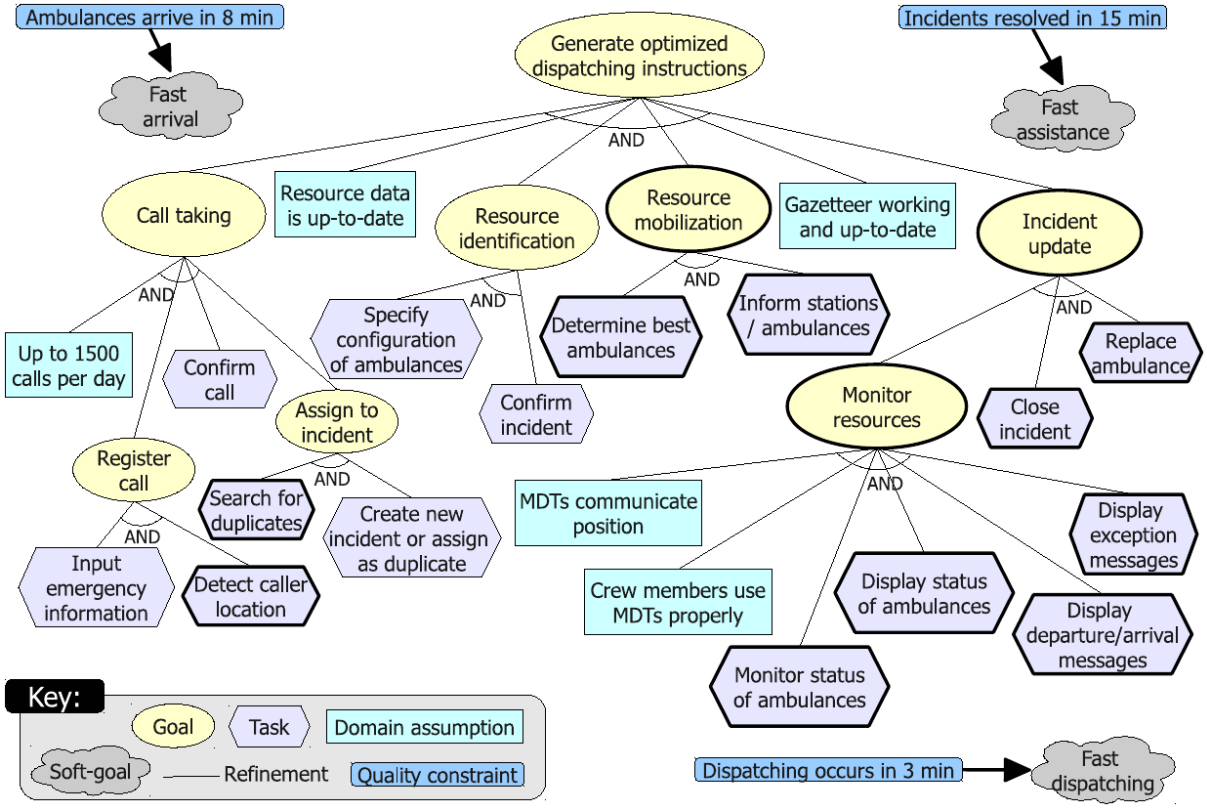

Figura 23 – Modelo de objetivos de um sistema de despacho de ambulâncias (SOUZA, 2012)

O objetivo principal do modelo "*Gerar instruções de despacho (de ambulâncias) otimizados*" é refinado em quatro subobjetivos e dois *domain assumptions*. Objetivos expressam estados intencionados do mundo, situações que não estão estabelecidas na realidade mas, que há comprometimento para alcançá-las. *Domain assumptions*, por outro lado, falam sobre suposições, crenças, de que determinado estado do mundo existe. Assim, é incompatível refinar um objetivo em uma suposição de domínio, uma vez que não há intencionalidade em *domain assumptions*.

Considerando o refinamento do objetivo principal somente para os quatro subobjetivos, observa-se que é utilizado o refinamento do tipo AND, denotando que a satisfação dos 4 subobjetivos referenciados satisfaz o objetivo principal. Em GORO, tal relação é interpretada como uma relação de **decomposição** em que o objetivo principal é decomposto em subpartes.

No modelo também podem ser vistos objetivos sendo refinados em tarefas. Uma tarefa descreve um plano de ação para se alcançar determinada situação na realidade. Um objetivo refinado em tarefa denota uma forma de operacionalizar tal objetivo (ou uma parte dele). Em GORO, então, este refinamento de objetivo em tarefa é interpretado como uma **operacionalização**.

No modelo há três *softgoals* com restrições de qualidade acoplados: *chegada rápida* (com a restrição *ambulâncias chegam em 8 minutos*), *assistência rápida* (com a restrição *incidentes resolvidos em 15 minutos*) e *despacho rápido* (com a restrição *despacho ocorre em 3 minutos*).

Em GORO **softgoals** referem-se a objetivos que não foram suficientemente analisados ainda, objetivos que se referem a uma região vaga de satisfabilidade. Neste caso, é o que o modelo analisado expressa com os três *softgoals*. Porém, há restrições de qualidade associados a esses objetivos. Tais restrições fazem com que seja bem delineada a região de satisfabilidade para tais *softgoals* — a ambulância chegar *rápido* significa chegar em até 8 minutos. Tal objetivo é, em GORO, um **softgoal**, **requisito não-funcional** e com uma restrição de qualidade (**quality constraint**) bem definida (em até 8 minutos) associada.

Apesar de objetivos descreverem um estado do mundo intencionado, é bastante comum encontrar na literatura a descrição de objetivos com verbos o que, a princípio, se refere a uma atividade, tarefa. O mesmo também é observado no modelo aqui apresentado como exemplo, em que o objetivo principal "*gerar instruções de despacho otimizadas*" descreve uma ação e, assim, uma tarefa. Porém, tratando-se de um objetivo, a intenção provavelmente é a de se referir a um estado do mundo em que *há geração de instruções de despacho de forma otimizada*.

4.3 Conclusões do Capítulo

No início deste capítulo a Tabela 2 apresenta as questões de competências levantadas no processo de elicitação de requisitos e como a GORO responde essas questões.

Na Seção 4.1, é feita uma análise embasada em GORO dos termos e conceituações adotadas por diferentes abordagens GORE. Nesta Seção os conceitos apresentados pelas diferentes abordagens são explicados à luz de GORO primeiramente para relacionar os conceitos das linguagens com os elementos de uma ontologia bem fundamentada sobre o domínio (GORO) mas também para revelar problemas semânticos que tais abordagens porventura apresentam. Por fim, na Seção 4.2 um modelo de objetivos real é analisado por intermédio da GORO.

5 Considerações Finais

A Engenharia de Requisitos Baseados em Objetivos tem se mostrado uma área de grande importância dentro da Engenharia de Requisitos. Porém, mesmo a grande quantidade de diferentes abordagens trazem uma conceituação frágil do domínio. GORO, então, é proposta como uma ontologia de referência de domínio com o intuito de prover uma conceituação formal sobre o domínio de requisitos baseados em objetivos.

Construída com uso do método SABiO, GORO é fundamentada em UFO e reutiliza ontologias que tratam de fragmentos do domínio: a ontologia de requisitos não-funcionais (representada em GORO com o prefixo NFR) e a ontologia de suposições (representada com o prefixo ASMP). O conhecimento utilizado para construção da ontologia foi extraído de diversos trabalhos disponíveis na literatura que tratam de GORE e domínios relacionados e também de extensa discussão com engenheiros de ontologias e especialistas no domínio no âmbito do grupo de pesquisa em que este autor está inserido. Desta forma, a base consensual sobre o que o domínio GORE trata foi construída e capturada pela ontologia proposta neste trabalho. Assim, GORO fornece uma descrição formal, ontologicamente bem fundamentada, do domínio de Engenharia de Requisitos Baseados em Objetivos, servindo de referência semântica do domínio, uma vez que provê um aprofundamento na natureza conceitual das entidades e suas relações.

Com a ontologia construída, a mesma foi utilizada como referência para análise semântica de três abordagens GORE populares na literatura: KAOS, $i^\star$ e Techne. Este estudo demonstrou problemas no uso de conceitos nas abordagens oriundos da frágil conceituação que tais abordagens fazem do domínio. A sobrecarga semântica de elementos e a definição vaga de conceitos possibilitam interpretação ambígua ou errônea, prejudicando o uso de tais abordagens.

GORO se mostrou eficaz como embasamento para análise do domínio GORE, esclarecendo e explicitando conceitos relacionados. Assim, serve também ao propósito de base para comunicação humana sobre Engenharia de Requisitos Baseados em Objetivos, melhorando o entendimento entre os envolvidos.

GORO também foi integrada por Ruy (2017) à SEON, uma rede de ontologias de Engenharia de Software cujo propósito é promover uma solução integrada para melhor lidar com o domínio de Engenharia de Software. O resultado pode ser visto em <http://dev.nemo.inf.ufes.br/seon/>.

O trabalho deu origem à publicação (NEGRI et al., 2017) que apresenta a GORO como ontologia do domínio GORE.

5.1 Contribuições

Dentre as principais contribuições deste trabalho pode-se destacar as listadas a seguir.

- Uma ontologia de referência do domínio de requisitos baseados em objetivos, fundamentada em UFO, que provê uma descrição aprofundada da natureza dos elementos que compõem o domínio;
- A compilação de distintos trabalhos que propõem conceituações sobre fragmentos do domínio GORE (como suposições e requisitos funcionais/não funcionais), reutilizando tais trabalhos em um proposta mais abrangente sobre o domínio;
- Análise da conceituação implícita em constructos utilizados em diferentes abordagens, dando embasamento ontológico na semântica de tais termos.

5.2 Dificuldades e Limitações

A principal dificuldade na realização deste trabalho foi relacionada justamente ao entendimento da semântica utilizada pelas diversas abordagens GORE estudadas, uma vez que as conceituações dadas por elas são fracas e muitas vezes insuficientes. Inicialmente, isto impactou na dificuldade de entendimento do domínio GORE, seu escopo, seus elementos e relações. Depois, com a ontologia GORO já construída, dificultou ainda entender os constructos utilizados nas diferentes abordagens para explicá-los à luz de GORO. Muitas vezes foi necessário buscar em diferentes trabalhos o que determinado constructo significa.

É uma limitação deste trabalho o fato de a ontologia proposta ainda não cobrir todo domínio GORE. As diferentes abordagens GORE utilizadas trazem conceitos que ainda não foram incorporados à ontologia. Ainda, o trabalho não define axiomas especificando restrições do universo de interesse, como sugerido pelo método SABiO, para formalização mais robusta da ontologia.

5.3 Perspectivas Futuras

Este trabalho abre possibilidades de outras contribuições na área, algumas delas apresentadas a seguir:

- Evoluir GORO, representando outros elementos presentes no domínio GORE que não foram tratados nesta primeira versão como o conceito de preferência, dependência entre agentes, delegação de tarefas, recursos, etc.;

- Desenvolver mapeamento entre linguagens GORE, utilizando a GORO como referência, para possibilitar interoperabilidade entre modelos descritos em diferentes abordagens;
- Expandir a análise conceitual dos elementos de outras abordagens GORE além das três feitas neste trabalho;
- Utilizar a GORO como referência semântica em práticas GORE na industria/mercado em um problema real. Assim, possibilitando o amadurecimento da ontologia pela visão de usuários reais e verificando a contribuição da ontologia em situações reais;
- Utilizar GORO para reengenharia, *redesign*, das linguagens GORE ou, ainda, como base para proposta de uma nova linguagem GORE.

Referências

AURUM, A.; WOHLIN, C. Requirements engineering: Setting the context. In: _____. *Engineering and Managing Software Requirements*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005. p. 1–15. ISBN 978-3-540-28244-0. Disponível em: <http://dx.doi.org/10.1007/3-540-28244-0_1>. Citado 2 vezes nas páginas 15 e 20.

BARESI, L.; PASQUALE, L.; SPOLETINI, P. Fuzzy goals for requirements-driven adaptation. In: *RE 2010, 18th IEEE International Requirements Engineering Conference, Sydney, New South Wales, Australia, September 27 - October 1, 2010*. [s.n.], 2010. p. 125–134. Disponível em: <https://doi.org/10.1109/RE.2010.25>. Citado na página 40.

BEECHAM, S.; HALL, T.; RAINER, A. Software process improvement problems in twelve software companies: An empirical analysis. *Empirical Software Engineering*, v. 8, n. 1, p. 7–42, Mar 2003. ISSN 1573-7616. Citado na página 15.

BOEHM, B. W.; PAPACCIO, P. N. Understanding and controlling software costs. *IEEE Trans. Softw. Eng.*, IEEE Press, Piscataway, NJ, USA, v. 14, n. 10, p. 1462–1477, 1988. ISSN 0098-5589. Disponível em: <http://dx.doi.org/10.1109/32.6191>. Citado na página 15.

BORGIDA, A.; ERNST, N.; JURETA, I. J.; LAPOUCHNIAN, A.; LIASKOS, S.; MYLOPOULOS, J. Techne: A (nother) Requirements Modeling Language. *Computer Systems Research Group. Toronto, Canada: University of Toronto*, 2009. Citado 4 vezes nas páginas 16, 23, 27 e 28.

BORST, W. N. *Construction of Engineering Ontologies for Knowledge Sharing and Reuse*. Tese (Doutorado) — University of Twente, 1997. Citado na página 29.

BRESCIANI, P.; PERINI, A.; GIORGINI, P.; GIUNCHIGLIA, F.; MYLOPOULOS, J. Tropos: An Agent-Oriented Software Development Methodology. *Autonomous Agents and Multi-Agent Systems*, Kluwer Academic Publishers, v. 8, n. 3, p. 203–236, 2004. ISSN 1387-2532. Citado na página 26.

BROOKS, F. P. No Silver Bullet Essence and Accidents of Software Engineering. *Computer*, IEEE Computer Society Press, Los Alamitos, CA, USA, v. 20, n. 4, p. 10–19, abr. 1987. ISSN 0018-9162. Disponível em: <http://dx.doi.org/10.1109/MC.1987.1663532>. Citado na página 19.

CARES, C. *From the i* Diversity to a Common Interoperability Framework*. Tese (PhD Thesis) — Universitat Politècnica de Catalunya Barcelona Tech, Barcelona, 2012. Citado 2 vezes nas páginas 43 e 56.

CARES, C.; FRANCH, X.; LÓPEZ, L.; MARCO, J. Definition and uses of the *i** metamodel. In: *Proceedings of the 4^{th} International* i* *Workshop, Hammamet, Tunisia, June 07-08, 2010*. [s.n.], 2010. p. 20–25. Disponível em: <http://ceur-ws.org/Vol-586/iStar10-paper04.pdf>. Citado 3 vezes nas páginas 16, 43 e 56.

DALPIAZ, F.; FRANCH, X.; HORKOFF, J. iStar 2.0 Language Guide. *CoRR*, 2016. Disponível em: <https://arxiv.org/abs/1605.07767>. Citado 7 vezes nas páginas 9, 26, 27, 28, 61, 63 e 65.

DARDENNE, A.; LAMSWEERDE, A. van; FICKAS, S. Goal-directed Requirements Acquisition. *Sci. Comput. Program.*, Elsevier North-Holland, Inc., Amsterdam, The Netherlands, The Netherlands, v. 20, n. 1-2, p. 3–50, abr. 1993. ISSN 0167-6423. Disponível em: <http://dx.doi.org/10.1016/0167-6423(93)90021-G>. Citado 5 vezes nas páginas 16, 23, 24, 62 e 64.

DECREUS, K.; POELS, G. A goal-oriented requirements engineering method for business processes. In: *INFORMATION SYSTEMS EVOLUTION (Lecture Notes in Business Information Processing)*. [S.l.: s.n.], 2011. v. 72, p. 29–43. ISBN 9783642177217. ISSN 1865-1348. Citado na página 56.

European Software Institute. *European user survey analysis*. [S.l.], 1996. Report USV_EUR 2.1. Citado na página 15.

FALBO, R. A. Experiences in using a method for building domain ontologies. In: *16th International Conference on Software Engineering and Knowledge Engineering (SEKE 2004), International Workshop on Ontology in Action (OIA 2004)*. Alberta, Canada: [s.n.], 2004. p. 474–477. Citado 2 vezes nas páginas 40 e 45.

FALBO, R. D. A.; BERTOLLO, G. A software process ontology as a common vocabulary about software processes. *International Journal of Business Process Integration and Management*, Inderscience Publishers, v. 4, n. 4, p. 239–250, 2009. Citado na página 30.

FALBO, R. de A. SABiO: Systematic Approach for Building Ontologies. In: *Proceedings of the 1st Joint Workshop ONTO.COM / ODISE on Ontologies in Conceptual Modeling and Information Systems Engineering co-located with 8th International Conference on Formal Ontology in Information Systems, ONTO.COM/ODISE@FOIS 2014, Rio de Janeiro, Brazil, September 21, 2014*. [s.n.], 2014. Disponível em: <http://ceur-ws.org/Vol-1301/ontocomodise2014_2.pdf>. Citado 4 vezes nas páginas 9, 30, 40 e 41.

FAYOUNI, A.; KAVAKLI, E.; LOUCOPOULOS, P. Towards a unified meta-model for goal oriented modelling. In: *Proceedings of the 2015 European, Mediterranean & Middle Eastern Conference on Information Systems*. [S.l.: s.n.], 2015. Citado 2 vezes nas páginas 16 e 43.

FRANCH, X. A method for the definition of metrics over i* models. In: *Proceedings of the 21st International Conference on Advanced Information Systems Engineering*. Berlin, Heidelberg: Springer-Verlag, 2009. (CAiSE '09), p. 201–215. ISBN 978-3-642-02143-5. Disponível em: <http://dx.doi.org/10.1007/978-3-642-02144-2_19>. Citado na página 56.

GREENSPAN, S. J. *Requirements Modeling: A Knowledge Representation Approach to Software Requirements Definition*. Tese (Doutorado), 1984. AAI0555105. Citado na página 19.

GRUBER, T. R. A translation approach to portable ontology specifications. *KNOWLEDGE ACQUISITION*, v. 5, p. 199–220, 1993. Citado na página 29.

GRÜNINGER, M.; FOX, M. S. Methodology for the design and evaluation of ontologies. In: *IJCAI95 Workshop on Basic Ontological Issues in Knowledge Sharing*. Montreal: [s.n.], 1995. Citado na página 41.

GUARINO, N. Formal ontology and information systems. IOS Press, p. 3–15, 1998. Citado 3 vezes nas páginas 9, 29 e 30.

GUARINO, N.; OBERLE, D.; STAAB, S. What is an ontology? In: *Handbook on Ontologies*. [s.n.], 2009. p. 1–17. Disponível em: <https://doi.org/10.1007/978-3-540-92673-3_0>. Citado na página 29.

GUIZZARDI, G. *Ontological Foundations for Structural Conceptual Models*. Tese (PhD Thesis) — University of Twente, The Netherlands, 2005. Citado 3 vezes nas páginas 29, 31 e 46.

GUIZZARDI, G. On ontology, ontologies, conceptualizations, modeling languages, and (meta)models. In: *Proceeding of the 2007 conference on Databases and Information Systems IV: Selected Papers from the Seventh International Baltic Conference DB&IS'2006*. Amsterdam, The Netherlands, The Netherlands: IOS Press, 2007. p. 18–39. ISBN 978-1-58603-715-4. Disponível em: <http://dl.acm.org/citation.cfm?id=1565421.1565425>. Citado 2 vezes nas páginas 29 e 30.

GUIZZARDI, G.; FALBO, R. de A.; GUIZZARDI, R. S. S. Grounding software domain ontologies in the unified foundational ontology (UFO): the case of the ODE software process ontology. In: *Memorias de la XI Conferencia Iberoamericana de Software Engineering (CIbSE 2008), Recife, Pernambuco, Brasil, February 13-17, 2008*. [S.l.: s.n.], 2008. p. 127–140. Citado 2 vezes nas páginas 35 e 46.

GUIZZARDI, G.; WAGNER, G.; ALMEIDA, J. P. A.; GUIZZARDI, R. S. S. Towards ontological foundations for conceptual modeling: The unified foundational ontology (UFO) story. *Applied Ontology*, v. 10, n. 3-4, p. 259–271, 2015. Disponível em: <https://doi.org/10.3233/AO-150157>. Citado 2 vezes nas páginas 31 e 46.

GUIZZARDI, G.; WAGNER, G.; FALBO, R. de A.; GUIZZARDI, R. S. S.; ALMEIDA, J. P. A. Towards Ontological Foundations for the Conceptual Modeling of Events. In: NG, W.; STOREY, V. C.; TRUJILLO, J. (Ed.). *ER*. Springer, 2013. (Lecture Notes in Computer Science, v. 8217), p. 327–341. ISBN 978-3-642-41923-2. Disponível em: <http://dblp.uni-trier.de/db/conf/er/er2013.html#GuizzardiWFGA13>. Citado 2 vezes nas páginas 35 e 62.

GUIZZARDI, R.; GUIZZARDI, G. Ontology-based transformation framework from TROPOS to AORML. *Social Modeling for Requirements Engineering, Cooperative Information Systems Series, MIT Press, Boston*, Citeseer, 2010. Citado na página 48.

GUIZZARDI, R. S. S.; FRANCH, X.; GUIZZARDI, G. Applying a foundational ontology to analyze means-end links in the i* framework. In: *Sixth International Conference on Research Challenges in Information Science, RCIS 2012, Valencia, Spain, May 16-18 2012*. [s.n.], 2012. p. 1–11. Disponível em: <https://doi.org/10.1109/RCIS.2012.6240425>. Citado 3 vezes nas páginas 26, 52 e 54.

GUIZZARDI, R. S. S.; FRANCH, X.; GUIZZARDI, G.; WIERINGA, R. Using a foundational ontology to investigate the semantics behind the concepts of the i* language.

In: *Proceedings of the 6th International* i* *Workshop 2013, Valencia, Spain, June 17-18, 2013*. [s.n.], 2013. p. 13–18. Disponível em: <http://ceur-ws.org/Vol-978/paper_3.pdf>. Citado 2 vezes nas páginas 53 e 65.

GUIZZARDI, R. S. S.; LI, F.; BORGIDA, A.; GUIZZARDI, G.; HORKOFF, J.; MYLOPOULOS, J. An Ontological Interpretation of Non-Functional Requirements. In: *Formal Ontology in Information Systems - Proceedings of the Eighth International Conference, FOIS 2014, September, 22-25, 2014, Rio de Janeiro, Brazil*. [s.n.], 2014. p. 344–357. Disponível em: <https://doi.org/10.3233/978-1-61499-438-1-344>. Citado 9 vezes nas páginas 9, 39, 40, 42, 44, 46, 50, 58 e 63.

HAREL, D.; RUMPE, B. *Modeling Languages: Syntax, Semantics and All That Stuff, Part I: The Basic Stuff*. Jerusalem, Israel, Israel, 2000. Disponível em: <http://portal.acm.org/citation.cfm?id=903627>. Citado 2 vezes nas páginas 16 e 43.

HEAVEN, W.; FINKELSTEIN, A. UML profile to support requirements engineering with KAOS. *IEE Proceedings - Software*, v. 151, n. 1, p. 10–28, 2004. Disponível em: <https://doi.org/10.1049/ip-sen:20040297>. Citado na página 16.

HORKOFF, J.; AYDEMIR, F. B.; CARDOSO, E.; LI, T.; MATÉ, A.; PAJA, E.; SALNITRI, M.; MYLOPOULOS, J.; GIORGINI, P. Goal-oriented requirements engineering: A systematic literature map. In: *24th IEEE International Requirements Engineering Conference*. [S.l.: s.n.], 2016. p. 106–115. Citado na página 46.

JACKSON, M. *Software Requirements &Amp; Specifications: A Lexicon of Practice, Principles and Prejudices*. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 1995. ISBN 0-201-87712-0. Citado 2 vezes nas páginas 23 e 50.

JACKSON, M. *Problem Frames: Analyzing and Structuring Software Development Problems*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2001. ISBN 0-201-59627-X. Citado na página 56.

JACKSON, M.; ZAVE, P. Deriving Specifications from Requirements: An Example. In: *Proceedings of the 17th International Conference on Software Engineering*. New York, NY, USA: ACM, 1995. (ICSE '95), p. 15–24. ISBN 0-89791-708-1. Disponível em: <http://doi.acm.org/10.1145/225014.225016>. Citado 2 vezes nas páginas 15 e 20.

JIANG, L.; TOPALOGLOU, T.; BORGIDA, A.; MYLOPOULOS, J. Incorporating goal analysis in database design: A case study from biological data management. In: *14th IEEE International Conference on Requirements Engineering (RE 2006), 11-15 September 2006, Minneapolis/St.Paul, Minnesota, USA*. [s.n.], 2006. p. 196–204. Disponível em: <https://doi.org/10.1109/RE.2006.33>. Citado na página 56.

JURETA, I.; BORGIDA, A.; ERNST, N. A.; MYLOPOULOS, J. Techne: Towards a New Generation of Requirements Modeling Languages with Goals, Preferences, and Inconsistency Handling. In: *RE 2010, 18th IEEE International Requirements Engineering Conference, Sydney, New South Wales, Australia, September 27 - October 1, 2010*. [s.n.], 2010. p. 115–124. Disponível em: <https://doi.org/10.1109/RE.2010.24>. Citado 2 vezes nas páginas 62 e 63.

JURETA, I.; MYLOPOULOS, J.; FAULKNER, S. Revisiting the core ontology and problem in requirements engineering. In: *16th IEEE International Requirements*

Engineering Conference, RE 2008, 8-12 September 2008, Barcelona, Catalunya, Spain. [s.n.], 2008. p. 71–80. Disponível em: <https://doi.org/10.1109/RE.2008.13>. Citado 2 vezes nas páginas 27 e 29.

JURETA, I.; MYLOPOULOS, J.; FAULKNER, S. Revisiting the Core Ontology and Problem in Requirements Engineering. IEEE Computer Society, p. 71–80, 2008. Citado na página 43.

JURETA, I. J.; MYLOPOULOS, J.; FAULKNER, S. A Core Ontology for Requirements. *Appl. Ontol.*, IOS Press, v. 4, n. 3-4, p. 169–244, 2009. ISSN 1570-5838. Citado 2 vezes nas páginas 27 e 46.

KOLIADIS, G.; GHOSE, A. Relating business process models to goal-oriented requirements models in kaos. In: _____. *Advances in Knowledge Acquisition and Management: Pacific Rim Knowledge Acquisition Workshop, PKAW 2006, Guilin, China, August 7-8, 2006, Revised Selected Papers*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006. p. 25–39. ISBN 978-3-540-68957-7. Disponível em: <https://doi.org/10.1007/11961239_3>. Citado na página 56.

LAMSWEERDE, A. van. Goal-oriented requirements engineering: A guided tour. In: *5th IEEE International Symposium on Requirements Engineering (RE 2001), 27-31 August 2001, Toronto, Canada*. [s.n.], 2001. p. 249. Disponível em: <https://doi.org/10.1109/ISRE.2001.948567>. Citado 5 vezes nas páginas 9, 14, 23, 24 e 25.

LAMSWEERDE, A. van. From system goals to software architecture. In: _____. *Formal Methods for Software Architectures: Third International School on Formal Methods for the Design of Computer, Communication and Software Systems: Software Architectures*. [S.l.]: Springer Berlin Heidelberg, 2003. p. 25–43. ISBN 978-3-540-39800-4. Citado 2 vezes nas páginas 62 e 64.

LAMSWEERDE, A. van. Reasoning about alternative requirements options. In: *Conceptual Modeling: Foundations and Applications - Essays in Honor of John Mylopoulos*. [s.n.], 2009. p. 380–397. Disponível em: <https://doi.org/10.1007/978-3-642-02463-4_20>. Citado 2 vezes nas páginas 24 e 62.

LAMSWEERDE, A. van; LETIER, E. From object orientation to goal orientation: A paradigm shift for requirements engineering. In: *Radical Innovations of Software and Systems Engineering in the Future, 9th International Workshop, RISSEF 2002, Venice, Italy, October 7-11, 2002, Revised Papers*. [s.n.], 2002. p. 325–340. Disponível em: <https://doi.org/10.1007/978-3-540-24626-8_23>. Citado 3 vezes nas páginas 14, 22 e 23.

LAPOUCHNIAN, A.; YU, Y.; MYLOPOULOS, J. Requirements-driven design and configuration management of business processes. In: *In BPM (2007*. [S.l.: s.n.], 2007. p. 246–261. Citado na página 56.

LEVESON, N. G. *Safeware - system safety and computers: a guide to preventing accidents and losses caused by technology*. [S.l.]: Addison-Wesley, 1995. I-XVII, 1-680 p. ISBN 978-0-201-11972-5. Citado 2 vezes nas páginas 23 e 50.

LI, F.; HORKOFF, J.; MYLOPOULOS, J.; GUIZZARDI, R. S. S.; GUIZZARDI,

G.; BORGIDA, A.; LIU, L. Non-functional requirements as qualities, with a spice of ontology. In: *IEEE 22nd International Requirements Engineering Conference, RE 2014, Karlskrona, Sweden, August 25-29, 2014*. [s.n.], 2014. p. 293–302. Disponível em: <https://doi.org/10.1109/RE.2014.6912271>. Citado na página 40.

LIU, L.; CHI, C. hung; JIN, Z.; YU, E. *Strategic Capability Modelling of Services*. 2006. Citado na página 56.

LÓPEZ, L.; FRANCH, X.; MARCO, J. Making explicit some implicit i* language decisions. In: *Conceptual Modeling - ER 2011, 30th International Conference, ER 2011, Brussels, Belgium, October 31 - November 3, 2011. Proceedings*. [s.n.], 2011. p. 62–77. Disponível em: <https://doi.org/10.1007/978-3-642-24606-7_6>. Citado 2 vezes nas páginas 16 e 56.

LUCENA, M.; SANTOS, E.; SILVA, C. T. L. L.; ALENCAR, F. M. R.; SILVA, M. J.; CASTRO, J. Towards a unified metamodel for i*. In: *Proceedings of the IEEE International Conference on Research Challenges in Information Science, RCIS 2008, Marrakech, Morocco, June 3-6, 2008*. [s.n.], 2008. p. 237–246. Disponível em: <https://doi.org/10.1109/RCIS.2008.4632112>. Citado 2 vezes nas páginas 16 e 43.

MASOLO, C.; BORGO, S.; GANGEMI, A.; GUARINO, N.; OLTRAMARI, A.; SCHNEIDER, L. Dolce: a descriptive ontology for linguistic and cognitive engineering. *WonderWeb Project, Deliverable D17 v2*, v. 1, 2003. Citado 3 vezes nas páginas 27, 43 e 46.

MATULEVICIUS, R.; HEYMANS, P. *Analysis of KAOS Meta-model*. Namur, Belgium, 2005. Citado 2 vezes nas páginas 16 e 43.

MUMFORD, S. *Dispositions*. Oxford University Press, 2003. ISBN 9780199259823. Disponível em: <https://books.google.com.br/books?id=z9EH9OO5EiUC>. Citado na página 62.

MYLOPOULOS, J. Conceptual modelling and telos. In: LOUCOPOULOS, P.; ZICARI, R. (Ed.). *Conceptual Modeling, Databases, and Case: An Integrated View of Information Systems Development*. New York, NY, USA: John Wiley & Sons, Inc., 1992. cap. 2, p. 46–68. ISBN 0471554626. Citado 2 vezes nas páginas 16 e 19.

NARDI, J. C.; FALBO, R. de A.; ALMEIDA, J. P. A.; GUIZZARDI, G.; PIRES, L. F.; SINDEREN, M. van; GUARINO, N. Towards a Commitment-Based Reference Ontology for Services. In: *17th IEEE International Enterprise Distributed Object Computing Conference, EDOC 2013, Vancouver, BC, Canada, September 9-13, 2013*. [s.n.], 2013. p. 175–184. Disponível em: <https://doi.org/10.1109/EDOC.2013.28>. Citado na página 39.

NEGRI, P. P.; SOUZA, V. E. S.; LEAL, A. L. d. C.; FALBO, R. A.; GUIZZARDI, G. Towards an Ontology of Goal-Oriented Requirements. In: *Proc. of the 20th Ibero-American Conference on Software Engineering, Requirements Engineering track*. Buenos Aires, Argentina: [s.n.], 2017. Disponível em: <http://wer.inf.puc-rio.br/WERpapers/papers_by_conference.lp?conference=WER17>. Citado na página 68.

NWOKEJI, J. C.; CLARK, T.; BARN, B. S. Towards a comprehensive meta-model for KAOS. In: *International Workshop on Model-Driven Requirements Engineering,*

MoDRE 2013, Rio de Janeiro, Brasil, July 15, 2013. [s.n.], 2013. p. 30–39. Disponível em: <https://doi.org/10.1109/MoDRE.2013.6597261>. Citado 2 vezes nas páginas 16 e 43.

OMG. *OMG Unified Modeling Language (OMG UML), Superstructure, Version 2.4.1*. 2011. Disponível em: <http://www.omg.org/spec/UML/2.4.1>. Citado na página 47.

PAECH, B.; KERKOW, D. Non-functional requirements engineering - quality is essential. In: *10th International Workshop on Requirements Engineering: Foundation for Software Quality, 2004. REFSQ '04. Riga, Latvia*. [S.l.: s.n.], 2004. p. 237–250. ISBN ISBN: 3-922602-91-6. Citado na página 39.

PATRÍCIO, P.; AMARAL, V.; ARAúJO, J. a.; RUI, M. Towards a unified goal-oriented language. *Proceedings - International Computer Software and Applications Conference*, p. 596–601, 2011. ISSN 07303157. Citado na página 43.

PIMENTEL, J.; CASTRO, J.; MYLOPOULOS, J.; ANGELOPOULOS, K.; SOUZA, V. E. S. From requirements to statecharts via design refinement. In: CHO, Y.; SHIN, S. Y.; KIM, S.-W.; HUNG, C.-C.; HONG, J. (Ed.). *SAC*. ACM, 2014. p. 995–1000. ISBN 978-1-4503-2469-4. Disponível em: <http://dblp.uni-trier.de/db/conf/sac/sac2014.html#PimentelCMAS14>. Citado na página 49.

QUARTEL, D.; ENGELSMAN, W.; JONKERS, H.; SINDEREN, M. V. A goal-oriented requirements modelling language for enterprise architecture. In: *Proceedings of the 13th IEEE International Conference on Enterprise Distributed Object Computing*. Piscataway, NJ, USA: IEEE Press, 2009. (EDOC'09), p. 1–11. ISBN 978-1-4244-4773-2. Disponível em: <http://dl.acm.org/citation.cfm?id=1719357.1719358>. Citado na página 56.

ROMAN, G. A Taxonomy of Current Issues in Requirements Engineering. *IEEE Computer*, v. 18, n. 4, p. 14–23, 1985. Disponível em: <https://doi.org/10.1109/MC.1985.1662861>. Citado na página 19.

RUY, F. *Software Engineering Standards Harmonization: an Ontology-based Approach*. Tese (Doutorado) — Universidade Federal do Espírito Santo, Brazil, 2017. Citado na página 68.

SEARLE, J. R. *Mind, Language and Society: Philosophy in the Real World*. [S.l.: s.n.], 1998. Citado na página 37.

SOMMERVILLE, I. *Software Engineering*. 9th. ed. USA: Addison-Wesley Publishing Company, 2010. ISBN 0137035152, 9780137035151. Citado na página 15.

SOUZA, V. E. S. *Requirements-based Software System Adaptation*. Tese (Doutorado) — University of Trento, Italy, 2012. Citado 2 vezes nas páginas 9 e 66.

WANG, X.; GUARINO, N.; GUIZZARDI, G.; MYLOPOULOS, J. Towards an Ontology of Software: a Requirements Engineering Perspective. In: *Formal Ontology in Information Systems - Proceedings of the Eighth International Conference, FOIS 2014, September, 22-25, 2014, Rio de Janeiro, Brazil*. [s.n.], 2014. p. 317–329. Disponível em: <https://doi.org/10.3233/978-1-61499-438-1-317>. Citado 10 vezes nas páginas 9, 46, 48, 49, 54, 55, 56, 57, 58 e 59.

WANG, X.; MYLOPOULOS, J.; GUIZZARDI, G.; GUARINO, N. How software changes the world: The role of assumptions. In: *Tenth IEEE International Conference on Research*

Challenges in Information Science, RCIS 2016, Grenoble, France, June 1-3, 2016. [s.n.], 2016. p. 1–12. Disponível em: <https://doi.org/10.1109/RCIS.2016.7549327>. Citado 8 vezes nas páginas 9, 20, 21, 22, 46, 50, 51 e 58.

YU, E. S.-K. *Modelling strategic relationships for process reengineering*. Tese (Doutorado) — University of Toronto, 1995. Citado 6 vezes nas páginas 16, 23, 25, 26, 56 e 63.

ZAVE, P.; JACKSON, M. Four Dark Corners of Requirements Engineering. *ACM Transactions on Software Engineering and Methodology*, v. 6, n. 1, p. 1–30, 1997. Citado 3 vezes nas páginas 15, 20 e 48.