UNIVERSIDADE FEDERAL DE PELOTAS
Centro de Desenvolvimento Tecnológico
Programa de Pós-Graduação em Computação

Tese

AgentDevLaw: Um Meta-Modelo e um *Middleware* para a Integração de Ontologias Legais e Sistemas Multiagentes

Fábio Aiub Sperotto

Pelotas, 2020

Fábio Aiub Sperotto

AgentDevLaw: Um Meta-Modelo e um *Middleware* para a Integração de Ontologias Legais e Sistemas Multiagentes

Tese apresentada ao Programa de Pós-Graduação em Computação do Centro de Desenvolvimento Tecnológico da Universidade Federal de Pelotas, como requisito parcial à obtenção do título de Doutor em Ciência da Computação.

Orientador: Prof. Dr. Marilton Sanchotene de Aguiar

Pelotas, 2020

Universidade Federal de Pelotas / Sistema de Bibliotecas
Catalogação na Publicação

S749a Sperotto, Fábio Aiub

AgentDevlaw : um meta-modelo e um *middleware* para a integração de ontologias legais e sistemas multiagentes / Fábio Aiub Sperotto ; Marilton Sanchotene de Aguiar, orientador. — Pelotas, 2020.

107 f. : il.

Tese (Doutorado) — Programa de Pós-Graduação em Computação, Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, 2020.

1. Sistemas multiagentes. 2. Ontologia. 3. Legislação brasileira. I. Aguiar, Marilton Sanchotene de, orient. II. Título.

CDD : 005

Elaborada por Maria Inez Figueiredo Figas Machado CRB: 10/1612

Fábio Aiub Sperotto

AgentDevLaw: Um Meta-Modelo e um *Middleware* para a Integração de Ontologias Legais e Sistemas Multiagentes

Tese aprovada, como requisito parcial, para obtenção do grau de Doutor em Ciência da Computação, Programa de Pós-Graduação em Computação, Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas.

Data da Defesa: 9 de junho de 2020

Banca Examinadora:

Prof. Dr. Marilton Sanchotene de Aguiar (orientador)
Doutor em Computação pela Universidade Federal do Rio Grande do Sul.

Prof. Dra. Ana Marilza Pernas
Doutora em Computação pela Universidade Federal do Rio Grande do Sul.

Prof. Dr. Antonio Carlos da Rocha Costa
Doutor em Computação pela Universidade Federal do Rio Grande do Sul.

Prof. Dra. Diana Francisca Adamatti
Doutora em Engenharia Elétrica pela Escola Politécnica da Universidade de São Paulo.

Dedico a todos, assim como eu, que acreditam na educação e na ciência como práticas para aprimorar o mundo.

AGRADECIMENTOS

Agradeço ao professor Dr. Marilton Sanchotene de Aguiar pela inestimável condução na orientação durante todas as etapas do doutorado. Obrigado por acreditar em mim e pela oportunidade da realização desta importante etapa em minha carreira. Agradeço as conversas, os incentivos, que me foram de grande importância para a conclusão deste trabalho.

Obrigado ao Dr. Mairon Belchior por todas as avaliações realizadas no modelo da ontologia. Pelas correções e dicas, desde a estrutura inicial até o modelo final. Agradeço pelas conversas que me guiaram em maiores estudos na área de ontologia.

Agradeço a todos os demais docentes e técnicos administrativos do Centro de Desenvolvimento Tecnológico onde reside o Programa de Pós-Graduação e Computação, da Universidade Federal de Pelotas. De forma direta ou indireta, são responsáveis pelas realizações dos discentes em suas formações.

Agradeço aos colegas da área de Informática, do campus Camaquã, do Instituto Federal Sul-rio-grandense (IFSul), os quais me apoiaram, direta ou indiretamente, no cumprimento dos estudos nesta formação. Agradeço ao IFSul pelo apoio concedido através de edital de licença de afastamento, importante para a qualidade da produção científica de seus docentes.

Agradeço aos meus pais, Flávio Juarez Sperotto, Regina Aiub Sperotto e a minha irmã, Dra. Fabiola Aiub Sperotto, pelo apoio de sempre. Família de docentes que me auxiliaram na jornada acadêmica e repassaram vários aspectos importantes em minha personalidade. Desta forma, colaboraram muito nesta minha escolha em ser professor e pesquisador.

Agradeço a minha querida Cynthia Zeni Refosco. Meu porto seguro, onde inúmeras vezes ouviu meus lamentos, reclamações, alegrias, sempre me ajudando para o sucesso de meus projetos pessoais e profissionais. O meu crescimento pessoal, espiritual, devo muito a ti.

Agradeço as docentes que fizeram parte da banca, Prof. Dra. Ana Marilza Pernas, Prof. Dr. Antonio Carlos da Rocha Costa e Profa. Dra. Diana Francisca Adamatti. Através de suas colocações, desde a proposta inicial, até a defesa final, enriqueceram este estudo.

Por fim, agradeço a todos os demais que passaram pela minha vida e contribuíram de forma direta ou indireta para ser quem eu sou hoje.

Não há como escapar a ordem imaginada. Quando derrubamos os muros da nossa prisão e corremos para a liberdade, estamos, na verdade, correndo para o pátio mais espaçoso de uma prisão maior.
— YUVAL NOAH HARARI

RESUMO

SPEROTTO, Fábio Aiub. **AgentDevLaw: Um Meta-Modelo e um *Middleware* para a Integração de Ontologias Legais e Sistemas Multiagentes**. Orientador: Marilton Sanchotene de Aguiar. 2020. 107 f. Tese (Doutorado em Ciência da Computação) – Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, Pelotas, 2020.

Com os sistemas multiagentes (SMA), as simulações sociais são desenvolvidas com o objetivo de compreender e desenvolver novas ideias sobre a sociedade. Estas simulações podem ocorrer através de variados temas, na área de jogos, de trânsito, resposta a sinistros, sobre políticas, entre outros. Estas pesquisas procuram entender como trocas sociais são construídas e como agentes podem interagir para resolver variados problemas. Os agentes podem necessitar conhecer mais sobre suas ações e sobre o meio em que operam (o que fazer, como fazer e quais regras são existentes). Neste sentido, os agentes utilizam ontologias para reconhecer e processar variados conhecimentos. As ontologias são descrições sobre o mundo como os seres humanos relacionam as coisas. Na área de Ciência da Computação existem esforços em modelar o conhecimento humano e distribuir estas informações em ontologia para que possam serem interpretadas em meio computacional. Em algum momento, é necessário realizar a regulação das ações destes agentes, restringindo os seus comportamentos. As ontologias legais são um tipo específico para lidar com modelagem de normas, de regulações, funções e papéis, oriundos da área jurídica. Assim, este trabalho desenvolve um modelo de ontologia legal e de um *middleware* que possa auxiliar as variadas plataformas de agentes em reconhecer e interpretar as informações legais destas ontologias. A base de conhecimento dessa ontologia é na lei brasileira e como ela pode ser reconhecida pelos sistemas multiagentes. Ao final, são realizados ensaios de aplicações desta proposta com simulações de regulação de ações e da criação de novas leis pelos agentes (proposição de novas leis pela Câmara e Senado). Os resultados dos testes demonstram a aplicabilidade e como o mecanismo auxilia na operação e compreensão das leis pelos agentes.

Palavras-chave: Sistemas Multiagentes. Ontologia. Legislação Brasileira.

ABSTRACT

SPEROTTO, Fábio Aiub. **AgentDevLaw: A Middleware Architecture for the Integration of Legal Ontologies and Multiagent Systems**. Advisor: Marilton Sanchotene de Aguiar. 2020. 107 f. Thesis (Doctorate in Computer Science) – Technology Development Center, Federal University of Pelotas, Pelotas, 2020.

With multi-agent systems (MAS), social simulations are developed with the aim of understanding and developing new ideas about society. These simulations can occur through various themes, in the area of games, traffic, response to claims, about policies, among others. These surveys seek to understand how social exchanges are built and how agents can interact to solve various problems. Agents may need to know more about their actions and the environment in which they operate (what to do, how to do it, and what rules are in place). In this sense, agents use ontologies to recognize and process a variety of knowledge. Ontologies are descriptions of the world as human beings relate things. In the area of Computer Science, there are efforts to model human knowledge and distribute this information in ontology so that they can be interpreted in a computational environment. At some point, it is necessary to regulate the actions of these agents, restricting their behavior. Legal ontologies are a specific type to deal with modeling of norms, regulations, functions, and roles, coming from the legal area. Thus, this work develops a model of legal ontology and middleware that can assist the various platforms of agents in recognizing and interpreting the legal information of these ontologies. The knowledge base of this ontology is in Brazilian law and how it can be recognized by multi-agent systems. In the end, tests of applications of this proposal are carried out with simulations of regulation of actions and the creation of new laws by the agents (proposal of new laws by the Chamber and Senate). The results of the tests demonstrate the applicability and how the approach helps in the operation and understanding of the laws by the agents.

Keywords: Multi-agent Systems. Ontology. Brazilian Legislation.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

API	Interface de Programação de Aplicação
FIPA	*Foundation for Intelligent Physical Agents*
GDPR	*General Data Protection Regulation*
JADE	*JAVA Agent DEvelopment Framework*
LCO	*Legal Core Ontology*
LKIF	*Legal Knowledge Interchange Format*
LRA	*Linked REST APIs*
OWL	*Web Ontology Language*
PLN	Processamento de Linguagem Natural
RDF	*Resource Description Framework*
REST	*Representational State Transfer*
SMA	Sistema Multiagente
SPARQL	*SPARQL Protocol and RDF Query Language*
SWRL	*Semantic Web Rule Language*
UFO	*Unified Fountational Ontology*
W3C	*World Wide Web Consortium*
XML	*Extensible Markup Language*
XSD	*XML Schema Datatype*

SUMÁRIO

1 INTRODUÇÃO

Na linha histórica de pesquisas em inteligência artificial distribuída, o conceito de agentes tornou-se algo inovador, o desenvolvimento de suas aplicações e linhas de pesquisas o fizeram ser considerado como uma alternativa interessante em relação a outros paradigmas. Os agentes originalmente são definidos como sistemas de computação situados em algum ambiente, capazes de realizar ações autônomas para alcançar os objetivos para os quais foram construídos (WOOLDRIDGE, 1999). Estes agentes são empregados em variados setores, desde problemas de distribuição de recursos elétricos (LOGENTHIRAN; SRINIVASAN; KHAMBADKONE, 2011), em gerenciamento de transporte (BAZZAN; KLÜGL, 2009), em melhores serviços para a *web* (LÜTZENBERGER et al., 2016), em gerenciamento de recursos naturais (HECKBERT; BAYNES; REESON, 2010) entre muitos outros estudos tanto sociais quanto industriais.

Estas entidades de *software* procedem em operar com suas atividades, reconhecendo e agindo sobre o ambiente em que estão alocadas. Este novo paradigma na construção de *software* favorece uma alternativa para sistemas de tomadas de decisão baseados na descentralização de funções (LEITAO; MARIK; VRBA, 2013). Uma entidade possui uma atividade bem definida e vários agentes podem, dependendo do contexto, receber uma parte de algum problema a ser resolvido. Estes agentes, trabalhando com a decomposição de problema, operam com suas funcionalidades e capacidades para resolver as missões do seu sistema, sejam quais forem.

Nestas capacidades reside também a necessidade da comunicação. Os agentes, quando reunidos, são classificados como um sistema multiagente (SMA) e nestes existe a necessidade da comunicação, como fator importante entre as entidades. Uma vez que agentes precisam resolver determinados problemas que são decompostos entre as entidades e, independentemente se são cooperativos ou competitivos, invariavelmente deverá existir a comunicação para o repasse de dados. Estas entidades, uma vez sendo reconhecidas como objetos de programação, existe a passagem de mensagens entre estes, transmitindo e recebendo dados (GENESERETH; KETCHPEL, 1994).

Através destes aspectos de autonomia e comunicação, seria uma questão de tempo até que estes agentes fossem considerados como uma sociedade de agentes. Neste sentido, existe uma intersecção nos estudos em ciências sociais, em computação baseada em agentes e em simulação de computadores, reunindo pesquisas para a concepção destes novos sistemas (DAVIDSSON, 2002).

A partir deste ponto, os objetos chamados agentes começam a receber uma correspondência com agentes da sociedade do mundo real (GILBERT, 2004). Os modelos de sistemas multiagentes permanecem como um conjunto de objetos que cooperam, mas agora cada objeto pode modelar um indivíduo. Nesta modelagem, crenças e intenções são codificadas nos agentes (ADAM; GAUDOU, 2016) que são organizados em grupos sociais tal e qual o ser humano. Isto favorece simulações onde é possível modelar conforme a realidade e simular hipóteses. Várias pesquisas apontam o uso de sistemas multiagentes para simulações socioeconômicas como apoio em tomadas de decisão (FURDÍK; SABOL; DULINOVÁ, 2010).

Entre os tipos de agentes, considera-se a divisão deste em entidades cognitivas e entidades reativas (SAWYER, 2003). Os agentes cognitivos possuem crenças sobre o ambiente e conhecimento sobre os estados das coisas (internos dos agentes e externos, no ambiente) e, de suas ações. É nesta categoria que, com a capacidade de raciocínio, os agentes também possuem a habilidade da comunicação. Os agentes reativos são contrários aos cognitivos, no sentido de que não possuem qualquer conhecimento interno ou externo (ambiente e de outros agentes).

No lado cognitivo, mais focado em sociedades complexas como a dos seres humanos, os agentes realizam trocas sociais, pois o foco permanece nas relações dos indivíduos e na resolução de problemas da sociedade. Como os agentes são autônomos em seus comportamentos e possuem suas próprias concepções de mundo, podem existir uma variedade de concepções diferentes dentro de um mesmo conjunto de entidades. Além disso, fatores como ações proibidas e consumos desenfreados de recursos podem levar a necessidade da regulação dos comportamentos dos agentes.

Neste fato, a ideia de uma sociedade social artificial surgiu pois não se possui mais a necessidade de resolver apenas a intenção de um agente e sim das intenções dentro de uma sociedade (MOSES; TENNENHOLTZ, 1995; HUHNS; STEPHENS, 1999). Isto acaba por alcançar o que se chama de sistemas normativos, que aplicam teorias sociais unindo as concepções de agente em software com as concepções do agente social da sociologia, filosofia, economia e ciências legais (BOELLA; TORRE; VERHAGEN, 2006). Estruturas como normas e agentes com papéis cada vez mais especializados começam a atuar na regulação dos agentes.

As regulações tendem a ser necessárias para atuar em agentes que possam violar determinadas regras sociais, entrar em conflitos com outros agentes, ou ainda, serem recompensados por atuações exemplares. Existem estudos sobre como pro-

ceder na criação de leis sociais e como estas podem inclusive diminuir a necessidade de negociações entre agentes (FITOUSSI; TENNENHOLTZ, 2000).

As leis podem serem utilizadas como abstrações em SMA abertos, na especificação de políticas que regulam as interações entre agentes (PAES et al., 2005). Naquele estudo, um *middleware* é fornecido como verificador das interações dos agentes, limitando a autonomia dos mesmos. Na relação de agentes mais especializados em verificar as sanções e planejar ações em relação a isso, pesquisas demonstram o uso de artefatos e modelos de normas para que os agentes sejam autuadores das ações no seu domínio de simulação (SANTOS; COSTA, 2012).

Os agentes que desempenham funções nestes sistemas sociais, precisam compreender o mundo da qual se encontram, o contexto de suas ações e dados relevantes sobre o mundo que os cerca. Neste sentido, fornecer conhecimento ou habilidades de raciocínio para estas entidades é fundamental na concretização de variados planos de simulações.

As ontologias procuram estruturar o conhecimento humano, de como os conceitos são criados e relacionados, apresentando uma visão formal do conhecimento. Na ciência da computação isto é materializado em estudos onde estruturas de bases de conhecimentos são desenvolvidas focando no compartilhamento de conhecimento estruturado (SMITH; WELT, 2001).

Em SMA as ontologias são reconhecidas como importantes aplicações para variadas necessidades (BERMEJO-ALONSO; SANZ; LOPEZ, 2006): i) em esclarecer o conhecimento para o domínio a ser utilizado para elaborar um vocabulário efetivo; ii) escalabilidade de conhecimento usando a ontologia como gerenciamento de conhecimento; iii) compartilhando ou reusando o conhecimento, associando termos com conceitos e relacionamentos em uma sintaxe utilizada pelos agentes; iv) robustez aos SMA pois os agentes podem recorrer as ontologias a fim de raciocinar sobre eventos atuais e novos; v) fornecer base de interoperabilidade entre agentes; e, vi) fornecer um foco no domínio da engenharia de software de SMA e auxiliar as equipes de desenvolvimento a ter uma base de compreensão cognitiva e de integração dos agentes.

De um lado, a necessidade de regulações de comportamentos de uma sociedade de agentes, de outro as ontologias estruturando conhecimento humano. Na realidade humana, os sistemas legislativos fornecem as regulações necessárias para que as trocas sociais sejam menos danosas e mais justas entre os indivíduos. O questionamento é de como operar este conhecimento para com os agentes.

Nos SMA, como as ontologias têm sido uma opção importante no fornecimento de conhecimento, as ontologias legais surgem como apoio nas resoluções destas necessidades. As ontologias legais favorecem estruturas específicas para comportar o conhecimento legal humano e fornecer computacionalmente estas restrições ou orientações legislativas (BREUKER; VALENTE; WINKELS, 2005).

Além dos modelos, há a necessidade de definição destes com ferramentas no apoio aos pesquisadores da área. As sociedades de agentes e seus sistemas computacionais são definidos e construídos para a resolução de inúmeros problemas. Somente neste caso existe a demanda por ferramentas especialistas que possam oferecer um ambiente de simulação que auxilie aos pesquisadores em suas resoluções. No caso da ontologia, sendo um outro campo de pesquisa, outros especialistas também continuam permanentemente fornecendo modelos e construindo relações de conhecimentos. Os questionamentos residem em como trazer estas duas vertentes para as diferentes plataformas de sistemas multiagentes. Um modelo de ontologia legal que reflete aspectos legislativos ou de regulações de agentes bem como um dispositivo nos ambientes em que os agentes possam operar com este modelo.

Através destes variados pontos de intersecção, a presente Tese visa propor a compreensão do fornecimento de conhecimento legal para os agentes através de uma ontologia e de um sistema pelo qual possa ser compreendido pelos SMA em suas plataformas de simulação. Este trabalho pretende viabilizar um modelo de uma ontologia legal da legislação brasileira dentro de uma concepção de *middleware* para que os agentes, em variadas plataformas, possam compreender e utilizar o conhecimento legislativo registrado na ontologia.

1.1 Objetivos do Trabalho

O objetivo geral desta Tese é o desenvolvimento de um modelo e de uma ferramenta que contemple a aplicação de normas legislativas em Sistemas Multiagentes.

Mais especificamente, tem-se os seguintes objetivos:

1. Proposta de um meta-modelo de Ontologia que possa comportar a definição de leis e normas, referenciando um sistema legal.

2. Desenvolvimento de um *middleware* tal que: i) facilite a interpretação das estruturas da ontologia, reconhecendo as suas normas, e a sua aplicação em um SMA; ii) viabilize, ao longo do tempo, adições de legislação sob algum sistema fornecido pela própria sociedade de agentes (votações, trocas sociais, entre outros); e, iii) permita ser reutilizado em diferentes plataformas de simulação de agentes.

3. Validação do sistema legal proposto sobre um estudo de caso, na área ambiental, a partir da modelagem do conhecimento de domínio, elaboração da ontologia e instanciação do *middleware*.

1.2 Estrutura da Tese

Esta Tese está organizada da seguinte maneira: o Capítulo 2 apresenta o referencial teórico, descrevendo e relacionando os estudos da área fim deste trabalho (ontologias, legislação e *middlewares*). O Capítulo 3 descreve a proposta, baseando-se no referencial, contendo os aspectos tecnológicos necessários para a sua construção dos objetivos propostos. O Capítulo 4 apresenta os ensaios de aplicação da proposta, isto é, fornece cenários de exemplo onde a proposta pode ser aplicada e como deve ser utilizada, considerando duas plataformas diferentes de simulação de agentes. Por fim, o Capítulo 5 apresenta as conclusões desta Tese.

2 REFERENCIAL TEÓRICO

O presente Capítulo trata sobre os trabalhos existentes nas áreas de ontologias legais e *middlewares*. Sendo que cada Seção pretende organizar os trabalhos relacionados, discutindo os conceitos e práticas nas resoluções de problemas. Sendo estes problemas de ordem do compartilhamento de conhecimento legislativo em meio computacional como, por exemplo, de lidar com características de comunicação entre tecnologias distintas na mesma área de SMA.

2.1 Ontologias Legais

As restrições comportamentais relacionam-se geralmente com a regulamentação entre agentes, sejam na interação entre si, no acesso a recursos ou, ainda, em atividades possíveis que os agentes assumem sob determinados papéis. Em comparação com a sociedade, sistemas legislativos tendem a serem concebidos para o fim de regulamentar as pessoas e as instituições, determinando o convívio entre indivíduos de uma mesma sociedade ou entre sociedades diferentes.

Na área legal, os estudos de ČYRAS et al. (2011); BREUKER et al. (2005); COSTA (2015), sobre as definições e conceitos sobre lei e moralidade, são construídos utilizando como base a Teoria Pura da Lei (KELSEN, 2009) e a Teoria Geral de Normas (KELSEN, 1990). Estes pesquisadores relatam que Kelsen propõe uma nova metodologia jurídica, devido as suas análises filosóficas sobre a lei e a moralidade, definindo novos conceitos, principalmente sobre as normas, suas tipificações e as ações legais dos indivíduos. Estes autores, de um lado, definem uma melhor visualização ontológica dos estudos de Kelsen (ČYRAS et al., 2011), enquanto no outro, apresenta uma reconstrução formal da teoria de sistemas legais de Kelsen (COSTA, 2015).

As normas basicamente, no contexto dos estudos comentados, são pontos específicos que definem uma regra ou uma ordem que, por sua vez, determina a base para um sistema legal. Os tipos destas normas podem ser de comando, autorização, permissão e derrogação (revogação) (BREUKER et al., 2005).

Normas de comando definem a proibição ou a obrigação de um certo comportamento. As normas de autorização (ou de empoderamento) definem as associações de papéis com poderes de postular e aplicar normas sob certas restrições. As normas de permissão são associadas aos comportamentos que não são proibidos nem comandados, mas sim, ativamente permitidos.

Nos estudos de KELSEN (2009), a norma de permissão é considerada uma espécie de exceção de uma norma de comando. Isto ocorre quando existe uma norma de comando sobre um certo comportamento e esta é revogada por uma norma de permissão. Por fim, as normas de revogação removem a validade de outras normas, mas não as remove em si[1]. As causas e feitos também são definidas como pertencentes a essa teoria ou nos sistemas legais descritos, sendo que o maior exemplo são as imputações.

As ações dos indivíduos são caracterizadas pelas suas imputações e, estas, são produzidas por ação do individuo onde o seu comportamento é reconhecido como condição e a sanção como consequência, daquilo prescrito em uma norma (ČYRAS et al., 2011).

Ainda, no trabalho de ČYRAS et al. (2011) é definido um conceito mais recente sobre hierarquia de normas, mas que permanece fiel à definição de Kelsen em seu conceito raiz. A hierarquia define uma norma superiora (norma base) que define uma constituição, onde as normas inferiores são instâncias destas que definem os comportamentos concretos sobre proibição, obrigação, permissão, entre outros.

Em conjunto com as definições e relações daqueles aspectos normativos e através de outras premissas filosóficas e conceituais, as ontologias surgem como alternativa para a necessidade de criar uma técnica para estruturar ou explicitar as “coisas” ou objetos que os humanos concebem sobre o mundo (SMITH; WELT, 2001). Esta técnica é desenvolvida através de um vocabulário de termos, ou conceitos, do qual é formalizada em uma estrutura legível, modelando as definições de mundo, tanto por humanos quanto por máquinas.

A ideia de toda ontologia é oferecer um caminho específico onde máquinas possam operar com bases de conhecimento. Estas bases devem ser especificadas por especialistas humanos para que os sistemas possam atuar de acordo com regras ou fragmentos de conhecimento sobre um determinado domínio.

A ontologia possui, como um dos seus vários argumentos, o objetivo de reunir o que se compreende sobre uma determinada parte do universo, das condições em que se vê a realidade (SMITH, 2004). Com isso, propostas de ontologias são fornecidas para que seja possível modelar as definições destes sistemas legais em um formato onde indivíduos (no sentido humano) e agentes (entidades de *software*) possam ope-

[1] Devido a isso, dentro de um sistema legal, normas depreciadas podem existir, ou seja, a norma existe mas não é mais válida.

rar referenciando sistemas legais.

Nas próximas Seções apresentam-se as ontologias fundamentais e as ontologias legais. Ao final, serão discutidos aspectos entre modelos de ontologias legislativas para com os sistemas multiagentes e as definições para a proposta apresentada nesta Tese.

2.1.1 Ontologias Legais Fundamentais

Estudos sobre ontologias para sistemas de conhecimento legal e gerenciamento de informações legais têm sido desenvolvidos pelo *Leibniz Center of Law* na Universidade de Amsterdam (BREUKER et al., 2005). Após a proposta de variadas metodologias para modelar a racionalização sobre conhecimento legal, os pesquisadores desenvolveram um *framework* relacionado a arquiteturas de sistemas de reutilização destes conhecimentos.

Os autores analisaram e modelaram variados domínios legais como regulamentações de trânsito, impostos, leis administrativas, criminais e internacionais com o objetivo de abstrair denominadores legais comuns entre aqueles variados domínios legais, procurando uma visão unificada destes domínios. Após este primeiro momento, buscou-se suporte à criação de ontologias centrais como a FOLaw e LRI-Core, que não são ontologias de domínio e nem suficientemente genéricas, mas tendem a atuar como intermediárias entre estas duas categorias, seja na aquisição de conhecimentos, indexação de ontologias ou outros propósitos.

A FOLaw (VALENTE; BREUKER; BROUWER, 1999; BREUKER et al., 2005) fornece estruturas que distinguem os vários tipos de conhecimento em racionalização legal[2]. Além disso, auxilia a organização e indexação de bibliotecas de ontologias de domínio suportando a aquisição de novos conhecimentos para construir novas ontologias. A FOLaw foi desenvolvida sob a perspectiva semântica sociológica dos papéis assumidos no sistema legal, isto é, a abordagem não é puramente na lei mas em uma visão legal-social, cujo ponto inicial para as modelagens é a sociedade em si.

De forma geral, a ontologia central é formada por fontes legais que são relacionadas às categorias de conhecimento legal (divide o conhecimento pelas suas funções legais na sociedade), sendo elas (VALENTE; BREUKER; BROUWER, 1999):

Conhecimento Normativo: categoria típica de conhecimento legal, possui definições legais e normativas, isto é, as definições de restrições comportamentais.

Conhecimento Meta-legal: possui definições sobre que conhecimento legal é válido e também possui verificadores de conflitos entre normas.

[2]Os termos raciocínio e racionalização serão utilizados nesta Tese e ambos denotam estruturas ontológicas onde é possível realizar inferências ou outros mecanismos computacionais que se baseiam em formalização de conhecimento humano.

Conhecimento de Mundo: define conhecimentos sobre terminologia, conceitos e causa, sobre um determinado domínio. Nesta categoria ou função, são especificados os conceitos sobre mundo. Um exemplo que poderia ser modelado nesta categoria é a especificação de conceitos, causas e efeitos de uma legislação de trânsito.

Conhecimento de Responsabilidade: conhecimento intermediário entre normativo e reativo, atuando na conexão entre causa e responsabilidade, sendo a conexão entre o agente responsável por violar a norma e a possibilidade de reação legal.

Conhecimento Reativo: na conclusão que uma certa situação é ilegal (conhecimento normativo) e que um agente deve ser culpado por isso (conhecimento de responsabilidade), o conhecimento reativo explicita que reação deve ser executada e como ela deve ser aplicada, onde persistem as descrições de sanções e recompensas.

Conhecimento Criativo: conhecimento necessário para que os legisladores dos agentes possam, em tempo de execução, criar novas entidades. Assim a lei não serve somente para classificar ou reagir sobre os agentes existentes, mas conceber novos agentes.

Estas categorias dividem os conhecimentos legais e atuam como funções dentro da ontologia que, reunidas, definem o sistema legal a qual está inserida a ontologia (VALENTE; BREUKER; BROUWER, 1999). Existem dependências entre essas categorias, como dependências entre funções, existindo informações que atuam como entradas e saídas destas respectivas categorias. Existem ainda outros tipos de dependências mas relacionadas à interação com a sociedade a qual está aplicada a ontologia. A premissa aqui é gerar um fluxo para raciocinar sobre a legalidade das ações e informações da sociedade, utilizando estas categorias como argumentos do racionalizador. Portanto a ideia não é criar uma taxonomia como outras ontologias de domínio, mas utilizar estas categorias como uma rede de dependências para auxiliar a avaliação de situações do mundo real.

Então, para o racionalizador, o ciclo começa a partir de uma situação de mundo real, onde é interpretada em uma descrição abstrata, processada em termos utilizados pelas fontes legais. A descrição abstrata é denominada como situação legal e o conhecimento utilizado nesta etapa é o conhecimento de mundo (formando o modelo legal abstrato).

A situação legal é confrontada com o conhecimento normativo para verificar a violação de normas, produzindo a classificação da situação (permitida ou não permitida). A situação é analisada com o conhecimento de mundo para verificar qual agente a causou (componente sobre causas). Estas informações são utilizadas como entrada

para o conhecimento de responsabilidade determinar qual agente assegura responsabilidade pela situação. Todos os resultados destas computações são utilizadas, por sua vez, como dados de entrada para a função que define a possível reação legal (usando conhecimento reativo). Ainda, neste mecanismo, a lei pode gerar uma entidade abstrata, pertencente ao sistema legal, usando o conhecimento criativo[3].

BREUKER et al. (2005) denotam que FOLaw está mais para um *framework* epistemológico do que uma ontologia concreta. Quando se refere às ontologias concretas, são formalizações conceituais sobre um domínio e não um ambiente para racionalização com evidências, hipóteses, entre outros. É comum a relação entre *framework* epistemológico e ontologias pois é comum que esta última forneça dados para a primeira.

Analisando trabalhos relacionados, os autores verificam que o domínio legal não parece possuir vocabulário específico. As análises relatam que é difícil separar totalmente vocabulário legal de atividades sociais. As noções de papéis, posição social, ações e relações sociais determinam importantes termos no domínio da lei positiva (entrando em concordância com a teoria de Kelsen e outros trabalhos citados anteriormente).

Além dessa predominância, os autores descrevem que as propriedades de relacionamentos, danos e individualidades são também recorrentes na lei. Com isso, determinam que invariavelmente os conceitos legais são fundados em cima do senso comum. Isto faz com que seja necessário, ao se modelar e entender algum domínio legal, incluir termos como agentes, ações, processos, tempo, espaço, entre outros.

Neste sentido, descrevem que aparenta ser necessário definir uma ontologia fundamental em cima de uma ontologia base (de termos típicos da área legal) porque os conceitos de leis estariam espalhados por quase todo o conjunto de conhecimento denominado de senso comum. Desta forma, propuseram a LRI-Core, onde a sua concepção contém duas camadas: i) a camada superior – de fundação; e, ii) a camada inferior – dos conceitos legais (BREUKER; VALENTE; WINKELS, 2005).

A camada superior define conceitos abstratos legais, ou seja, aquilo que oriunda do senso comum, como: conceitos mentais sobre intenção e conteúdo, papel, conceitos sociais e, concepções físicas de objeto e de processos. A camada inferior expressa a ontologia legal através dos conceitos anteriores, definindo ação e documento, norma e organização, organização judicial, ação legal e código legal. Através destas concepções e destas duas camadas, a ontologia é definida na Figura 1.

[3]Uma espécie de entidade suporte aos agentes legislativos, talvez como um agente policial é suporte do agente legislador.

Figura 1 – As duas camadas principais da ontologia LRI-Core.

A Figura 1 define a proposta LRI-Core onde são visualizadas as duas camada superiores. Para a sua utilização em alguma modelagem jurídica, os conceitos específicos de domínio devem ser extensões dos conceitos mais a direita da Figura (a camada superior fica mais à esquerda da Figura). Assim, esta se torna aplicável como ontologia de base e não como concepção epistemológica mais associada a mecanismos de racionalização[4].

DOLCE e DOLCE-CORE também são opções dentro do cenário de ontologias fundacionais, cuja visão é de que ontologias universais para representação do conhecimento não podem ser viabilizadas (BORGO; MASOLO, 2009; GANGEMI et al., 2002). Não são ontologias específicas para modelagens jurídicas, mas são encontradas nos mesmos estudos, referenciadas como ontologia base para definições lógicas e de construção de conceitos para quaisquer áreas. Estas também não fornecem a taxonomia pronta pra uso, mas a construção lógica para aplicar em um domínio a fim de gerar a ontologia-fim.

Neste sentido, estas ontologias pretendem fornecer uma perspectiva ontológica clara (um guia de modelagem), uma aplicação correta para os domínios (com descrições explícitas de suas suposições básicas sobre conceitos), além de manter interoperabilidade com outros sistemas ontológicos.

A DOLCE (GANGEMI et al., 2002) reside também na mesmas perspectivas de capturar suas categorias através da análise linguística e do senso comum possibilitando a análise da natureza social dos domínios. As categorias definidas nesta ontologia são relacionadas diretamente com percepções humanas, impressões culturais e convenções sociais.

A Figura 2 representa as principais estruturas superiores da taxonomia da DOLCE. As entidades de *endurant* e *perdurant* relacionam a durável e perdura (preserva ao longo do tempo), respectivamente. As entidades *endurant* são descritas como classes equivalentes a de perduro. A diferença entre estas entidades é o seu comportamento no tempo.

Quando se lê "entidades" a interpretação deve ser de "conceitos que devem ser estendidos" que não geram indivíduos[5]. Isto ocorre para a maioria dos estudos de ontologias fundacionais (ou base), devido aos seus autores determinarem que os conceitos dos seus modelos podem não gerar indivíduos e, sim, para gerar ou classificar outros conceitos do domínio de aplicação.

[4] Ainda que seja possível fornecer mecanismos de inferência com esta ontologia

[5] Indivíduos e instâncias devem ser considerados como sinônimos nestes estudos sobre ontologia, quando for referido sobre os indivíduos sociais, estes serão referenciados por agentes (ainda assim deve-se considerar o contexto do termo empregado).

Figura 2 – As principais categorias da taxonomia da DOLCE.

As entidades de *endurant* estão totalmente presentes nos tempos onde estas são apresentadas. As entidades *perdurant* se estendem ao longo do tempo sendo que em um dado momento bem específico, estão parcialmente presentes[6]. Esta também recebe a denominação de ocorrências, que por sua vez geram os conceitos de eventos e estados. As ocorrências compreendem os conceitos variáveis sobre eventos, processos, fenômenos, atividades e estados.

As qualidades são os conceitos mais básicos que pode-se perceber e medir (formas, cores, tamanhos, sons, entre outros). Segundos os autores, em outras taxonomias, qualidade pode ser utilizada como sinônimo de propriedade, mas não é o caso desta ontologia (GANGEMI et al., 2002). As qualidades são particulares de uma entidade, como o cheiro e a cor. As propriedades tendem a ser universais e podem constar em mais de uma entidade (que as qualidades não). É possível definir as qualidades como características específicas de uma entidade em um sentido mais restrito (não existindo em outras entidades). Se um indivíduo possui a mesma cor de outro, quer dizer que no conjunto de cores estes possuem o mesmo espaço de definição, mas os indivíduos possuem cores distintas (numericamente distintas). O espaço e tempo também são considerados tipos de qualidades.

O termo substância (*substantial*), designa objetos que podem estender de substâncias físicas e não físicas, caracterizando o prelúdio lógico para a construção de conceitos sobre objetos de agentes físicos (como uma pessoa), objetos físicos não agentes (uma casa, um martelo), objetos sociais não físicos e agentivos (como a própria lei, sistema econômico, a moeda, entre outros), objetos sociais agentivos como o agente social (pessoa jurídica) e sociedade (instituições, bancos, empresas).

Os objetos são unidades que não dependem de outras unidades, são independentes de ocorrências (eventos) e podem ter partes temporárias. Os objetos que são chamados de agentivos são os que não formam os não agentivos, isto é, o objeto agentivo é formado por não agentivos. Neste sentido, um exemplo desta relação são “as pessoas são constituídas de organismos”. Agentes sociais não são constituídos por objetos físicos agentivos (mesmo que dependa deles), mas podem constituir sociedades (instituições, empresas) (GANGEMI et al., 2002). Objetos sociais não agentivos (isto é, leis, tratados de paz, entre outros) dependem da sociedade para a sua criação.

A ontologia DOLCE-CORE (BORGO; MASOLO, 2009) limita-se a entidades que existem no tempo[7], onde estas são definidas como entidades temporais que podem ser criadas, adotadas, abandonadas, entre outros. Diferentemente ocorre na DOLCE, regiões e espaços são entidades abstratas (fora do tempo e espaço). Na sua concepção lógica, as entidades existem em diferentes tempos, são persistentes através do

[6]Os autores denotam como exemplo o próprio artigo, o documento está totalmente presente mas algumas partes não estão, estão em partes (as citações).

[7]Particularidades temporais, em tradução direta do artigo referenciado.

tempo.

O aspecto estrutural de particularidades temporais é dividido nas seguintes categorias: objetos, eventos, qualidades individuais, regiões, conceitos e somas arbitrárias. Estas categorias são rígidas, de modo que as entidades não podem alterar de categoria durante uma linha de tempo (tempo de execução). Estas categorias são formalizadas e refatoradas através da lógica de predicados. A principal contribuição é melhorar o sistema ontológico do modelo DOLCE mas não define maiores alterações na taxonomia apresentada na Figura 2.

Ainda nas ontologias fundamentais, são referenciadas as ontologias baseadas na UFO (*Unified Fountational Ontology*), construída por lógica filosófica, filosofia da linguagem, linguística e psicologia cognitiva (GUIZZARDI; FALBO; GUIZZARDI, 2008). Esta ontologia é dividida em três fragmentos: UFO-A, UFO-B e UFO-C.

A UFO-A é o núcleo da ontologia, sendo basicamente para modelar a natureza das entidades. A modelagem é realizada inicialmente pela tipificação do termo *Entity* (entidade) entre os conceitos de *Particular* e *Universal*. O *Particular* registra entidades que possuem na realidade uma identidade única enquanto *Universals* definem padrões e características dos diferentes *Particulars*.

Depois segue com a modelagem de outras expressões como *Moments* (indivíduo que existe somente em outros indivíduos como cores, conexões, sintomas) e *Relations* (entidades que conectam entidades). No *Relation* as propriedades fornecem uma base para determinar relacionamentos legais, onde o indivíduo poderá receber relações legais se participar de uma relação que possa derivar tais definições. A UFO-A também define o conceito de *Situations* que também segue na lógica do conceito de *endurant* (como em outros modelos comentados anteriormente), modelando estados das coisas. Todo esse núcleo é disponibilizado como estruturas para desenhar e modelar o conhecimento em um nível mais filosófico.

A UFO-B é distinta da UFO-A em relação a temporalidade dos indivíduos. Enquanto na UFO-A as modelagens tem como referência conceitos *endurant*, na UFO-B são referenciados os *perdurants*, isto é, indivíduos que são compostos de partes temporais, existindo em um período de tempo específico. Conceitos como *Starts*, *Meets*, *Before*, *Event*, *TimePoint* são aplicados na mesma base da UFO-A e ajudam a denotar a temporalidade de um processo, como por exemplo um “jogo de futebol” com os seus inícios e finais dos tempos.

A terceira camada, chamada de UFO-C, é construída em cima da UFO-A e UFO-B e define as entidades sociais. Nesta camada são distinguidos e modelados os *Particulars Agentive* e *Non-agentive*. Com isto são definidos os conceitos de *Agents* e *Objects*. Os *Agents* são agentes físicos (pessoas) e sociais (uma organização, uma sociedade). Os *Objects* também se dividem entre físicos (um livro, um carro) e sociais (dinheiro, idioma e descrições normativas).

As descrições normativas nesta ontologia definem uma ou mais normas reconhecidas, pelo menos, por um agente social que define *Universals*. Neste sentido, um agente define um contrato social (momento universal), objetos sociais (senado ou câmara dos deputados) e papéis sociais como o de presidente, primeiro ministros e até mesmo pedestres. As legislações nesta ontologia são modeladas a partir desta terceira camada.

Os demais conceitos da UFO-C também se relacionam com os agentes, como as definições de *Intentional Moment*, onde definem-se os conceitos de crenças, desejos, intenções e outros conceitos como as ações (*Action*) que contenham planos de ações ou atos comunicativos dos agentes e os seus objetivos (*Goal*).

A UFO é apresentada como um modelo de avaliação e de redesenho sobre semânticas do mundo real dentro de um processo de engenharia de software. Através de fundamentos filosóficos fornece uma infraestrutura para lidar com variações semânticas entre ontologias e linguagens de construção de modelos. Cada camada, da UFO-A para UFO-C, determina um tipo de modelagem avaliando conceitos e seus tipos, de algum domínio de aplicação.

A LKIF (*Legal Knowledge Interchange Format*) é uma proposta de ontologia cujo objetivo é fornecer uma tradução entre bases legais escritas em diferentes representações e formalismos (ALEXANDER, 2009). Além de ser, por si só, uma representação formal de conhecimento para processadores de raciocínio em serviços de sistemas legais.

Esta proposta utiliza de estratégias de avaliações cognitivas, análises de textos jurídicos e concepção de conceitos legais básicos. Para representar estes conceitos básicos legais, reúne a compreensão legislativa de três públicos: cidadãos, profissionais da área jurídica e acadêmicos da área jurídica. Assim, estudam cada forma de expressão utilizada pelos interessados destes públicos e evidenciam os conceitos que fazem mais sentido e que possuem maior frequência no meio jurídico.

São realizadas pesquisas com cada parceiro, representando cada tipo de público, para que estes forneçam uma lista de 20 conceitos mais relacionados a área. Posteriormente é realizada uma apuração e suas correlações, resultando em uma lista de 250 termos. A lista é retornada aos pesquisados para avaliações de nível de abstração, relevância para o domínio legal, grau de expressividade legal ou de senso comum, grau de que um termo é um termo legal (não apenas de um subdomínio legal e sim mais generalista) e grau de importância para a ontologia.

Todos estes critérios conduzem para uma pontuação sobre os termos. Após as avaliações, o processo gera uma lista de conceitos mais pontuados onde os 50 primeiros termos serão considerados para com a LKIF-Core. A expressão *Core* (núcleo) define o caráter desejado para o modelo: servir como ontologia de referência para termos comuns, senso comum, não específicos a um domínio de aplicação.

Os 15 conceitos mais pontuados de acordo com a sua importância são: *Law*, *Right*, *Jurisdiction*, *Permission*, *Prohibition*, *Rule*, *Sanction*, *Violation*, *Power*, *Duty*, *Legal Position*, *Norm*, *Obligation*, *Permissive Right* e *Argument* (ALEXANDER, 2009).

Apesar deste processo, os autores conduziram avaliações posteriores pois indicam que mesmo que um termo e suas relações tenham alto grau de importância pode não ser relevante sua inserção na ontologia. A utilização dependerá da intenção do que é definido pelo conceito, que realmente possa formalizar como nós interpretamos o mundo. Então usam de metodologias para definir que a aceitação de um conceito e suas relações são permitidas se suas combinações são sistemáticas e possuem contexto independente com os demais.

A organização dos conceitos elencados pelo estudo, são definidos através de 10 módulos (*expression*, *norm*, *process*, *action*, *role*, *place*, *time*, *mereology*, legal_action, *legal_role*) e dois *frameworks* (*time_modification* e *rules*). Cada módulo contém os conceitos e suas relações de acordo com as camadas superior, intencional e legal. Cada camada se relaciona a instância desejada para a elaboração, ou seja, qual o domínio de discurso que está sendo focado para o uso com a LKIF[8].

Na camada superior constam os conceitos mais primitivos reutilizados por todos, fornecendo noções de ocorrências temporais, localização (mundo físico), como: *Process*, *Mereology*, *Time* e *Place*. Na camada de intenção residem os conceitos relacionados ao comportamento, sendo a atitude definida em *Action* executada através de um papel em particular (*Role*). Sendo que é definido quem são potencialmente os responsáveis pelos efeitos causados pelas ações (processos de reação, criação e reação definidos na ontologia).

A forma como ocorre as atitudes dos agentes é externalizada pelas suas comunicações e o conceito *Expression* se responsabiliza por estas definições, relacionando com outros conceitos como *Statement* e *Declaration* que são considerados os meios pelos quais ocorrem os atos comunicativos.

A última camada nomeada como legal, define um local onde formaliza o conhecimento sobre as afirmações normativas. Estas afirmações reúnem mais conceitos sobre os agentes pelo termo *Agent* (seguido pelos tipos *Natural_Person* e *Legislative_Person*). Através de cada tipo de agente e de seus mandatos legais ou delegações pela sociedade, os direitos e poderes são formalizados ao longo do restante da ontologia.

Os autores descrevem que as capacidades de crenças e desejos dos agentes devem ser conceituados nesta camada, uma vez que em sua formalização esta ramificação de conceitos legais ocorre através do conceito *Mental_Object*. Esta ramificação atravessa conceitos como *Norm* (com seus sub-conceitos de *Right* e *Permission*).

[8]Leis físicas, como leis que expressam a gravidade, ou outras leis da sociedade que descrevem regulações entre indivíduos e o poder.

Sendo que no *Permission* existem ainda outros sub-conceitos que são *Obligation* e *Prohibition*.

Ao final do estudo os autores demonstram através de artigos de lei a aplicação dessa ontologia. Cada artigo de lei deve ser compreendido como uma afirmação normativa e deste, sentenças lógicas definidas por OWL 2[9] (que é uma versão mais aprimorada da linguagem de ontologias para a *web* semântica).

2.1.2 Ontologias Legais e SMA

Apesar das variadas aplicações de ontologias na área de SMA, o foco desta Seção dar-se-á em ontologias normativas, ou aplicadas na regulamentação comportamental dos agentes. Em FREITAS (2017), apresenta-se a utilização de uma ontologia como meta-modelo do SMA, tendo como alvo principal o ambiente JaCaMo[10] de desenvolvimento de SMA. Nesta ontologia, chamada de OntoMAS, são definidos os conceitos sobre quem são os agentes, suas crenças, missões, planos, normas, regras e relacionamentos entre estas estruturas.

Neste estudo, devido a organização da ferramenta de simulação ser dividida entre dimensões de agente, ambiente e organização, o autor propõe ontologias para cada uma destas. Na dimensão de organização, é onde os aspectos normativos são estruturados. A Figura 3 exibe a ontologia desta dimensão.

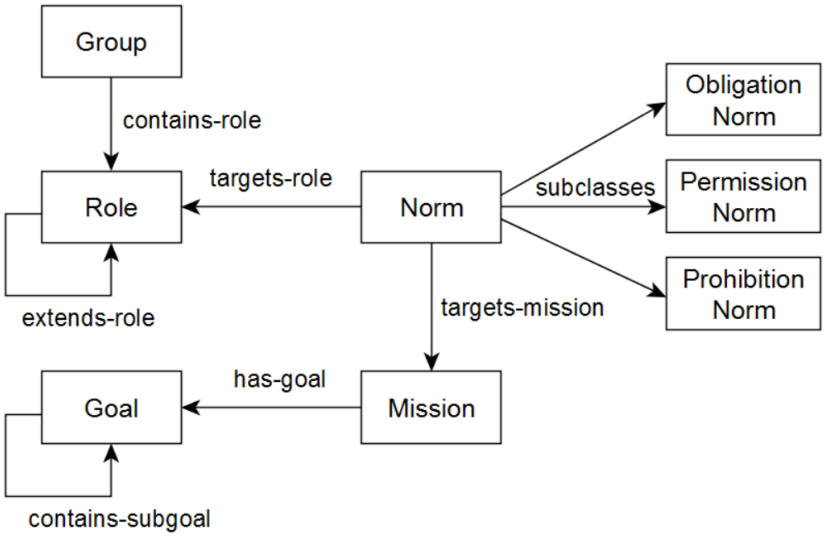


Figura 3 – Ontologia OntoMAS na dimensão organizacional proposta por FREITAS (2017).

Na OntoMAS, o conceito *role* (papel), define a função que um agente pode adotar e o que deve obedecer, respeitando as restrições que estiverem vinculadas ao papel. Como faz referência aos papéis extraídos do ambiente Moise[11] (que trabalha com

[9] https://www.w3.org/TR/owl2-overview
[10] http://jacamo.sourceforge.net
[11] http://moise.sourceforge.net

a organização dos agentes) da plataforma JaCaMo, o autor define que não existe necessidade de criar sub-conceitos (ou subclasses) deste conceito.

Os grupos, objetivos e missões também são inerentes à plataforma e descrevem a organização de um sistema multiagente nas suas atribuições e resoluções de problemas. Conforme visualizado na Figura 3, uma missão possui objetivos (e estes sub-objetivos) e os papéis dos agentes podem ser agrupados no conceito de grupo.

As normas em OntoMAS seguem como regulação na organização de comportamento dos agentes. Neste caso podem ser um dos três diferentes tipos. Uma *ObligatioNorm* é uma norma que precisa ser feita/concluída. A *PermissionNorm* é uma norma que é permitida (não proibida), enquanto a *ProhibitionNorm* é algo definitivamente proibido no sistema.

Na Figura 3 também é possível visualizar os predicados ou propriedades entre os objetos onde, por exemplo, é indicado que uma norma está relacionado a missões e a papéis. A ideia é reunir os níveis estruturais com os funcionais da plataforma, onde posteriormente é desenvolvido sob um *plugin* para a mesma plataforma.

O *plugin* denominado OntoJaCaMo, interpreta as instâncias da ontologia e gera código para a plataforma JaCaMo (será descrita melhor na Seção 4.1), respeitando a codificação para cada uma das unidades envolvidas na infraestrutura (agentes, organização e ambiente).

Em outros estudos, existem modelos que tentam cobrir modelagens de organizações de agentes, onde instituições são formalmente modeladas (DIGNUM; VÁZQUEZ-SALCEDA; DIGNUM, 2005). A ontologia é utilizada como descritor do conteúdo e da linguagem de comunicação dos agentes. Neste caso, os estudos focam na especificação formal e na organização de agentes, definindo um meta-modelo de ontologia. Neste meta-modelo são estruturados os conceitos do *framework* sobre normas, regras, papéis, grupos, violações e sanções.

As chamadas ontologias normativas, exploram estas questões de modelar regras que, posteriormente, podem se tornar normas dentro de um SMA. No estudo apresentado em FELICíSSIMO et al. (2005), propõe-se uma ontologia base, onde as ontologias de domínio devem estendê-la para refletir as intenções normativas. A ontologia pode ser visualizada na Figura 4.

A Figura 4 descreve os conceitos de normas, papéis, penalidade, lugar, obrigação, proibição, permissão e ação. O conceito papel possui relação com várias instâncias de normas e uma relação com local (onde é válido o papel para com a norma).

O conceito de norma possui atributo denotando se está ativo ou não (se a norma está válida), relação com uma instância de penalidade. A norma também regula as ações possíveis instanciadas nesta ontologia. As normas nessa ontologia possuem três sub-conceitos: obrigação, proibição e permissão.

O conceito de penalidade possui os três sub-conceitos conforme são visualizados

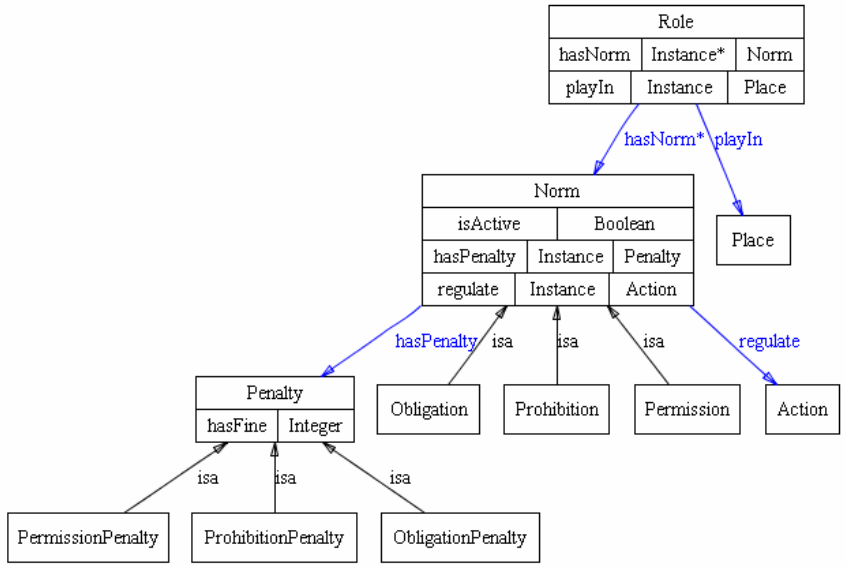


Figura 4 – Ontologia normativa proposta por FELICíSSIMO et al. (2005).

na Figura 4, a ideia dos autores é o de tipificar ou categorizar as ações que violam as normas, mas o estudo em FELICíSSIMO et al. (2005) não fornece maiores detalhes destes e dos motivos que justificam os sub-conceitos do conceito norma.

A representação de conhecimento humano nas ontologias, expressando funcionalidades legislativas, é o centro das análises de VALENTE; BREUKER (1994). A ontologia funcional de leis abrange categorias de conhecimento normativo, responsabilidade de causa e conhecimentos de mundo, reativo, posição e criativo.

Todas essas categorias fornecem funcionalidades de normas de empoderamento, comando e ainda responsabilidades causais e legais. A ideia dos autores é preconizar um sistema legal acoplado acima da sociedade de agentes onde, através de mecanismos de raciocínio da ontologia, forneça as estruturas legislativas para aquela sociedade.

Seguindo as pesquisas nas ontologias alinhadas aos aspectos de normas e regulamentações, existem outros modelos que são extensões daquelas ontologias fundamentais descritas anteriormente. Uma extensão da UFO focada em aspectos legais é a LCO (*Legal Core Ontology*) que propõe uma camada acima das demais da UFO, chamada UFO-L (GRIFFO; ALMEIDA; GUIZZARDI, 2015).

Em contraste aos modelos anteriores, que referenciam a Teoria da Lei de Kelsen, este modelo utiliza a Teoria dos Direitos Fundamentais de Alexy (ALEXY, 2010). A contraposição definida pelos autores é que o Positivismo Legal de Kelsen e as expressões dadas por Alexy acabam gerando modelos diferentes para um mesmo domínio e modelos baseados em Kelsen não introduzem conceitos modernos de leis na sociedade.

Inclui também conceitos sobre normas mas as classifica em regras e princípios. A justificativa dessa referência, segundo os autores, é que Alexy fornece os princípios

como graus de satisfação ao invés de regras que são cumpridas ou não, além de utilizar trabalhos anteriores sobre definição do que é uma relação legal.

O modelo UFO-L estende os conceitos definidos na UFO para formalizar uma base conceitual legal que possa ser reutilizada em diversos outros domínios de ontologias e bases legais de conhecimento. Nesta extensão são adicionados e aperfeiçoados os conceitos de relações legais com os direitos e deveres dos agentes.

Uma relação legal é um vínculo entre duas entidades através de um fato legal e é tipificada em uma norma legal, introduzindo no modelo o conceito de *Legal Relator*. Este conceito é derivado do *Social Relator* definido na UFO onde um relator social é um relator composto de dois ou mais pares de momentos sociais associados (contratos sociais) (GUIZZARDI; FALBO; GUIZZARDI, 2008). Desta forma um relator legal é uma especialização do relator social mas que depende de outros indivíduos que atuam com papéis legais.

Cada relator legal é dividido entre *Simple Legal Relator* e *Complex Legal Relator*. O *Simple Legal Relator* é baseado nas teorias de Alexy e representa os direitos sobre alguma coisa e utiliza pares de conceitos fundamentais legais como *Right-Duty* e *No-right-permission* (que neste caso são sub-conceitos de Simple *Legal Relator*).

O *Right-Duty Relator* define as relações legais entre os vínculos de titular do direito e o detentor do direito. O titular de direito é aquele que possui o direito em algo contra um dever (cidadão tem o título de direito a votar contra o Estado) enquanto o detentor do direito é alguém que tem o dever de materializar o direito do titular de direito que, neste caso, é o próprio Estado (GRIFFO; ALMEIDA; GUIZZARDI, 2015).

No conceito de *No-right–permission Relator* utiliza-se a relação legal que conecta quem possui o detentor da permissão e o permissor. O detentor da permissão é alguém que tem a permissão recebida do permissor. Um exemplo, segundo os autores, é dada pela expressão “permissão em fumar em ambientes abertos”, onde o detentor da permissão é o fumante e o permissor é o Estado.

A ontologia legal PrOnto, alinhada às questões de direitos e deveres, busca modelar a GDPR[12] (*General Data Protection Regulation*) para reutilização e aplicação do conhecimento no domínio da privacidade (PALMIRANI et al., 2018). A GDPR é um *framework* de proteção de dados aplicado pela União Europeia e regulamenta os princípios legais de seus variados Estados membros, no seu chamado Mercado Digital Único[13].

A regulamentação é aplicada nos processos que envolvem dados pessoais, definindo uma série de obrigações para os envolvidos neste processo. As entidades envolvidas naqueles processos visam avaliar os riscos e adaptar suas funções com

[12] `https://gdpr-info.eu`

[13] A ideia é reunir todos os mercados digitais dos Estados membros em um único lugar, removendo barreiras tecnológicas e facilitando as trocas comerciais e acesso aos serviços (com segurança e qualidade).

base nestas avaliações de impacto, especificando medidas para a salvaguarda dos sujeitos e os direitos fundamentais (dignidade) destes mesmos (Artigo 35 da GDPR). Também especifica instrumentos tais como auditorias e verificação de conformidade a fim de garantir a aplicação dos princípios de proteção de dados (Artigo 25 da GDPR).

A ontologia legal PrOnto é criada através da modelagem dessa legislação e formaliza as normas de proteções de dados. Chamada de ontologia da privacidade, PrOnto possui em seu núcleo os conceitos de *data types*, *documents*, *agents*, *roles*, *processing purposes*, *legal bases*, *processing operations*, e operações deônticas para modelar *rights* (direitos) e *duties* (deveres). Todos estes conceitos são organizados por módulos que agrupam as diferentes modelagens que dependem da intenção, ou seja, quando se referem às pessoas envolvidas como em *agents* ou quando a modelagem se refere aos processos legais (neste caso os artigos de lei que motivam as regulamentações e que vão permear entre *processing purposes* e *legal bases*).

A ontologia faz análise das listas de direitos sobre dados por toda a GDPR: direito de acesso, direito de ser esquecido, direito à portabilidade, direito de apagar, entre outros. Neste sentido, realiza a modelagem através de operadores deônticos tais como direito, obrigação, proibição e permissão entre outras customizações particulares da GDPR para obrigações e direitos. O módulo que trata especificamente disso determina um conjunto de predicados para implementar regras legais através da LegalRuleML (ATHAN et al., 2015) utilizada como linguagem para modelar a ontologia.

Estes predicados na PrOnto definem os relacionamentos de direitos e obrigações dos atores envolvidos, permissões e violações ou conformidades. Incluem também um *status* sobre se as obrigações ou proibições são violadas ou mantidas. Todos estes itens são requisitados pela GDPR e são implementados na ontologia para que seja possível realizar consultas como “me retorne todos os processos de atividades que têm sido violadas por alguns atores em um dado momento”. De acordo com este exemplo também são definidos na ontologia parâmetros temporais e de jurisdição em consideração aos direitos que são efetivos somente em uma regulação doméstica específica.

Cada obrigação possui um direito conectado com os dados do titular chamado *Bearer*. O conceito que relaciona o *Bearer* para com as obrigações e direitos é o *DeonticSpecification* que também agrega o relacionamento do conceito de tempo (Interval), *LegalRule* (que traz a formalização da legislação GDPR para a ontologia) e *AuxiliaryParty* (subclasse de *Roles* ou papéis que são apontados pela especificação).

A proposta de EL GHOSH et al. (2016) também analisa legislações na construção de ontologias legais implementando uma ontologia no domínio da legislação Libanesa. A definição da construção desta ontologia ocorre através de Processos de Modelagem Conceitual (estratégia *top-down*) e Processo de Aprendizado de Ontologia (estratégia

bottom-up). Nesta proposta, os autores apresentam uma ontologia de referência de domínio e não uma ontologia operacional (implementada para utilização).

Após realizarem um levantamento de estratégias para a construção de ontologia, definem que esta será organizada em camadas (superiora, núcleo e de domínio) utilizando como referência ontologias fundamentais tais como: DOLCE, UFO, LRI-Core e LKIF-Core. Além disso, reutilizam conceitos da UFO e da LKIF-Core tais como: *Agent*, *Intentional_Moment*, *Action*, *Event*, *Normative_Description*, *Goal*, *Situation*, *Medium*, *Document*, *Legal_Source*, *Legal_Document* e *Code*.

Após esta organização, para a construção *top-down*, seguem para processos *bottom-up* utilizando metodologias de aprendizado de ontologia e técnicas semi-automatizadas de Processamento de Linguagem Natural (PLN). A última etapa é a definição da modelagem do conhecimento em um domínio específico ou não, resultando nos módulos finais da ontologia.

Na camada superiora desta arquitetura são definidos os conceitos mais gerais e as relações que cobrem todos os domínios desejados. O conceito *Act* (ação) é exemplo de um conceito que consta em quaisquer domínios. Na camada de núcleo residem as estruturas de conhecimento específicos sobre o domínio legal. Um *Legal-Act* é um conceito que representa qualquer ação legal em todos os domínios legais (civil, penal). A camada de domínio descreve a conceitualização do domínio desejado, que neste caso é a legislação penal. Nesta camada de domínio tem como conceito *Infraction* (infração) que representa especificamente uma ação para o sistema penal. Na camada mais inferior, de aplicação, se localizam as instâncias da infração ou de outros conceitos da camada de domínio. Os conceitos *Crime*, *Offense* e *Violation* são instâncias de *Infraction* (sendo que instâncias aqui se referem a sub-conceitos).

Aqueles conceitos na última camada de aplicação são oriundos de técnicas semi-automáticas de processamento de linguagem natural. No estudo em EL GHOSH et al. (2016), foram extraídos dos 300 artigos de lei penal Libanês cerca de 1200 termos e foram verificadas a alta frequência dos termos que resultaram na escolha destes como conceitos. Assim, a proposta dos autores do estudo é entregar um processo de construção de ontologias com diferentes estratégias a fim de reutilizar conceitos legais existentes em outras ontologias e minerar os textos jurídicos da nação no qual pertencem para fornecer uma ontologia legal base.

O modelo proposto por SANTOS; COSTA (2012) fornece um sistema de políticas públicas para os SMA baseados na plataforma JaCaMo. As políticas públicas neste modelo são definidas através do conjunto de normas e planos. Na estrutura do modelo, as normas são categorizadas como normas de obrigação e normas de proibição, onde se descrevem as políticas endereçadas a todos os tipos de agentes dentro da simulação. Nos planos de ação procura-se estruturar ações aos agentes do governo para executarem determinadas atividades relacionadas ao cumprimento das normas

pela sociedade. Ainda, o trabalho de SANTOS; COSTA (2012) não utiliza uma fonte externa (uma ontologia, por exemplo) para fornecer o conhecimento. O conhecimento é estruturado através de artefatos nativos da plataforma onde opera os agentes.

Em MOREIRA; COSTA; AGUIAR (2017) é apresentado um modelo aplicando SMA para desenvolver uma camada de aplicação de legislação em ambiente virtual de ensino. Mais especificamente, sobre legislações na área de ensino à distância, com a ajuda dos agentes é proposta a definição de um modelo que incorpora agentes e legislação ao ambiente de ensino virtual. O objetivo é o de auxiliar professores, tutores, coordenadores e equipes multidisciplinares em variadas tarefas reguladas por legislações existentes. Para isso, trata das definições de sistema jurídico também inspiradas por KELSEN (2009), definições de normas, validações de normas e sobre o sistema normativo de agentes. Neste estudo em específico não realizam a modelagem da legislação por ontologias e sim por diagramações de possibilidades de ações dos agentes. Esse modelo é aplicado em um ambiente virtual de ensino (Moodle) com o objetivo de verificar a conformidade do papel assumido com as ações executadas no sistema, com apoio das interações entre os agentes no SMA acoplado.

Tabela 1 – Comparativo entre trabalhos sobre ontologia e legislação.

Trabalhos	Normas	Funções Legais	Elementos de Conceito	Conhecimento reativo	Ontologia	Implementação	SMA
FOlaw (VALENTE; BREUKER; BROUWER, 1999)	Sim	Sim	Não	Não	Não	Outros	Não
LRI-Core (BREUKER et al., 2005)	Não	Sim	Sim	Não	Sim	Não	Não
Dolce (GANGEMI et al., 2002)	Não	Não	Sim	Não	Sim	Não	Não
Dolce-CORE (BORGO; MASOLO, 2009)	Não	Não	Sim	Não	Não	Não	Não
UFO (GUIZZARDI; FALBO; GUIZZARDI, 2008)	Sim	Sim	Sim	Breve	Sim	Não	Não
LKIF (ALEXANDER, 2009)	Médio	Sim	Sim	Não	Sim	Ontologia	Não
FREITAS (2017)	Sim	Não	Não	Não	Sim	Ont./Outros	Sim
DIGNUM; VÁZQUEZ-SALCEDA; DIGNUM (2005)	Sim	Não	Não	Não	Sim	Ontologia	Sim
FELICíSSIMO et al. (2005)	Sim	Não	Não	Não	Médio	Ontologia	Sim
VALENTE; BREUKER (1994)	Parcial	Sim	Não	Sim	Sim	Não	Não
GRIFFO; ALMEIDA; GUIZZARDI (2015)	Médio	Sim	Sim	Não	Sim	Não	Não
PALMIRANI et al. (2018)	Sim	Sim	Sim	Sim	Sim	Não	Não
EL GHOSH et al. (2016)	Não	Sim	Sim	Não	Parcial	Não	Não
SANTOS; COSTA (2012)	Sim	Não	Sim	Sim	Não	Outros	Sim
MOREIRA; COSTA; AGUIAR (2017)	Sim	Não	Sim	Não	Não	Outros	Sim

Na Tabela 1 apresenta-se um sumário consolidando as principais características dos trabalhos relatados nesta Seção, comparando os estudos de ontologias legislativas ou outras abordagens de legislação em SMA. Os critérios utilizados na comparação visam conceber o nível de atendimento daqueles estudos e são descritos abaixo:

Normas: cria ou estende o conceito de normas, para que dentro da ontologia se determinem mecanismos de controle/restrição de comportamento dos agentes. Caso o estudo atenda este critério, provavelmente realiza a modelagem de normas em detalhes. Os valores são de acordo com o atendimento e detalhamento as normas de orientação a possíveis agentes: Sim, Não ou Parcial (sem aprofundamento de conceituação e aplicação).

Funções legais: amparado por conceitos de teoria da lei, define categorias (ou funcionalidades) de conhecimento que são tipos de conhecimento que guiam a interpretação do conhecimento legal. Aqui também podem ser desenvolvidos conceitos sobre responsabilidade legal que especifica a conexão entre normas e agentes no sentido de definir a responsabilidade pela norma ou a culpabilidade pela violação da mesma.

Elementos de conceito: determinam elementos necessários para a aplicação do conceito na ontologia ou na programação dos agentes.

Conhecimento reativo ou criativo: se o estudo define conhecimentos de reação, sobre reagir a determinadas sanções, recompensas ou até mesmo criação de departamentos legais através da criatividade de algum agente.

Ontologia: se fornece uma ontologia para a sociedade de agentes para racionalização sobre o conceito de lei. Os valores podem ser Sim, Não ou Parcial.

Implementação: possui alguma implementação, seja em ontologia ou em outro componente. Estes componentes devem ser artefatos de software para implementar o modelo/estudo nos agentes. Os valores aqui podem ser Não, Ontologia ou Outros. O valor Outros denota que os pesquisadores desenvolveram outros artefatos de software que não é uma ontologia. Este critério visa se diferenciar do anterior uma vez que possam existir estudos sobre formalização semântica de ontologias sem a definição tecnológica[14].

SMA: indica o uso em simulações de agentes, através de uso direto de ferramentas de SMA ou como estes podem se beneficiar do estudo.

Fez-se necessária a separação das ontologias legais a partir das ontologias fundamentais (apresentadas na Seção anterior) pois o interesse desta pesquisa está sobre ontologias que modelam legislações. Durante o levantamento do referencial, notou-se que existe o primeiro grupo de ontologias que fornecem modelos bases ou fundamentais, ou ainda, de núcleo, denotando expressões comuns do meio legal, formando ontologias de uso geral. Nesta Seção, foram apresentadas ontologias que em sua maioria modelam domínios específicos e fornecem exemplo para aplicações de ontologias legais.

Existem outras categorias de ontologias mas que não interessam a esta Tese, pois não residem no escopo legislativo. As ontologias aqui apresentadas, por sua vez, referenciam muitas outras, no objetivo de procurar resolver necessidades de formalização de legislações. Estas formalizações podem ser em sistemas legais de países específicos ou sobre teorias da lei, berço do estudo do Direito.

[14] Documento ontológico para uso por *software*.

2.2 *Middlewares*

Os sistemas distribuídos são desenvolvidos através de duas faces para uma mesma necessidade, ou para um mesmo conjunto de problemas a serem resolvidos. Um conceito inicial de um sistema distribuído é aquele em que seus componentes estão distribuídos em rede e se comunicam, se coordenam, através de mensagens (COULOURIS et al., 2011).

A inteligência artificial distribuída vem nesse viés para estudar como um grupo inteligente de agentes pode se combinar para resolver problemas globais (FERBER; WEISS, 1999). Os agentes precisam se comunicar para efetivamente realizar a cooperação da resolução de suas missões ou problemas. Além disso, os agentes precisam de informação sobre o seu meio e podem necessitar também de conhecimento adquirido por fontes externas, como o capítulo anterior procurou descrever.

Através destas concepções, observa-se ao longo dos anos uma crescente produção de componentes de *software*, intitulados agentes, que precisam dividir um problema complexo, maior, em problemas menores de melhor processamento ou resolução. Neste mesmo período, a indústria de software têm fornecido sistemas com diferentes tecnologias e que em variados momentos precisam se comunicar, fazendo com que "software pontes" sejam necessários para a interligação (BLAIR et al., 2011).

Esta Seção pretende revisar aspectos destas "pontes" reconhecidos por *middleware* e como estes têm sido relacionados em sistemas que precisam destas intercomunicações. Esta revisão irá se deter o tanto quanto possível em modelos envolvendo ontologia e agentes, ainda que outros modelos correlacionados a interoperabilidade de serviços semânticos possam também serem considerados. Os presentes estudos servem de base para as definições futuras dos aspectos de arquitetura do modelo proposto (Capítulo 3).

2.2.1 *Middlewares* e Ontologias

Inicialmente, em BLAIR et al. (2011) discute-se o desenvolvimento de sistemas heterogêneos e os desafios impostos nestes quando precisam realizar a interoperabilidade. Neste caso avaliam a postura de duas comunidades de pesquisadores que pesquisam e procuram resolver esta interoperação: na de *middleware* e web semântica.

Em relação à comunidade de *middleware*, estes se preocupam na confecção de padrões de software para realizar estas comunicações entre diferentes sistemas. Os *middlewares* e serviços web atuam como principais atores no desenvolvimento de sistemas desde sua distribuição. Os autores descrevem as capacidades destes componentes tentarem resolver os desafios de comunicação um-para-um (ou intermediário). A comunicação um-para-um visa conceber um componente intermediário entre duas

tecnologias diretamente, apenas para trafegar os dados. Os componentes intermediários são aqueles em que existe uma tradução entre os dados trafegados dos sistemas. Isto é, ao receber a comunicação de um sistema A para um sistema B, realiza a tradução de dados do A para um modelo de dados que B possa compreender.

Em relação a web semântica, os autores descrevem que *middlewares* semânticos são desenvolvidos como suporte para expressar recursos de serviços, as solicitações destes recursos e a interoperabilidade de dados. As ontologias atuam auxiliando na verificação da disponibilidade do serviço e da própria interoperabilidade de dados.

Mais focado em serviços web e integração entre diferentes fontes de dados, existem estudos e modelos que tentam resolver a comunicação entre diferentes sistemas. Um exemplo é o LRA (*Linked REST APIs*), que implementa um sistema que procura resolver problemas de serviços *web* que forneçam dados sobre autores e seus trabalhos acadêmicos (SERRANO et al., 2017). O problema reside nas diferentes fontes que fornecem dados sobre nomes e títulos de trabalhos acadêmicos através de diferentes expressões (denominações). Um exemplo são as expressões *name* e *title* que são utilizados para a mesma informação, que é o título de um artigo.

A fim de resolver esse problema utilizam a ontologia VIVO[15] para reconhecer automaticamente as diferentes expressões para um mesmo significado. Resolvido este problema de integrar diferentes fontes de dados, fornecem consultas SPARQL[16] como entrada para as pesquisas. Cada consulta procura fornecer uma entrada em alto nível para que usuários possam descrever o que gostariam de pesquisar nas diferentes fontes de dados.

O *middleware* proposto procura então serviços chamados APIs (acrônimo para Interface de Programação de Aplicação) REST[17] que, possam ser candidatos de fontes úteis para corresponder aos dados que estão sendo desejados para a pesquisa. De forma automática avalia com seus algoritmos internos e com a ontologia VIVO fontes de dados existentes pela web. Esta procura é chamada de descoberta de serviços que possam corresponder com a consulta SPARQL do usuário. O *middleware* possui uma lista de APIs que fornecem conjuntos de dados relacionados ao domínio em questão. O sistema proposto realiza o raciocínio do que se deseja na consulta SPARQL e faz inferências sobre cada terminologia encontrada. Estas inferências procuram resolver rótulos de dados que são diferentes mas apontam para os de mesmo significado para a pesquisa desejada inicialmente.

Em seguida, realiza os processamentos, raciocínios sobre os dados, agregação e por fim retornam dados ao usuário. Basicamente utilizam ontologias RDF[18] para

[15] `https://wiki.duraspace.org/display/VIVODOC110x/VIVO+Ontology+Domain+Definition`

[16] SPARQL é uma linguagem de consulta que opera em estruturas RDF de informações (`https://www.w3.org/TR/rdf-sparql-query`).

[17] `https://restfulapi.net`

[18] RDF é um padrão de modelo para intercâmbio de dados pela *web* (`https://www.w3.org/RDF`).

semanticamente fazerem anotações sobre APIs REST. Este modelo fornece uma solução interessante pois remove a necessidade de busca manual de fontes de dados pelo usuário, facilitando suas pesquisas por dados na web.

O projeto *Smart and Networking Underwater Robots in Cooperation Meshes* (SWARMs) fornece estudos sobre os desafios de comunicação entre veículos autônomos submarinos (LI et al., 2017). Os autores descrevem a importância da comunicação entre estes veículos para que possam realizar atividades de cooperação. Neste quesito da cooperação, os veículos podem ser construídos por diferentes fabricantes e no momento da comunicação podem ter problemas de significado. Por exemplo, mesmo quando a terminologia é compartilhada, o seu significado pode variar dependendo da semântica utilizada pelo veículo. A expressão Posição, segundo os autores, pode ser relacionado a localização georreferenciada de um robô A, enquanto o robô B usa este termo para expressar suas coordenadas angulares.

Além disso, ferramentas de comunicação (controle de missão) para estes veículos são construídas a fim de que um operador possa enviar informações, missões e controlar aspectos de rede como a inscrição de novos veículos que estiverem ao alcance. Neste sentido, SWARMs apresenta um *middleware* semântico, dividido entre nível superior, sistemas de comunicação e nível inferior. O *middleware* interopera a ferramenta de controle de missão, o gerenciamento de dados e as consultas semânticas com um barramento de comunicação. Ainda, existe o nível inferior que opera com os dispositivos e sensores dos veículos.

A ontologia SWARMs é uma rede de diferentes ontologias que reside no nível superior do *middleware* e procura resolver as necessidades do intercâmbio de dados dos veículos, suas capacidades, progresso de missão e informações sobre o ambiente. Após realizar uma revisão de diferentes ontologias para cooperação de veículos submarinos, apresenta a ontologia que reúne o senso comum a ser utilizado pelos veículos a fim de compreenderem informações sobre missões, planejamento de ações, reconhecimento e sensoriamento de ambiente, sobre veículos robôs, comunicação e rede. Todas estas categorias de informações listadas são ontologias por si só. Uma ontologia núcleo reúne todos os diferentes conhecimentos, interligando em um ponto o acesso aos mesmos.

A ontologia foi produzida utilizando o editor Protégé e formalizada em um documento OWL[19]. Para relações complexas utilizada SWRL (*Semantic Web Rule Language*)[20] para expressar regras complexas e relações que, por si só, a ontologia não consegue inferir. Estas regras e relações são fornecidas por especialistas na área fim e registradas na ontologia. Através de todos estes detalhes, o *middleware* proposto em LI et al. (2017) fornece um meio operacional de modo que o *middleware* recebe as

[19] `https://www.w3.org/OWL`

[20] `https://www.w3.org/Submission/SWRL`

missões de um operador central e os comunica aos veículos. Estes veículos heterogêneos utilizam as capacidades ontológicas de inferência para compreender o seu meio e os seus pares para o cumprimento das tarefas. Assim, são resolvidos os problemas para estes tipos de robôs, que mesmo sendo de diferentes construções, poderão receber ordens, compreender sua posição no ambiente e sua relação com os demais veículos presentes.

2.2.2 *Middlewares* e Agentes

Em sistemas multiagentes os *middlewares* são encontrados de formas variadas, tanto em modelo de arquitetura quanto em diferentes contextos (acesso a itens externos aos agentes, melhoria de funcionalidades internas, entre outros). Sempre houve a necessidade de que agentes pudessem acessar recursos do ambiente onde estão operando (WEYNS et al., 2009).

Dependendo da perspectiva, os próprios agentes podem ser utilizados como componentes de um *middleware* (OLARU; FLOREA; SEGHROUCHNI, 2013) ou novos componentes podem ser agregados em plataformas de simulação para que agentes recebam novos dados, informações ou mecanismos sobre a realidade (SCHULDT; GEHRKE; WERNER, 2008).

No primeiro estudo, OLARU; FLOREA; SEGHROUCHNI (2013) fornecem um modo de reconhecimento de contexto onde os agentes formam uma camada onde resolve a problemática da comunicação entre usuários e dispositivos de sensores pelo ambiente. Neste caso a camada é descrita como bidirecional, pois recebe e envia dados entre dispositivos e sensores de atuação no ambiente. Os agentes são utilizados devido a sua capacidade de autonomia e auto-organização, literalmente aplicando ações em dispositivos conectado com um ambiente dito inteligente[21].

Uma outra perspectiva aborda a construção de *middlewares* para inserção nas plataformas de simulação de agentes com o objetivo de fornecer mais mecanismos que não são naturais aos agentes internamente codificados. Um exemplo é a criação de um *middleware* para sincronismo temporal, onde os autores fornecem um mecanismo onde aspectos temporais de execução são fornecidos (SCHULDT; GEHRKE; WERNER, 2008). Isto é importante no sentido de que o desenvolvimento de agentes pode não compor os aspectos necessários de simulação de sistemas (tempo e sincronismo) entre as ações e as entidades.

Estes autores descrevem que inserir um *middleware* internamente na plataforma de simulação de agentes pode não ser a melhor resolução pois as versões subsequentes da mesma plataforma precisam também possuir esta camada para correto funcionamento. Ainda nesta visão, um outro estudo é o *middleware* MISIA a ser apli-

[21] Ambientes inteligentes são pesquisas sobre interconexão entre dispositivos e sensores de um ambiente para medições e ações, no contexto de computação pervasiva e também na computação ubíqua.

cado entre diferentes plataformas (GARCÍA et al., 2011). Sendo que o contexto é uma camada intermediária entre uma plataforma que contenha os comportamentos dos agentes para um outro ambiente que execute simulação dos mesmos agentes.

Uma outra opção nesta linha de componentes para as plataformas de simulação é o componente de intermediação de conhecimento proposto em SPEROTTO; ADAMATTI (2012), onde o modelo fornece tratamento de informações imprecisas através de ontologias para o sistemas multiagentes. Neste trabalho, os conceitos e suas imprecisões são tratados através de sinônimos com sistema de avaliação *fuzzy* para retornar aos agentes uma determinada expressão ou palavra que faça mais sentido para as trocas de mensagens realizadas entre os mesmos (caso não encontre a expressão desejada, busca pelos sinônimos registrados). Neste modelo, todos os agentes possuem acesso a ontologia, consultando-a sempre que necessário.

Para compreensão das funcionalidades encontradas nos estudos realizados, a Tabela 2 apresenta um comparativo entre determinados critérios. Estes critérios demonstram a capacidade da resolução de problemas de comunicação entre entidades nas pontas dos *middlewares* e na distribuição de conhecimento para com os agentes.

Tabela 2 – Comparativo entre trabalhos sobre *middlewares*.

Trabalhos	Comunicação	Web	Ontologia	Inferência	Agentes	Multiplataforma	Software
SERRANO et al. (2017)	Intermediário	Sim	Sim	Sim	Possível	Sim	Não
LI et al. (2017)	Intermediário	Não	Sim	Sim	Possível	Sim	Não
OLARU; FLOREA; SEGHROUCHNI (2013)	Intermediário	Sim	Não	Possível	Sim	Sim	Não
SCHULDT; GEHRKE; WERNER (2008)	Intermediário	Não	Não	Não	Sim	Não	Não
GARCÍA et al. (2011)	Um-para-um	Não	Não	Não	Sim	Não	Não
SPEROTTO; ADAMATTI (2012)	Intermediário	Não	Sim	Sim	Sim	Sim	Sim

Os critérios determinados para a comparação, descritos a seguir, visam organizar os trabalhos encontrados em determinados pontos de aplicação, isto é, da capacidade de atendimento de funcionalidades:

Comunicação: pode conter os valores um-para-um ou intermediário. O primeiro apenas trafega dados entre os diferentes sistemas, no segundo, realiza também a tradução de dados entre as diferentes tecnologias.

Web: denota a possibilidade do uso para a web do mecanismo proposto, utilizando serviços da Internet. Pode conter os valores Sim e Não.

Ontologia: indica se utiliza uma ontologia para aplicação e distribuição do conhecimento. Pode conter os valores Sim ou Não.

Inferência: denota se existe a funcionalidade de realizar deduções, automáticas ou não, das informações trafegadas (geralmente demanda uma ontologia ligada, mas não é obrigatório). Pode conter os valores Sim, Não ou Possível.

Agentes: informa se a proposta é focada na comunicação entre agentes. Pode conter os valores Sim, Não ou Possível, caso exista a possibilidade de portar o estudo para a área de agentes sociais (ou aplica em outros tipos de entidades de agentes de *software* ou de robôs).

Multiplataforma: descreve se o mecanismo oferecido é multiplataforma de simulação de agentes ou é fabricado para uma ferramenta específica. Pode conter os valores Sim ou Não.

Software: Sim para o caso de fornecer um componente de software a ser aplicado nos sistemas e Não se for um modelo teórico, *software* proprietário ou se nenhum componente foi disponibilizado na publicação.

Existem outros formas de integrações entre dados e agentes que não são formalmente relacionados a *middleware* mas também realizam integrações entre agentes e recursos externos como bases de conhecimento. As plataformas de simulação de agentes como JaCaMo, JADE[22], Repast[23], entre outras, podem fornecer componentes para que os agentes possam interagir, reconhecer a si mesmos e ao seu ambiente (KRAVARI; BASSILIADES, 2015). É comum que as plataformas se preocupem com a simulação de agentes e não forneçam mecanismos extras para suporte a decisão ou acesso a banco de dados ou ontologias (pelo menos não de forma nativa).

O trabalho de FREITAS et al. (2015) apresenta a possibilidade de fornecer acesso dos agentes diretamente com a ontologia por dentro da plataforma JaCaMo. Neste estudo os autores utilizam os artefatos CArtAgO da plataforma para implementar o acesso com as ontologias através da API OWL (HORRIDGE; BECHHOFER, 2011). Os autores fornecem implementações para buscar e manipular os conceitos e relações em ontologias OWL. Neste caso, emprega a ontologia como apoio na tomada de decisão dos agentes e pode substituir o sistema de crenças nativo utilizado pela plataforma para que os agentes dependam somente do conhecimento da ontologia.

O PlaSMA (WARDEN et al., 2010) é uma plataforma de simulação de agentes que tem por base a plataforma JADE e utiliza ontologias para a elaboração de planos de ação e fornecimento de conhecimento aos agentes. Apesar do foco ser em sistemas logísticos, o sistema pode ser reutilizado para outros domínios. Implementa um modelo que explica o mundo físico por ontologias OWL. Ainda, utiliza uma ontologia com conceitos gerais da área logística e, como suporte, se relaciona com outras ontologias que explicam conceitos de transporte, comunicação e bens (materiais e não materiais). Adicionalmente, divide os agentes em dois grupos: os de ambiente e os atores. Os de ambiente são os agentes que intimamente se relacionam com as on-

[22] `https://jade.tilab.com`
[23] `https://repast.github.io`

tologias para fornecer informações (sobre modal de transportes, previsão do tempo, entre outros) aos agentes atores da simulação.

Em relação a outras formas de integração, o trabalho de RANI; NAYAK; VYAS (2015) apresenta um sistema de aprendizado eletrônico com assistência de agentes e ontologias. Os autores definem um sistema de aprendizado online que personaliza e exibe o seu conteúdo de acordo com a utilidade do mesmo pelos seus usuários. Este mesmo trabalho utiliza também a API OWL, mas neste caso aplica a plataforma JADE na operação com os agentes. Existem agentes monitores que verificam o sistema de aprendizado e atualizam a ontologia que possui um modelo de aprendizado preparado para auxiliar nas personalizações.

Esta ontologia contém regras e validações necessárias utilizadas pelo sistema, o agente monitor compreende ações do usuário e atualiza a ontologia refletindo as personalizações. Apesar dos autores utilizarem uma API externa ao ambiente JADE, a plataforma possui algumas manipulações nativas para acessar uma ontologia, criar classes em Java que possam refletir com os conceitos pertencentes a ontologia, entre outros componentes que podem ser agregados para com a plataforma fornecendo funcionalidades ontológicas mais complexas[24].

[24]Documentações JADE em `https://jade.tilab.com/documentation/tutorials-guides`

3 A ARQUITETURA AgentDevLaw

O referencial teórico desta Tese (Capítulo 2) focou em apresentar como os conceitos e arquiteturas têm sido abordados nas resoluções de problemas nas áreas fim desta pesquisa. Em uma parte, ontologias e normas coexistindo até a confecção de ontologias legais para fornecer conhecimento de restrições comportamentais aos agentes. Em outra, arquiteturas baseadas em *middlewares* que visam auxiliar na comunicação de informações entre sistemas de agentes, de tecnologias variadas, indo além do tráfego de dados, fornecendo também novos mecanismos de processamento.

Neste Capítulo, a partir dos estudos realizados, serão descritas as concepções que culminam na abordagem proposta pela Tese, a criação do AgentDevLaw. A nomenclatura AgentDevLaw é composta das intenções para o qual o presente modelo visa atender:

Agent: pois é um modelo que possui foco para agentes (em inglês, *agent*);

Dev: pois é uma ferramenta auxiliar aos desenvolvedores (em inglês, *developers*) no que tange o acesso ao conhecimento; e,

Law: pois designa qual tipo de conhecimento é compartilhado pelo modelo, o conhecimento sobre leis (em inglês, *laws*).

3.1 A Estrutura das Leis Brasileiras

Para a definição do modelo de ontologia desta Tese, faz-se necessário o conhecimento estrutural das leis, neste caso, das leis nacionais. Pois é através deste conhecimento, desta análise, em conjunto com outros estudos apresentados anteriormente, que está definida a proposta de organização de um modelo de ontologia legal.

Na estrutura jurídica das leis brasileiras, a parte normativa é onde descreve o texto que trata as normas propostas (PENNA; MACIEL, 2002). Estas normas são divididas em artigos que, por conseguinte, são divididos em parágrafos e, então, em incisos, alíneas e itens. Os artigos podem ser agrupados em seções, capítulos e livros de acordo com a sua finalidade (artigos do código civil, criminal, de trânsito, entre outros).

No *caput* de cada artigo (primeira informação do artigo), deve ser descrita a norma geral, geralmente são explicações sobre o que irá reger, as restrições e exceções são discriminados pelos parágrafos (descrevendo como é regido ou regulado). Cada parágrafo deve ser numerado descrevendo um único objeto alvo de regulação. Caso exista desdobramento dos parágrafos, detalhamento sobre algumas questões, estas são registradas em incisos. Caso ainda um inciso precise de desdobramentos, estes são listados em alíneas e por itens (também se necessário) (PENNA; MACIEL, 2002).

A redação destes instrumentos de lei são regulamentados pela Lei Complementar nº 95 (BRASIL, 1998a) onde é definido um padrão de organização e são relatadas possíveis exceções nos textos. Conforme comentado anteriormente, podem existir desdobramentos e condições de registro diferenciadas dependendo das necessidades. De forma geral, um texto de lei é dividido em três partes: a parte preliminar, a normativa e a final. A parte preliminar define a ementa da lei, indicando o espaço para o qual se destina a lei. Na parte normativa e final são onde os registros de regulação e as medidas de implementação são definidas (norma regulatória com suas informações de aplicação e revogação). Os artigos, parágrafos, incisos, alíneas ou itens, são todos relacionados como dispositivos da lei.

Cada aplicação de lei é condicionada a ser mais específica quanto o possível do que acerca o conhecimento técnico ou científico sobre o assunto regulado. Isto é visualizado quando certos dispositivos são desdobrados a fim de melhor especificar ou condicionar aplicações legais. Um exemplo para melhor observação pode ser visto na legislação ambiental, o Artigo 27 da Lei nº 7653 que altera a redação dos Artigos 18, 27, 33 e 34 da Lei nº 5.197 (BRASIL, 1988):

Art. 27 Constitui crime punível com pena de reclusão de 2 (dois) a 5 (cinco) anos a violação do disposto nos arts. 2º, 3º, 17 e 18 desta lei.

§ 1º É considerado crime punível com a pena de reclusão de 1 (um) a 3 (três) anos a violação do disposto no artigo 1º e seus parágrafos 4º, 8º e suas alíneas a, b e c, 10 e suas alíneas a, b, c, d, e, f, g, h, i, j, l e m, e 14 e seu § 3º desta lei.

§ 2º Incorre na pena prevista no *caput* deste artigo quem provocar, pelo uso direto ou indireto de agrotóxicos ou de qualquer outra substância química, o perecimento de espécimes da fauna ictiológica existente em rios, lagos, açudes, lagoas, baías ou mar territorial brasileiro.

§ 3º Incide na pena prevista no § 1º deste artigo quem praticar pesca predatória, usando instrumento proibido, explosivo, erva ou substância química de qualquer natureza.

§ 4º Fica proibido pescar no período em que ocorre a piracema, de 1º de outubro a 30 de janeiro, nos cursos d'água ou em água parada ou mar territorial, no

período em que tem lugar a desova e/ou a reprodução dos peixes; quem infringir esta norma fica sujeito à seguinte pena:

a) se pescador profissional, multa de 5 (cinco) a 20 (vinte) Obrigações do Tesouro Nacional - OTN e suspensão da atividade profissional por um período de 30 (trinta) a 90 (noventa) dias;

b) se a empresa que explora a pesca, multa de 100 (cem) a 500 (quinhentas) Obrigações de Tesouro Nacional - OTN e suspensão de suas atividades por um período de 30 (trinta) a 60 (sessenta) dias;

c) se pescador amador, multa de 20 (vinte) a 80 (oitenta) Obrigações do Tesouro Nacional - OTN e perda de todos os instrumentos e equipamentos usados na pescaria.

§ 5º Quem, de qualquer maneira, concorrer para os crimes previstos no *caput* e no 1º deste artigo incidirá nas penas a eles cominadas.

§ 6º Se o autor da infração considerada crime nesta lei for estrangeiro, será expulso do País, após o cumprimento da pena que lhe for imposta, (Vetado), devendo a autoridade judiciária ou administrativa remeter, ao Ministério da Justiça, cópia da decisão cominativa da pena aplicada, no prazo de 30 (trinta) dias do trânsito em julgado de sua decisão.

Nesta listagem de informações da lei, verifica-se pelo artigo 27 que as descrições de texto das normas podem orientar ou explicar sobre certas situações e regramentos. Os parágrafos 2, 3 e 4, do mesmo artigo, são exemplos de detalhamentos para o que e para quem se aplica a punição, previamente descrita no caput. No parágrafo 4, constam detalhes sobre determinada situação em que é definido a restrição de uma ação (pesca proibida) e nas alíneas a, b e c suas respectivas punições dependendo de quem é o autor da infração. O parágrafo 6 do artigo 27 define uma orientação de ação ao legislador ou forças policiais, quando o infrator não for brasileiro.

Um outro exemplo de redação pode ser conferido através dos artigos 54 e 55 da Lei nº 9605 (BRASIL, 1998b), onde:

Art. 54 Causar poluição de qualquer natureza em níveis tais que resultem ou possam resultar em danos à saúde humana, ou que provoquem a mortandade de animais ou a destruição significativa da flora:

Pena - reclusão, de um a quatro anos, e multa.

§ 1º Se o crime é culposo:

Pena - detenção, de seis meses a um ano, e multa.

§ 2º Se o crime:

I - tornar uma área, urbana ou rural, imprópria para a ocupação humana;

> II - causar poluição atmosférica que provoque a retirada, ainda que momentânea, dos habitantes das áreas afetadas, ou que cause danos diretos à saúde da população;
>
> III - causar poluição hídrica que torne necessária a interrupção do abastecimento público de água de uma comunidade;
>
> IV - dificultar ou impedir o uso público das praias;
>
> V - ocorrer por lançamento de resíduos sólidos, líquidos ou gasosos, ou detritos, óleos ou substâncias oleosas, em desacordo com as exigências estabelecidas em leis ou regulamentos:
>
> Pena - reclusão, de um a cinco anos.
>
> **§ 3º** Incorre nas mesmas penas previstas no parágrafo anterior quem deixar de adotar, quando assim o exigir a autoridade competente, medidas de precaução em caso de risco de dano ambiental grave ou irreversível.

> **Art. 55** Executar pesquisa, lavra ou extração de recursos minerais sem a competente autorização, permissão, concessão ou licença, ou em desacordo com a obtida:
>
> Pena - detenção, de seis meses a um ano, e multa.
>
> Parágrafo único. Nas mesmas penas incorre quem deixa de recuperar a área pesquisada ou explorada, nos termos da autorização, permissão, licença, concessão ou determinação do órgão competente.

Neste caso existem as aplicações de regulações pelos parágrafos com itens (como apresentado no artigo 54) ou diretamente pelo artigo (apresentado no artigo 55). Desta forma, nem sempre é especificado o detalhamento da sanção e autoria pelos parágrafos, sendo que variavelmente a definição da legislação reside somente no artigo. Estes tipos de artigos descrevem toda a informação necessária sobre a regra vigente sem a necessidade de dispor em maiores detalhes em subsequentes parágrafos e, por consequência, em alíneas.

No meio jurídico existem duas terminologias relacionadas a expressão sanção: sanção premial e sanção negativa (BENEVIDES FILHO, 2013). A sanção negativa está relacionada com a consideração feita a atos não desejados por alguma determinada regra, punições ou coações ao agente praticante da ação. A sansão premial está relacionada com premiações para quando o agente executa uma conduta que é aprovada ou foi esperada. No âmbito desta Tese, optou-se pelo emprego das terminologias de sanção ou recompensas, isto é, agrupando em sanções tudo o que for oposto à recompensa. Se existirem casos especiais a estes termos, serão devidamente explicitados.

Os casos de exceções de detalhamentos pelos subelementos são observados através de outros dispositivos de lei, inclusive, no mesmo contexto de leis ambientais. Na

Lei nº 5197 (BRASIL, 1967), o artigo 18 define que é proibida a exportação para o Exterior, de peles e couros de anfíbios e répteis, em bruto. Não define como deve ser punido tal infração. O artigo 27, descrito melhor pela Lei nº 7653, informa a punição logo na descrição do artigo: “Constitui crime punível com pena de reclusão de 2 (dois) a 5 (cinco) anos a violação do disposto nos arts. 2º, 3º, 17 e 18 desta lei”.

Existe ainda um terceiro caso, quando o artigo de lei possui parágrafos mas destes não apontam diretamente para sanções, ficando a cargo de outro mecanismo explicitar estas mesmas regulações. Na Lei nº 5197 (BRASIL, 1967) tem-se o seguinte artigo:

Art. 3º. É proibido o comércio de espécimes da fauna silvestre e de produtos e objetos que impliquem na sua caça, perseguição, destruição ou apanha.

§ 1º Excetuam-se os espécimes provenientes legalizados.

§ 2º Será permitida mediante licença da autoridade competente, a apanha de ovos, lavras e filhotes que se destinem aos estabelecimentos acima referidos, bem como a destruição de animais silvestres considerados nocivos à agricultura ou à saúde pública.

Este artigo possui dois parágrafos regulando o comércio de espécimes e ferramentas de caça, sendo que o artigo 27 da Lei nº 7653 é quem condiciona a sanção de reclusão de 2 a 5 anos. Observa-se que nestes parágrafos existe uma possível permissão de tais atividades através de licenças, que também acabam por discorrer em outros artigos na mesma lei (art. 20, por exemplo).

A parte final de cada lei descreve aspectos de período de publicação. Cada lei possui uma data de publicação, entretanto, conforme especificações do artigo 8 da Lei nº 95 (BRASIL, 1998a), a aplicação da mesma pode variar. Se existe a necessidade de amplo conhecimento da redação, um prazo de aplicação é informado. Este prazo de aplicação fará com que os efeitos normativos somente serão válidos a partir da expiração desta cláusula e não da publicação.

Caso for reconhecido, por algum consenso parlamentar que, tanto a redação quanto os seus efeitos são de simples entendimento, é publicado na parte final que a lei vigora a partir da sua publicação. Podem existir variações como período de validade, onde a normativa somente existe dentro de um intervalo de datas. Um exemplo disso é a regulação de pesca em época de reprodução dos peixes (piracema) (BRASIL, 1988).

É comum encontrar relações entre leis diferentes pois atualizações ocorrem ao longo dos anos. A seção III da Lei nº 95 (BRASIL, 1998a) define as especificações de alterações de leis. Existem algumas formas para que atualizações ocorram, podendo ser necessária a criação de novo texto normativo se a alteração for considerável. Caso existam alterações parciais, estas podem ser dadas por leis complementares ou revogadas parcialmente da lei original.

Não é permitido a reusabilidade de numerações, um parágrafo ou alínea, por exemplo. Caso exista a revogação ou atualização, estas serão substituídas por outro dispositivo de numeração diferenciada (também não é permitida a reordenação dos itens). Um artigo pode vetar ou complementar outro. Existe a possibilidade de centralizar todas as modificações em um documento, através da visualização da lei na sua forma compilada. Independente disso, mesmo na compilação, as correções são aprovadas e publicadas separadamente, em outras leis.

Neste sentido, observa-se nos documentos legislativos certas estruturas organizacionais que culminam em um padrão sistemático de trabalhos jurídicos. Em outros elementos, no entanto, existem relações mais complexas entre artigos de lei que não possuem parágrafos, nem alíneas e podem definir regulações para si ou para outros artigos. Ainda assim, é possível extrair elementos destas estruturas a fim de organizá-los em uma lista conceitual de conceitos e de relacionamentos entre estes.

3.2 Uma Ontologia Legal

A partir das concepções apresentadas sobre a estrutura da legislação brasileira, para a presente Tese considerou-se que todo artigo descreve uma legislação pertinente e seus subelementos são descritores de ações e punições, definindo ao menos uma restrição comportamental (norma). A exceção deste regramento é quando o próprio artigo é uma norma, conforme evidências na Seção 3.1. Se cada artigo foca em um objeto de lei, então considerou-se que cada objeto legislativo é uma norma.

Por um lado, análises das instruções normativas utilizando exemplos da legislação ambiental brasileira, de outro, estudos sobre ontologias legais e normativas. É possível observar nas últimas concepções um elemento predominante chamado *Norma*. Este conceito, como descritor da regulamentação comportamental dos agentes, torna-se um elo importante em quaisquer definições sobre ontologias jurídicas.

Esta concepção, de que norma é um mecanismo prescritivo de regulação, é comum em diversos trabalhos da literatura como, por exemplo, os de FELICíSSIMO et al. (2005); FREITAS (2017); ALEXANDER (2009). Além disso, a obra de VALENTE; BREUKER (1994), baseado fortemente na Teoria da Lei de Kelsen (KELSEN, 2009), já definia que existem diferentes tipos de normas no conhecimento normativo (ainda que sejam apenas diferenças de termos linguísticos e não de função).

Assim, de acordo com estes estudos e suas bases teóricas, não há a necessidade de elaboração de novos termos para normas, pois existe um determinado padrão de aplicação das mesmas. Baseando-se nestes estudos, serão empregados os mesmos termos para normas de obrigação, proibição e permissão. Aliado a estes termos, são empregadas nesta Tese outras definições como recompensa e punição, baseando-se na possibilidade de diferentes categorizações de termos em consequências dos dife-

rentes atos dos agentes. Isto aliado também ao conjunto de possíveis construções de conhecimentos reativos que estes possam vir a possuir em suas modelagens (VALENTE; BREUKER, 1994; PALMIRANI et al., 2018).

Apesar das ontologias fundacionais procurarem fornecer base para que quaisquer outras ontologias de domínios possam ser aplicadas, nesta Tese empregou-se o conceito de objeto social agentivo da DOLCE, proposto por GANGEMI et al. (2002), que define informações sobre aspectos da sociedade em papéis e instituições existentes.

Entretanto, não será utilizado da mesma forma como em DOLCE, onde existem os conceitos de agente social (como uma pessoa legal) e de sociedade (as instituições da sociedade). A partir da definição social de DOLCE, considerou-se para esta Tese a relação direta de papéis desempenhados na sociedade pelos agentes. Esta relação direta é similar às definições de FREITAS (2017), onde um papel é regulado por uma norma e um agente, ao assumir tal especificação, ativará o seu compromisso com as restrições impostas pelo seu papel.

Em conjunto disso, a especificação de papel na LRI-CORE fornecida por BREUKER; VALENTE; WINKELS (2005) foca nesta definição de que os papéis são identificações individuais das pessoas. Em relação às leis brasileiras, esta ideia é bastante similar pois, dependendo do contexto, diferentes punições são escalonadas para diferentes papéis, assumindo que diferentes cargos na sociedade possuem diferentes normativas[1]. Portanto, estas correlações tornaram-se relevantes e, então, foram considerados para o modelo proposto os termos *papel* e suas subdivisões *agente governamental* e *agente social*, permitindo o registro das informações particulares de papel de cada indivíduo.

Sobre aspetos gerais das leis brasileiras, existe a possibilidade de extração de variados termos e relações para o modelo ontológico (e que serão empregados ao longo desta Seção). O artigo 27 (BRASIL, 1988) é um exemplo de dispositivo com desdobramentos a ser utilizado, pois em seu parágrafo número 4 é possível extrair que: durante a piracema (período de tempo de validade), torna-se proibida (restrição) a pesca (ação) em cursos d'água ou água parada em território nacional (lugar). Apesar de outros artigos exemplificados na Seção 3.1 evidenciarem que tais informações possam estar organizados de outras formas dentre os dispositivos de lei, esta lógica e relação de termos (validade, restrição, entre outros) serão considerados para a definição desta proposta.

As normas regulam as ações realizadas pelos agentes, por isso, esta definição também precisa constar no modelo ontológico. Neste trabalho, quem define as normas é a legislação brasileira, que especifica as normas. Neste sentido, cada norma regula uma ou mais ações, enquanto a ação é regulada por alguma legislação, ocor-

[1] Punições escalam se o indivíduo é uma pessoa comum ou se é uma empresa, no caso de alguns artigos compilados da Lei nº (BRASIL, 1967), por exemplo.

rendo uma espécie de ciclo de relações: ação regulada por legislação especificada por norma que regula a ação.

A seguir serão sumarizados os conceitos da ontologia proposta. Esta organização de conceitos visa respeitar as obras de ontologias da área (Seção 2.1.1), compactuando com as análises realizadas sobre as informações mais importantes da construção das leis brasileiras (Seção 3.1). Para adequar o texto da Tese aos modelos encontrados na literatura, a partir deste momento, das definições até o modelo de ontologia construído, será escolhido o idioma inglês para a conceitualização.

***Legislation*:** Descreve o conceito de leis expressado pelo caput de cada artigo de lei ou por um parágrafo quando recai nas exceções estudas.

***Norm*:** Informação que descreve um regulamento sobre o comportamento de uma entidade sob um dado contexto. Uma ou mais normas registram o detalhamento de parágrafos e alíneas legais. A norma pode ser especializada em obrigação, permissão, proibição, punição ou de recompensa.

- ***Permission*:** Descreve normas de permissão.
- ***Reward*:** Descreve normas sobre ações premiadas dos agentes, isto é, ações cometidas pelos agentes que são louváveis, merecedoras de algum tipo de compensação.
- ***Prohibition*:** Descreve normas de ações que não podem ser executadas.
- ***Obligation*:** Descreve normas sobre obrigações que os agentes precisam desempenhar.

***Action*:** Podem descrever as atividades a serem cumpridas, tanto pelos legisladores, policiais ou outras entidades da sociedade.

***Punishment*:** Define, através dos artigos, parágrafos e alíneas, descrições sobre punições. As punições descrevem os efeitos quando alguma regra é violada. As punições podem ser classificadas em reclusão, multa e apreensão.

- ***Confinement*:** Se uma norma especificar punição por reclusão, esta deverá conter valor mínimo e máximo de tempo de duração (em meses ou anos).
- ***PayAFine*:** Se uma norma especificar punição por multa, deverá informar valores e condições da mesma.
- ***Seizure*:** Recolhimento de recursos materiais dos infratores.

***Social*:** Descreve os objetos sociais do ambiente, generalizando como padrão os papéis da sociedade, das quais são agentes governamentais e agentes sociais.

***Role*:** Descreve papéis assumidos pelos integrantes da sociedade, divididos entre governamental e social.

***GovernmentAgent*:** Papel assumido por servidores públicos.

***SocialAgent*:** Papel assumido pelos cidadãos da sociedade que não se enquadram no funcionalismo público.

***Environment*:** Descreve objetos físicos da sociedade.

***Place*:** Região física onde deve atuar a definição de norma.

***Resource*:** Recursos existentes no ambiente de execução dos agentes, entre materiais e não materiais.

***Non-material*:** Descreve os recursos intangíveis como o dinheiro, por exemplo.

***Material*:** Descreve os recursos tangíveis, podendo ser desde objetos comuns a ambientes destinados a serem recursos (minas de minério, trechos de água para a pesca, entre outros).

Para a construção da ontologia, inicialmente devem ser definidos os axiomas desta, onde são visualizados os relacionamentos e restrições entre os conceitos. A lógica de primeira ordem foi empregada como técnica de formalização desta proposta:

1. As categorias legislação, ambiente, social e ação definem as principais ramificações da ontologia.

$$(Legislation, Environment, Social, Action) \in Thing$$

2. Os lugares e recursos são membros de ambientes.

$$(Place, Resource) \in Environment$$

3. Todo recurso é dividido entre recurso material e não material.

$$(\forall_x)(Resource(x) \rightarrow Material(x)|Non\text{-}Material(x)$$

4. Todo instrumento normativo é uma norma.

$$normative \equiv Norm$$

5. Toda norma é subparte de uma legislação.

$$(\forall_x)(Norm(x)) \subset Legislation$$

6. Toda punição, ou recompensa, ou obrigação, ou permissão, ou proibição é uma subparte de uma norma.
   $(\forall_x)(Punishment(x) \vee Reward(x) \vee Obligation(x) \vee Permission(x) \vee Prohibition(x)) \subset Norm$

7. Toda multa, apreensão e reclusão são tipos de uma punição.

$$(\forall_x)(PayAFine(x), Seizure(x), Confinement(x)) \subset Punishment$$

8. Todo agente social ou agente governamental é um papel dentro de uma noção social da sociedade.

$$(\forall_x)(socialAgent(x) \vee governmentAgent(x)) \subset Role \equiv Social$$

9. Uma ação pode modificar um ambiente.

$$(\exists_{x,y})(Action(x) \wedge Environment(y)) \rightarrow (modify(x, y))$$

10. As ações são reguladas por alguma legislação.

$$regulatedBy(x, y)(Action(x), Legislation(y))$$

11. As legislações regulam uma ou mais ações.

$$regulate(x, y)(Legislation(x), Action(y))$$

12. Toda recompensa possui um recurso a ser fornecido.

$$(\forall_{x,y})(Reward(x) \wedge Resource(y)) \rightarrow has(x, y)$$

13. Existem normas que aplicam punições ou proibições ou obrigações ou recompensas ou permissões.
    $(\exists_{x,y})(Norm(x) \wedge (Punishment(y) \vee Prohibition(y) \vee Obligation(y) \vee Reward(y) \vee Permission(y)) \rightarrow apply(x, y)$

14. As normas especificam uma legislação.

$$specifiedBy(x, y)(Legislation(x), Norm(y))$$

15. As normas são relacionadas a papéis.

$$relates(x, y)(Norm(x), Role(y))$$

A hierarquia de conceitos da ontologia, extraída dos Axiomas 1 a 8 estipulados anteriormente, pode ser visualizada na Figura 5. Para a edição da ontologia foi utilizada a ferramenta Protégé[2] (MUSEN, 2015) na sua versão 5. Essa ontologia deve ser considerada como base e pode receber mais conceitos dependendo do domínio, sua intenção é focar na parte jurídica do sistema.

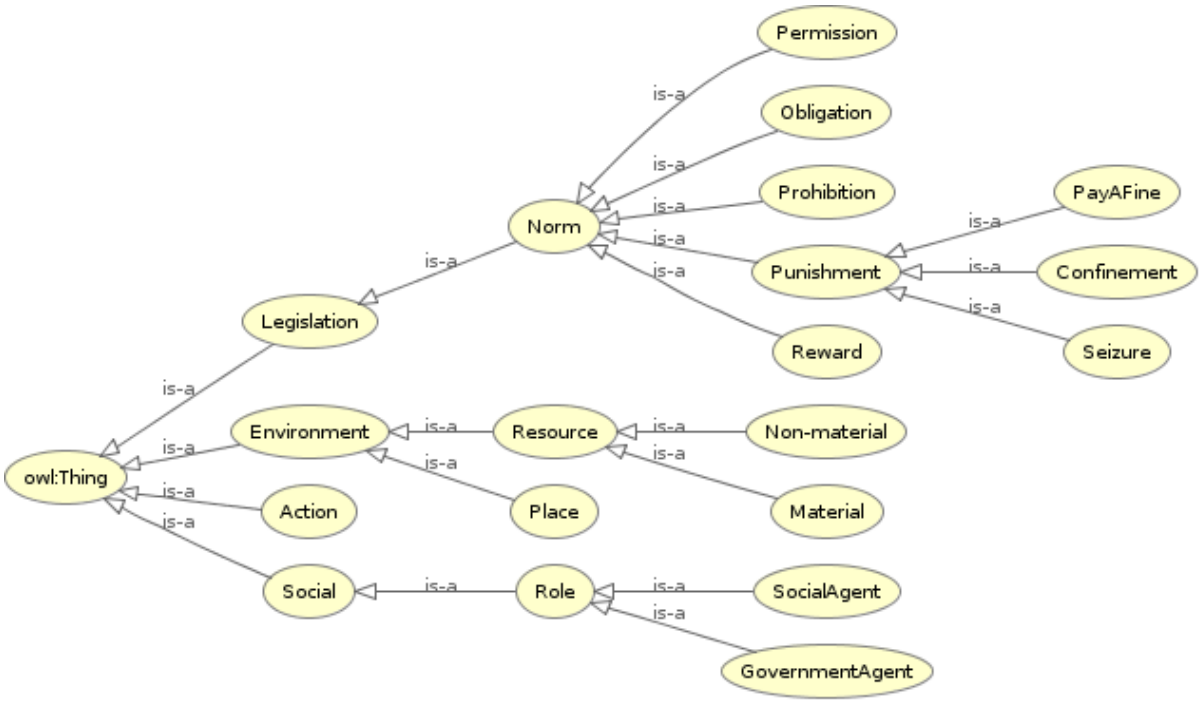


Figura 5 – Ontologia proposta (hierarquia de conceitos).

O conceito normativo refere-se às explicitações de instruções normativas, as leis do sistema. Cada lei pode ser representada por uma ou mais normas, que possuem como sub-conceitos recompensa, punição, obrigação, permissão. Estes são considerados como subtipos de norma devido sua aplicação dentro da sociedade (não existe punição ou permissão sem uma normativa).

Do mesmo modo, as multas, apreensão e reclusão também são subtipos de uma punição que também existem somente através das normas. O conceito de ambiente refere-se aos conhecimentos sobre itens internos ao ambiente, sejam estes lugares (regiões) e recursos existentes. Os recursos podem ser materiais, quando têm o sentido de recursos físicos encontrados geograficamente (como os rios) ou não materiais, quando referem-se aos recursos mentais ou monetários (moeda).

[2] `https://protege.stanford.edu/`

Conforme comentado neste Capítulo, as leis podem possuir um determinado período de validade, não somente na sua vacância bem como no período de atuação, como é o caso da regulação de pesca em época de piracema (BRASIL, 1988). Esta lei, apesar de possuir uma data de publicação, é ativa durante a reprodução de peixes (1 de outubro a 30 de janeiro).

Assim, é conferido que toda lei possui uma data em que define o marco da sua aplicação (ações que antecedem a data não recebem as mesmas regulações previstas). Esta data pode ser da sua publicação, após o período de vacância ou ainda, dentro de um período estipulado (caso da piracema). Obrigatoriamente será considerado no modelo proposto uma data inicial e, se existir, uma data final, porque a especificação tem um período de validade e o modelo como um todo deve responder como tal.

Como analisado na Seção 2.1.1, verificou-se que existem ontologias bases que são fornecidas para que os especialistas em seus domínios possam estender e incorporar mais detalhes, refletindo suas necessidades. Neste estudo, o domínio refletido é o da legislação brasileira.

Para continuar na modelagem de domínio, incorporando mais detalhes apresentados até o presente momento, será empregado novamente o exemplo do artigo 27 parágrafo 4 (BRASIL, 1988). Suas alíneas descrevem que:

a. se pescador profissional, multa de 5 (cinco) a 20 (vinte) Obrigações do Tesouro Nacional - OTN e suspensão da atividade profissional por um período de 30 (trinta) a 90 (noventa) dias;

b. se a empresa que explora a pesca, multa de 100 (cem) a 500 (quinhentas) Obrigações de Tesouro Nacional - OTN e suspensão de suas atividades por um período de 30 (trinta) a 60 (sessenta) dias;

c. se pescador amador, multa de 20 (vinte) a 80 (oitenta) Obrigações do Tesouro Nacional - OTN e perda de todos os instrumentos e equipamentos usados na pescaria.

Este parágrafo fornece uma restrição de comportamento aos pescadores no consumo de recursos no ambiente em que se encontram. Novos relacionamentos e predicados podem surgir no sentido de explicar que, dependendo de uma condição, uma punição é aplicada (conforme modelagem da ontologia base). Além disso, existirão instâncias ou indivíduos na ontologia final que representam os conceitos concretos, ou objetos concretos. A instância é a representação prática do conceito da ontologia naquele domínio.

As alíneas referidas descrevem multas, apreensão e suspensão das atividades, portanto devem ser instâncias dos referidos conceitos na ontologia. Os predicados definem geralmente o relacionamento entre instâncias específicas. Utilizando a onto-

logia base (Figura 5) e considerando este mesmo artigo 27, parágrafo 4, para a norma na ontologia, as extensões podem ser definidas como:

1. Uma instância de norma deve explicitar o referido artigo e parágrafo, para servir de base de conhecimento na ontologia. Com isso, se pode ter uma instância de legislação com nome de `law-7653_article27_4` que é especificada pela instância de norma `7653_article27_4_line-c`, através da propriedade *aplica* que aponta para os indivíduos que forneçam conhecimento da punição a ser aplicada.

   Se x = `7653_article27_4_line-c` tem-se que:

   $(\exists!_{x,y})(Norm(x) \land apply(y)) \rightarrow Punishment(PayAFine) \land Seizure(equipment\_loss)$

   O mesmo deverá ser relacionado a outros tipos de punições, permissões, obrigações e apreensões que existirem no texto jurídico. Em caso do artigo não possuir parágrafos em sua descrição e, por consequência, sem alíneas, a única alteração permitida é no nome dado para a instância.

2. Cada multa deve ser instanciada como indivíduo na ontologia que contém a informação do valor em X e é do tipo multa.

   $$individuals(PayAFine_X) \subset concept(PayAFine)$$

3. Para toda legislação, esta obrigatoriamente deve possuir uma data de homologação ou uma data inicial de quando estiver ativa. Entretanto, pode de forma não obrigatória ter uma data final de validade:

   $(\forall_{x,y})(Legislation(x) \land dateTime(y) \land dateTime(z) \rightarrow Legislation(x) \land starts\_at(y) \lor ends\_at(z)$

Os conhecimentos sobre atores na sociedade devem seguir a ramificação iniciada pelo conceito social. Inicialmente os papéis dos agentes assumidos nesta sociedade são modelados aqui e podem se relacionar com outras entidades na ontologia. A Figura 6 descreve a mesma ontologia com as propriedades de objeto.

É possível visualizar na Figura 6 as propriedades de objeto que geram as conexões entre os indivíduos de uma ontologia, isto é, são os axiomas definidos para a relação entre conceitos (itens 9 a 15 da primeira listagem de axiomas e todos os demais de relacionamentos de conceitos da segunda lista). Nesta mesma Figura 6, é possível verificar que toda norma possui papéis relacionados (legislador, infrator, policial). A norma de forma geral, regula uma ação que pode modificar o ambiente da sociedade. Uma ou mais ações podem ser reguladas por uma ou mais legislações. A recom-

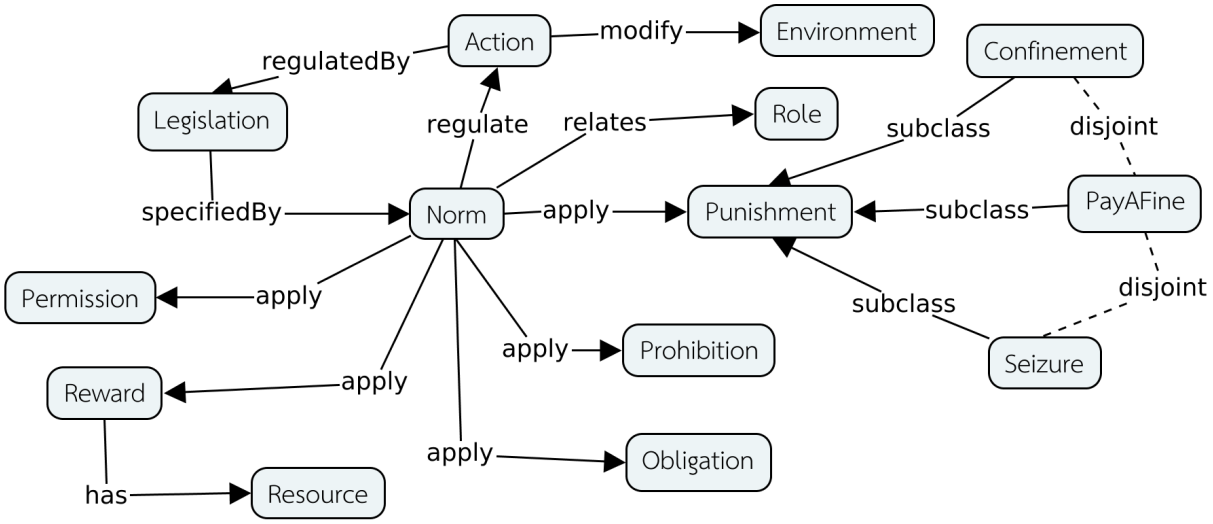


Figura 6 – Ontologia proposta com as propriedades de objetos (relações entre instâncias).

pensa possui um recurso pois as recompensas podem ser materiais ou imateriais aos agentes.

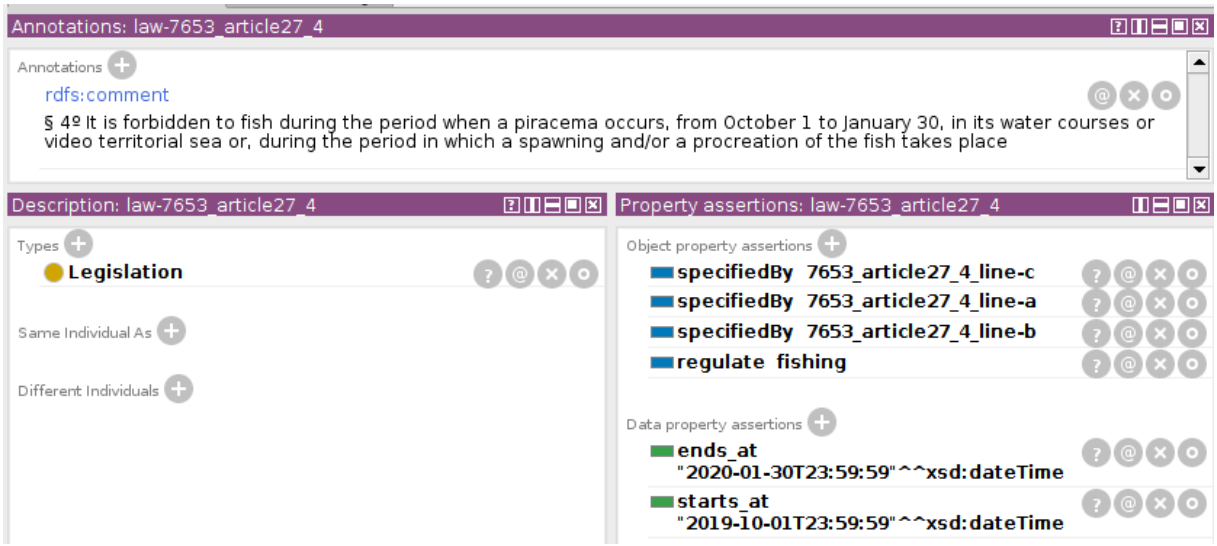


Figura 7 – Uma instância da legislação com suas propriedades de relacionamentos com outros objetos e dados.

Na Figura 7 é descrito um exemplo dos itens formalizados anteriormente sobre toda Legislação ser especificada por *Norms* (neste caso serão três normas de acordo com as alíneas a, b e c). A instância `law-7653_article27_4` descreve o artigo de lei na sua *tag* `rdfs:comment` e é especificado pelos indivíduos `7653_article27_4_line-a`, `7653_article27_4_line-b` e `7653_article27_4_line-c`.

Ainda nesta mesma instância, notam-se duas propriedades de dados do tipo `dateTime` definida como `starts_at` e `ends_at`. Somente a data inicial é uma proprie-

dade obrigatória conforme formalização número 3 (do segundo conjunto de axiomas), informando a partir de qual data a lei é válida. Caso exista uma periodicidade de validade da lei, a propriedade `ends_at` deve ser informada com a data de expiração da lei. Na mesma Figura 7 o axioma número 11 da primeira lista é obedecido, fazendo com que seja definido que esta instância `law-7653_article27_4` regula (regulate) uma outra instância de *Action* chamada *fishing*.

Figura 8 – Formato de cadastro da instância *fishing* de *Action*.

Para satisfazer o axioma 10, é necessário que ao se registrar as instâncias de Action, as mesmas sejam relacionadas com as legislações que as regulam. Neste sentido, a Figura 8 demonstra a ação fishing relacionada com a legislação da Figura 7. Não existe um limite nesta afirmação, portanto, mais de uma legislação poderá estar relacionada a esta ação através do uso do predicado *regulatedBy*.

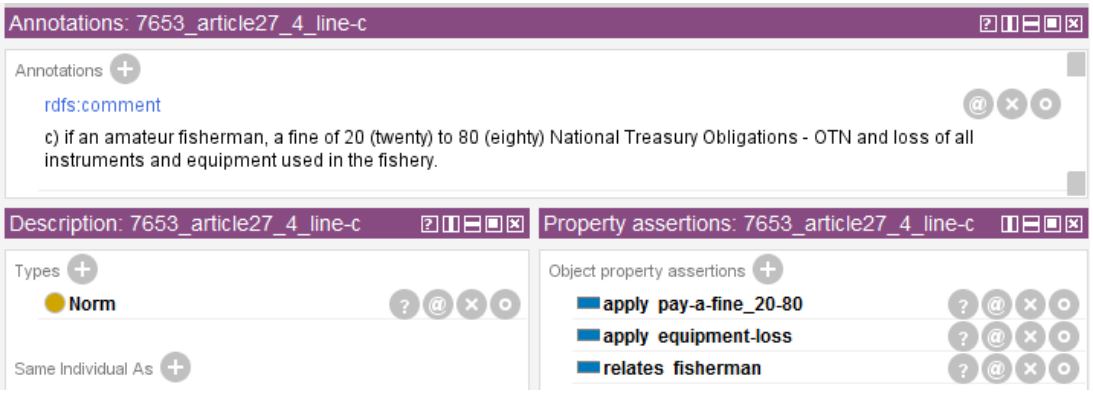


Figura 9 – Uma instância de norma com suas propriedades de relacionamentos com sanções e papéis.

A Figura 9 demonstra a instanciação de `7653_article27_4_line-c`, relacionada como especificação da lei demonstrada na Figura 7. Contém também seu texto legal na *tag* `rdfs:comment` e, neste caso, possui dois tipos de propriedades de objeto, aplica (`apply`) e relaciona (`relates`).

A Figura 6, bem como suas formalizações anteriores, denota que toda norma relaciona um ou mais papéis e pode aplicar variados subtipos de normas, isto é, indivíduos de proibição, obrigação, permissão, punições e recompensa.

No exemplo da Figura 9 percebe-se que esta instância se relaciona para aplicabilidade da norma para com as instâncias `pay_a_fine_20-80` (pagamento de multa)

e `equipment-loss` (perda de equipamentos). Conforme a descrição nos comentários da instância, esta alínea descreve que tais sanções são relacionadas para a entidade pescador (amador), por isso o registro da propriedade `relates` apontando para a instância `fisherman`.

A Figura 10 exibe como a instância `pay_a_fine_20-80` encontra-se registrada na ontologia. Como os documentos OWL e RDF possuem as especificações dos tipos de dados em XML Schema Datatype[3] (XSD) as propriedades de dados em cada instância deverão ser informadas de acordo com a necessidade. Neste caso foi definida a propriedade `fine_value` com o valor de 40 unidades e com a especificação de que é do tipo *float*. As propriedade de dados têm por objetivo discriminar melhor as características de cada indivíduo da ontologia. Em outros casos, como nas instâncias do conceito *Confinement*, haverá outros tipos de dados descrevendo os valores mínimo e máximo de prisão, por exemplo.

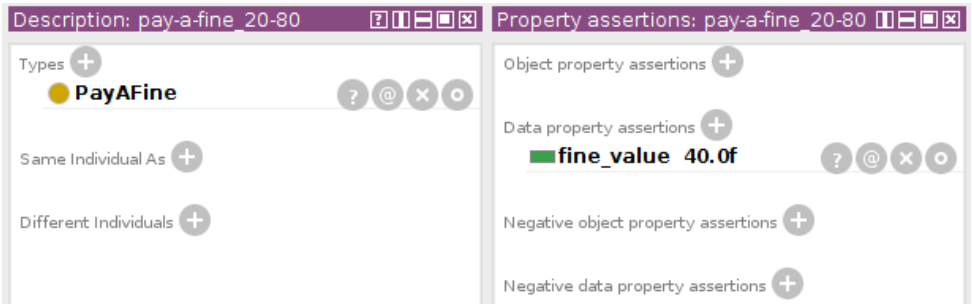


Figura 10 – Exemplo de instância com especificação de tipos de dados.

Na Figura 11 são exibidos os arcos que conectam cada um dos conceitos e instâncias que existem especificamente para a norma do artigo exemplificado (artigo 27). Dado o formato de nomenclatura das instâncias de *Legislation*, percebe-se a correlação com o número do parágrafo (número 4), onde suas alíneas tornam-se instâncias de normas.

Dentre os exemplos de artigos apresentados na Seção 3.1, quando é encontrado um caso onde um artigo de lei não possuir parágrafos, ou se os tem, mas não possui as alíneas, opta-se em seguir com o padrão completo de descrição na ontologia: artigo em *Legislation*, parágrafos/alíneas/itens em *Norm*.

Se um artigo possuir parágrafo único, já informando as regulações, é necessário registrar a mesma sequência de indivíduos: define o artigo em *Legislation*, o mesmo deve ser especificado por uma instância de *Norm* (ao invés do número do parágrafo, utiliza-se a letra u, de parágrafo único, na nomenclatura da instância). Isto não implica em perdas semânticas no modelo proposto, porém demanda ao(s) especialista(s) atenção na interpretação, durante a coleta de informações ou o registro delas na ontologia.

[3] `https://www.w3.org/TR/swbp-xsch-datatypes/`

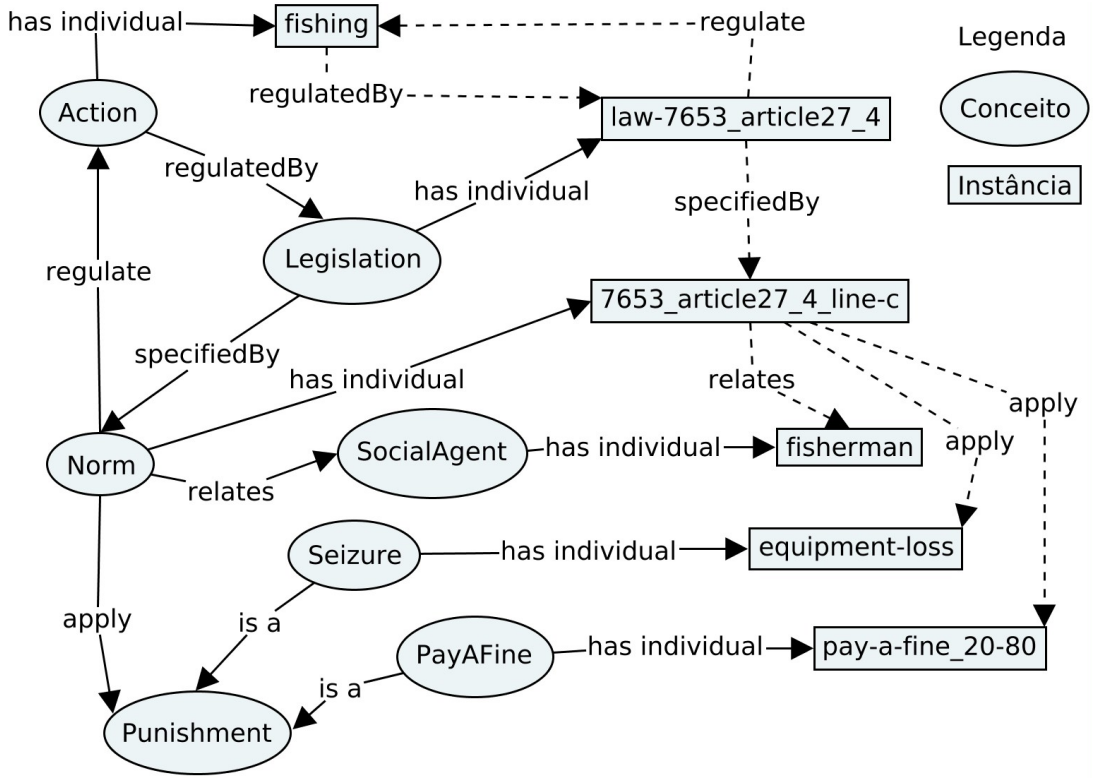


Figura 11 – Um recorte exibindo os tipos de relações entre conceitos (linhas contínuas) e entre instâncias (linhas tracejadas) da ontologia.

Independente dos casos encontrados, as regulamentações devem seguir o padrão de instâncias de *Legislation* e *Norm* (vinculadas pela propriedade `specifiedBy`). Apenas não haverá a obrigatoriedade do registro do texto legal na *tag* de `rdfs:comment` das instâncias de *Norm*. É importante, porém também não obrigatório, nesta proposta, manter a descrição jurídica na *tag* `rdfs:comment` das instâncias de *Legislation*. Isto poderá ser obrigatório dependendo da opção de busca configurada no *middleware* (isto será tratado na Seção 3.3).

Além das propriedades de relacionamentos entre conceitos já definidas anteriormente, específicas do domínio legal (norma relaciona papéis ou norma aplica punição, entre outros), outras propriedades mais avançadas podem ser pertinentes nas ontologias.

Sistema de raciocinadores como Pellet (SIRIN; PARSIA, 2004), FaCT++ (TSARKOV; HORROCKS, 2006), HermiT (GLIMM et al., 2014), entre outros, aproveitam alguns tipos de propriedades para realizar inferências nas ontologias. O significado de inferir no aspecto lógico sugere que tais sistemas podem realizar deduções e novas afirmações a partir de ligações existentes na ontologia.

Estas ligações são os relacionamentos entre as instâncias de classes de uma ontologia, sendo que os tipos reconhecidos são classes disjuntas, propriedades inversas, propriedades transitivas, simétricas, assimétricas, reflexivas, irreflexiva, funcionais e funcionais reversas (ANTONIOU; VAN HARMELEN, 2004; MOTIK et al., 2012).

As classes disjuntas definem que seus indivíduos só podem estar relacionados a um tipo de classe por vez. As instâncias só podem estar relacionadas a uma única classe, sendo diferentes dos demais conceitos independentemente de possuírem um mesmo ancestral na hierarquia, uma mesma superclasse.

Na ontologia proposta, as subclasses de *Punishment* devem ser registradas como disjuntas uma vez que os indivíduos das mesmas não podem transitar de um conceito para outro. Uma instância de *PayAFine*, por exemplo, pode ser um indivíduo de *Obligation* sem perda semântica, dependendo somente do contexto legal. Entretanto, os conceitos subclasses de *Punishment* são disjuntos, pois instâncias que descrevem multas, prisão e perda de equipamentos devem ser consideradas categorias individuais muito específicas (não existe a possibilidade de considerar que uma multa seja do mesmo tipo que uma prisão).

A ideia aqui é diminuir possíveis problemas gerados por um dado indivíduo que acabe por pertencer a mais de uma classe, o que poderia em algum momento conflitar os tipos de sanções definidos via *Norm*. As subclasses de papéis não são disjuntas, uma vez que um mesmo agente pode possuir múltiplos papéis na sociedade. Um fiscal do governo permanece com o seu papel base de cidadão e recebe as sanções relacionadas não somente ao seu papel base na sociedade mas novas regras de acordo com o seu papel de fiscalização.

As propriedades inversas indicam o relacionamento inverso de uma propriedade de relação. Neste caso, para uma dada relação entre um conceito A e B pode existir outra conectando B com A, inversamente. Na presente ontologia são definidas algumas propriedades inversas.

A propriedade *publishedBy* define o inverso da *apply*, ou seja, se as Normas aplicam Punições, Obrigações, Recompensas, Proibições, Permissões, o inverso determina que na existência de uma Punição, esta é publicada por uma Norma. De forma similar a propriedade *ruledBy* é inversa de *relates*, pois a Norma relaciona um ou mais Papéis e cada Papel deve ser regrado por uma Norma. Um exemplo de regramento são as atribuições de um chefe de Estado que são determinadas por uma legislação específica, todos por um artigo de lei.

A ontologia do domínio deve possuir instâncias sobre os agentes, papéis, recursos e lugares. Esses elementos na ontologia permitem à ontologia a capacidade de servir como referencial de mundo legislativo completo, sem a necessidade de outras bases, ensinando aos agentes sobre o seu espaço. Por fim, esta Seção forneceu uma ontologia legal, atendendo ao primeiro objetivo específico descrito na Seção 1.1.

3.3 O *Middleware*

Nas Seções anteriores foi discutida a importância da ontologia em formalizar e distribuir conhecimento. Independente das entidades serem agentes, ou outros tipos de componentes, é reconhecida a importância de fornecimento de informação semântica para os agentes. Além de compartilhar uma compreensão sobre algo, as ontologias fornecem capacidades de raciocínios e sistemas de suporte a decisão para estas entidades de software.

De acordo com o apresentado nas Seções 2.1.2 e 2.2.2, é perceptível a gama de implementações e integrações entre agentes e ontologias. Dependendo da plataforma a ser utilizada é possível conceber um agente acessando diretamente a ontologia, ou utilizando componentes intermediários para este fim. A variância das integrações ficam limitadas às capacidades que os agentes possuem dentre as plataformas comentadas anteriormente.

Enquanto algumas plataformas possuem acesso nativo à ontologia, outras demandam que bibliotecas ou API sejam conectadas intermediando o acesso às bases ontológicas. Um benefício técnico de possuir acesso às ontologias de dentro da plataforma é que estas implementações tendem a não corromperem quando a plataforma recebe alguma atualização, uma vez que fazem parte do pacote de programação, diminuindo a dependência de suporte de terceiros. Uma vez que o componente nativo faz parte da suíte de desenvolvimento da plataforma, sempre receberá manutenção e será testado em conjunto com outros componentes no lançamento de cada nova versão. O malefício é a necessidade de conhecimento sobre manipulações de ontologias.

Na maioria das vezes, nos projetos classificados como “não” na coluna de SMA da Tabela 1, para fornecer aos agentes os benefícios da ontologia, se faz necessário uma certa especialidade do(a) pesquisador(a) na área de ontologias. Um problema ocorre quando o desejado é conectar uma ontologia com o sistema multiagente e seguir para com a sua utilização, sem a necessidade do conhecimento de manipulações OWL/RDF e da lógica de consultas SPARQL ou outros mecanismos comuns na área de ontologia.

Além disso, fornecer uma implementação nativa de integração com a ontologia pode ocasionar na dependência com a plataforma, fazendo que sejam necessárias muitas adaptações para que a mesma lógica possa ser utilizada em outras plataformas. Este é um ponto negativo no uso de ontologias, uma vez que a prerrogativa ontológica é compartilhar conhecimento comum em meio computacional.

Nesta Tese, a ideia é fornecer um meio termo onde o mecanismo de acesso a ontologia possa ser simplificado aos desenvolvedores de agentes, de modo que o foco seja no componente em si e não nas plataformas onde este possa ser utilizado. Este modelo tem como inspiração a arquitetura de integração vertical de WEYNS et al. (2009),

onde a preocupação da plataforma de agentes é maior com o envio e a recepção de dados com o *middleware*. Em adição, há o nível de aderência à outras plataformas, semelhantes às integrações diretas como em FREITAS et al. (2015), com foco em componentes integrados para melhor interoperabilidade em diferentes plataformas fornecendo funcionalidades extras até então nativamente inexistentes, bem preconizado por SCHULDT; GEHRKE; WERNER (2008).

Devido a isto, a Figura 12 apresenta esquematicamente a proposta de um *middleware* encarregado do conhecimento sobre a ontologia legal fornecida nesta Tese. De acordo com a Figura 12, os grandes processos do *middleware* proposto são: i) receber ou capturar a ação do agente e o seu papel na sociedade simulada; ii) consultar a ontologia conectada, através de consultas SPARQL, em busca de leis que regulam tal atividade do agente; e, iii) retorno para a plataforma de SMA com as informações encontradas sobre normas.

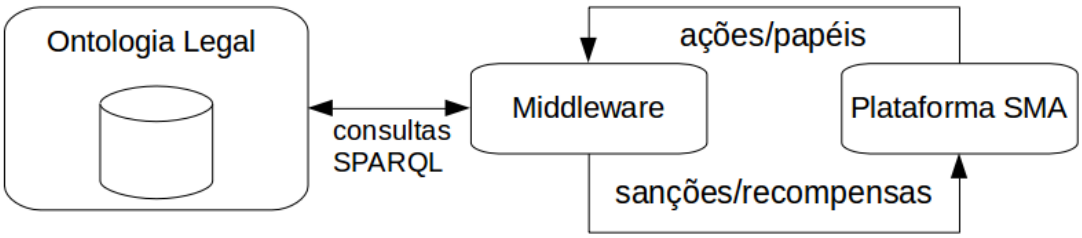


Figura 12 – Esquema do *Middleware* proposto.

A entrada deste modelo é composta pela ação e pelo papel do agente, pois as atividades que o agente estiver executando podem violar alguma legislação inerente. Em algumas legislações, existem leis que focam em regular a atividade de certos indivíduos na sociedade, isto é, alguns papéis nas sociedades podem possuir mais regulações que outros. Estas leis refletem esta característica quando definem diferente sanções para diferentes agentes em uma mesma violação, conforme descrito na Seção 3.2. O mecanismo de captura ou recepção das ações dos agentes é dependente das plataformas e será demonstrado com exemplos no Capítulo 4.

Em seguida, o modelo realiza pesquisas na ontologia. Existem duas formas de buscas de informações pelo *middleware* na ontologia e ambas ocorrem por consultas SPARQL. A consulta principal pode ser visualizada na Figura 13. Esta consulta realiza um processamento para encontrar por toda a ontologia, legislações que regulam a ação do agente. Nas linhas 10 e 11, a consulta se preocupa em iniciar a busca por instâncias ou indivíduos de *Action* com o foco na propriedade de relacionamento regulatedBy, visando instâncias de *Legislation*. Nas linhas 20 a 23, são verificadas as consequências existentes daquela legislação encontrada (normas), sendo que alguns campos como data final (linha 14) e comentários são opcionais, pois dependerá da lei e de como esta foi registrada pelos especialistas da ontologia.

```
1  PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2  PREFIX owl: <http://www.w3.org/2002/07/owl#>
3  PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
4  PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
5  PREFIX law: <http://www.semanticweb.org/fsp/ontologies/2018/11/agentdevlaw#>
6  SELECT ?action ?law ?description ?starts_at ?ends_at ?predicative ?norm
7          ?norm_desc ?norm_applied ?norm_type ?roles
8  WHERE {
9  {
10     ?action rdf:type law:Action .
11     ?action law:regulatedBy ?law .
12     ?law rdfs:comment ?description .
13     ?law law:starts_at ?starts_at.
14     OPTIONAL { ?law law:ends_at ?ends_at } .
15     FILTER(IF(EXISTS{ ?law law:ends_at ?ends_at },
16         "2020-01-30T10:44:36"^^xsd:dateTime >= ?starts_at &&
17         "2020-01-30T10:44:36"^^xsd:dateTime < ?ends_at ,
18         "2020-01-30T10:44:36"^^xsd:dateTime >= ?starts_at )
19     ) .
20     ?law ?predicative ?norm .
21     OPTIONAL { ?norm rdfs:comment ?norm_desc } .
22     ?norm law:apply ?norm_applied .
23     ?norm_applied rdf:type ?norm_type .
24     ?norm law:relates ?roles .
25 }
26   FILTER regex(str(?action), "agent|action", "i") .
27   FILTER(?roles = law:agentRole || ?roles = law:allRoles) .
28   FILTER(?norm_type != owl:NamedIndividual) .
29 }
30 ORDER BY ASC(?starts_at)
```

Figura 13 – Exemplo de consulta SPARQL na ontologia proposta.

Ainda, nas linhas 15 a 19 é apresentada uma estrutura condicional sobre a data final. Neste trecho da consulta é empregada uma busca em leis considerando a data vigente relacionada às datas nas leis existentes da ontologia. Isto ratifica a necessidade das leis possuírem data inicial de validade, mas algumas possuem uma data final pois podem existir leis válidas em um dado período de tempo (esta análise e formalização lógica foi abordada na Seção 3.2). Com isso, a opcionalidade bem como a união de registros com e sem datas finais, faz com que seja possível realizar consultas em todas as leis registradas, indiferente do período do ano e como é realizada a redação da legislação. As datas são preenchidas automaticamente pelo *middleware*, sempre a partir da data da consulta.

Nas linhas 26 e 28 da Figura 13, são definidos os filtros finais da consulta, uma vez que não se deseja encontrar todas as ações e, para cada uma, suas legislações relacionadas. O mecanismo recebe a atividade do agente e o tipo do agente, respectivamente, para filtrar as informações da ontologia com foco na específica atividade do agente. É esperado uma *string* descrevendo esta ação, geralmente representada por um verbo como correr, nadar, pescar, entre outros (pode ser um termo ou uma frase).

O filtro da linha 26 da Figura 13 procura por nomes de instâncias de *Action* (*?action*)

que combinem com o informado, através de uma pesquisa por expressões regulares. Mesmo que o usuário informe uma frase explicando a ação do agente, o *middleware* irá substituir os espaços em branco da frase pelo operador lógico *OR* (|) para que as consultas por expressões regulares não procure pela frase exata e sim pelas palavras que a compõe (por exemplo, “exportar peles” será automaticamente substituída por “exportar|peles”).

Quanto maior a quantidade de palavras inseridas no filtro, menor será o escopo para encontrar legislações pertinentes. Isto ocorre pelo fato do filtro realizar pesquisas por expressões regulares nos textos de lei inseridos na ontologia. Entretanto, para este funcionamento, é necessário que as ações sejam vinculadas na ontologia de forma correta (conforme exemplo na Figura 8).

Um problema característico são as variações dos verbos das ações, mesmo em inglês, que podem serem cadastrados de formas variadas, dependendo de como é redigida a legislação. Exemplos disso são as variações de pescar ou pescando (fish ou fishing), exportando (exports ou exporting), que por mais que sejam palavras diferentes, nesta ontologia, seriam ações reguladas pelas mesmas legislações. Porém, uma pode estar cadastrada enquanto outra não e isto pode causar falhas nas buscas, refletindo em uma menor compreensão do uso da ontologia legal pelos desenvolvedores de agentes.

Com o o objetivo de lidar com isso, foi desenvolvido dentro do *middleware* uma segunda forma de de percorrer as informações na ontologia. Uma segunda consulta SPARQL que foca nas *tags* `rdfs:comment` das instâncias do conceito *Legislation*. Com isso, ao invés de procurar inicialmente pelas ações existentes na ontologia, o *middleware* se preocupa em combinar os textos informados dentro daquelas *tags* com as ações dos agentes, onde o texto puro dos artigos de lei são anotados. Como o filtro de pesquisa de ações ocorre por busca de expressões regulares, também pode ocorrer discrepâncias entre os termos informados e os termos existentes nas anotações de comentário de *Legislation*. Na tentativa de minimizar tais efeitos, é aplicado o o cálculo de distância cosseno[4], empregado em estudos de similaridades de texto, para ranquear o retorno da lista de legislações. Assim as legislações do início da lista, tendem a fazerem mais sentido com as ações pesquisadas.

Para ilustrar as duas formas de busca, considera-se como exemplo a seguinte informação de ação, repassada ao *middleware* pelos SMA: *“hunting animals and exports skins”*. Neste caso, por padrão, pretende-se encontrar ações da ontologia relacionadas a *hunting* e *exports*.

A Figura 14 denota o *log* do sistema, exibindo que encontrou a instância `law-5197_article1_0`. Somente foi encontrado uma única informação porque na ontologia esta legislação está definida como única reguladora da ação *hunting*.

[4] `https://www.sciencedirect.com/topics/computer-science/cosine-similarity`

```
Law -> law-5197_article1_0
Description -> Animals of any species, at any stage of their development and living
naturally out of captivity, constituting the wildlife, as well as
their nests, shelters and natural breeding grounds are State property,
and their use, pursuit, destruction, hunting are prohibited or catch
it.
Norm -> _article1_par-1
Norm consequence -> detention-1_3 (Confinement), The consequence detention-1_3 has properties values:
Variable = min_jail, Data = 1, Data type = integer
Variable = max_jail, Data = 3, Data type = integer
```

Figura 14 – Log contendo exemplo de busca de legislações por ações pelo *middleware*.

Se configurado para utilizar a segunda forma de pesquisa, pelos textos informados nos comentários de Legislation, aumenta a probabilidade de retornar mais informações. Na Figura 15 além da instância `law-5197_article1_0` também foram encontradas as instâncias `law-5197_article18_0` e `law-5197_article3_0`. Isto ocorreu pois as demais legislações não possuem relações com as ações informadas pelo SMA. Porém, como pode ser visualizado nas linhas de `Description` da Figura 15, a lista é ranqueada de acordo com o nível de similaridade calculado.

```
Law -> law-5197_article18_0
Description -> Art. 18. Exports of raw amphibian and reptile skins and hides abroad
are prohibited.
Norm -> _article18
Norm consequence -> detention-2_5 (Confinement), The consequence detention-2_5 has properties values:
Variable = min_um_type, Data = year,integer, Data type = string
Variable = min_jail, Data = 2, Data type = integer
Variable = max_um_type, Data = year,integer, Data type = string
Variable = max_jail, Data = 5, Data type = integer
Law -> law-5197_article3_0
Description -> Art. 3º The trade in wildlife specimens and products and objects that
entail hunting, stalking, destroying or harvesting is prohibited.
Norm -> _article3_par-1
Norm consequence -> detention-2_5 (Confinement), The consequence detention-2_5 has properties values:
Variable = min_um_type, Data = year,integer, Data type = string
Variable = min_jail, Data = 2, Data type = integer
Variable = max_um_type, Data = year,integer, Data type = string
Variable = max_jail, Data = 5, Data type = integer
Norm consequence -> license (Permission), The consequence license don't have property values
Law -> law-5197_article1_0
Description -> Animals of any species, at any stage of their development and living
naturally out of captivity, constituting the wildlife, as well as
their nests, shelters and natural breeding grounds are State property,
and their use, pursuit, destruction, hunting are prohibited or catch
it.
Norm -> _article1_par-1
Norm consequence -> detention-1_3 (Confinement), The consequence detention-1_3 has properties values:
Variable = min_jail, Data = 1, Data type = integer
Variable = max_jail, Data = 3, Data type = integer
```

Figura 15 – Log contendo exemplo de busca de legislações por similaridades de texto, pelo *middleware*.

Devido ao termo *skins*, a instância `law-5197_article18_0` ficou no topo da lista, denotando sua provável maior importância de legislação para os itens informados pelo SMA. Mesmo que esta instância do topo não seja reconhecida pelos desenvolvedores do SMA como a mais pertinente, toda a lista é retornada. A ideia central é tentar

melhorar as buscas de informações legais. A busca por ações, que procede conforme consulta SPARQL da Figura 13, é a padrão. A troca de tipos de buscas ocorre através de um parâmetro no arquivo de configuração, fornecendo liberdade de escolha, em tempo de execução, aos usuários deste modelo.

Não é obrigatório informar ao *middleware* o tipo do agente para as consultas, pois o filtro da linha 27 (Fig. 13) contém posteriormente ao seu operador *OR* a especificação *allRoles*. Isto faz com que sejam filtradas leis para o tipo de agente especificado pela plataforma de agentes ou quaisquer outras leis que são endereçadas a todos os agentes, pois é uma instância padrão de *Roles* da ontologia, uma lei é para todos ou especificada para alguns.

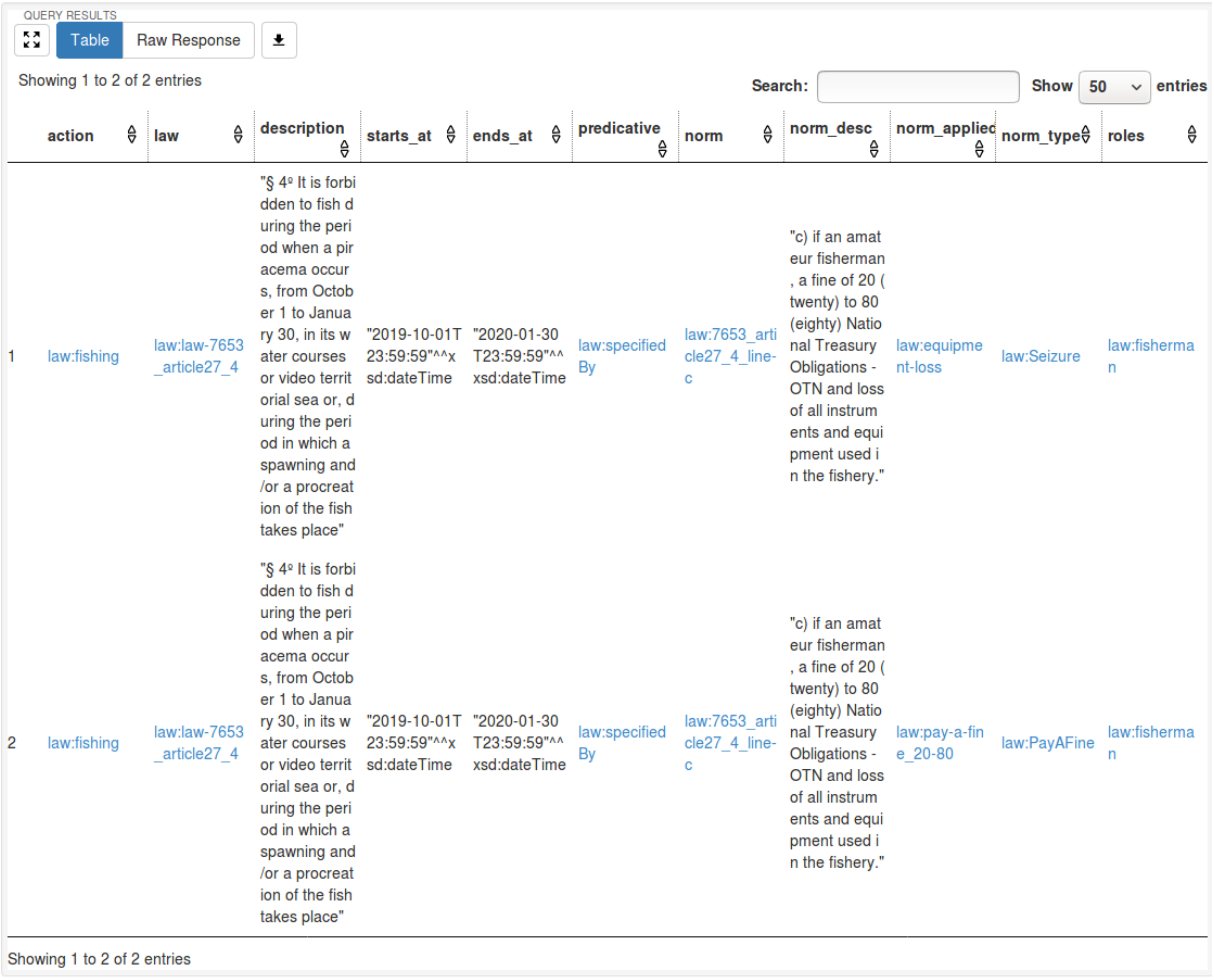
O artigo 27 da lei 7653 (BRASIL, 1988), discutida na Seção 3.2, faz parte de um conjunto de artigos que foram vetados de uma lei anterior 5197 (BRASIL, 1967), pois possuem uma nova redação. Enquanto a lei anterior de 1967 foi a primeira em publicar o artigo 27, a lei que contém o artigo com nova redação (atualização) é de 1988. O artigo na lei mais nova passa a vigorar enquanto o mesmo na lei antiga é removido (na lei antiga o artigo é tachado e a menção para a lei mais nova é fornecida). Como é esperado a probabilidade de mais de uma lei ser retornada pelas pesquisas na ontologia, incluiu-se na busca uma forma de apresentar as leis da mais recente para a mais antiga (linha 30 da Figura 13).

A Figura 16 demonstra os resultados da consulta SPARQL relacionada a procura do termo *fish* (pescar) com o papel (role) de *fisherman* (pescador). Este exemplo reflete a especificação do parágrafo 4 do artigo 27 da lei 7653 (BRASIL, 1988). Conforme descrito na Seção 3.2, este dispositivo legal menciona que em caso da época de piracema, se um pescador amador realizar a ação pesca, deverá receber como sanção multa e perda de equipamentos. Os resultados da Figura 16 alcançam estas informações das relações na ontologia proposta.

A ontologia legal deste trabalho é fornecida de duas maneiras: i) o arquivo OWL da ontologia em conjunto com o componente de intermediação; e, ii) através de um serviço web. O serviço web empregado é o Fuseki[5] onde é possível fornecer um *end-point* de consultas de ontologia (inclusive executar consultas SPARQL diretamente, como visualizado nas Figuras 13 e 16).

Isto é um benefício pois os desenvolvedores em suas simulações não precisam se preocupar em manter cópias das ontologias junto de suas plataformas. Há um benefício também em possuir acesso aos arquivos das ontologias, que é o de ter o acesso quando não existe conexão com a Internet (este é considerado como padrão para o uso do *middleware*). De qualquer forma as duas opções são descritas na documentação do modelo em seu repositório.

[5] `https://jena.apache.org/documentation/fuseki2`



QUERY RESULTS

Table | Raw Response

Showing 1 to 2 of 2 entries

Search: Show 50 entries

	action	law	description	starts_at	ends_at	predicative	norm	norm_desc	norm_applied	norm_type	roles
1	law:fishing	law:law-7653_article27_4	"§ 4º It is forbidden to fish during the period when a piracema occurs, from October 1 to January 30, in its water courses or video territorial sea or, during the period in which a spawning and/or a procreation of the fish takes place"	"2019-10-01T23:59:59"^^xsd:dateTime	"2020-01-30T23:59:59"^^xsd:dateTime	law:specifiedBy	law:7653_article27_4_line-c	"c) if an amateur fisherman, a fine of 20 (twenty) to 80 (eighty) National Treasury Obligations - OTN and loss of all instruments and equipment used in the fishery."	law:equipment-loss	law:Seizure	law:fisherman
2	law:fishing	law:law-7653_article27_4	"§ 4º It is forbidden to fish during the period when a piracema occurs, from October 1 to January 30, in its water courses or video territorial sea or, during the period in which a spawning and/or a procreation of the fish takes place"	"2019-10-01T23:59:59"^^xsd:dateTime	"2020-01-30T23:59:59"^^xsd:dateTime	law:specifiedBy	law:7653_article27_4_line-c	"c) if an amateur fisherman, a fine of 20 (twenty) to 80 (eighty) National Treasury Obligations - OTN and loss of all instruments and equipment used in the fishery."	law:pay-a-fine_20-80	law:PayAFine	law:fisherman

Showing 1 to 2 of 2 entries

Figura 16 – Exemplo de resultados retornados pela consulta SPARQL.

Na Seção 3.2, a ontologia fornecida neste modelo define que uma *Legislation* é especificada por uma ou mais normas (*Norm*) que aplicam uma série de sanções ou recompensas (as consequências de ações). Uma ação de um agente pode ocasionar a violação de uma ou mais leis, dependendo tanto da atividade ocorrida quanto do papel que esta entidade desempenha dentro da sociedade de agentes. Neste sentido, uma ação de um agente pode ocasionar o encontro de uma ou mais leis e de uma ou mais normas aplicando as sanções. Neste caso o *middleware* processa uma lista de cada, no formato orientado a objeto, para então entregar em Java as informações ao sistema multiagente desejado (o que facilitaria o uso pelos desenvolvedores de agentes).

Para implementar esta situação, foram criadas duas classes: *Law* e *Norm*. Estas classes refletem as informações oriundas da ontologia, focada em processar os dados a fim de entregá-los aos agentes das plataformas. A classe *Law* do *middleware* não possui o mesmo nome que o conceito da ontologia (*Legislation*) por decisão de

implementação, uma vez que o nome do pacote dos componentes é legislation[6]. Assim, é entregue ao SMA uma lista de objetos da classe *Law* que possui uma lista de objetos de *Norm* e, por consequência, uma lista explicativa de instâncias da classe *Consequence*.

Os tipos de dados do retorno das consultas com a ontologia e do retorno do componente para com o sistema multiagente são diferentes. A ideia é manter a simplicidade, dentro da orientação a objeto, sem a necessidade do utilizador conhecer os tipos de dados praticados internamente pela ontologia[7].

Esta é a responsabilidade da classe *Consequence*, enquanto *Norm* descreve as normas, *Consequence* descreve os tipos de dados que estas normas podem ter. O conceito de multa será entregue por *Norm*, entretanto as informações sobre o tipo de dado de multa (inteiro ou ponto flutuante) fica a cargo de *Consequence* informar. O mesmo é considerado para outras consequências. Uma pena de reclusão de 2 meses a 2 anos é informado por uma norma, entretanto o descritor sobre a prisão ser em meses ou anos, também fica a cargo de *Consequence*. Esta decisão de projeto em dividir entre instâncias diferentes e não centralizar os descritores em *Norm* se justifica em tentar diminuir a quantidade de contextos múltiplos (o que é da ontologia e o que é entregue ao SMA) das classes, seguindo o princípio de responsabilidade única do *design* de orientação a objetos (WAMPLER, 2007).

Para possibilitar a instanciação de novas leis dentro da ontologia, a partir do SMA, é fornecida uma funcionalidade para viabilizar a rápida definição de novas relações a partir do modelo proposto (um exemplo de uso da funcionalidade é encontrado no repositório da aplicação[8]). A premissa é se basear nas regulamentações de redação das leis brasileiras descritas na Seção 3.1.

Independente de ser uma nova lei ou atualização de uma antiga, o modelo deve receber as informações e criar um nova lei na ontologia (com nova numeração). Desta forma, são esperadas pelo usuário do modelo as informações sobre uma nova *Legislation*, para qual papel é destinado a legislação (*Role*), uma ou mais normas (*Norm*) contendo os detalhes das sanções (multa, prisão, entre outros).

Nenhuma destas informações precisam necessariamente existir previamente na ontologia, indivíduos e predicados de relações serão construídos automaticamente se for necessário. Ainda não há a inserção pelo modelo de tipificação de dados, conforme explicações da função da classe *Consequence*. Entretanto, novas informações legais na ontologia são possíveis de serem enviadas do SMA (uma simulação deste procedimento será apresentada na Seção 4.3).

[6] https://github.com/fabiosperotto/agentdevlaw/tree/master/src/br/com/agentdevlaw/legislation

[7] https://www.w3.org/TR/rdf-schema

[8] https://github.com/fabiosperotto/agentdevlaw/blob/master/src/br/com/agentdevlaw/NewLawExample.java

O desenvolvimento de todos os componentes envolvidos com a ontologia utiliza como base o projeto Apache Jena[9]. Jena reúne um conjunto de componentes importantes para manipular documentos semânticos desenvolvidos sob o padrão de recomendação web W3C[10]. Este *framework* manipula grafos OWL/RDF, fornece *parsers* para processar os documentos e componentes de consultas para utilizar SPARQL nas ontologias (CARROLL et al., 2004), além da capacidade de incluir raciocinadores na aplicação para inferências adicionais (AMEEN; KHAN; RANI, 2014).

Assim, o *middleware* é fornecido através de um componente que engloba todos recursos com o arquivo da ontologia legal anexado, para que possa ser reutilizado em plataformas variadas da área de SMA (atendendo o objetivo específico 2 da Seção 1.1) no formato de uma biblioteca (neste caso um arquivo .jar).

O Capítulo a seguir será destinado para os ensaios e testes em cenários de simulação social, com diferentes plataformas, para comprovação de sua utilização. Todos os componentes e mais informações de utilização do *middleware* encontram-se no repositório do projeto em `https://github.com/fabiosperotto/agentdevlaw`.

[9] `https://jena.apache.org/`
[10] `https://www.w3.org/RDF`

4 ENSAIOS DE APLICAÇÕES

Os variados ambientes de simulações de agentes fornecem benefícios e funcionalidades que podem resolver as necessidades de pesquisadores (em variadas linguagens de programação). Neste Capítulo será apresentado, com exemplos de simulação, como o modelo proposto pode ser integrado nestes ambientes, favorecendo as aplicações com conhecimento das leis da sociedade (neste ponto, leis brasileiras).

A simulação principal, aqui referida, será da atividade pesqueira de um pescador e um sistema governamental. O pescador é um agente comum que procura realizar a sua tarefa na simulação. O sistema governamental pode ser um ou mais agentes envolvidos, como governantes e fiscais, na regulação das atividades e recursos na sociedade de agentes. Os exemplos descritos a seguir não são únicos e visam apenas fornecer a aplicação e validação prática do modelo proposto nesta Tese. Ainda, conterá outras simulações relacionadas aos testes de alterações de leis na ontologia pelos agentes. Portanto, este Capítulo condicionado a aplicação da proposta, tem por objetivo testá-la e responder ao objetivo específico 3, descrito na Seção 1.1.

4.1 JaCaMo

O ambiente JaCaMo fornece uma plataforma de programação orientada a multiagentes e tem sido utilizado em projetos de mundo real baseado em agentes (BOISSIER et al., 2013). A plataforma JaCaMo combina outras três tecnologias no desenvolvimento de sistemas multiagentes: Jason[1] para programação de agentes autônomos, CArtAgO[2] para programação de artefatos de ambiente e Moise[3] para programação de organização de agentes. Existem algumas possibilidades da utilização do presente modelo a ser aplicado nas simulações nesta plataforma, pois dependerá da lógica de implementação de cada desenvolvedor(a). A principal utilização ocorre através de artefatos CArtAgO. A Figura 17 demonstra um exemplo de integração entre o *middleware* e o ambiente destino de programação.

[1] `http://jason.sourceforge.net`
[2] `http://cartago.sourceforge.net`
[3] `http://moise.sourceforge.net`

```
1  import java.util.*;
2  import br.com.agentdevlaw.legislation.Law;
3  import br.com.agentdevlaw.legislation.Norm;
4  import br.com.agentdevlaw.middleware.QueryProcess;
5  import br.com.agentdevlaw.ontology.OntologyConfigurator;
6  import cartago.*;
7
8  public class Legislacao extends Artifact {
9
10   List<ObsProperty> list_aux_leis = new ArrayList<ObsProperty>();
11
12   void init() {
13
14   }
15
16   @OPERATION
17   void checkAction(String action) {
18
19     OntologyConfigurator ontology = new OntologyConfigurator();
20     ontology.setOrigin(OntologyConfigurator.SERVER);
21     QueryProcess middleware = new QueryProcess(ontology);
22     List<Law> laws = middleware.searchAction(action,getCurrentOpAgentId().getAgentName());
23
24     if(!laws.isEmpty()) {
25       for(int i = 0; i < laws.size(); i++) {
26         List<Norm> norms = laws.get(i).getNorms();
27         for(int j = 0; j < norms.size(); j++) {
28
29           System.out.print("Consequência da norma " +
30           norms.get(j).getConsequence() + " (" +
31           norms.get(j).getConsequenceType() + "), ");
32           signal("legal", norms.get(j).getConsequenceType(),
33           norms.get(j).getConsequence() );
34         }
35       }
36     }
37   }
38 }
```

Figura 17 – Artefato CArtAgO com aplicação do *middleware*.

As linhas 17 a 38 da Figura 17 descrevem uma ação do artefato onde disponibiliza o método `checkAction()` que recebe como parâmetro a ação do agente. Em seguida realiza a configuração e ativação do *middleware* que retornará uma lista contendo as leis e regulações encontradas (linhas 19 a 22). A linha 22 demonstra o envio da ação do agente e o seu papel exercido na sociedade, para as verificações com a ontologia, via método `searchAction()`. Na linha 32 a função `signal()` envia um sinal de "legal" para todos os agentes que estiverem focando[4] neste artefato. Isto é útil para que quando for encontrada alguma informação sobre aquela ação do agente, os agentes recebem um alerta e acreditam naquela informação (que existe alguma regulação, por exemplo).

[4]Em JaCaMo, o foco significa que o agente permanece prestando atenção em quaisquer modificações realizadas (ou informações enviadas) no artefato CArtAgO.

Após o artefato codificado, na tecnologia Jason, pode ser concebido um agente Governo (Figura 18), a fim de que possa ativar o acesso à legislação. No plano `!start` o agente Governo irá ativar a legislação (linha 6) disponibilizando-o para todo o sistema através da comunicação da linha 8.

```
1  /* Initial goals */
2  !start.
3  /* Plans */
4  +!start : true <-
5    .println("governo fazendo as leis.");
6    makeArtifact("legislacao", "simulalei.Legislacao", [], Legislacao);
7    focus(Legislacao);
8    .broadcast(tell, legislacao).
9
10 +violacao: true <-
11   .println("Recebi violacao").
```

Figura 18 – Agente Governo em Jason.

Ao receber a crença sobre alguma violação (linha 10) o mesmo poderá realizar alguma ação sobre esta informação (ver agente Sistema na Figura 19).

```
1  /* Initial goals */
2  !observe.
3  /* Plans */
4  +!observe : true <-
5    ?existeLegislacao(L).
6
7  +?existeLegislacao(LegId): true <-
8    lookupArtifact("legislacao", LegId);
9    focus(LegId);
10   println("Achei legislacao").
11
12 -?existeLegislacao(LegId): true <-
13   .wait(10);
14   ?existeLegislacao(LegId).
15
16 +legal(Conceito, Valor) : true <-
17   println("Recebi uma comunicacao sobre sancao do tipo = ", Conceito, " e de valor = ", Valor)
       ;
18   +sancao(Valor);
19   .perceive.
```

Figura 19 – Agente Sistema em Jason.

Ao receber a crença da ativação da legislação para toda a sociedade, o agente Pescador (Figura 20) utiliza o método `checkAction` para avaliar sua ação. O artefato, codificado na Figura 17, caso encontre alguma legislação pertinente sobre a ação, envia um sinal (na sua linha 32) para todos os agentes que estão focados neste artefato. Com isso, o agente Sistema na sua linha 16 (Figura 19) percebe que existiu uma infração cometida por um agente (neste caso a proibição da pesca).

```
1 /* Initial goals */
2 !start.
3 /* Plans */
4 +!start : true <-
5   .print("Pescador online").
6
7 +legislacao: true <-
8   checkAction("fish"). //ou pescar
```

Figura 20 – Agente Pescador em Jason.

Neste exemplo, optou-se em utilizar um outro agente chamado Sistema, sendo que este poderia agir como um fiscal ou assumir algum outro papel na sociedade, em que centraliza as informações de ações de agente e aplica as normas reconhecidas pelas leis vigentes na ontologia. Aqui, o agente Sistema apenas informa que possui conhecimento de infrações cometidas. As demais codificações existentes no agente Sistema (Figura 19) existem somente para que quando o governo ativar o artefato de legislação, este agente possa permanecer focado no mesmo artefato.

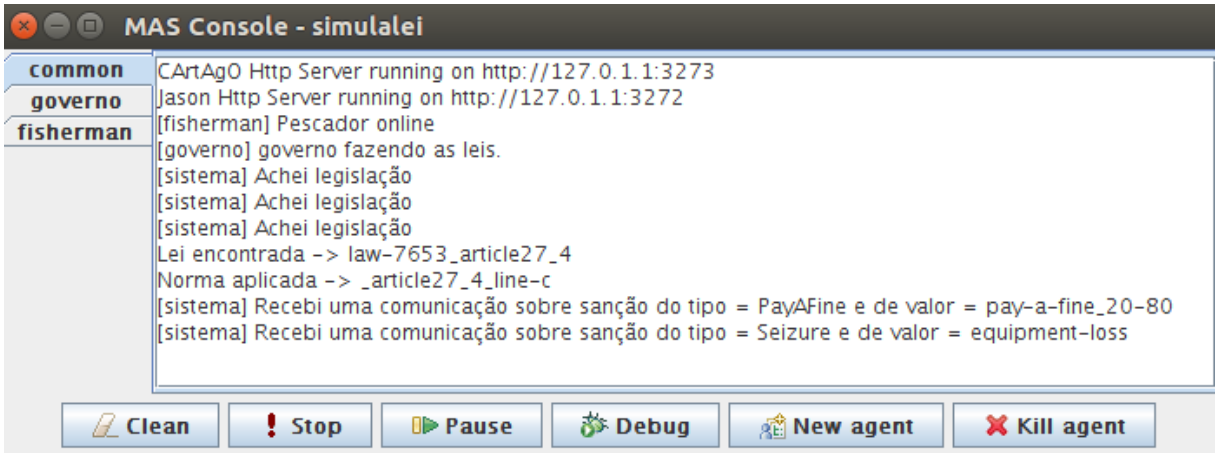


Figura 21 – Resultado da simulação com os agentes Governo, Pescador e Sistema.

A Figura 21 exibe os resultados desta simulação. Como o pescador estava em época de piracema, ocorreram duas infrações: a perda de equipamentos e o pagamento de multa. Neste caso, foi reconhecido como pescador amador, caso for outros papéis como pescador profissional ou empresa, outros tipos de sanções seriam comunicados.

Uma outra forma de integração ocorre através de ações internas da camada Jason da plataforma JaCaMo. Esta integração é útil caso as simulações não utilizem as demais camadas do sistema como o ocorrido anteriormente com artefato CArtAgO. A Figura 22 demonstra como criar uma ação interna onde através do método `execute()` (linha 10 a 47), são realizadas também as buscas na ontologia via o *middleware* proposto.

```
1  import java.util.List;
2  import br.com.agentdevlaw.legislation.*;
3  import br.com.agentdevlaw.middleware.QueryProcess;
4  import br.com.agentdevlaw.ontology.OntologyConfigurator;
5  import jason.*;
6
7  public class action extends DefaultInternalAction {
8
9      @Override
10     public Object execute(TransitionSystem ts, Unifier un, Term[] args) throws Exception {
11
12       //extraindo o tipo (papel) do agente
13       String[] agentTypePath = ts.getAg().toString().split(".*/");
14       String[] agentType = agentTypePath[1].split("\\.");
15
16       try {
17           StringTerm acao = (StringTerm)args[0];
18
19           OntologyConfigurator ontology = new OntologyConfigurator();
20         ontology.setOrigin(OntologyConfigurator.SERVER);
21         QueryProcess middleware = new QueryProcess(ontology);
22         List<Law> laws =  middleware.searchAction(acao.getString(), agentType[0]);
23
24         StringTerm result =  new StringTermImpl("empty");
25
26         if(!laws.isEmpty()) {
27
28           for(int i = 0; i < laws.size(); i++) {
29             System.out.println("Lei encontrada -> " + laws.get(i).getIndividual());
30             List<Norm> norms = laws.get(i).getNorms();
31             System.out.println("Norma aplicada -> " + norms.get(0).getIndividual());
32             for(int j = 0; j < norms.size(); j++) {
33
34               StringTerm positive =  new StringTermImpl(norms.get(j).getConsequenceType());
35               return un.unifies(positive, args[1]);
36             }
37           }
38         }
39
40           return un.unifies(result, args[1]);
41
42       }catch (ArrayIndexOutOfBoundsException e) {
43       throw new JasonException("A acao interna 'action' não recebeu 1 parametro obrigatório");
44     }catch (Exception e) {
45       throw new JasonException("Algum erro na execucao de internal action 'action'");
46     }
47     }
48 }
```

Figura 22 – Ação interna Jason com integração ao middleware.

No uso pelos agentes, um exemplo é exibido na Figura 23, onde o agente realiza a busca pela ação interna `eval.action()`[5] na linha 6. Assim, poderá acreditar que deve ou não algo para o Estado (infrações), através da crença `devendoEstado()` na linha 7. Todos estes detalhes e exemplos da Seção, podem ser encontrados no repositório `https://github.com/fabiosperotto/legislative-simulation`.

[5]Denota o nome do pacote onde se encontra a ação, seguido de '.' e do nome da ação.

```
1 /* Initial goals */
2 !start.
3 /* Plans */
4 +!start : true <-
5   .print("Pescador online");
6   eval.action("fish", A); //ou pescar
7   +devendoEstado(A).
```

Figura 23 – Agente pescador acessando ação interna.

4.2 JADE

A plataforma JADE (*JAVA Agent DEvelopment Framework*) é um *framework* completamente implementado na linguagem Java e que também implementa as especificações da FIPA (*Foundation for Intelligent Physical Agents*)[6]. Através de uma série de camadas internas de *middlewares* e componentes fornece um ambiente com capacidade de realizar qualquer tipo de simulação distribuída de agentes, incluindo até mesmo plataformas móveis com sistemas Android[7].

Para a simulação da aplicação da Tese, optou-se pela implementação de dois agentes: Governo (*Government*) e Pescador (*Fisherman*). Como esta plataforma não demanda uma estrutura orientada de simulação com artefatos e crenças (como no JaCaMo), dois cenários base para testar as hipóteses foram estipuladas. A Figura 24 demonstra a interface gráfica do JADE com os agentes participantes.

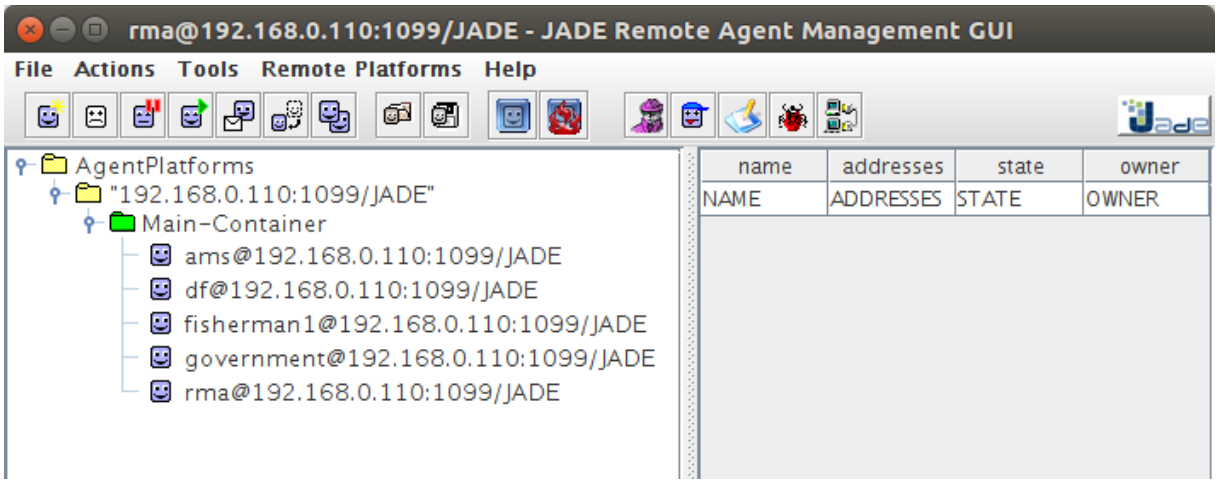


Figura 24 – Interface gráfica com os agentes participantes.

Nos dois cenários, o rio é fornecido como um recurso a ser disponibilizado nas *yellow pages* da plataforma JADE. As *yellow pages* são uma especificação de diretórios de serviços e JADE implementa a especificação *Agent Management Specification* da

[6] http://www.fipa.org
[7] https://jade.tilab.com/doc/tutorials/JadeAndroid-Programming-Tutorial.pdf

FIPA[8], onde os agentes podem registrar seus serviços enquanto outros podem procurar serviços a serem consumidos. Nesta plataforma, cada agente deve possuir um ou mais comportamentos (*behaviour*) que podem ser executados dependendo de ações ou por um período de tempo (a cada 10 segundos, por exemplo).

```
1  import jade.core.Agent;
2  import jade.domain.DFService;
3  import jade.domain.FIPAException;
4  import jade.domain.FIPAAgentManagement.DFAgentDescription;
5  import jade.domain.FIPAAgentManagement.ServiceDescription;
6
7  public class Government extends Agent{
8
9    protected void setup() {
10
11     System.out.println("Government Agent is ready!");
12     //registrando recurso de rio nas yellow pages
13     DFAgentDescription dfd = new DFAgentDescription();
14     dfd.setName(getAID());
15     ServiceDescription sd = new ServiceDescription();
16     sd.setType("river");
17     sd.setName("River resource");
18     dfd.addServices(sd);
19     try {
20       DFService.register(myAgent, dfd);
21     }catch (FIPAException fe) {
22       fe.printStackTrace();
23     }
24   }
25 }
```

Figura 25 – Agente Governo registrando recurso nas Yellow Pages.

Na Figura 25, o serviço rio é fornecido através do registro nas *yellow pages*, nas linhas 13 a 18. No primeiro cenário de simulação, o agente Governo fornece apenas o rio, simulando o fornecimento do acesso ao rio para que outros agentes possam consumir os recursos. O agente Pescador deve encontrar este recurso e realizar a pesca, se e somente se, verificado com a legislação se possui permissão para tal.

O pescador, cuja implementação pode ser vista na Figura 26, tem um comportamento de encontrar o recurso rio a cada 10 segundos (linha 11). Se conseguir acessar o recurso (linha 21), o mesmo irá utilizar o *middleware* para pesquisar se sua atividade é legalmente permitida (linhas 22 a 26). Caso não encontre nenhum retorno da legislação a respeito de sua ação, adiciona um segundo comportamento que é a ação propriamente dita (`FishPerformer()` na linha 28). Este segundo comportamento é implementado apenas como testes a partir da linha 53.

[8] `http://www.fipa.org/specs/fipa00023/XC00023H.html`

```
1  public class Fisherman extends Agent{
2    private String role = "fisherman";
3    public String getRole() {
4      return role;
5    }
6    public void setRole(String role) {
7      this.role = role;
8    }
9    protected void setup() {
10     System.out.println("Agent "+this.role+" is ready.");
11     addBehaviour(new TickerBehaviour(this, 10000) {
12       protected void onTick() {
13         System.out.println("Trying to fish");
14         DFAgentDescription template = new DFAgentDescription();
15         ServiceDescription sd = new ServiceDescription();
16         sd.setType("river");
17         template.addServices(sd);
18         try {
19           DFAgentDescription[] result = DFService.search(myAgent, template);
20           if(result.length > 0) { //existe recurso de rio nas yellow pages
21             System.out.println("Found a river resource!");
22             OntologyConfigurator ontology = new OntologyConfigurator();
23             ontology.setOrigin(OntologyConfigurator.MODEL);
24             QueryProcess middleware = new QueryProcess(ontology);
25             List<Law> laws =  middleware.searchAction("fish", role);
26             if(laws.isEmpty()) {
27               System.out.println("Theres no results against agent action 'fish'");
28                 myAgent.addBehaviour(new FishPerformer());
29             }else {
30               for(int i = 0; i < laws.size(); i++) {
31                 System.out.println("Law -> " + laws.get(i).getIndividual());
32                 List<Norm> norms = laws.get(i).getNorms();
33                 System.out.println("Norm -> " + norms.get(0).getIndividual());
34                 for(int j = 0; j < norms.size(); j++) {
35                   System.out.println("Norm consequence -> " + norms.get(j).getConsequence() + " (" + norms.get(j).
       getConsequenceType() + "), ");
36                 }
37               }
38               System.out.println("Fishing is not allowed");
39             }
40           }else {
41             System.out.println("River resource not found");
42           }
43         }
44         catch (FIPAException fe) {
45           fe.printStackTrace();
46         }
47       }
48     });
49   }
50   protected void takeDown() {
51     System.out.println("Fisherman "+this.role+" terminating.");
52   }
53   private class FishPerformer extends Behaviour {
54
55     public void action() {
56       System.out.println("Fisherman is fishing");
57     }
58
59     public boolean done(){
60       System.out.println("Terminating fishing");
61       return true;
62     }
63   }
64 }
```

Figura 26 – Agente Pescador verificando legalidade de seu ato.

A Figura 27 fornece o *log* dos resultados da simulação. Pode-se verificar que caso o Pescador receba a informação de que não é permitido pescar, retorna um ‘‘`fishing is not allowed`’’ sem proceder para a parte que recebe o comportamento da pesca. Se caso verificado pela legislação que não existem infrações, então seguirá para o fluxo do comportamento de pesca.

O segundo cenário remove a preocupação do agente Pescador em ter que verificar por si só se sua ação é legal. O agente Governo realiza a verificação na legislação e adiciona ou remove o recurso rio nas *yellow pages* quando necessário.

```
Agent fisherman is ready.
Government Agent is ready!
Trying to fish
Found a river resource!
Law -> law-7653_article27_4
Norm -> _article27_4_line-c
Norm consequence -> pay-a-fine_20-80 (PayAFine),
Norm consequence -> equipment-loss (Seizure),
Fishing is not allowed
```

Figura 27 – Resultado da simulação com os agentes Governo, Pescador.

```
1  import java.util.List;
2  import br.com.agentdevlaw.legislation.Law;
3  import br.com.agentdevlaw.middleware.QueryProcess;
4  import br.com.agentdevlaw.ontology.OntologyConfigurator;
5  import jade.core.Agent;
6  import jade.core.behaviours.TickerBehaviour;
7  import jade.domain.DFService;
8  import jade.domain.FIPAException;
9  import jade.domain.FIPAAgentManagement.DFAgentDescription;
10 import jade.domain.FIPAAgentManagement.ServiceDescription;
11
12 public class Government extends Agent{
13
14   protected void setup() {
15
16     System.out.println("Government Agent is ready!");
17
18     DFAgentDescription dfd = new DFAgentDescription();
19     ServiceDescription sd = new ServiceDescription();
20
21     addBehaviour(new TickerBehaviour(this, 6000) {
22       protected void onTick() {
23
24         OntologyConfigurator ontology = new OntologyConfigurator();
25         ontology.setOrigin(OntologyConfigurator.MODEL);
26         QueryProcess middleware = new QueryProcess(ontology);
27         List<Law> laws =  middleware.searchAction("fish", "fisherman");
28
29         if(laws.isEmpty()) {
30           //registrando recurso de rio nas yellow pages
31
32           dfd.setName(getAID());
33           sd.setType("river");
34           sd.setName("River resource");
35           dfd.addServices(sd);
36           try {
37             DFService.register(myAgent, dfd);
38           }
39           catch (FIPAException fe) {
40             fe.printStackTrace();
41           }
42
43         }else {
44           dfd.removeServices(sd);
45           System.out.println("Fishing is not allowed");
46         }
47       }
48     });
49   }
50 }
```

Figura 28 – Agente Governo registrando recurso nas *Yellow Pages* com permissão da legislação.

A implementação do agente Governo com estas habilidades são descritas na Figura 28. A cada 6 segundos (linha 21) o agente determina se for legalmente permitida a pesca segundo a legislação (linhas 24 a 29) registra o recurso rio (linhas 32 a 37). Remove o mesmo recurso quando verifica que não é mais legalmente permitido. Assim o agente Pescador simplesmente ao possuir acesso ao recurso (permissão), usa-o até que o mesmo não esteja mais disponível (permissão revogada).

Uma outra possibilidade de implementação é que a própria legislação possa ser disponibilizada e removida nas *yellow pages*, fornecendo conhecimento legal sobre determinadas ações. As implementações para esta plataforma podem ser encontradas em `https://github.com/fabiosperotto/legislative-simulation-jade`.

4.3 Simulação da Proposição de Novas Leis

Até aqui, demonstrou-se a utilização que agentes possam ter em aplicar o *middleware* em seus domínios e procurar por uma legislação relacionada às suas atividades. Além deste uso, é provável que uma determinada sociedade de agentes necessite criar suas leis e atualizar a ontologia legal, mantendo o registro destas alterações mesmo quando encerradas suas simulações.

A criação de leis brasileiras segue o rito determinado pelo regimento da república[9]. O processo legislativo determina que para a criação de novas leis, sejam criadas comissões específicas na área da lei para estudar e estruturar a mesma. As novas leis na área do meio ambiente, por exemplo, são debatidas na comissão de meio ambiente e desenvolvimento sustentável[10].

Quando a proposta estiver suficientemente madura, o parlamentar relator envia para a câmara dos deputados para avaliação e votação, podendo ir também para o senado. As duas casas podem aprovar ou reprovar o projeto de lei. Caso a proposta seja aprovada pelas casas legislativas, ela pode ser sancionada e publicada pela presidência.

Estes preceitos devem ser conduzidos dentro da simulação desejada, caso o objetivo dos agentes seja da proposição de novas leis e o *middleware* deve atender aos agentes em poder incluir novas leis na ontologia legal. No exemplo escrito nesta Seção é utilizada a plataforma JaCaMo apresentada na Seção 4.1 (todas as implementações estão no mesmo repositório descrito na Seção) e um conjunto de 15 (quinze) agentes. Neste exemplo, é necessário que um agente relator inicie e envie a proposta, que a nova lei seja aprovada nas duas casas legislativas e que seja sancionada pelo agente presidente.

A proposta de lei, na simulação, utiliza como exemplo a descrição do artigo 41

[9] `https://www.camara.leg.br/entenda-o-processo-legislativo/`

[10] `https://www2.camara.leg.br/atividade-legislativa/comissoes/comissoes-permanentes/cmads`

da lei nº 9605 (BRASIL, 1998b) que determina sobre o ato de “Provocar incêndio em mata ou floresta” com as possíveis sanções: Pena – reclusão, de dois a quatro anos, e multa. Inicialmente um agente relator procura na ontologia legal se esta proposta existe, conforme a Figura 29.

Na Figura 29 o agente relator utiliza uma ação interna (ver Seção 4.1) na linha 4 que aplica o *middleware* para averiguar se encontra uma legislação com a mesma descrição que a proposta. Neste ponto, a ideia é simplificar as rodadas de estudos e debates realizadas pela comissão responsável pela proposta. Para efeitos de exemplificação, o relator determina sobre o envio da proposta.

```
1  !start.
2  +!start : true <-
3    .print("Researching about law proposal: 'Cause a fire in the woods or forest'");
4    chamber.research("Cause a fire in the woods or forest", "allRoles", "firing", "Confinement>
       detention-2_4@PayAFine>pay-a-fine-500", P, D, Norm, Consequences, Action, Role);
5    !send_proposal(P, D, Norm, Consequences, Action, Role).
6
7  +!send_proposal(P, D, Par, S, A, R) : P = "no" <-
8      .print("Proposal that does not exist in current legislation: ", D);
9      .send(committee_chairman, tell, proposal(D, Par, S, A, R)).
10
11 +!send_proposal(P, D, Par, S, A, R) : P \== "no" <-
12     .print("Proposal '",D,"' exist in current legislation").
```

Figura 29 – Agente Relator pesquisando sobre a proposta de lei.

Caso encontrar uma lei que combine com esta, comunica isso na simulação e encerra a mesma (linha 11). Caso não encontrar, envia a mesma para o agente Presidente da Câmara dos Deputados (chamada na linha 5 da Figura 29, implementação nas linhas 7 a 9 da mesma Figura). Todas as informações sobre a lei são enviadas, os parâmetros da linha 7, por exemplo, determinam que a descrição, um parágrafo legal e uma *string*, contendo as sanções, sejam devidamente comunicadas.

A lista de sanções é uma *string* que contém os nomes do conceitos de normas da ontologia legal pertencente a este estudo e de seus valores, no formato `Conceito>valor` (isto será detalhado melhor ao final da simulação).

Ao receber a proposta de nova lei, o agente Presidente da Câmara recebe a comunicação (Figura 30) na linha 8, onde se entende que a comissão aprovou a proposta e está pronta para votação pelos deputados. Na linha 12 o agente presidente enviar uma comunicação para todos os agentes da simulação que a determinada proposta está em votação.

```
1  !start.
2
3  +!start : true <-
4    makeArtifact("voteBoard","congress.VoteBoard",[], VoteBoardID);
5    lookupArtifact("voteBoard", VoteBoard);
6    focus(VoteBoard).
7
8  +proposal_approved(D, Par, S, A, R) : true <-
9    .print("I received a new law proposal: ", D, " let's vote!");
10   !voting_law(D, Par, S, A, R).
11
12 +!voting_law(D, Par, S, A, R): true <-
13   .broadcast(tell, vote_proposal(D, Par, S, A, R)). //this will start the election
14
15 +polling(D, Par, S, A, R, V)[source(O)] : true <-
16   board(V, O, D, Par, S, A, R)[artifact_name("voteBoard")]. //here the votes are computed
17
18 +vote_ended(D, Par, S, A, R) : law_approved(X) & X = "yes" <-
19   .print("All voted, law is approved!");
20   .send(committee_chairman, tell, law_approved(D,"yes"));
21   .send(senator_president, tell, law_approved_chamber(D, Par, S, A, R)).
22
23 +vote_ended(D, Par, S, A, R) : law_approved(X) & X \== "yes" <-
24   .print("All voted, law is not approved");
25   .send(committee_chairman, tell, law_approved(D,"no")).
26
27 +law_published(N, D, Par, S, A, R) : true <-
28   .print("I received the law", N, ". Thank you Mr./Mrs. President").
```

Figura 30 – Agente Presidente da Câmara dos Deputados.

A implementação apresentada na Figura 31 demonstra que quando o Presidente dos Deputados enviou a comunicação, este percebe e crê que precisa fazer a votação relacionada. A linha 2 gera um número entre 0 e 1 que ao final será recebido como voto a favor (1) ou voto contra (0). Este número sempre é aleatório para fins de simulação e não tem relação com o sistema de crenças dos agentes. A linha 3 envia o voto para o presidente, sendo que este o recebe na linha 15 da implementação (Figura 30).

```
1 +vote_proposal(D, Par, S, A, R)[source(O)]: true <-
2   .term2string(N, math.random * 1 + 0);
3   .send(O, tell, polling(D, Par, S, A, R, N)).
```

Figura 31 – Implementação do agente Deputado.

Ao todo, neste exemplo, cinco deputados realizam a votação. O líder dos deputados recepciona os votos pelo artefato `congress.VoteBoard` onde armazena e computa os votos. Na linha 15 percebe-se uma configuração da plataforma onde é possível receber o nome do agente emissor da mensagem (`source(A)`), isto faz com que seja possível a votação nominal (voto possui nome do votante).

```
1  package congress;
2  import java.util.ArrayList;
3  import java.util.List;
4  import cartago.*;
5
6  public class VoteBoard extends Artifact {
7
8    List votes = new ArrayList();
9    int votesInFavor;
10   int votesAgainst;
11
12   void init() {
13     this.votesAgainst = 0;
14     this.votesInFavor = 0;
15     defineObsProperty("voters",5);
16     defineObsProperty("law_approved", "no");
17   }
18
19   @OPERATION
20   void board(Double vote, String politician, String lawDescription, String paragraph, String
       sanctionList, String action, String role) {
21
22     int voteProcessed = (int) Math.round(vote);
23     boolean infavor = false;
24
25     if(voteProcessed == 1) {
26       infavor = true;
27       this.votesInFavor++;
28       System.out.println("Politician = "+politician+" voted YES");
29     }
30
31     if(voteProcessed == 0) {
32       infavor = false;
33       this.votesAgainst++;
34       System.out.println("Politician = "+politician+" voted NO");
35     }
36
37     Vote voteCell = new Vote(politician, infavor);
38     this.votes.add(voteCell);
39     ObsProperty prop = getObsProperty("voters");
40       prop.updateValue(prop.intValue()-1);
41       if(prop.intValue() == 0) this.voteResults(lawDescription, paragraph, sanctionList, role)
       ;
42
43   }
44
45   void voteResults(String lawDescription, String paragraph, String sanctionList, String action
       , String role) {
46
47     System.out.println("Votes in favor = "+this.votesInFavor);
48     System.out.println("Votes against = "+this.votesAgainst);
49     if(this.votesInFavor > this.votesAgainst) {
50       ObsProperty prop = getObsProperty("law_approved");
51       prop.updateValue("yes");
52     }
53     signal("vote_ended", lawDescription, paragraph, sanctionList, action, role);
54   }
55 }
```

Figura 32 – Implementação do artefato `congress.VoteBoard`.

A implementação do artefato `VoteBoard` pode ser conferida na Figura 32. O artefato define algumas propriedades observáveis pelos agentes que focarem nele a respeito de quantos votantes faltam pra finalizar a votação e se a proposta foi aprovada (linhas 15 e 16). A operação ocorre pela função `board` na linha 20, onde recebe

o nome do parlamentar e processa se o seu voto é a favor ou contra (linhas 22 a 38 da Figura 32). Ao final da função, caso se encerrem a quantidade de votantes, é determinado se a proposta de lei foi aprovada ou reprovada (linhas 39 a 41 da Figura 32). Caso a proposta tenha sido reprovada pela maioria, é encerrada a simulação (linha 23 da Figura 30). Se a proposta for aprovada pelos deputados, a linha 50 (Figura 32) atualiza a propriedade que gera a crença da aprovação e envia um sinal de que estão encerradas as votações.

O Presidente dos Deputados ao receber esta confirmação dos votos e da aprovação da proposta, envia a mesma para o Presidente do Senado (linha 18 da Figura 30), que envia também uma comunicação da aprovação para o presidente da hipotética comissão de onde saiu a proposta (linha 20 da Figura 30).

```
1  !start.
2
3  +!start : true <-
4    makeArtifact("voteBoardSenate","congress.VoteBoard",[], VoteBoardSenateID);
5    lookupArtifact("voteBoardSenate", VoteBoardSenate);
6    focus(VoteBoardSenate).
7
8  +law_approved_chamber(D, Par, S, A, R) : true <-
9    .print("A received a law approved by Deputies: '", D, "' lets vote in Senate!");
10   !voting_in_senate(D, Par, S, A, R).
11
12 +!voting_in_senate(D, Par, S, A, R) : true <-
13   .broadcast(tell, vote_proposal_in_senate(D, Par, S, A, R)).
14
15 +polling_senate(D, Par, S, A, R, V)[source(O)] : true <-
16   board(V, O, D, Par, S, A, R)[artifact_name("voteBoardSenate")].
17
18 +vote_ended(D, Par, S, A, R) : law_approved(X) & X = "yes" <-
19   .print("All voted, law is approved!");
20   .send(president, tell, law_approved_senate(D, Par, S, A, R)).
21
22 +vote_ended(D, Par, S, A, R) : law_approved(X) & X \== "yes" <-
23   .print("All voted, law is not approved");
24   .send(deputies_president, tell, law_approved(D,"no")).
25
26 +law_published(N, D, Par, S, A, R) : true <-
27   .print("I received the law", N, ". Thank you Mr./Mrs. President").
```

Figura 33 – Agente Presidente do Senado.

As implementações do Presidente, do Senado e, dos senadores, são similares ao Presidente dos Deputados e dos deputados, respectivamente. A Figura 33 descreve as implementações do líder dos senadores. Este agente irá proceder da mesma forma que o líder dos deputados: realizar a votação da proposta recebida pela Câmara. Os cinco agentes senadores votarão exatamente da mesma estrutura que os deputados.

Caso os senadores votarem em sua maioria contra a proposta, a simulação é encerrada. Se votarem a favor, o Presidente dos Senadores envia uma comunicação ao Presidente da república para lidar com a parte final da proposição (linhas 18 a 20 da Figura 33).

```
1  !start.
2
3  +!start : true <-
4    makeArtifact("publication", "congress.Publication", [], PublicationID);
5    lookupArtifact("publication", Publication);
6    focus(Publication).
7
8  +law_approved_senate(D, Par, S, A, R) : true <-
9    .print("My attention is needed, a law was approved on senate");
10   newLaw(D, Par, S, A, R)[artifact_name("publication")].
11
12 +law_published(N, D, Par, S, A, R) : true <-
13   .print("I approve and sign this new law");
14   .broadcast(tell, law_published(N, D, Par, S, A, R)).
15
16 +law_publishing_error(N, D, Par, S, A, R) : true <-
17   .print("Something is wrong with this law '",D,"' and I cannot approve").
```

Figura 34 – Agente Presidente da República.

O agente Presidente da República, na implementação da Figura 34, ao receber a informação que a proposta está aprovada pelo senado (linha 8), determina a publicação desta através do artefato `congress.Publication` (linha 10).

É na codificação apresentada na Figura 35 onde o agente Presidente aplica o *middleware* proposto para se comunicar com a ontologia legal e atualizar a mesma. Após realizar todo o processamento das informações recebidas pela proposta de lei, tenta realizar a atualização da ontologia, via operação `newLaw` (linha 45). O presidente gera no artefato o número da lei, que neste exemplo foi implementado como aleatório nas linhas 24 e 25. Em caso de sucesso, emite um sinal (linha 45) avisando que a ontologia se encontra atualizada.

A classe da linha 37 da implementação (Figura 35), realiza o processamento da *string* de sanções, para processar a lista de normas, baseando-se nos separadores para determinar os conceitos e valores de normas que são as punições para quem infringir esta lei. É apenas uma decisão de simplificação do processamento para não obrigar a existência de listas ou conjunto de dados ao agentes desde a proposta do relator.

O relator envia somente a *string* com o formato padronizado que depois este será devidamente processado. No exemplo "`Confinement > detention-2_4@PayAFine > pay-a-fine-500`", o indivíduo `detention-2_4` (detenção de 2 a 4 anos) que é relacionado ao conceito *Confinement*, assim como o indivíduo `pay-a-fine-500` (pagamento de multa de 500 unidades monetárias) relacionada ao conceito *PayAFine*. Assim a lista de normas encontra-se pronta para ser atualizada com a ontologia.

```
1  package congress;
2  import java.util.ArrayList;
3  import java.util.Calendar;
4  import java.util.List;
5  import java.util.Random;
6  import br.com.agentdevlaw.legislation.Law;
7  import br.com.agentdevlaw.legislation.Norm;
8  import br.com.agentdevlaw.middleware.QueryProcess;
9  import br.com.agentdevlaw.misc.OntologyDate;
10 import br.com.agentdevlaw.ontology.OntologyConfigurator;
11 import cartago.*;
12
13 public class Publication extends Artifact {
14
15   void init() {
16
17   }
18
19   @OPERATION
20   void newLaw(String description, String paragraph, String sanctions, String action, String
       role) {
21
22     System.out.println("Processing a new law to the middleware");
23     Random r = new Random();
24     int lawNumber = r.nextInt((9999 - 1111) + 1) + 1111;
25     Law newLaw = new Law("law-"+lawNumber, description);
26
27     Calendar calendar = Calendar.getInstance();
28     newLaw.setStartDate(OntologyDate.createDateFormat(calendar.get(Calendar.YEAR),
29         calendar.get(Calendar.MONTH),
30         calendar.get(Calendar.DAY_OF_MONTH),
31         calendar.get(Calendar.HOUR),
32         calendar.get(Calendar.MINUTE),
33         calendar.get(Calendar.SECOND)));
34     List<String> actions = new ArrayList<String>();
35     actions.add(action);
36     newLaw.setActions(actions);
37     NormProcess normProcessor = new NormProcess(sanctions, paragraph, lawNumber, role);
38     List<Norm> norms = new ArrayList<Norm>();
39     norms = normProcessor.processListNorms();
40     newLaw.setNorms(norms);
41     OntologyConfigurator ontology = new OntologyConfigurator();
42     ontology.setOrigin(OntologyConfigurator.MODEL);
43     QueryProcess middleware = new QueryProcess(ontology);
44
45     if(middleware.insertNewLaw(newLaw)) {
46       System.out.println("A new law was published in legal ontology");
47       signal("law_published", lawNumber, description, paragraph, sanctions, action role);
48     }else {
49       signal("law_publishing_error", lawNumber, description, paragraph, sanctions, action,
       role);
50     }
51   }
52 }
```

Figura 35 – Artefato *congress.Publication*.

Durante o processo de atualização, o *middleware* irá adicionar a nova lei como um novo trecho em OWL, conforme consta neste exemplo, na codificação da Figura 36. O sinal é recebido pelo Presidente na linha 8 da implementação da Figura 34 onde o mesmo emite uma mensagem para todos os agentes alertando que assinou e publicou a nova lei. Um exemplo de recepção dessa comunicação pode ser vista na linha 26 do agente Presidente do Senado (Figura 33) onde o mesmo agradece pela publicação.

```
<owl:NamedIndividual rdf:about="http://www.semanticweb.org/fsp/ontologies/2018/11/agentdevlaw#firing">
  <law:regulatedBy>
    <law:Legislation rdf:about="http://www.semanticweb.org/fsp/ontologies/2018/11/agentdevlaw#law-9774">
      <law:specifiedBy>
        <law:Norm rdf:about="http://www.semanticweb.org/fsp/ontologies/2018/11/agentdevlaw#9774-_article1">
          <law:apply>
            <owl:NamedIndividual rdf:about="http://www.semanticweb.org/fsp/ontologies/2018/11/agentdevlaw#pay-a-fine-500">
              <rdf:type rdf:resource="http://www.semanticweb.org/fsp/ontologies/2018/11/agentdevlaw#PayAFine"/>
            </owl:NamedIndividual>
          </law:apply>
          <law:apply>
            <law:Confinement rdf:about="http://www.semanticweb.org/fsp/ontologies/2018/11/agentdevlaw#detention-2_4">
              <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#NamedIndividual"/>
            </law:Confinement>
          </law:apply>
          <law:relates>
            <owl:NamedIndividual rdf:about="http://www.semanticweb.org/fsp/ontologies/2018/11/agentdevlaw#allRoles">
              <rdf:type rdf:resource="http://www.semanticweb.org/fsp/ontologies/2018/11/agentdevlaw#Role"/>
            </owl:NamedIndividual>
          </law:relates>
        </law:Norm>
      </law:specifiedBy>
      <law:starts_at rdf:datatype="http://www.w3.org/2001/XMLSchema#dateTime"
      >2020-05-31T01:40:09</law:starts_at>
      <rdfs:comment>Cause a fire in the woods or forest</rdfs:comment>
    </law:Legislation>
  </law:regulatedBy>
  <rdf:type rdf:resource="http://www.semanticweb.org/fsp/ontologies/2018/11/agentdevlaw#Action"/>
</owl:NamedIndividual>
```

Figura 36 – Nova lei em OWL/RDF criado na ontologia pelo *middleware*.

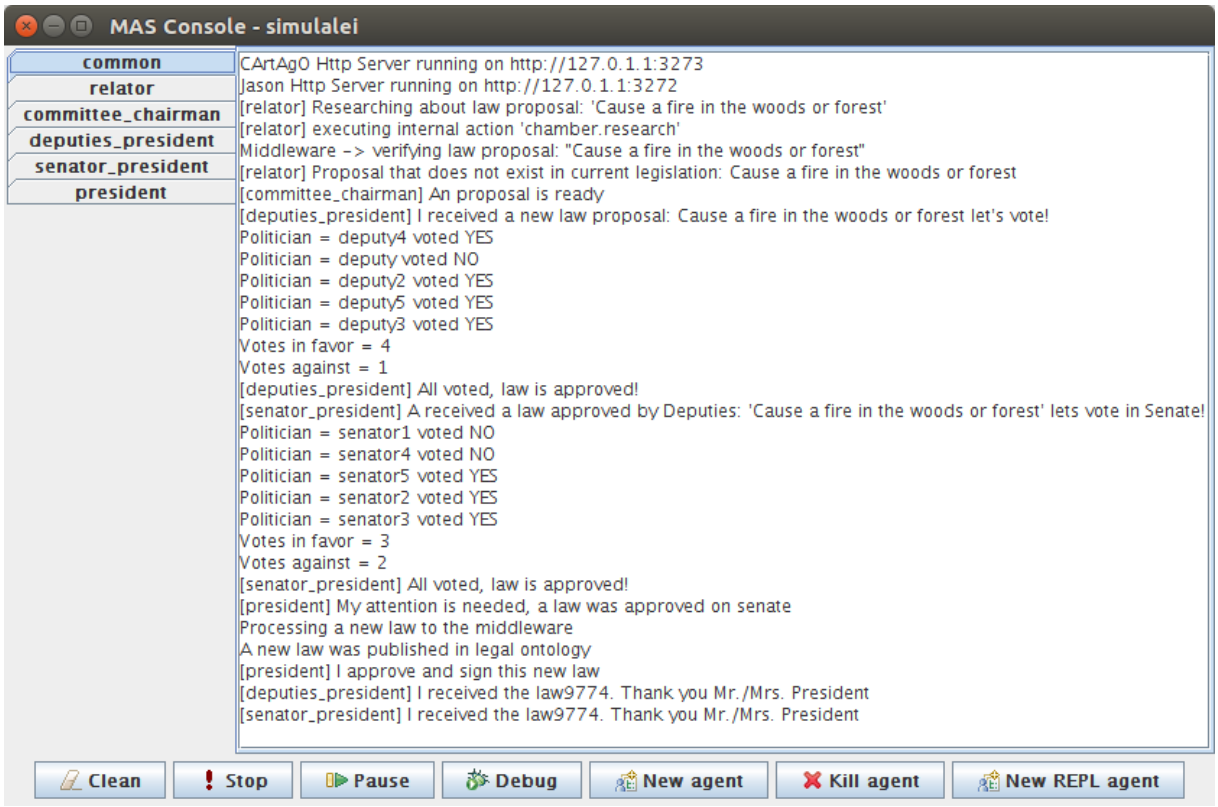


Figura 37 – Resultados da simulação de congresso.

A Figura 37 apresenta os resultados das interações quando todos os envolvidos aprovam por maioria e, então, a lei é publicada. Até o presente momento não existem funcionalidades para atualizações ou modificações nas leis, apenas a busca e a

inserção são as funcionalidades na versão 2.0.0-beta[11]. Novas funcionalidades serão fornecidas em futuras versões do *middleware*.

4.4 Discussões sobre Simulações e Plataformas

Para validar aspectos de ações dos agentes com uma legislação vigente, é necessário avaliar suas ações, como descrito nas simulações das Seções anteriores. O maior problema é encontrado na captura destas informações, sendo que em muitos casos é necessário que o agente realize o ato comunicativo de sua operação para as autoridades legais.

Dependendo dos aspectos de modelagem social aplicados nas simulações, há possibilidade do agente omitir suas operações (ato de mentir) para desviar de penalidades que possam ser creditadas para aquela sociedade. Nem sempre um agente social será obrigado a comunicar seus atos. Isto pode ser resolvido como uma obrigação na programação, mas não é uma obrigação social, uma vez que um agente criminoso não irá publicizar seus atos. O mais próximo da resolução deste problema é manter a crença do ato comunicativo para todas as operações, como uma via natural dos agentes no atos de fala (SINGH, 1991) e capturar estas mensagens sobre as ações para avaliação pela autoridade.

Uma outra alternativa de resolução deste problema, no ambiente JaCaMo, seria utilizar um *sniffer*[12] para capturar as mensagens de todos os agentes da plataforma. Nesta plataforma, exemplos de codificação de interceptação são fornecidas pelos mantenedores do projeto. Uma autoridade suprema poderia ser implementada e fiscalizar as atividades do cidadãos monitorando, por exemplo, os verbos de ação e verificando as prerrogativas legais de tais atos.

Entretanto, apesar da resolução técnica da intermediação nas comunicações dos agentes, isto pode gerar novos problemas de ordem social na sociedade simulada. Uma vez que toda e qualquer comunicação puder ser interceptada, surgem questões éticas como a violação de privacidade, das quais não são tratadas nesta Tese. Além disso, estes componentes são próprios da plataforma. Outras plataformas podem ou não conter o mesmo componente implementado em tecnologias iguais ou diferentes, inviabilizando uma possível integração e reuso associado ao *middleware* proposto.

Conforme exibido, a plataforma JADE realiza a centralização de todas as implementações na linguagem Java, além de possuir uma interface visual onde os agentes e informações podem ser manipulados em tempo de execução. A vantagem é de centralizar as simulações em uma única linguagem de programação. Outras formas de implementações que poderiam ser fornecidas para esta simulação seriam a troca de

[11] `https://github.com/fabiosperotto/agentdevlaw/releases`

[12] `http://jason.sourceforge.net/Jason/Examples/Entries/2007/6/20_Sniffer_(Jomi_Hubner).html`

mensagens, onde a plataforma segue à risca as especificações da FIPA. Entretanto, esta situação demandaria que o agente pescador enviasse uma mensagem ao agente governo (ato comunicativo), ou ao sistema, declarando que estaria realizando a pesca (vide simulação em JaCaMo).

A interceptação automática das ações demandaria também a aplicação de um *sniffer*[13] próprio existente, o que também não foi considerado pelas mesmas implicações sociais de violação de privacidade.

Apesar destas limitações, um outro benefício desta plataforma é uma API robusta para utilização com ontologias[14]. Classes inteiras podem ser implementadas e refletirem aspectos da ontologia legal deste estudo, sem demandar uma camada adicional para a comunicação com serviços de ontologia. Uma terceira ideia de implementação seria o consumo da ontologia legal e quais ações são legalmente permitidas ou violadas, diretamente dentro da plataforma. Como isto demanda conhecimento ontológico de quem apenas utiliza a plataforma e, pelo fato de que todas as implementações seriam nativas do ambiente JADE, então estas não serão abordadas como opção nesta Tese.

Conforme verificados nas Seções 4.1 e 4.2, o resultado das buscas retornaram indivíduos que descrevem, por exemplo, o pagamento de multa (`pay-a-fine_20-80` instância do conceito PayAFine). O valor dessa multa está compreendido entre 20 e 80 unidades (nomenclatura da instância), entretanto, para a utilização da informação da ontologia, a instância deve especificar melhor estes dados através de propriedades de dados (Figura 10 da Seção 3.2).

No middleware existe o método `getConsequenceValues()` onde este explica estes tipos de dados das instâncias. O método retorna informações de tipo e outras descrições sobre aquele determinado valor encontrado na instância. O objetivo é aumentar as compreensões pelos desenvolvedores de agentes, a respeito dos tipos de dados em esquema *XML* encontrados nestas propriedades de dados.

4.5 Comparando o AgentDevLaw com os Trabalhos Relacionados

Os trabalhos relacionados, apresentados na Tabela 1, descrevem a capacidade de resolução ou de atendimento sobre certas necessidades. Estas necessidades são refletidas pelos critérios distribuídos nas colunas: normas, funções legais, elementos de conceito, ontologia, implementação e SMA. A Tabela 2 também fornece um comparativo de trabalhos em relação a *middlewares*, através dos critérios: comunicação, *web*, ontologia, inferência, agentes, multiplataforma e *software*. Observa-se que os Trabalhos Relacionados não contemplam todos os critérios descritos.

[13] `https://jade.tilab.com/documentation/tutorials-guides/sniffer/introduction/`
[14] `https://jade.tilab.com/doc/api/jade/content/onto/Ontology.html`

Em relação à normas, alguns trabalhos descrevem a origem ou a definição do seu uso. No AgentDevLaw é deixado clara a origem e a aplicação das normas na ontologia. As normas advém das leis brasileiras e de alguns dos estudos referenciados na Seção 3.2 e, neste caso, as normas podem ser consideradas similares aos trabalhos UFO-C (GUIZZARDI; FALBO; GUIZZARDI, 2008), FREITAS (2017), FELICíSSIMO et al. (2005) e LKIF (ALEXANDER, 2009). Entretanto, o diferencial do AgentDevLaw reside em basear-se também no domínio jurídico brasileiro.

Os modelos Dolce (GANGEMI et al., 2002), Dolce-CORE (BORGO; MASOLO, 2009) e UFO (GUIZZARDI; FALBO; GUIZZARDI, 2008) são ontologias fundacionais (ontologias universais, de ampla descrição) e atendem bem as questões de conceituação de mundo. Entretanto, a ontologia legal do AgentDevLaw não visa contemplar todos os aspectos de uma sociedade (instituições, categorias, funções, entre outros). Além disso, realizar uma extensão daqueles trabalhos relacionados seria permanecer condicionado à mesma visão de mundo daqueles especialistas. A ontologia do AgentDevLaw visa ser um modelo de conhecimento jurídico diferenciado, podendo estar alinhado à outras ontologias mais abrangentes sobre a sociedade.

A LKIF fornece um método de coleta de conceitos jurídicos por profissionais da área, no AgentDevLaw isto é realizado através de análises da redação de leis, bem como na aceitação de conceitos legais comuns de outras ontologias. À primeira vista, este pode ser um ponto de vulnerabilidade no AgentDevLaw, a não avaliação por especialistas na área de Direito, que no campo da ontologia são chamados de especialistas de domínio. Porém, o trabalho de EL GHOSH et al. (2016) realiza de forma similar o raciocínio de sua construção ontológica sobre redações de leis (leis libanesas).

As funções legais, que determinam a responsabilidade legal de atos realizados pelos agentes, são fortemente especificadas em VALENTE; BREUKER; BROUWER (1999) e LKIF (ALEXANDER, 2009), onde o próprio modelo ontológico fornece tais mecanismos. No AgentDevLaw esta capacidade de modelagem é simplificada, quando comparada a VALENTE; BREUKER; BROUWER (1999), porque as funções legais entre papéis não são tão robustas no modelo. O foco está na transmissão das normas brasileiras sem fazer com que existam camadas de complexidades nas relações internas da ontologia, facilitando a disponibilização desta para outros domínios.

Os aspectos de modelagem de atos comunicativos da LKIF (ALEXANDER, 2009) são considerados como padrão interno aos agentes (e de suas plataformas), sendo que o AgentDevLaw não pretende replicar isso. O *middleware* deve ser empregado nas plataformas obtendo atitudes e papéis dos agentes, indiferente de como estão modelados. Especificamente, neste critério, AgentDevLaw apresenta-se como uma positiva alternativa em simplificar a modelagem para o ganho de adequação de uso em múltiplas plataformas de agentes.

A proposta do FOLaw (VALENTE; BREUKER; BROUWER, 1999) fornece, por

exemplo, as capacidades de indexação de diferentes ontologias e de categorização de conhecimentos, enquanto que na LRI-Core (BREUKER et al., 2005) existem camadas dentro da mesma ontologia. Estes itens podem ser úteis para dividir ou receber ontologias de domínios legais diferentes e isto não é contemplado nesta Tese. O foco do AgentDevLaw está no aspecto jurídico em uma única camada e, entretanto, o *middleware* poderia auxiliar em processar mais de uma ontologia, se necessário. Contudo, isto geraria outros aspectos a serem analisados e resolvidos, das quais não estão no escopo deste trabalho, como união e diferenciação de informações em ontologias.

Alguns estudos relacionados na Tabela 1 possuem a marcação de "não" no critério de ontologia por fornecerem todos os fundamentos, sem fornecer a ontologia para aplicação dos estudos. O mesmo ocorre na questão de implementação, os estudos também marcados como "não" residem nos aspectos conceituais dos modelos sem fornecimento de tecnologia.

A proposta de FREITAS (2017) realiza a geração de código para a plataforma JaCaMo diretamente pelo seu modelo. O AgentDevLaw não possui tais funcionalidades pois o foco é na obtenção do conhecimento legal, no seu uso dentro do SMA e no fornecimento das funcionalidades para múltiplas plataformas. Neste quesito são duas opções válidas e suas considerações de uso estão relacionadas ao nível de facilidade para os desenvolvedores de SMA (pelo menos do ambiente JaCaMO) e às necessidades da estruturação de bases de crenças dos agentes. A Tabela 3 visa incluir e posicionar o AgentDevLaw nos critérios comparativos entre ontologias que abordam o contexto legal (Seção 2.1).

Tabela 3 – Comparativo do AgentDevLaw com os trabalhos sobre ontologias e legislação.

Trabalhos	Normas	Funções Legais	Elementos de Conceito	Conhecimento reativo	Ontologia	Implementação	SMA
AgentDevLaw	**Sim**	**Não**	**Sim**	**Sim**	**Sim**	**Sim**	**Sim**
FOlaw (VALENTE; BREUKER; BROUWER, 1999)	Sim	Sim	Não	Não	Não	Outros	Não
LRI-Core (BREUKER et al., 2005)	Não	Sim	Sim	Não	Sim	Não	Não
Dolce (GANGEMI et al., 2002)	Não	Não	Sim	Não	Sim	Não	Não
Dolce-CORE (BORGO; MASOLO, 2009)	Não	Não	Sim	Não	Não	Não	Não
UFO (GUIZZARDI; FALBO; GUIZZARDI, 2008)	Sim	Sim	Sim	Breve	Sim	Não	Não
LKIF (ALEXANDER, 2009)	Médio	Sim	Sim	Não	Sim	Ontologia	Não
FREITAS (2017)	Sim	Não	Não	Não	Sim	Ont./Outros	Sim
DIGNUM; VÁZQUEZ-SALCEDA; DIGNUM (2005)	Sim	Não	Não	Não	Sim	Ontologia	Sim
FELICíSSIMO et al. (2005)	Sim	Não	Não	Não	Médio	Ontologia	Sim
VALENTE; BREUKER (1994)	Parcial	Sim	Não	Sim	Sim	Não	Não
GRIFFO; ALMEIDA; GUIZZARDI (2015)	Médio	Sim	Sim	Não	Sim	Não	Não
PALMIRANI et al. (2018)	Sim	Sim	Sim	Sim	Sim	Não	Não
EL GHOSH et al. (2016)	Não	Sim	Sim	Não	Parcial	Não	Não
SANTOS; COSTA (2012)	Sim	Não	Sim	Sim	Não	Outros	Sim
MOREIRA; COSTA; AGUIAR (2017)	Sim	Não	Sim	Não	Não	Outros	Sim

Em relação as capacidades de *middlewares* (para com ontologias e agentes) determinados pelos critérios na Tabela 2, também há a comparação entre os pontos de atendimento, dentre os quesitos estipulados.

Na parte da comunicação o AgentDevLaw preocupou-se em realizar não somente

o tráfego de dados entre sistemas (ontologias e agente) mas também com a tradução necessária. Esta tradução ocorre tanto pelas questões de acessar conhecimentos na ontologia em formatos não compreensíveis diretamente pelos agentes (OWL/RDF) tanto quanto no suporte aos desenvolvedores de agentes em reconhecer tipos de dados ontológicos (reconhecer o que é um tipo inteiro, se o dado é em dias, meses, entre outros aspectos). Neste quesito, é similar a todos da Tabela 2, porém se sobressai ao modelo proposto em GARCÍA et al. (2011).

No quesito *web*, o AgentDevLaw também se sobressai, principalmente frente aos estudos de LI et al. (2017), SCHULDT; GEHRKE; WERNER (2008), GARCÍA et al. (2011) e SPEROTTO; ADAMATTI (2012). Apesar do presente modelo não ser integralmente disponibilizado como serviço *web*, as suas configurações permitem o uso de serviços de consultas de ontologias (SPARQL) e, assim, permitindo o uso em aplicações com ou sem conexão a Internet.

Em relação a empregar ou não as ontologias com o *middleware*, os trabalhos relacionados se dividem em suas aplicações. Os benefícios da aplicação de se guiar pelo conhecimento fornecido pelas ontologias, nos estudos relacionados, são considerados cruciais para o sucesso dos modelos e o AgentDevLaw é mais uma opção a seguir o quesito.

Considerando o uso de diferentes ontologias, aqui também encontra-se outro fator em que outro modelo atende melhor quando existe a necessidade de múltiplas ontologias. O modelo proposto em LI et al. (2017) opera com uma rede de ontologias, sendo que o AgentDevLaw realiza a interpretação de conhecimento somente com o seu modelo de visão sobre a legislação.

No critério de inferência, da capacidade de processamento para deduções automáticas das informações trafegadas pelo *middleware*, existem estudos que fornecem plenamente tais capacidades, como de SERRANO et al. (2017) e de LI et al. (2017). O modelo AgentDevLaw proposto não fornece tais mecanismos.

Tabela 4 – Comparativo do AgentDevLaw com os trabalhos sobre *middlewares*.

Trabalhos	Comunicação	Web	Ontologia	Inferência	Agentes	Multiplataforma	Software
AgentDevLaw	**Intermediário**	**Sim**	**Sim**	**Não**	**Sim**	**Sim**	**Sim**
SERRANO et al. (2017)	Intermediário	Sim	Sim	Sim	Possível	Sim	Não
LI et al. (2017)	Intermediário	Não	Sim	Sim	Possível	Sim	Não
OLARU; FLOREA; SEGHROUCHNI (2013)	Intermediário	Sim	Não	Possível	Sim	Sim	Não
SCHULDT; GEHRKE; WERNER (2008)	Intermediário	Não	Não	Não	Sim	Não	Não
GARCÍA et al. (2011)	Um-para-um	Não	Não	Não	Sim	Não	Não
SPEROTTO; ADAMATTI (2012)	Intermediário	Não	Sim	Sim	Sim	Sim	Sim

No critério de multiplataforma, os trabalhos de SCHULDT; GEHRKE; WERNER (2008) e de GARCÍA et al. (2011) não demonstram a aplicação em diferentes ambientes de desenvolvimento. No critério de software, somente o trabalho de SPEROTTO; ADAMATTI (2012) realiza o fornecimento do software, tanto em descrição nos arti-

gos quanto em componentes para livre uso. O AgentDevLaw possui como premissa apresentada nos objetivos específicos (Seção 1.1) o atendimento destes itens, ser multiplataforma e de não ser software proprietário. A Tabela 4 visa incluir e posicionar o AgentDevLaw nos critérios comparativos entre os trabalhos que apresentam *middleware* (Seções 2.2.1 e 2.2.2).

O modelo AgentDevLaw reúne a intenção de duas opções para a área de agentes e de ontologias. Uma ontologia legal e um componente que realiza a integração de conhecimento com os agentes. Desta forma, apresenta-se como uma opção relevante dentre os dois mundos de propostas para trazer não só o conhecimento legal para a regulação dos agentes em multiplataformas, mas também fornecer um domínio especializado em leis brasileiras.

5 CONCLUSÃO

O papel dos agentes em simulações sociais ocorrem sob a perspectiva das sociedades humanas, sendo estas organizadas por arranjos sociais e instituições. Há a perspectiva dos cidadãos participando de trocas sociais com suas crenças, ou de autoridades preocupados com o bem estar e o equilíbrio nestas trocas.

Em sistemas multiagentes, através da simulação social, os estudos determinam métodos ou técnicas para que seja possível modelar as relações e utilizar como estimativas de planejamento. Sejam estes estudos para entes da esfera de planejamento público ou para estudos de movimentações sociais e suas análises estatísticas.

O modelo proposto procurou preencher aspectos de reconhecimento de informações legislativas sobre regulamentações de ações de agentes. A realização deste estudo evidenciou o panorama de regulação de ações em sistemas sociais, a avaliação das ações dos agentes em seu papel como cidadão dentro de uma sociedade.

Após as análises sobre aspectos da estruturação das leis brasileiras, bem como nos estudos relacionados, o presente estudo fornece uma ontologia legal para a área de SMA. Esta ontologia compõe conceitos e relações presentes nas estruturas de leis brasileiras, bem como alinhado a outros conceitos de normas e restrições nos comportamentos de agentes. A premissa foi a de fornecer o entendimento sobre leis em uma ontologia acessível aos agentes. A ontologia legal pode ser reutilizada em outros sistemas de acordo com outras necessidades.

Além da ontologia legal, também compreendeu-se no presente modelo um mecanismo de *middleware* para efetivar as devidas comunicações entre agentes e ontologia. O componente, aplicado em diferentes plataformas, visa a entrega do conhecimento da ontologia para os agentes. Os agentes conseguem buscar informações sobre leis e conferir suas ações com estas. Avaliando da possibilidade de variadas regulações e sanções atreladas as suas ações.

O estudo foi testado em diferentes cenários nas plataformas JaCaMo e Jade. Sendo que cada cenário procurou fornecer testes de possibilidades de uso do modelo proposto. Sejam cenários onde o próprio agente tem acesso as informações da ontologia a respeito de suas ações, ou, por outros agentes, intermediando e avaliando

as atitudes no SMA. Há também o fornecimento de simulações para a elaboração ou criação de leis pelos agentes, utilizando como referência os processos empregados pelas casas legislativas brasileiras (Câmara e Senado).

Os estudos no contexto da concepção de sociedade de agentes, sistemas multi-agentes, simulação social e demais expressões ou aspectos relacionados, tendem a continuar em evolução. Sejam elas na melhoria contínua de avaliações socioeconômicas em pesquisas e planejamento, assim como seu uso por entidades e pessoas interessadas em regulações automáticas de ações. Mais desdobramentos e correções são encontradas no modelo proposto, favorecendo novas concepções de aplicação de conhecimento da sociedade com os agentes.

Esta Tese fomenta também futuras pesquisas interdisciplinares, reforçando integrações entre as áreas computacionais e sociais, não se restringindo à área do Direito.

A contribuição da organização ontológica do conhecimento, diferente de outros métodos, visou clarear as relações e compreensões de conceitos tanto para os agentes computacionais quanto aos agentes desenvolvedores ou pesquisadores. O uso de outros métodos tais como sistemas de armazenamento de ontologias ou até sistemas de banco de dados (relacionais ou não), não podem ser descartados. Podem, futuramente serem considerados, principalmente em soluções híbridas com ontologias, uma vez que estas podem escalar em tamanho e complexidade de suas informações (bem como a sua necessidade de gerenciamento e manutenção).

As relações somente da parte de ontologia não são consideradas como imutáveis ou bala de prata. A proposição indica o estudo legislativo brasileiro, com isso, é imperativo seu contínuo desenvolvimento e amadurecimento, principalmente através de análises interdisciplinares (mais na área de Direito do que na Ciência da Computação). Incluindo-se o desenvolvimento de soluções comparativas entre sistemas jurídicos de outras nações e podendo, ainda, definir futuros modelos que representem a legislação de blocos geopolíticos.

A ontologia proposta pode demandar correções e melhorias, aumentando sua qualidade conforme o seu amadurecimento, através de mais análises, testes e aplicações. Em ontologias, as informações são as instâncias dos conceitos e, por isso, processá-las de forma adequada aumenta a qualidade da aplicação com ontologias.

Nos aspectos de raciocínio, por parte da ontologia, os predicados entre os conceitos podem receber melhorias em relação ao domínio de aplicação. Um exemplo são as exceções em normas da legislação. Existem casos em que leis podem proibir determinada ações, exceto se os atores destas possuírem permissões (licença). Com isso, tanto a ontologia legal e o *middleware*, poderiam receber mais informações do SMA a fim de aumentar o nível de interpretação das leis e de comunicação com os agentes.

Ainda, neste aspecto de compartilhamento de conhecimento, o modelo não rea-

liza a combinação de múltiplas bases ontológicas que possam constituir diferentes arranjos legais. Bases estas que podem refletir aspectos penais, civis, legislação de trânsito, entre outros, cada qual constituída através de especialistas, mas que mesmo individualmente, poderiam ser fornecidas de forma organizada e calculada frente as diferentes atitudes dos agentes.

O modelo proposto poderia fornecer de forma centralizada todos os aspectos legais de cada categoria, entretanto, conhecimentos jurídicos geralmente são divididos em bases diferentes, incluindo o acesso às legislações internacionais. Neste caso, o próprio *middleware* poderia calcular as atitudes e reconhecer padrões, pesquisando por legislações diferentes, nacionais e internacionais, através de serviços *web* ou de múltiplas bases ontológicas (benefício do modelo já ser aderente a serviços *web* de ontologia como Fuseki). Técnicas de similaridades entre ontologias podem ser aplicadas em versões futuras deste modelo, caso seja o interesse de uso com múltiplas bases.

Em conjunto com estes aspectos de diferentes categorias existe também as especificações de níveis de legislação que devem ser consideradas para futuras melhorias no projeto. Existem legislações municipais, estaduais e federais (única empregada no modelo), um complicador está na sobreposição de regulações e qual destas devem serem consideradas para aplicação.

No caso dos agentes legisladores, ao criarem novas leis, os seus papéis na hierarquia da estrutura política poderiam serem utilizadas para determinação de domínio. Se em uma dada simulação social, agentes legisladores vereadores desenvolverem novas leis, o modelo deverá agir de acordo e realizar esta compreensão que a nova lei tem seu campo de atuação naquele determinado alcance e não deve ser considerado como conhecimento federalizado. Portanto, para esta questão, os papéis dos agentes e novas estruturas de esferas legais devem ser viabilizadas nas próximas versões do presente modelo.

Um outro ponto relacionado com os serviços pela Internet é o fornecimento integral deste modelo através deste meio. Atualmente confere a necessidade de inserir um componente no sistema multiagente, como benefício de acesso *offline* ao conhecimento legislativo. Somente a ontologia anexa ao *middleware*, poderá estar em local diferente. Entretanto, em um futuro breve este mesmo modelo poderá ser fornecido inteiramente via serviço *web*, facilitando aspectos de configuração.

A parte de detecção de ações dos agentes fomenta a inclusão deste modelo nos estudos sobre privacidade de agentes. Abrindo opções de pesquisa sobre como as instituições de agentes poderiam aplicar ou gerenciar tais componentes, principalmente em sistemas não de simulação, mas de uso por órgãos governamentais, das quais têm demandando entidades computacionais na avaliação de ações e documentos em meio digital.

Este trabalho preocupou-se em focar no acesso dos agentes à ontologia, de múltiplas plataformas. Entretanto, como opera nas áreas de ontologias e SMA, os estudos poderão ramificar e estabelecer novos componentes para as duas áreas. Na ontologia, tanto novos modelos jurídicos quanto novas estruturas para raciocínio automático, poderão serem desenvolvidas. Em sistemas multiagentes, novos componentes ou modelagens poderão exercer novas opções ou configurações em simulações sociais. Além disso, novas pesquisas sobre mecanismos de conhecimento podem ir para áreas diferentes, estimulando inovações, que demandem regulações em informações e ações.

REFERÊNCIAS

ADAM, C.; GAUDOU, B. BDI agents in social simulations: a survey. **The Knowledge Engineering Review**, Shaftesbury Road, Cambridge, CB2 8BS, UK, v.31, n.3, p.207–238, 2016.

ALEXANDER, B. LKIF core: Principled ontology development for the legal domain. **Law, ontologies and the semantic web: channelling the legal information flood**, Nieuwe Hemweg 6B, 1013 BG Amsterdam, Netherlands, v.188, p.21, 2009.

ALEXY, R. **A theory of constitutional rights**. Great Clarendon Street, OX2 6DP, Oxford, New York, USA: Oxford University Press, 2010.

AMEEN, A.; KHAN, K. U. R.; RANI, B. P. Reasoning in semantic web using Jena. **Computer Engineering and Intelligent Systems**, USA, v.5, n.4, p.39–47, 2014.

ANTONIOU, G.; VAN HARMELEN, F. Web ontology language: Owl. In: **Handbook on ontologies**. Berlin, Heidelberg: Springer, 2004. p.67–92.

ATHAN, T. et al. LegalRuleML: Design Principles and Foundations. In: FABER, W.; PASCHKE, A. (Ed.). **Reasoning Web. Web Logic Rules**: 11th International Summer School 2015, Berlin, Germany, July 31- August 4, 2015, Tutorial Lectures. Cham: Springer International Publishing, 2015. p.151–188.

BAZZAN, A. L.; KLÜGL, F. **Multi-agent systems for traffic and transportation engineering**. Chocolate Avenue 701, Hershey, New York, USA: IGI Global, 2009.

BENEVIDES FILHO, M. O que é Sanção? **Revista da Faculdade de Direito**, Rua Meton de Alencar, s/n, Fortaleza, CE, Brasil, v.34, n.1, p.355–373, 2013.

BERMEJO-ALONSO, J.; SANZ, R.; LOPEZ, I. A survey on ontologies for agents. **From Theory to Practice**, ASLab-ICEA, v.1, 2006.

BLAIR, G. S. et al. The Role of Ontologies in Emergent Middleware: Supporting Interoperability in Complex Distributed Systems. In: MIDDLEWARE 2011, 2011, Berlin, Heidelberg. **Proceedings...** Springer Berlin Heidelberg, 2011. p.410–430.

BOELLA, G.; TORRE, L. van der; VERHAGEN, H. Introduction to normative multiagent systems. **Computational & Mathematical Organization Theory**, Switzerland, v.12, n.2-3, p.71–79, Oct. 2006.

BOISSIER, O. et al. Multi-agent oriented programming with JaCaMo. **Science of Computer Programming**, Radarweg 29, 1043 NX Amsterdam, The Netherlands, v.78, n.6, p.747–761, 2013.

BORGO, S.; MASOLO, C. Foundational Choices in DOLCE. In: STAAB, S.; STUDER, R. (Ed.). **Handbook on Ontologies**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. p.361–381.

BRASIL. Lei nº 5.197 de 3 de Janeiro de 1967. Dispõe sobre a proteção à fauna e dá outras providências. **Diário Oficial [da] República Federativa do Brasil**, Brasília, DF, 1967.

BRASIL. Lei nº 7.653 de 12 de Fevereiro de 1988. Altera a redação dos arts. 18, 27, 33 e 34 da Lei nº 5.197, de 3 de janeiro de 1967, que dispõe sobre a proteção à fauna, e dá outras providências. **Diário Oficial [da] República Federativa do Brasil**, Brasília, DF, 1988.

BRASIL. Lei nº 95 de 26 de Fevereiro de 1998. Dispõe sobre a elaboração, a redação, a alteração e a consolidação das leis, conforme determina o parágrafo único do art. 59 da Constituição Federal, e estabelece normas para a consolidação dos atos normativos que menciona. **Diário Oficial [da] República Federativa do Brasil**, Brasília, DF, 1998.

BRASIL. Lei nº 9.605 de 12 de Fevereiro de 1998. Dispõe sobre as sanções penais e administrativas derivadas de condutas e atividades lesivas ao meio ambiente, e dá outras providências. **Diário Oficial [da] República Federativa do Brasil**, Brasília, DF, 1998.

BREUKER, J. et al. Law and the Semantic Web. In: BENJAMINS, V. R.; CASANOVAS, P.; BREUKER, J.; GANGEMI, A. (Ed.). . Berlin, Heidelberg: Springer-Verlag, 2005. p.36–64.

BREUKER, J.; VALENTE, A.; WINKELS, R. Use and Reuse of Legal Ontologies in Knowledge Engineering and Information Management. In: BENJAMINS, V. R.; CASANOVAS, P.; BREUKER, J.; GANGEMI, A. (Ed.). **Law and the Semantic Web**: Legal Ontologies, Methodologies, Legal Information Retrieval, and Applications. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005. p.36–64.

CARROLL, J. J. et al. Jena: Implementing the Semantic Web Recommendations. In: INTERNATIONAL WORLD WIDE WEB CONFERENCE ON ALTERNATE TRACK PAPERS AND POSTERS, 13., 2004, New York, NY, USA. **Proceedings...** Association for Computing Machinery, 2004. p.74–83. (WWW Alt. 04).

COSTA, A. C. d. R. Situated legal systems and their operational semantics. **Artificial Intelligence and Law**, Berlin, Heidelberg, v.23, n.1, p.43–102, 2015.

COULOURIS, G.; DOLLIMORE, J.; KINDBERG, T.; BLAIR, G. **Distributed Systems**: Concepts and Design (5th Ed.). Wokington, England: Addison-Wesley, 2011. (International Computer Science Series).

ČYRAS, V. et al. Visualization of Hans Kelsen's Pure Theory of Law. In: FUNDAMENTAL CONCEPTS AND THE SYSTEMATIZATION OF LAW, FCASL 2011, WORKSHOP AT JURIX, 2011, Vienna, Austria. **Proceedings...** Springer, 2011. p.112–122.

DAVIDSSON, P. Agent based social simulation: A computer science view. **Journal of artificial societies and social simulation**, UK, v.5, n.1, 2002.

DIGNUM, V.; VÁZQUEZ-SALCEDA, J.; DIGNUM, F. OMNI: Introducing Social Structure, Norms and Ontologies into Agent Organizations. In: **Lecture Notes in Computer Science**. Berlin: Springer Berlin Heidelberg, 2005. v.3346, p.181–198.

EL GHOSH, M.; NAJA, H.; ABDULRAB, H.; KHALIL, M. Towards a Middle-out Approach for Building Legal Domain Reference Ontology. **International Journal of Knowledge Engineering**, Singapore, v.2, n.3, p.109–114, 2016.

FELICíSSIMO, C.; LUCENA, C.; CARVALHO, G.; PAES, R. **Normative Ontologies to Define Regulations Over Roles in Open Multi-Agent Systems**. Palo Alto, California: AAAI Fall Symposium "Roles, an Interdisciplinary Perspective: Ontologies, Programming Languages, and Multiagent Systems, 2005.

FERBER, J.; WEISS, G. **Multi-agent systems**: an introduction to distributed artificial intelligence. Boston, Massachusetts, EUA: Addison-Wesley Reading, 1999. v.1.

FITOUSSI, D.; TENNENHOLTZ, M. Choosing social laws for multi-agent systems: Minimality and simplicity. **Artificial Intelligence**, Amsterdã, Países Baixos, v.119, n.1, p.61 – 101, 2000.

FREITAS, A. et al. Integrating ontologies with multi-agent systems through CArtAgO artifacts. In: IEEE/WIC/ACM INTERNATIONAL CONFERENCE ON WEB INTELLIGENCE AND INTELLIGENT AGENT TECHNOLOGY (WI-IAT), 2015, Singapore. **Proceedings...** IEEE, 2015. v.2, p.143–150.

FREITAS, A. L. S. C. **Model-driven engineering of multi-agent systems based on ontology**. 2017. Tese (Doutorado em Ciência da Computação) — Pontificial Catholic University of Rio Grande do Sul.

FURDÍK, K.; SABOL, T.; DULINOVÁ, V. Policy modelling supported by e-participation ICT tools. In: METTEG'10), 4., 2010. **Proceedings...** University of Applied Sciences: Northwestern Switzerland: Olten, 2010. p.135–146.

GANGEMI, A. et al. Sweetening Ontologies with DOLCE. In: **Knowledge Engineering and Knowledge Management**: Ontologies and the Semantic Web. Berlin: Springer Berlin Heidelberg, 2002. p.166–181.

GARCÍA, E. et al. MISIA: Middleware Infrastructure to Simulate Intelligent Agents. In: **Advances in Intelligent and Soft Computing**. Berlin: Springer Berlin Heidelberg, 2011. p.107–116.

GENESERETH, M. R.; KETCHPEL, S. P. Software Agents. **Commun. ACM**, New York, NY, USA, v.37, n.7, p.48–ff., July 1994.

GILBERT, N. Agent-based social simulation: dealing with complexity. **The Complex Systems Network of Excellence**, UK, v.9, n.25, p.1–14, 2004.

GLIMM, B. et al. HermiT: an OWL 2 reasoner. **Journal of Automated Reasoning**, Switzerland, v.53, n.3, p.245–269, 2014.

GRIFFO, C.; ALMEIDA, J. P. A.; GUIZZARDI, G. Towards a legal core ontology based on Alexy's theory of fundamental rights. In: MULTILINGUAL WORKSHOP ON ARTIFICIAL INTELLIGENCE AND LAW, ICAIL 2015, 2015. **Proceedings...** ACM Digital Library, 2015.

GUIZZARDI, G.; FALBO, R. A.; GUIZZARDI, R. S. S. Grounding Software Domain Ontologies in the Unified Foundational Ontology (UFO): The case of the ODE Software Process Ontology. In: XI CONFERENCIA IBEROAMERICANA DE SOFTWARE ENGINEERING (CIBSE 2008), RECIFE, PERNAMBUCO, BRASIL, FEBRUARY 13-17, 2008, 2008, Recife, PE, Brasil. **Memorias...** Editora da UFPE, 2008. p.127–140.

HECKBERT, S.; BAYNES, T.; REESON, A. Agent-based modeling in ecological economics. **Annals of the New York Academy of Sciences**, Hoboken, Nova Jersey, EUA, v.1185, n.1, p.39–53, 2010.

HORRIDGE, M.; BECHHOFER, S. The OWL API: A Java API for OWL Ontologies. **Semant. Web**, NLD, v.2, n.1, p.11–21, Jan. 2011.

HUHNS, M. N.; STEPHENS, L. M. Multiagent Systems and Societies of Agents. In: WEISS, G. (Ed.). . Cambridge, MA, USA: MIT Press, 1999. p.79–120.

KELSEN, H. **General Theory of Norms**. Oxônia, Reino Unido: Oxford University Press, 1990.

KELSEN, H. **Pure Theory of Law**. United States: Lawbook Exchange, Ltd., 2009.

KRAVARI, K.; BASSILIADES, N. A survey of agent platforms. **Journal of Artificial Societies and Social Simulation**, UK, v.18, n.1, p.11, 2015.

LEITAO, P.; MARIK, V.; VRBA, P. Past, Present, and Future of Industrial Agent Applications. **IEEE Transactions on Industrial Informatics**, Piscataway, New Jersey, US, v.9, n.4, p.2360–2372, 2013.

LI, X. et al. SWARMs ontology: A common information model for the cooperation of underwater robots. **Sensors**, Switzerland, v.17, n.3, p.569, 2017.

LOGENTHIRAN, T.; SRINIVASAN, D.; KHAMBADKONE, A. M. Multi-agent system for energy resource scheduling of integrated microgrids in a distributed system. **Electric Power Systems Research**, Amsterdã, Países Baixos, v.81, n.1, p.138–148, 2011.

LÜTZENBERGER, M.; KÜSTER, T.; MASUCH, N.; FÄHNDRICH, J. Multi-Agent System in Practice: When Research Meets Reality. In: INTERNATIONAL CONFERENCE ON AUTONOMOUS AGENTS & MULTIAGENT SYSTEMS, 2016., 2016, Richland, SC. **Proceedings...** International Foundation for Autonomous Agents and Multiagent Systems, 2016. p.796–805. (AAMAS '16).

MOREIRA, M. I. G.; COSTA, A. C. d. R.; AGUIAR, M. S. d. A legislation-oriented VLE-MAS system applied to MOODLE. In: INTERNATIONAL CONFERENCE ON INFORMATION TECHNOLOGY BASED HIGHER EDUCATION AND TRAINING (ITHET), 16., 2017. **Proceedings...** IEEE, 2017. p.1–8.

MOSES, Y.; TENNENHOLTZ, M. Artificial social systems. **Computers and Informatics**, Bratislava, Slovakia, v.14, n.6, p.533–562, 1995.

MOTIK, B. et al. OWL 2 web ontology language profiles. **W3C recommendation**, MIT, v.11, p.43, 2012.

MUSEN, M. A. The Protégé Project: A Look Back and a Look Forward. **AI Matters**, New York, NY, USA, v.1, n.4, p.4–12, June 2015.

OLARU, A.; FLOREA, A. M.; SEGHROUCHNI, A. E. F. A context-aware multi-agent system as a middleware for ambient intelligence. **Mobile Networks and Applications**, Switzerland, v.18, n.3, p.429–443, 2013.

PAES, R. et al. Specifying laws in open multi-agent systems. In: **Agents, Norms and Institutions for Regulated Multiagent Systems - ANIREM**. Switzerland: Springer, 2005.

PALMIRANI, M. et al. Legal Ontology for Modelling GDPR Concepts and Norms. In: ANNUAL CONFERENCE JURIX 2018, 31., 2018, Nieuwe Hemweg 6B, Amsterdam, Netherlands. **Proceedings...** IOS Press BV, 2018. p.91–100.

PENNA, S.; MACIEL, E. C. d. A. Técnica Legislativa: Orientação para Padronização de Trabalhos. **Brasília: Senado Federal, Secretaria de Especial de Editoração e Publicações**, Consultoria Legislativa, Anexo II, Bloco B, Brasília, Brasil, 2002.

RANI, M.; NAYAK, R.; VYAS, O. An ontology-based adaptive personalized e-learning system, assisted by software agents on cloud storage. **Knowledge-Based Systems**, Amsterdã, Países Baixos, v.90, p.33–48, 2015.

SANTOS, I. A. d. S. d.; COSTA, A. C. d. R. Toward a Framework for Simulating Agent-Based Models of Public Policy Processes on the Jason-CArtAgO Platform. In: SECOND INTERNATIONAL WORKSHOP ON AGENT-BASED MODELING FOR POLICY ENGINEERING (AMPLE 2012), 2012, Montpellier, France. **Proceedings...** TU Delft, 2012. p.45–59.

SAWYER, R. K. Artificial societies: Multiagent systems and the micro-macro link in sociological theory. **Sociological methods & research**, Thousand Oaks, CA, USA, v.31, n.3, p.325–363, 2003.

SCHULDT, A.; GEHRKE, J. D.; WERNER, S. Designing a simulation middleware for FIPA multiagent systems. In: IEEE/WIC/ACM INTERNATIONAL CONFERENCE ON WEB INTELLIGENCE AND INTELLIGENT AGENT TECHNOLOGY, 2008., 2008. **Proceedings...** IEEE Computer Society, 2008. v.2, p.109–113.

SERRANO, D.; STROULIA, E.; LAU, D.; NG, T. Linked rest APIs: a middleware for semantic rest API integration. In: IEEE INTERNATIONAL CONFERENCE ON WEB SERVICES (ICWS), 2017. **Proceedings...** IEEE, 2017. p.138–145.

SINGH, M. P. Towards a Formal Theory of Communication for Multi-agent Systems. In: TWELFTH INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE (IJCAI-91), 1991. **Proceedings...** Morgan Kaufmann, 1991. v.91, p.69–74.

SIRIN, E.; PARSIA, B. Pellet: An owl dl reasoner. In: DESCRIPTION LOGIC WORKSHOP (DL 2004), 2004, Whistler, British Columbia, Canada. **Proceedings...** CEUR Workshop Proceedings, 2004. v.104, p.212–213.

SMITH, B. Beyond concepts: ontology as reality representation. In: OF THE 3RD INTERNATIONAL CONFERENCE ON FORMAL ONTOLOGY IN INFORMATION SYSTEMS, 2004. **Proceedings...** IOS Press Amsterdam, 2004. p.73–84.

SMITH, B.; WELT, C. FOIS Introduction: Ontology—towards a New Synthesis. In: INTERNATIONAL CONFERENCE ON FORMAL ONTOLOGY IN INFORMATION SYSTEMS - VOLUME 2001, 2001, New York, NY, USA. **Proceedings...** ACM, 2001. p..3–.9. (FOIS '01).

SPEROTTO, F. A.; ADAMATTI, D. F. A proposal for interoperability to agent communication using synonyms. In: THIRD BRAZILIAN WORKSHOP ON SOCIAL SIMULATION, 2012. **Proceedings...** IEEE, 2012. p.39–43.

TSARKOV, D.; HORROCKS, I. FaCT++ Description Logic Reasoner: System Description. In: FURBACH, U.; SHANKAR, N. (Ed.). **Automated Reasoning, IJCAR 2006**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006. p.292–297.

VALENTE, A.; BREUKER, J. A Functional Ontology of Law. **Towards a global expert system in law**, Padua, Italy, p.112–136, 1994.

VALENTE, A.; BREUKER, J.; BROUWER, B. Legal modeling and automated reasoning with ON-LINE. **International Journal of Human-Computer Studies**, Amsterdã, Países Baixos, v.51, n.6, p.1079–1125, 1999.

WAMPLER, D. Aspect-oriented design principles: Lessons from object-oriented design. In: SIXTH INTERNATIONAL CONFERENCE ON ASPECT-ORIENTED SOFTWARE DEVELOPMENT, 2007, New York, NY, USA. **Proceedings...** Association for Computing Machinery, 2007.

WARDEN, T. et al. Towards Ontology-Based Multiagent Simulations: The Plasma Approach. In: EUROPEAN CONFERENCE ON MODELLING AND SIMULATION, (ECMS), 2010, Kuala Lumpur, Malaysia. **Proceedings...** European Council for Modeling and Simulation, 2010. p.50–56.

WEYNS, D.; HELLEBOOGH, A.; HOLVOET, T.; SCHUMACHER, M. The agent environment in multi-agent systems: A middleware perspective. **Multiagent and Grid Systems**, Nieuwe Hemweg 6B, 1013 BG, Amsterdam, The Netherlands, v.5, n.1, p.93–108, 2009.

WOOLDRIDGE, M. Intelligent agents. In: WEISS, G. (Ed.). **Multiagent Systems**. 2.ed. 55 Hayward Street, Cambridge, London: MIT Press London, 1999. v.6, p.3–45.