Pós-Graduação em Ciência da Computação

Adriano Ferraz da Costa

Arandu, um *Chatbot* para construção de ontologias guiado por uma ontologia de topo.

Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
http://cin.ufpe.br/~posgraduacao

Recife
2020

Adriano Ferraz da Costa

Arandu, um *Chatbot* para construção de ontologias guiado por uma ontologia de topo.

Tese de Doutorado apresentada ao Programa de Pós-graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco, como requisito parcial para obtenção do título de Doutor em Ciência da Computação.

Área de Concentração: Inteligência Computacional
Orientador: Frederico Luiz Gonçalves de Freitas

Recife
2020

Catalogação na fonte
Bibliotecária Monick Raquel Silvestre da S. Portes, CRB4-1217

C837a Costa, Adriano Ferraz da
Arandu, um *Chatbot* para construção de ontologias guiado por uma ontologia de topo / Adriano Ferraz da Costa. – 2020.
104 f.: il., fig., tab.

Orientador: Frederico Luiz Gonçalves de Freitas.
Tese (Doutorado) – Universidade Federal de Pernambuco. CIn, Ciência da Computação, Recife, 2020.
Inclui referências.

1. Inteligência computacional. 2. Ontologia. I. Freitas, Frederico Luiz Gonçalves de (orientador). II. Título.

006.31 CDD (23. ed.) UFPE - CCEN 2020 -88

Adriano Ferraz da Costa

"**Arandu, um *Chatbot* para construção de ontologias guiado por uma ontologia de topo.**"

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Doutor em Ciência da Computação.

Aprovado em: 02/03/2020.

Orientador: Frederico Luiz Gonçalves de Freitas

BANCA EXAMINADORA

Prof. Dr. André Menezes Marques das Neves
Departamento de Design / UFPE

Prof. Dr. Cedric Luiz de Carvalho
Instituto de Informática - INF / UFG

Prof. Dr. Evandro de Barros Costa
Instituto de Computação / UFAL

Prof. Dra. Flávia de Almeida Barros
Cin / UFPE

Prof. Dr. Yuri de Almeida Malheiros Barbosa
Departamento de Computação Científica / UFPB

Dedico esse trabalho ao meu filho Miguel Henrique e ao meu sobrinho Fernando Alonso, que ele sirva de inspiração para ambos.

AGRADECIMENTOS

Agradeço a Deus primeiramente.

Agradeço a toda minha família na pessoa do meu irmão Alonso.

Agradeço a todos os Professores que fizeram parte da minha formação, citando a Profa. Marise e a Profa. Sulamita que foram fundamentais na minha alfabetização.

Agradeço a todos os meus amigos, que não são poucos, amizades formadas na infância e amizades formadas durante a caminhada da vida.

Agradeço a Universidade Estadual de Goiás - UEG unidade de Santa Helena, por permitir a minha qualificação e na oportunidade agradeço a todos os colegas de trabalho, farei de tudo para repassar o conhecimento adquirido.

Agradeço ao meu orientador Fred Freitas que inspira pelo conhecimento e pela humanidade, espero ser reflexo do aprendizado obtido com você. Estendo esse agradecimento aos meus orientadores acadêmicos que foram alicerce para esta caminhada.

Agradeço a minha paixão Wanessa, seu companheirismo foi fundamental nesse desafio. Estendo esse agradecimento à sua família.

RESUMO

Com a grande quantidade de informações atualmente disponíveis na Web, representar conhecimento de maneira semântica e formal torna-se uma tarefa essencial para a comunicação entre sistemas. Ontologias fornecem uma maneira reutilizável de construírem bases de conhecimento. Ontologias permitem a construção de axiomas lógicos que são capazes de raciocinar e inferir novos fatos. No entanto, a construção de ontologias ainda é uma tarefa desafiadora, devido à necessidade de especialistas de domínio aprenderem uma linguagem formal, bem como reutilizar definições de ontologias de uso geral, conhecidas como ontologias de topo. O reuso de ontologias, principalmente as de topo, garante o uso de termos que são consenso entre a comunidade do domínio representado, além de dar maior qualidade e robustez a ontologia que está sendo criada. Porém, somente adicionar termos de outra ontologia não necessariamente garantirá coerência no modelo que foi expandido, fazer isso sem a validação de um especialista pode acarretar a inclusão de termos que não fazem sentido para o domínio representado. Este trabalho propõe uma metodologia para construção de ontologias por meio de um *Chatbot*, chamado de Arandu. Diferente de outras metodologias, Arandu guia o usuário no processo de construção de uma ontologia, sugere classes advindas de uma ontologia de topo, sugere quantificadores lógicos, raciocina novos fatos e informa inconsistências lógicas. Através de técnicas de Processamento de Linguagem Natural (PLN), Arandu converte sentenças informadas pelo usuário em linguagem natural para axiomas lógicos formalizados em linguagem OWL. Arandu permite que o especialista de domínio valide quais classes devem ser reutilizadas, eliminando ambiguidades ou incoerência de classes sugeridas. Reutilizar conhecimento de uma ontologia de topo não é uma tarefa trivial e não compõe técnicas de construção automática de ontologias a partir de textos, essa característica traz originalidade e se destaca como principal avanço científico desta tese. Os resultados obtidos com testes realizados com usuários, apontam que Arandu contribui para que especialistas de domínio, com pouco conhecimento técnico sobre ontologias, construa uma ontologia utilizando linguagem natural e reutilize conhecimento de uma ontologia de topo. Por meio do paradigma de pesquisa *Design Science Research* (DSR), validamos o artefato tecnológico desenvolvido e concluímos que Arandu converte sentenças informadas pelo usuário em linguagem natural para axiomas lógicos e sugere definições provenientes de uma ontologia de nível superior, o que permite a criação de ontologias em linguagem OWL por meio de diálogos, expandindo o modelo esperado.

Palavras-chaves: Ontologia. Representação do conhecimento. Web Semântica. Chatbot.

ABSTRACT

With a large amount of information currently available on the Web, representing knowledge semantically and formally becomes an essential task for communication between systems. Ontologies provide a reusable way to build knowledge bases. Ontologies allow the construction of logical axioms that are capable of reasoning and inferring new facts. However, the creation of ontologies is still a challenging task, due to the need for domain experts to learn a formal language, as well as to reuse definitions of general-purpose ontologies, known as top ontologies. The reuse of ontologies, especially the top ones, guarantees the use of terms that are consensus among the community of the represented domain, in addition to giving higher quality and robustness to the ontology that creates. However, just adding terms from another ontology will not necessarily guarantee coherence in the model that has expanded, doing so without the validation of a specialist can lead to the inclusion of terms that do not make sense for the domain represented. This work proposes a methodology for building ontologies using a Chatbot, called Arandu. Unlike other methodologies, Arandu guides the user in the process of building an ontology, suggests classes from a top ontology, suggests logical quantifiers, reasons new facts, and reports logical inconsistencies. Through Natural Language Processing (PLN) techniques, Arandu converts user-informed sentences into natural language for formalized logical axioms in OWL language. Arandu allows the domain specialist to validate which classes should reuse, eliminating ambiguities or inconsistencies of suggested classes. Reusing knowledge of a top ontology is not a trivial task and does not compose techniques of automatic construction of ontologies from texts; this characteristic brings originality. It stands out as the main scientific advance of this thesis. The results obtained with tests carried out with users, point out that Arandu contributes so that domain experts, with little technical knowledge about ontologies, build an ontology using natural language and reuse knowledge from a top ontology. Using the Design Science Research (DSR) research paradigm, we validate the technological artifact developed and conclude that Arandu converts user-informed sentences into natural language into logical axioms and suggests definitions from a higher-level ontology, which allows the creation of ontologies in OWL language by through dialogues, expanding the expected model.

Keywords: Ontology. Knowledge Representation. Semantic Web. Chatbot.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Ontologias têm sido amplamente utilizadas e se mostraram importantes nos mais diversos domínios (LIMA; ESPINASSE; FREITAS, 2018) (MARAN et al., 2017). Ontologias podem ser definidas como vocabulários formais e compartilhados, que permitem interoperabilidade entre softwares, reuso e raciocínio sobre o conhecimento representado (GRUBER, 1995). No entanto, criar ontologias ainda não é uma tarefa trivial, exigindo conhecimento de uma linguagem formal de construção de ontologias.

Ontologias podem ser classificadas e divididas de acordo com a sua proposta (GUARINO; WELTY, 2002). Por exemplo, uma ontologia pode ser classificada como de topo, ser ampla e tentar representar todas as entidades e eventos do mundo. Pode ser classificada como de domínio e representar um domínio específico como biologia, finanças, etc., ou ser de aplicação, sendo bem específica e representar conhecimento sobre os animais de uma região, por exemplo. Ontologias são construídas por especialistas de domínio, que utilizam o conhecimento adquirido na área, livros, textos, artigos, entrevistas e/ou ontologias que já foram criadas para reutilizar conhecimento (HEIJST; SCHREIBER; WIELINGA, 1997).

Uma ontologia é formada por triplas (sujeito, objeto e predicado) porém, construir boas ontologias vai além da construção de triplas (GUARINO, 1998). Boas ontologias são construídas com o uso de axiomas lógicos que proveem quantificadores existenciais ou universais, disjunções, conjunções, complemento, entre outros. O uso destes operadores amplia a expressividade da ontologia e permite checagem de consistência e descoberta de novos fatos (FREITAS; OTTEN, 2016).

Embora modelar conhecimento e torná-lo explícito através de ontologias seja vantajoso, criar ontologias ainda é uma tarefa onerosa, que depende não só de conhecimento do domínio a ser representado, mas de ferramentas e técnicas de modelagem que permitam uma representação coerente e organizada. O reuso de outras ontologias também não é uma tarefa trivial, pois exige conhecimento da ontologia a ser reutilizada (BLOMQVIST; HAMMAR; PRESUTTI, 2016).

Reutilizar uma ontologia, principalmente uma ontologia de topo, amplia a capacidade de correspondência com outras ontologias (PAVEL; EUZENAT, 2013). O reuso de uma ontologia é um passo comum em várias metodologias para construção de ontologias (GRUBER, 1995) (CIMIANO; VÖLKER; BUITELAAR, 2010) (PINTO; STAAB; TEMPICH, 2004).

Ontologia é um dos principais componentes da pilha de tecnologias que compõem a Web Semântica (BERNERS-LEE; HENDLER; LASSILA, 2001). Por meio de triplas que representam classes, instâncias e relações é possível construir um grafo de conhecimento sobre um determinado domínio.

Devido à grande quantidade de dados disponíveis na web e à necessidade de interoperabilidade entre softwares, é necessário estruturar conhecimento para que ele possa ser

disponibilizado em uma linguagem formal. Por isso, criar ferramentas ou metodologias que permitam que especialistas de um domínio representem conhecimento é fundamental (RODRIGUES et al., 2019).

Tanto as metodologias mais conhecidas para desenvolvimento de ontologias (NOY; MCGUINNESS, 2001) (FERNANDEZ-LOPEZ; GOMEZ-PEREZ; JURISTO, 1997) (SUáREZ-FIGUEROA; GOMEZ-PEREZ; FERNáNDEZ-LóPEZ, 2012), como o Protégé, software mais utilizado para modelagem de ontologias (MUSEN, 2015), recomendam e permitem que o especialista de domínio reutilize outra ontologia.

Iniciar a construção de uma ontologia com uma técnica de construção automática a partir de textos, conhecida como aprendizagem de ontologias, é uma ótima opção. Porém, necessita de validação de um especialista de domínio para garantir sua coerência e não dispõe de mecanismos automáticos de reuso (CIMIANO, 2006).

Trabalhos como os de (AZEVEDO et al., 2014a) (LIMA, 2017) (SILVA, 2019) utilizaram com sucesso técnicas de aprendizagem de ontologias para converter um diálogo em linguagem natural para uma ontologia, uma ótima estratégia, visto que usuários demonstram interesse por interfaces de conversação (FØLSTAD; BRANDTZÆG, 2017). Contudo, esses trabalhos não propõem o reuso de uma ontologia.

Segundo (FERNáNDEZ-LóPEZ et al., 2019), a prática de reuso de ontologias em um mesmo domínio não é consolidada, contradizendo, inclusive, uma das definições mais difundidas sobre ontologias, que a define como um vocabulário compartilhado. Ainda segundo (VERDONCK et al., 2019), reusar ontologias de topo dá maior qualidade às ontologias.

Propomos nesta tese uma metodologia apoiada por um Chatbot para que um especialista de domínio construa uma ontologia a partir de um diálogo em linguagem natural. Interagindo com um Chatbot em um processo gamificado, conseguimos converter sentenças em linguagem natural para axiomas lógicos em linguagem OWL e reutilizar conhecimento de outra ontologia.

Arandu[1] encoraja o usuário a informar definições que serão transformadas em axiomas lógicos na ontologia em desenvolvimento. Arandu sugere a inserção de quantificadores existenciais ou universais para as relações informadas.

Utilizando uma ontologia de topo, Arandu sugere classes que possam ser reutilizadas, isto amplia a cobertura da ontologia em desenvolvimento. Arandu também faz checagem de consistência e informa ao usuário, caso haja alguma definição contraditória.

Utilizando a abordagem epistemológica *Design Science Research* - DSR (SIMON, 1996), validamos o artefato de software desenvolvido. Arandu, de forma gamificada guia o usuário no processo de construção de uma ontologia utilizando linguagem natural e fazendo reuso de uma ontologia superior.

[1] Arandu, o nome do Chatbot desenvolvido, é uma palavra do dicionário *Guarani* que significa aquele que sabe, que tem conhecimento.

1.1 QUESTÃO DE PESQUISA

Como construir uma ontologia de domínio baseada em definições de uma ontologia de topo, a partir de um sistema de diálogo?

1.2 HIPÓTESES

Para responder a questão de pesquisa proposta nesta tese, formulamos duas hipóteses:

- ***H1 - É possível construir uma ontologia e reutilizar conhecimento de uma ontologia de topo por meio de um Chatbot.***
- ***H2 - Utilizar um Chatbot para construir uma ontologia torna o processo de construção e reuso mais fácil.***

1.3 OBJETIVOS

Essa tese tem como objetivo geral:

- Criar um *Chatbot* que guie o usuário no processo de construção de uma ontologia e ajude a reutilizar conhecimento de uma ontologia de topo.

Para alcançar este objetivo geral, propomos os seguinte objetivos específicos:

- Implementar um *Chatbot* com tecnologias que permitam converter uma sentença em linguagem natural para axiomas lógicos;
- Criar um serviço de comunicação com uma ontologia de topo;
- Propor uma metodologia que guie o usuário no processo de construção de uma ontologia;
- Analisar o uso do artefato tecnológico desenvolvido por usuários.

1.4 ORGANIZAÇÃO DO TRABALHO

Esse trabalho esta organizado da forma como se segue:

No capítulo 2 definimos o que é uma ontologia, um componente da *Web* Semântica, e apresentamos suas bases lógico formais.

No capítulo 3 discutimos sobre a engenharia de ontologias, onde apresentamos metodologias, tipos de ontologias e técnicas relacionadas a esta tarefa.

No capítulo 4 tratamos da área de processamento de linguagem natural, componente indispensável para a construção de *chatbots*, e que provê as tarefas necessárias para a conversão de linguagem natural para axiomas lógicos. Apresentamos também neste capítulo os trabalhos relacionados e o paradigma de pesquisa utilizado.

No capítulo 5 apresentamos o Arandu *Chatbot* e toda a tecnologia utilizada para a sua construção.

No capítulo 6 demonstramos o uso do Arandu e os resultados obtidos com um teste realizado com usuários.

No capítulo 7 temos as conclusões deste trabalho, propomos alguns trabalhos futuros e apresentamos as publicações obtidas com esta pesquisa.

2 ONTOLOGIAS

Ontologia é uma palavra derivada dos vocabulários gregos *ontos* (ser) e *logos* (palavra). Os primeiros estudos sobre ontologias foram iniciados por Aristóteles, em um contexto filosófico onde ele atribuía esse termo a um ramo da Metafísica para classificar as coisas, ou seja, como uma ciência para descrever "o ser", isto é, o estudo dos atributos que pertencem às coisas por causa de sua própria natureza (GUARINO; OBERLE; STAAB, 2009).

Em Ciência da Computação, uma ontologia define um conjunto de primitivas representacionais específicas para modelar um domínio do conhecimento. Essas primitivas representacionais são classes, instâncias e relações que formam um grafo, onde os nós representam classes ou instâncias e os vértices representam as relações. Ontologias têm se tornado objeto de estudo em diversos domínios, com o objetivo de representar conhecimento (ISOTANI; BITTENCOURT, 2015).

A criação de ontologias permite que sejam construídas bases de dados e de conhecimento compartilháveis e reutilizáveis (GRUBER, 1995). Um pequeno exemplo de um grafo que representa uma ontologia pode ser visto na Figura 1, onde modelamos o cenário de uma classe chamada disciplina com relações e instâncias de classe.

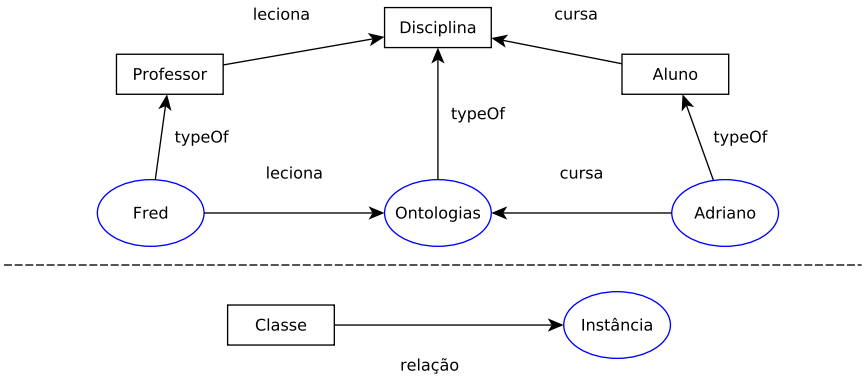

Figura 1 – Exemplo de uma ontologia que representa o cenário de uma classe chamada "disciplina"com possíveis relações e instâncias de classe.

No contexto dos sistemas de banco de dados, uma ontologia pode ser vista como um nível de abstração de modelos de dados, análogo aos modelos hierárquicos e relacionais, mas destinado a modelar o conhecimento sobre os indivíduos, seus atributos e suas relações com outros indivíduos.

As ontologias são tipicamente especificadas em linguagens que permitem a abstração de estruturas de dados e estratégias de implementação. Na prática, as linguagens das ontologias estão mais próximas em poder expressivo à lógica de primeira ordem do que as linguagens usadas para modelar bancos de dados. Por esta razão, as ontologias são

consideradas no nível "semântico", enquanto que os esquemas de banco de dados são modelos de dados no nível "lógico"ou "físico".

Ontologias também são usadas para integrar bancos de dados heterogêneos, possibilitando a interoperabilidade entre sistemas diferentes e especificando interfaces para serviços independentes baseados no conhecimento (GRUBER, 1995).

Especificamente no ramo da Inteligência Artificial, uma ontologia O pode ser definida como um relacionamento de quatro elementos, representado por $O = (C, R, I, A)$, onde (KIRYAKOV; OGNYANOV; MANOV, 2005):

- C - é o conjunto de classes que representam os conceitos em um dado domínio de interesse;
- R - é o conjunto de relações ou associações entre os conceitos do domínio;
- I - é o conjunto de instâncias derivadas das classes, ou ainda, os exemplos de conceitos representados em uma ontologia;
- A - é o conjunto de axiomas do domínio, que servem para modelar restrições e regras inerentes às instâncias e conceitos.

Segundo (GUARINO, 1998) uma ontologia pode ser classificada em três níveis:

- *Ontologia de Topo* - As ontologias de Topo ou alto nível descrevem conceitos gerais como espaço, tempo, matéria, objeto, evento, ação, etc. Elas são independentes de um determinado problema ou domínio. Esta classificação será discutida com mais detalhes na Seção 3.5.
- *Ontologia de domínio* - São ontologias que expressam um vocabulário relacionado a um domínio genérico (como medicina ou automóveis) ou uma tarefa ou atividade genérica (como diagnosticar ou vender), especializando os termos de uma ontologia de alto nível. Segundo (PEASE, 2011) as ontologias de domínio cobrem diversas áreas específicas, incluindo governo mundial, finanças e economia, vírus biológicos, entre outros.
- *Ontologias de Aplicação* - São ontologias apropriadas para uma aplicação em questão, que fazem sentido no contexto da aplicação. São pouco reusáveis, por terem um objetivo bem específico.

As ontologias normalmente são escritas em uma linguagem ontológica formal, embora algumas ontologias não usem formatos padronizados para esse fim. O W3C[1] reconhece

[1] W3C ou *World Wide Web Consortium*, o W3C é a principal organização de padronização da *World Wide Web*, atualmente é dirigida por Tim Berners-Lee.

como um dos formatos padrão para o desenvolvimento de ontologias a linguagem OWL (*Web Ontology Language*), que será tratada na Seção 2.3.

Ontologias são consideradas o pilar da Web Semântica, que será descrita na próxima Seção. 2.1.

2.1 WEB SEMÂNTICA

O termo Web Semântica foi proposto por Tim Berners-Lee, James Hendler e Ora Lassila em 2001 no artigo "*The Semantic Web*"(BERNERS-LEE; HENDLER; LASSILA, 2001). Web Semântica ou Web 3.0 é o nome dado a uma série de tecnologias que têm como objetivo estruturar as informações da Web, para que estas se tornem acessíveis, padronizadas e interoperáveis por softwares e agentes de software (BERNERS-LEE; HENDLER; LASSILA, 2001).

As tecnologias que compõem a Web Semântica são validadas pelo *World Wide Web Consortium* (W3C), que atualmente é dirigido por Berners-Lee. O W3C é quem define quais tecnologias estão maduras o suficiente para compor a Web Semântica.

As camadas que compõem a Web Semântica podem ser vistas na Figura 2. Algumas dessas camadas já foram implementadas e possuem suas tecnologias oficiais, algumas ainda estão em fase de implementação e dependem da implementação e maturidade de outras camadas inferiores. Essas camadas são descritas nas seções seguintes.

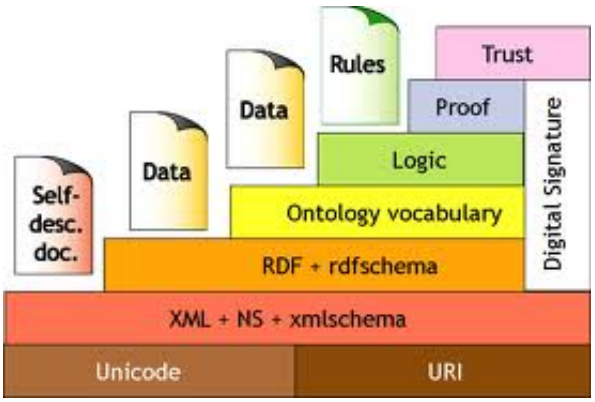

Figura 2 – Camadas que compõem a Web Semântica, fonte: *www.w3.org*

2.1.1 UNICODE+URI

A primeira camada refere-se à codificação **UNICODE+URI**. A camada UNICODE tem por objetivo prover uma codificação em nível de caracteres universal válida e sem ambiguidade, acessível por qualquer plataforma.

Unicode é um padrão para a indústria da computação, é um sistema de codificação projetado para suportar diversos idiomas mantido pelo *Unicode Technical Committee*[2].

[2] *http://www.unicode.org*

URI - *Uniform Resource Identifier* é a referência universal e única para representar um recurso. Uma URL - *Uniform Resource Locator* de um site, também é uma URI. Um exemplo de URI é *"http://www.cin.ufpe.br/sword/#Arandu"*, onde *"http"* identica o protocolo de comunicação utilizado, *"www.cin.ufpe"* identifica a autoridade da URI, *"sword"* é o caminho que aponta para um recurso, e *"#Arandu"* é o recurso em questão.

2.1.2 XML+ *Name spaces*+ XML *Schema*

A camada **XML+ *Name spaces*+ XML *Schema*** provê a sintaxe básica para descrição dos recursos de uma maneira independente de plataforma.

A linguagem de marcação XML não possui marcadores ou *tags* fixos, isso permite que um mesmo recurso possa ser descrito de várias maneiras. O XML *Schema* garante que um recurso seja definido com os mesmos marcadores em documentos diferentes, ou também que um recurso possa ser diferenciado de outro pelo seu *namespace*.

Um *namespace* é definido em um documento XML dentro de uma *tag* pelo atributo **"xmlns"**. Um exemplo de código XML com declaração de *namespaces* pode ser visto na Figura 3.

```
<h:table xmlns:h="http://www.w3.org/TR/html4/">
  <h:tr>
    <h:td>Apples</h:td>
    <h:td>Bananas</h:td>
  </h:tr>
</h:table>

<f:table xmlns:f="https://www.w3schools.com/furniture">
  <f:name>African Coffee Table</f:name>
  <f:width>80</f:width>
  <f:length>120</f:length>
</f:table>
```

Figura 3 – Exemplo de código XML com declaração de *namespaces*

2.1.3 RDF + RDF *Schema*

A camada **RDF + RDF *Schema***, também conhecida como **camada relacional**, possui como padrão oficial a linguagem RDF - *Resource Description Framework*.

Baseada em XML, RDF provê um padrão de representação dos metadados através de triplas. RDF foi projetado como uma maneira padrão para descrever dados e possibilitar que aplicações se comuniquem.

Na Figura 4 temos um exemplo de código RDF/XML, e na figura 5 temos o grafo gerado por esse código RDF/XML.

```
<?xml version="1.0"?>

<RDF>
  <Description about="http://www.cin.ufpe.br/sword#Arandu">
    <author>Adriano Ferraz</author>
    <homepage>http://www.cin.ufpe.br/sword</homepage>
  </Description>
</RDF>
```

Figura 4 – Exemplo de código RDF/XML

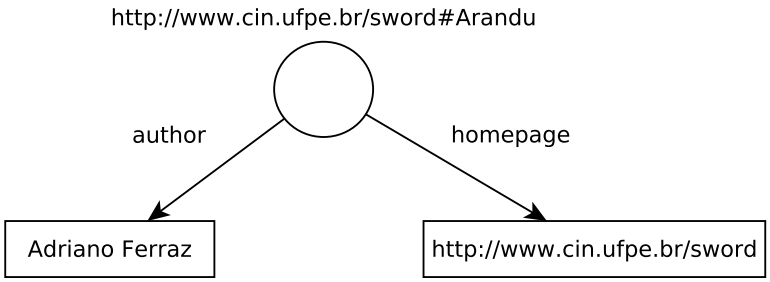

Figura 5 – Grafo do código RDF/XML da Figura 4

2.1.4 *Camada de ontologias*

A camada ***Ontology vocabulary***, ou **camada de ontologias**, é a camada que tem como finalidade dar uma maior expressividade e suprir as necessidades de representação de recursos da camada RDF. Esta camada tem como linguagem oficial a linguagem OWL - *Web Ontology Language*, e dada a sua importância principalmente no contexto deste trabalho, será descrita com detalhes na seção 2.3.

As outras camadas não possuem uma tecnologia oficial ou ainda não foram implementadas.

A próxima seção trata de Lógica de Descrições - DL, onde serão apresentados sua sintaxe, semântica e seu papel nas tarefas de raciocínio.

2.2 LÓGICA DE DESCRIÇÕES - DL

Lógica de Descrições ou *Description Logic* (DL) do termo em inglês, refere-se a uma família de linguagens formais de representação do conhecimento. DLs são usadas na Inteligência Artificial para descrever e raciocinar sobre conceitos relevantes de um domínio de aplicação.

No contexto da Web Semântica, DLs proveêm o formalismo lógico para linguagem OWL, permitindo que se criem axiomas lógicos para ontologias. O uso de DL em ontologias OWL permite que se tenham alguns perfis em OWL de acordo com a sua expressividade em DL. Nas seções a seguir, serão apresentadas características referentes a sintaxe,

semântica e tarefas de racíocinio em DL.

2.2.1 Sintaxe e Famílias de DL

Alguns construtores em DL também fazem parte dos construtores de Lógica de Primeira Ordem (FOL)[3], são eles: interseção ou conjunção, união ou disjunção, negação ou complemento, restrição universal e restrição existencial. Outros construtores não possuem construção correspondente em FOL, incluindo restrições sobre papéis, inversa, transitividade e funcionalidade (BAADER et al., 2003).

DL pode ser caracterizada em várias classes, também conhecidas como famílias, que irão definir qual seu poder de expressividade. A sintaxe dos membros de uma família DL, dependerá de quais construtores estão sendo utilizados e declarados. Na tabela 1 temos a sintaxe de DL e as possibilidades de linguagens que podemos formar de acordo com os construtos utilizados.

Tabela 1 – Sintaxe e Famílias de DL.

Descrição	Sintaxe	Linguagem
interpretação	$\mathcal{I}$	
domínio não vazio	$\Delta^{\mathcal{I}}$	
conceito	A, C, U	$\mathcal{FL}$
papel	R	
conjunção ou intersecção de conceitos	$C \sqcap U$	
quantificador universal	$\forall R.C$	
quantificador existencial	$\exists R$	
conceito superior	$\top$	$\mathcal{AL}^*$
conceito inferior	$\bot$	
negação	$\neg A \neg C$	
disjunção ou união de conceitos	$A \sqcup D$	
restrição existencial	$\exists R.C$	
restrição numérica	$(\geq n\ R)(\leq n\ R)$	
coleção de indivíduos	$\{a_1 ... a_n\}$	
hierarquia de papéis	$R \sqsubseteq S$	$\mathcal{H}$
inversão de papéis	R^-	$\mathcal{I}$
restrição numérica qualificada	$(\geq n\ R.C)(\leq n\ R.C)$	$\mathcal{Q}$

2.2.1.1 Lógica $\mathcal{ALC}$

Uma das famílias de DL que serve como base para DLs mais expressivas é a $\mathcal{ALC}$.

Os construtos que formam $\mathcal{ALC}$ são descritos na tabela 2.

[3] Lógica de Primeira Ordem ou *First Order Logic*, é também uma linguagem de representação baseada em lógica, porém com menor poder de expressividade do que DL.

Tabela 2 – Tabela de construtos $\mathcal{ALC}$

Sintaxe	Semântica	Descrição
A	$A^{\mathcal{I}} \sqsubseteq \Delta^{\mathcal{I}}$	conceito primitivo
R	$R^{\mathcal{I}} \sqsubseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$	papel primitivo
$\top$	$\Delta^{\mathcal{I}}$	conceito superior
$\bot$	$\emptyset$	conceito inferior
$\neg C$	$\Delta^{\mathcal{I}} \backslash C^{\mathcal{I}}$	complemento
$C \sqcap D$	$C^{\mathcal{I}} \sqcap D^{\mathcal{I}}$	conjunção
$C \sqcup D$	$C^{\mathcal{I}} \sqcup D^{\mathcal{I}}$	disjunção
$\forall R.C$	$x \mid \forall y.R^{\mathcal{I}}(x,y) \rightarrow C^{\mathcal{I}}(y)$	quantificador universal
$\exists R.C$	$x \mid \exists y.R^{\mathcal{I}}(x,y) \wedge C^{\mathcal{I}}(y)$	quantificador existencial

2.2.2 Bases de Conhecimento

Bases de conhecimento possuem como característica a ênfase na representação de conhecimento explicito para raciocínio automático. O raciocínio permite inferir o conhecimento representado que está explicitamente contido na Base de Conhecimento(KB[4]) (BRACHMAN; LEVESQUE, 2004).

DL suporta padrões de inferência que ocorrem em muitas aplicações de sistemas de processamento de informação inteligentes, e que também são usados pelos seres humanos para classificação de conceitos e indivíduos. Um sistema de Representação de Conhecimento baseado em DL tem como base a arquitetura apresentada na Figura 6 (BAADER et al., 2003).

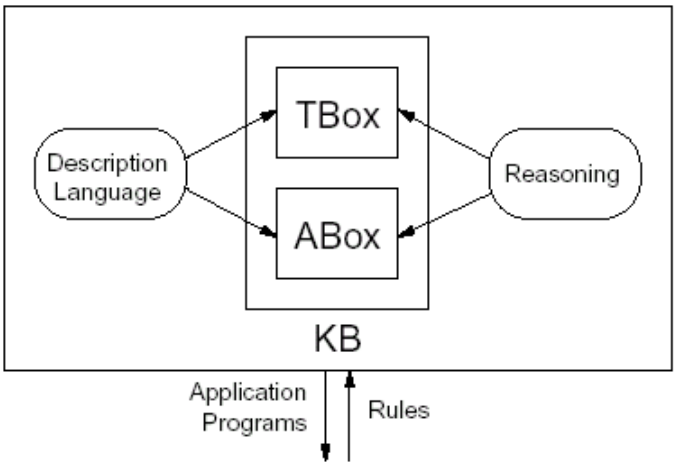

Figura 6 – Arquitetura base de um sistema de Representação do Conhecimento baseado em DL.

Uma base de conhecimento compreende dois componentes, o TBox $\mathcal{T}$ e o ABox $\mathcal{A}$.

[4] do termo em inglês *Knowledge Base* - KB

O TBox introduz a terminologia, ou seja, o vocabulário de um domínio de aplicação. Consiste em conceitos que denotam conjuntos de indivíduos e papéis, que denotam relações binárias entre indivíduos. Tarefas típicas de raciocínio para uma terminologia são determinar se uma descrição é satisfatória ou se uma descrição é mais geral do que outra.

Exemplo de componente TBox:

$Student \equiv Person \sqcap \exists studiesAt.University$

$PhdStudent \sqsubseteq Student \sqcap Researcher$

O ABox contém afirmações sobre indivíduos. Um problema importante para um ABox é descobrir se o conjunto de asserções é consistente, ou seja, se ele possui um modelo, e se as afirmações sobre um indivíduo específico são uma instância de uma descrição de um conceito.

Exemplo de componente ABox:

$PhdStudent(filipe)$

$studiesAt(filipe, UFPE)$

2.2.3 Tarefas de Raciocínio

As principais tarefas de Raciocínio em DL são satisfiabilidade, subsunção, equivalência e disjunção, e podem ser definidas formalmente como (BAADER et al., 2003):

- **Satisfiabilidade**: Um conceito C é *satisfatível* em relação a $\mathcal{T}$ se existir um modelo de $\mathcal{I}$ para $\mathcal{T}$ tal que $C^{\mathcal{I}}$ não seja vazio. Nesse caso dizemos que $\mathcal{I}$ é um modelo de C.

  Exemplo:

  TBox

  $Mulher \equiv Pessoa \sqcap Femea$

  $Mae \equiv Mulher \sqcap \exists temFilho.Pessoa$

  Teste de Satisfatibilidade

  $Mae(a) \equiv Mulher(a) \sqcap temFilho(a, b)$

  $Mae(a) \equiv (Pessoa(a) \sqcap Femea(a)) \sqcap temFilho(a, b)$

  O teste de satisfatibilidade é realizado adicionando instâncias e substituindo as regras, a fim de que o modelo continue válido.

- **Subsunção**: Um conceito C é *subconceito* de D em relação a $\mathcal{T}$ se $C^{\mathcal{I}} \sqsubseteq D^{\mathcal{I}}$ para cada modelo $\mathcal{I}$ de $\mathcal{T}$. Neste caso escrevemos $C \sqsubseteq_{\mathcal{T}} D$ ou $\mathcal{T} \models C \sqsubseteq D$.

  Exemplo:

  $Vertebrado \equiv Animal \sqcap \exists temParte.Osso$

  $Passaro \equiv Animal \sqcap \exists temParte.Osso \sqcap \exists temParte.Pena$

$AraraAzul \equiv Vertebrado \sqcap \forall temCor.Azul \sqcap \exists temParte.Pena$

$\models$

$AraraAzul \sqsubseteq Passaro$

- **Equivalência**: Dois conceitos C e D são *equivalentes* em relação a $\mathcal{T}$ se $C^{\mathcal{I}} = D^{\mathcal{I}}$ para cada modelo $\mathcal{I}$ de $\mathcal{T}$. Neste caso escrevemos $C \equiv_{\mathcal{T}} D$ ou $\mathcal{T} \models C \equiv D$

  Exemplo:

  TBox

  $Mulher \equiv Pessoa \sqcap Femea$

  $Homem \equiv Pessoa \sqcap \neg Mulher$

  $Masculino \equiv Pessoa \sqcap \neg Femea$

  Expansão do TBox e Simplificação

  $Homem \equiv Pessoa \sqcap \neg Mulher$

  $Homem \equiv Pessoa \sqcap \neg(Pessoa \sqcap Femea)$

  $Homem \equiv Pessoa \sqcap (\neg Pessoa \sqcup \neg Femea)$

  $Homem \equiv (Pessoa \sqcap \neg Pessoa) \sqcup (Pessoa \sqcap \neg Femea)$

  $Homem \equiv Pessoa \sqcap \neg Femea$

  Relacionamento entre os conceitos

  $Homem^I = Masculino^I$

- **Disjunção**: Dois conceitos C e D são *disjuntos* em relação a $\mathcal{T}$ se $C^{\mathcal{I}} \sqcap D^{\mathcal{I}} = \emptyset$ para cada modelo $\mathcal{I}$ de $\mathcal{T}$.

Dado o seguinte exemplo de KB:
$Homem \sqcup Mulher \equiv Pessoa$
$Homem \sqsubseteq \neg Mulher$
$Homem(Jose)$
Neste exemplo de KB, um raciocinador deve ser capaz de verificar:

1. a classe $Homem \sqcap Mulher$ é insatisfatível.
2. $Homem$ é subclasse de $Pessoa$.
3. esse KB é consistente.
4. inferir que $Pessoa(Jose)$.

DL fornece o formalismo necessário para que se possa definir um domínio. Na próxima Seção 2.3, será apresentada a linguagem OWL que possui seu formalismo em DL.

2.3 WEB ONTOLOGY LANGUAGE - OWL

OWL é uma linguagem desenvolvida para representar ontologias (vide seção 2) na Web. OWL atualmente é a linguagem padrão para este tipo de representação, recomendada pela W3C e considerada uma peça chave para a implementação da Web Semântica. OWL possui uma semântica formal que permite que aplicações processem o conteúdo das informações ao invés de apenas apresenta-las aos humanos.

Com OWL é possível criar ferramentas capazes de realizar inferências lógicas e formalizar conteúdo da Web. A linguagem OWL se encontra na versão 2 (OWL 2)[5], lançada em 2009 e revisada em 2012.

A linguagem OWL é uma linguagem baseada em lógica de descrições e permite expressar conhecimento que possa ser processado por computadores. Com ela, é possível raciocinar e verificar consistência.

OWL foi especialmente projetada para representar conhecimento complexo sobre coisas, grupos de coisas e relações, e pode ser definida formalmente como (KRÖTZSCH, 2012):

- Uma interpretação $\mathcal{I}$ consiste em um conjunto não vazio $\Delta^{\mathcal{I}}$ (chamado domínio) e uma função de interpretação $.^{\mathcal{I}}$ que atribui:
  - cada individuo a como um elemento $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$.
  - cada classe C como um conjunto $C^{\mathcal{I}} \sqsubseteq \Delta^{\mathcal{I}}$.
  - cada propriedade P como uma relação binária $P^{\mathcal{I}} \sqsubseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$

Restringindo os recursos de $\mathcal{ALCI}$ (tabela 3), é possível definir os três perfis de OWL 2, respectivamente, onde **IName**, **CName** e **PName** são individuos, classes e propriedades em nosso vocabulário e uma expressão $C \sqsubseteq C$ representa qualquer axioma da forma $D \sqsubseteq E$ com $D, E \in C$, incluindo $D \neq E$ (KRÖTZSCH, 2012).

Tabela 3 – Construtos $\mathcal{ALCI}$

$\mathcal{ALCI}$
Axiom $::= C \sqsubseteq C \mid C(IName) \mid P(IName, IName)$
$C ::= CName \mid \top \mid \bot \mid C \sqcap C \mid C \sqcup C \mid \neg C \mid \exists P.C \mid \forall P.C$
$P ::= PName \mid PName^{-}$

A linguagem OWL 2 pode ser definida em vários perfis que definem a sua complexidade. Esses perfis são OWL EL, OWL QL e OWL RL. A seguir detalhamos algumas características típicas destes perfis.

OWL 2 EL - Este é um perfil de OWL 2 que é definido para fornecer um comportamento altamente otimizado para ontologias grandes e complexas que dependem de definições de classes complexas. Os construtos de OWL 2 EL são apresentados na tabela 4, onde foi definida uma ontologia exemplo chamada $\mathcal{EL}tiny$.

[5] *https://www.w3.org/TR/owl2-overview/*

Tabela 4 – Construtos OWL 2 EL

$\mathcal{EL}tiny$
Axiom $::= C \sqsubseteq C \mid C(IName) \mid P(IName, IName)$
$C ::= CName \mid \top \mid \bot \mid C \sqcap C \mid \exists P.C$
$P ::= PName$

Exemplo de ontologia OWL 2 EL:

$Gato \sqsubseteq \exists captura.(Animal \sqcap Pequeno)$

$Animal \sqcap \exists captura.Animal \sqsubseteq Predador$

$Gato \sqsubseteq Animal$

Esses axiomas expressam que alguns gatos capturam animais pequenos, que animais que capturam animais são predadores e que gato é um animal.

OWL 2 RL - Este perfil de OWL 2 foi construído para otimizar a interseção de programas com regras em Lógicas de Descrição. OWL 2 RL fornece um perfil para a implementação de sistemas de raciocínio com de motores de inferência existentes. Os construtos de OWL 2 RL são apresentados na tabela 5, onde foi definida uma ontologia exemplo chamada $\mathcal{RL}tiny$.

Tabela 5 – Construtos OWL 2 RL

$\mathcal{RL}tiny$
Axiom $::= CL \sqsubseteq CR \mid CR(IName) \mid P(IName, IName)$
$CL ::= CName \mid \bot \mid CL \sqcap CL \mid CL \sqcup CL \mid \exists P.CL$
$CR ::= CName \mid \bot \mid CR \sqcap CR \mid \neg CL \mid \forall P.CR$
$P ::= PName \mid PName^{-}$

Exemplo de ontologia OWL 2 RL:

$Gato \sqsubseteq \forall captura.(Animal \sqcap Pequeno)$

$Animal \sqcap \exists captura.Animal \sqsubseteq Predador$

$Gato \sqsubseteq Animal$

Esses axiomas expressam que todos os gatos capturam somente animais pequenos, que animais que capturam animais são predadores e que gato é um animal.

OWL 2 QL - Este perfil de OWL 2 foi formulado especificamente como uma forma de capturar a semântica de modelos de bases de dados UML (*Unified Modeling Language*) e uso de OWL como uma linguagem de integração de dados. Os construtos de OWL 2 QL são apresentados na tabela 6, onde foi definida uma ontologia exemplo chamada $\mathcal{QL}tiny$.

Exemplo de ontologia OWL 2 QL:

$Gato \sqsubseteq \exists captura.(Animal \sqcap Pequeno)$

$\exists captura.\top \sqsubseteq Predador$

$\exists captura^{-}.\top \sqsubseteq Animal$

$Gato \sqsubseteq Animal$

Tabela 6 – Construtos OWL 2 QL

$\mathcal{QL}tiny$
Axiom ::= $CL \sqsubseteq CR \mid CR(IName) \mid P(IName, IName)$
$CL ::= CName \mid \top \mid \bot \mid \exists P.\top$
$CR ::= CName \mid \top \mid \bot \mid CR \sqcap CR \mid \neg CL \mid \exists P.CR$
$P ::= PName \mid PName^{-}$

Esses axiomas expressam que gato captura algum animal pequeno, que alguém que captura qualquer coisa é um predador, que o inverso de quem captura alguma coisa é um animal, e que gato é um animal.

A formalização de uma ontologia, como visto, permite que o conhecimento possa ser processado por computadores. No próximo capítulo trataremos da engenharia de ontologias uma sub-área da representação do conhecimento.

3 ENGENHARIA DE ONTOLOGIAS

Engenharia de ontologias é o nome dado a área destinada ao estudo de técnicas para a construção de ontologias, surgiu como um avanço para a área de Engenharia do Conhecimento. Engenharia do Conhecimento estuda teorias e práticas para representação do conhecimento e raciocínio, muito utilizada por sistemas especialistas (ISOTANI; BITTENCOURT, 2015).

O processo de construção de uma ontologia é um processo evolutivo e pode ser caracterizado pelos seguintes estágios: **Especificação, Conceitualização, Formalização, Implementação** e **Manutenção**, mostrados na Figura 7.

Figura 7 – Ciclo comum de construção de uma ontologia.

O processo de desenvolvimento de uma ontologia pode ter algumas destas fases realizadas simultaneamente, esse ciclo de construção garante que ela seja avaliada, documentada e codificada em uma linguagem formal, diminuindo a propagação de erros (FERNANDEZ-LOPEZ; GOMEZ-PEREZ; JURISTO, 1997). Nas seções seguintes serão descritos alguns critérios fundamentais para representação de conhecimento.

3.1 CRITÉRIOS PARA *DESIGN* DE ONTOLOGIAS

O projeto de construção de uma ontologia, ou *design* de uma ontologia, é uma tarefa que pode ser guiada por várias metodologias ou critérios. Quando escolhemos como representar algo em uma ontologia, estamos tomando decisões de *design* (GRUBER, 1995).

Para orientar e avaliar projetos de ontologias, precisamos de critérios objetivos que sejam baseados na finalidade do artefato resultante, em vez de se basearem somente em noções de naturalidade ou verdade. A seguir podemos ver um conjunto preliminar de critérios de *design* para ontologias cujo propósito é o compartilhamento de conhecimento e a interoperação entre programas baseados em uma conceituação compartilhada, esses critérios foram propostos por (GRUBER, 1995).

1. **Clareza**: Uma ontologia deve efetivamente comunicar o significado pretendido de termos definidos. As definições devem ser objetivas. Enquanto a motivação para definir um conceito pode surgir de situações sociais ou requisitos computacionais, a definição deve ser independente do contexto social ou computacional. O formalismo é fundamental para esse processo. É preferível o uso de axiomas lógicos para as definições, e um predicado definido por condições necessárias e suficientes é também preferível do que uma definição parcial. Todas as definições devem ser documentadas com linguagem natural.

2. **Coerência**: Uma ontologia deve ser coerente, isto é, deve sancionar inferências que sejam consistentes com as definições. No mínimo, os axiomas de definição devem ser logicamente consistentes. A coerência também deve aplicar-se aos conceitos que são definidos informalmente, como os descritos na documentação e exemplos de linguagem natural. Se uma sentença que pode ser inferida a partir dos axiomas contradiz uma definição ou exemplo dado informalmente, então a ontologia é incoerente.

3. **Extensibilidade**: Uma ontologia deve ser projetada prevendo o uso de vocabulário compartilhado. Ela deve oferecer recursos para que seja possível definir novos termos para usos especiais com base no vocabulário existente, sem exigir a revisão das definições existentes.

4. **Viés mínimo na codificação**: A conceituação deve ser especificada no nível de conhecimento sem depender de uma codificação simbólica específica. As escolhas de representação não devem ser feitas puramente para a conveniência de notação ou implementação. A codificação da ontologia deve favorecer ao máximo a possibilidade de agentes desenvolvidos em outros sistemas de representação poderem compartilhar as informações.

5. **Compromisso ontológico mínimo**: Uma ontologia deve exigir o comprometimento ontológico mínimo, suficiente para apoiar as atividades de compartilhamento e de conhecimento pretendidas. Uma ontologia deve fazer o menor número possível de afirmações sobre o mundo a ser modelado, permitindo que as partes comprometidas com a liberdade da ontologia se especializem e se instanciem conforme necessário.

3.2 METODOLOGIAS PARA CONSTRUÇÃO DE ONTOLOGIAS

A tarefa de construir uma ontologia não é trivial. A construção envolve muitos passos, por isso várias metodologias têm sido desenvolvidas na área de engenharia de ontologias. Segundo Uschold e King (USCHOLD; KING, 1995) uma metodologia deve compreender os seguintes passos:

- Identificar o propósito da ontologia;
- Capturar a ontologia em nível de conhecimento;
- Codificar;
- Integrar a ontologia com outras existentes;
- Avaliar;
- Documentar.

Esses passos inspiraram a criação de várias metodologias, como por exemplo a metodologia TOVE (FOX; GRÜNINGER, 1997) e a *metodologia unificada* de Uschold (USCHOLD, 1996). Esses trabalhos seminais guiaram também metodologias como METHONTOLOGY (FERNANDEZ-LOPEZ; GOMEZ-PEREZ; JURISTO, 1997), DILIGENT (PINTO; STAAB; TEMPICH, 2004), entre outros.

Identificar o propósito da ontologia é uma tarefa crucial. Realizar entrevistas com especialistas do domínio, analisar textos específicos do domínio, definir questões de competência (questões que a ontologia deva responder), são essenciais para definir o escopo e propósito da ontologia (CIMIANO; VÖLKER; BUITELAAR, 2010).

Capturar a ontologia em nível de conhecimento está relacionado a identificar os conceitos e relacionamentos chave, desambiguar definições, identificar os termos que irão pertencer aos conceitos e relacionamentos, e por fim concordar com o que foi definido (CIMIANO; VÖLKER; BUITELAAR, 2010).

Na etapa de codificação, o modelo informal concebido será convertido para uma linguagem formal. Segundo Uschold e King (USCHOLD; KING, 1995), uma representação ontológica deve separar a representação em nível de conhecimento da representação em uma linguagem formal. Essa separação evita viés de codificação na representação.

Na etapa de integração, Uschold e King citam ainda que integrar ontologias não é uma tarefa trivial, porém reusar e integrar ontologias é uma tarefa crucial (USCHOLD; KING, 1995).

Nas seções seguintes, descreveremos algumas metodologias que, assim como as já citadas, fundamentaram a metodologia utilizada por Arandu.

3.2.1 Ontology Development 101

A metodologia de desenvolvimento de ontologias 101 é uma metodologia interativa e flexível. Criada por pesquisadores da Universidade de Stanford, é uma das mais conhecidas (ISOTANI; BITTENCOURT, 2015).

As fases que compõem esta metodologia são (NOY; MCGUINNESS, 2001):

- Determinar o escopo do domínio da ontologia, bem como definir questões de competência;

- Considerar reuso de ontologias existentes;
- Enumerar termos importantes na ontologia;
- Definir classes e hierarquia de classes;
- Definir propriedades das classes;
- Definir as características das propriedades;
- Criar instâncias.

Essas fases formam um ciclo, que pode ser iterado várias vezes durante o processo de construção. Uma representação visual desta metodologia pode ser vistas na figura 8.

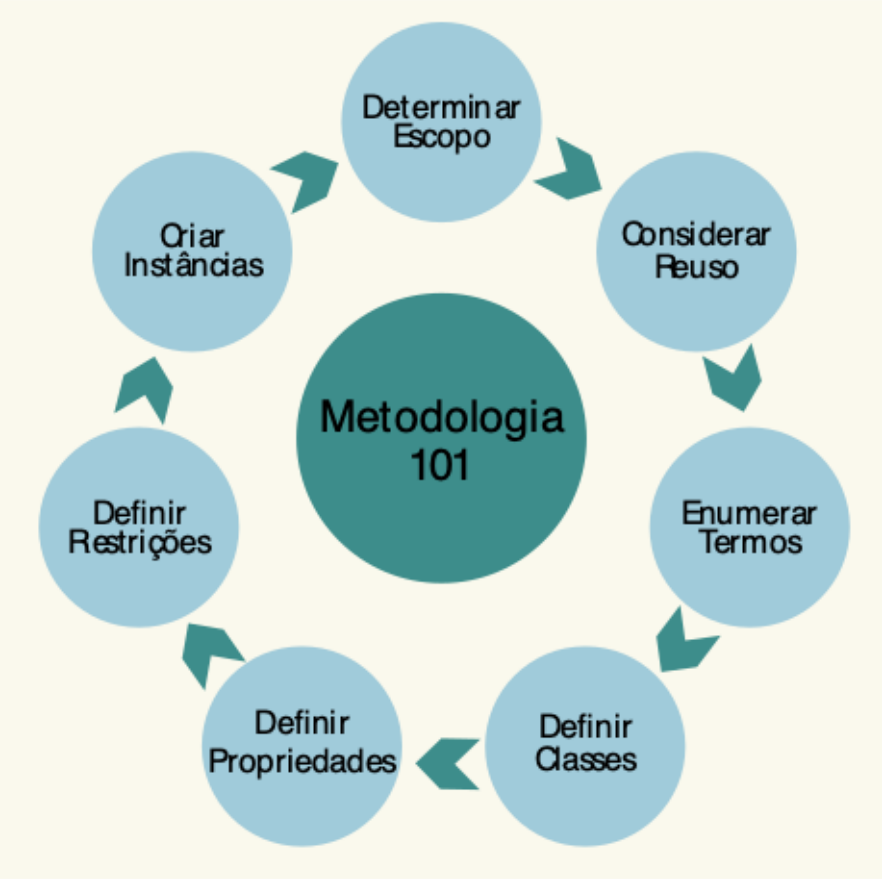

Figura 8 – Representação visual das fases que compõem a metodologia 101.

3.2.2 *METHONTOLOGY*

METHONTOLOGY é um método estruturado para construir ontologias. Foi baseado na experiência adquirida pelos seus autores desenvolvendo uma ontologia no domínio de produtos químicos (FERNANDEZ-LOPEZ; GOMEZ-PEREZ; JURISTO, 1997).

METHONTOLOGY é uma metodologia bem estruturada para construir ontologias a partir do zero. Ela inclui em seus processos técnicas para entregas de atividades produzidas e fomenta o reuso de ontologias existentes. A figura 9 apresenta visualmente as etapas desta metodologia, que a seguir são detalhadas.

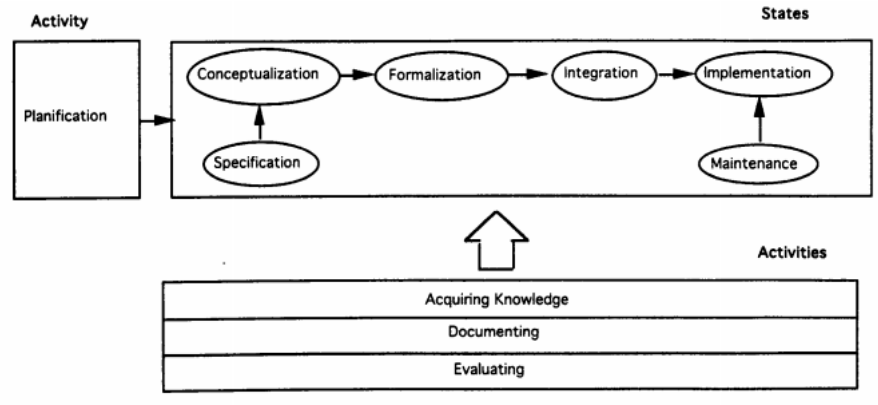

Figura 9 – Etapas que compõem a metodologia *METHONTOLOGY*. Fonte: (FERNANDEZ-LOPEZ; GOMEZ-PEREZ; JURISTO, 1997).

Especificação: o objetivo desta fase é produzir um documento em linguagem natural, formal, semi-formal ou informal com as especificações da ontologia. Este processo pode ser feito com um conjunto de representações intermediárias ou questões de competência.

Aquisição de Conhecimento: deve ser realizada em paralelo com a especificação dos requisitos da ontologia, e acompanha todo o processo de desenvolvimento da ontologia, sendo menor conforme ela avança. Livros, especialistas, manuais, outras ontologias, técnicas de *brainstorm* e entrevistas são ferramentas utilizadas nesta fase.

Integração: o objetivo desta fase é acelerar o processo de construção da ontologia, fazendo reuso de definições existentes em outras ontologias.

Implementação: é o processo de codificação da ontologia em uma linguagem formal com suporte de aplicação.

Avaliação: é o processo de checagem de consistência da ontologia. Significa realizar um julgamento técnico da ontologia e documentação em relação à especificação de requisitos durante cada fase e entre as fases do ciclo de vida. A avaliação envolve os termos verificação e validação.

Documentação: é um processo que METHONTOLOGY propõe que seja feito durante toda a construção da ontologia, relatando todas as fases de construção.

3.2.3 Neon Methodology

A Metodologia Neon para construção de ontologias é baseada em cenários, que suporta os aspectos colaborativos do desenvolvimento, reuso de ontologias e evolução em ambientes distribuídos (SUáREZ-FIGUEROA; GOMEZ-PEREZ; FERNáNDEZ-LóPEZ, 2012).

Neon é composta por um conjunto de nove cenários para a construção de ontologias, promovendo reuso de recursos ontológicos e não ontológicos. Neon provê ainda um glossário de processos e atividades envolvidos no desenvolvimento de ontologias, dois modelos de ciclo de vida de ontologias e um conjunto de diretrizes metodológicas para diferentes processos e atividades (SUáREZ-FIGUEROA; GOMEZ-PEREZ; FERNáNDEZ-LóPEZ, 2012). Na figura 10 é apresentado o fluxo de atividades que compõem os nove cenários da metodologia Neon.

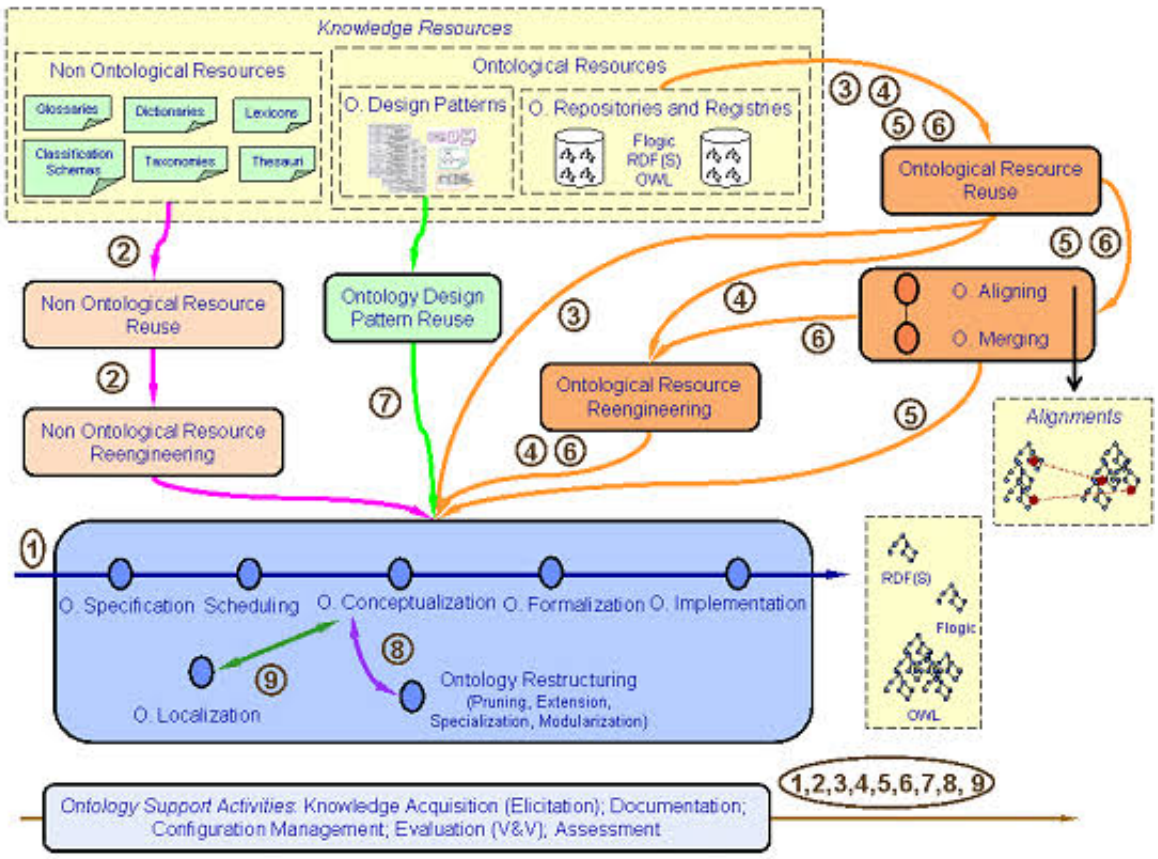

Figura 10 – Fluxo de cenários que compõem a metodologia Neon.

3.2.4 RapidOWL

RapidOWL é uma metodologia adaptativa e leve para engenharia de conhecimento colaborativo baseada em um refinamento iterativo. O principal objetivo de RapidOWL é fazer a elicitação, estruturação e processamento de conhecimento e, portanto, de cooperação de especialistas de um domínio e engenheiros de conhecimento (AUER; HERRE, 2007).

RapidOWL é uma metodologia incremental baseada em metodologias ágeis de desenvolvimento de software, que prega a divisão do processo de construção não em fases, mas em dimensões, baseadas em valores, princípios e práticas. Essas divisões podem ser vistas na figura 11 e detalhadas a seguir.

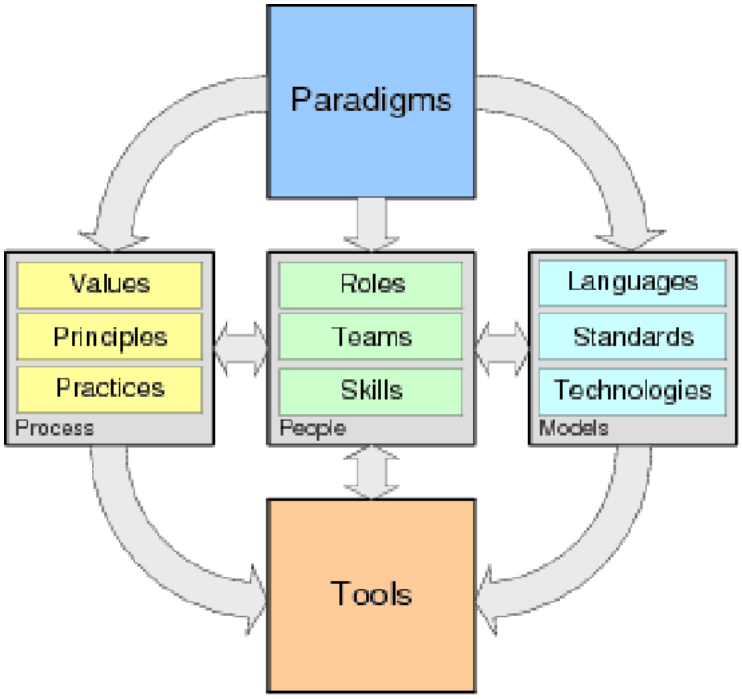

Figura 11 – Representação das dimensões que compõem a metodologia RapidOWL.

Valores são atividades relacionadas a comunicação, *feedback*, etc., que promovem colaboração, evolução e simplicidade da ontologia.

Princípios referem-se ao processo de desenvolvimento baseado em meio prazo, considerando mundo aberto, mudanças incrementais, autoria uniforme, métodos para modelagem e aquisição de instâncias, desenvolvimento observável e *feedback* rápido.

RapidOWL se baseia em práticas inspiradas em metodologias ágeis em desenvolvimento de software, exemplo *eXtreme Programming* (XP). As entregas de ontologias nessa metodologia são menores e o modelo se amplia ao longo do tempo.

3.2.5 Discussão sobre Metodologias para construção de ontologias

As metodologias para construção de ontologias são comparáveis às metodologias para construção de softwares e, assim como as metodologias de construção de software, também evoluíram. Porém, a construção de uma ontologia é um pouco mais ampla e abrange aspectos de representação do conhecimento que vão além da formalização de código.

Todas as metodologias citadas nas seções anteriores possuem suas etapas bem definidas e bem organizadas sequencialmente. Embora, todas elas proponham iteração durante as fases, a metodologia RapidOWL se destaca por prever uma maior revisão das etapas, com previsão de entregas e versões mais rápidas da ontologia. RapidOWL com sua proposta ágil se assemelha muito às metodologias ágeis de desenvolvimento de software, que têm sido bastante adotadas atualmente.

Uma atividade comum à todas metodologias apresentadas é o fomento ao reuso de outras ontologias. O reuso de outras ontologias provê agilidade no processo de construção

de uma ontologia (dentre outras vantagens que serão tratadas na próxima seção).

Nesta tese, propomos uma metodologia para construção de ontologias, inspirada nas etapas das metodologias apresentadas, com uma proposta ágil semelhante a RapidOWL.

A metodologia proposta, chamada de Arandu, possui como diferencial ser uma metodologia apoiada por software que guia o usuário em todas as suas etapas de construção da ontologia. Arandu permite a entrega de versões da ontologia a cada interação, a documentação é realizada em paralelo à construção e tem como principal avanço o reuso de outra ontologia de forma automática.

A metodologia Arandu permite que haja o reuso de termos de uma ontologia de topo de forma coerente. Com a introdução de uma etapa chamada de **desambiguação**, o especialista de domínio validará o reuso dos termos propostos pelo software que apoia a metodologia.

Arandu demonstrou ser possível que especialistas de domínio leigos (com pouco conhecimento de construção de ontologias) consigam construir e reutilizar uma ontologia de topo. Reutilizar uma ontologia, conforme está previsto em todas as metodologias apresentadas, não é uma tarefa trivial. Na seção a seguir, trataremos do reuso de ontologias.

3.3 REUSO DE ONTOLOGIAS

Alguns autores definem uma ontologia como um "vocabulário formal e compartilhado" (GRUBER, 1995). Essa definição bastante difundida remete ao conceito de criar uma ontologia que possa ser reutilizada e muitas vezes desenvolvida de forma colaborativa. Porém, este desenvolvimento deve ser guiado por uma metodologia para que seja eficiente e respeite o processo de *design* da ontologia (CIMIANO; VÖLKER; BUITELAAR, 2010).

Um dos propósitos de uma ontologia é poder ser reutilizada em vários contextos (CIMIANO; VÖLKER; BUITELAAR, 2010). Segundo (PINTO; STAAB; TEMPICH, 2004) podemos distinguir o reuso de uma ontologia em dois tipos:

- **Fusão ou *Merge***, este processo consiste em unificar várias ontologias que tenham similaridade ou pertençam ao mesmo domínio e criar uma nova ontologia.
- **Composição ou integração**, consiste em selecionar partes de ontologias existentes e integrar em uma nova ontologia, como um módulo ou submódulo.

No processo de fusão, não é possível identificar qual é a contribuição de cada ontologia para a nova ontologia. No processo de integração já é possível distinguir claramente qual a parte que a ontologia reutilizada integra a nova ontologia.

Outro método para reutilizar ontologias, é o alinhamento de ontologias. Métodos de alinhamento de ontologia, são baseados na similaridade semântica entre as ontologias. Segundo (CIMIANO; VÖLKER; BUITELAAR, 2010) podemos distingui-los em três tipos:

- Baseado na estrutura, compara as classes similares;
- Baseado nas instâncias, compara as instâncias e propriedades.
- Baseado em métodos léxicos, compara a distância/similaridade entre classes e propriedades.

Uma outra técnica de reuso de ontologias é a integração e *merge* a partir da identificação de ontologias apropriadas, que podem ser encontradas em serviços *web* como Swoogle[1] (CIMIANO; VÖLKER; BUITELAAR, 2010).

Selecionar uma ontologia apropriada para uma aplicação em particular às vezes pode ser uma tarefa difícil, uma solução para esse problema é selecionar ontologias fundacionais ou ontologias de topo (HEIJST; SCHREIBER; WIELINGA, 1997). Porém, ontologias de topo são grandes e complexas. Na tabela 7, apresentamos algumas vantagens e desvantagens em se reutilizar uma ontologias

Tabela 7 – Tabela com algumas vantagens e desvantagens em se reutilizar uma ontologia de topo.

Vantagens	amplia a quantidade classes
	torna a ontologia mais robusta
	mapeamento para termos que são consenso
	maior possibilidade de ser reutilizada
Desvantagens	classes que não possuem relação com o domínio
	torna a ontologia mais pesada, com classes desnecessárias
	processo complexo, necessita conhecer a ontologia de topo
	pouco automatizado, ferramentas de consulta a ontologias são pouco integráveis

De acordo com a tabela 7, podemos identificar que reutilizar uma ontologia de topo é benéfico, porém é uma tarefa difícil para leigos. Na próxima seção apresentaremos algumas técnicas utilizadas para construção de ontologias.

3.4 APRENDIZAGEM DE ONTOLOGIAS - *ONTOLOGY LEARNING*

O termo aprendizagem de ontologias, ou *Ontology learning* (do inglês), se refere ao conjunto de técnicas e métodos para geração automática de ontologias a partir de textos. Ontologias tendem a evoluir ao longo do tempo ou necessitam ser adaptadas para diferentes aplicações. Realizar isso de forma dinâmica ou automática é uma tarefa viável principalmente através de textos (MAEDCHE; STAAB, 2004).

Extrair termos, relações e uma classificação taxonômica de um texto, é uma tarefa complexa que envolve várias sub-tarefas. As tarefas que envolvem o processo de geração

[1] http://swoogle.umbc.edu/.

automática de uma ontologia podem ser divididas em camadas. Na figura 12 é apresentada uma visualização dessas camadas que compõem a aprendizagem de ontologias (CIMIANO, 2006).

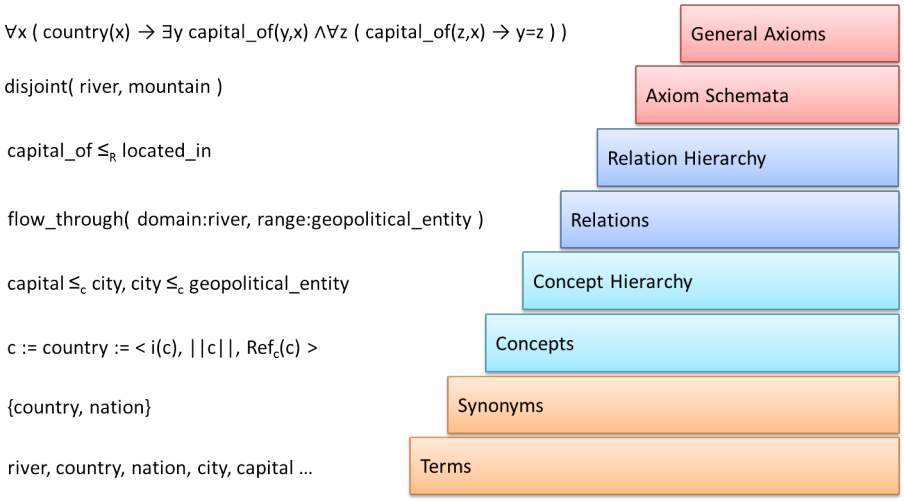

Figura 12 – Camadas que compõem a Aprendizagem de Ontologias (CIMIANO, 2006).

Na base do conjunto de camadas que compõem a aprendizagem de ontologias temos **terms**, ou extração de termos, uma tarefa que é o pré-requisito para todas as tarefas da aprendizagem de ontologias. A extração de termos é uma tarefa de PLN, especificamente alguma implementação de um etiquetador de classe gramatical sobre o corpus do domínio escolhido para identificação dos termos relevantes (MANNING et al., 2014).

No nível de **synonyms** é realizada a aquisição das variações semânticas dos termos extraídos. A aquisição dos sinônimos é feita com o apoio de alguma ferramenta como por exemplo o WordNet[2], e pode englobar tanto termos dentro do mesmo idioma como traduções.

Na camada **concepts** são extraídos os conceitos, que devem ser a sua própria definição no contexto, ou uma extensão de suas possíveis instâncias, ou ainda um conjunto de instâncias que o compõem em uma perspectiva linguística. Segundo (CIMIANO; VÖLKER; BUITELAAR, 2010), os conceitos devem incluir a geração de definições formais e informais, onde definições informais formarão um glossário e definições formais formarão a base para extrair relações e axiomas.

No nível **concept hierarchy** temos o cerne de uma ontologia, a definição hierárquica dos conceitos. Independente da linguagem formal que se utilize para representar uma ontologia, a parte mais importante é esta relação taxonômica. Em aprendizagem de ontologias, a maioria das técnicas se baseia na extração de padrões linguísticos de hiponímias para construir essa estrutura. A seguir temos alguns exemplos de regras linguísticas que podem ser utilizadas para extrair essa hierarquia de conceitos (HEARST, 1992).

[2] https://wordnet.princeton.edu/

Nas camadas **relations** e **relation hierarchy**, há a extração de relações. Ela depende do aspecto sintagmático da linguagem, isto é, até que ponto as unidades linguísticas estão conectadas dentro de uma frase, parágrafo ou discurso mais extenso. Portanto, a extração de relações está relacionada ao problema de adquirir restrições de seleção para argumentos, por exemplo a análise de sujeito, verbo e predicado em uma frase.

Nas camadas **axiom** e **general axioms**, são formadas as regras lógicas e axiomas lógicos. Segundo (CIMIANO; VÖLKER; BUITELAAR, 2010), esta é uma tarefa que tem atraído muito a atenção da comunidade de aprendizagem de ontologias. Isso se dá devido à dificuldade de extrair essas regras lógicas a partir de um texto. Alguns trabalhos em aprendizagem de ontologias tratam apenas de axiomas de disjunção, por exemplo (HAASE; VÖLKER, 2005) e (VÖLKER et al., 2007).

Contudo, (CIMIANO; VÖLKER; BUITELAAR, 2010) citam que o desafio mais importante é integrar essas diferentes técnicas de forma que sistemas de engenharia de ontologia eficazes (de alto desempenho) e eficientes (robustos e rápidos) possam ser projetados, com técnicas de PLN integradas e transparentes ao engenheiro.

3.5 ONTOLOGIAS DE TOPO

Uma ontologia de topo, ou uma ontologia superior, descreve conceitos gerais que são independentes de domínio. Essas ontologias representam, no mais alto nível conceitual, as coisas que compõem o Mundo. Um exemplo de conceitos que podem ser definidos em uma ontologia de topo são conceitos de espaço, tempo, eventos, objetos, etc.

Nas seções seguintes, serão mostradas algumas ontologias de topo e suas características.

3.5.1 SUMO - *Suggested Upper Merged Ontology*

A ontologia *Suggested Upper Merged Ontology* (SUMO) é uma ontologia formal. Começou como uma ontologia de topo com cerca de 1000 termos e 4000 axiomas usando declarações SUO-KIF (PEASE, 2011).

SUO-KIF é uma variante da linguagem KIF (*Knowledge Interchange Format*[3]), uma linguagem declarativa capaz de expressar declarações arbitrárias em lógica de primeira ordem com suporte a raciocinadores.

Os axiomas de SUMO incluem cerca de 750 regras. SUMO abrange noções gerais de senso comum, tais como tempo, relações espaciais, objetos físicos, eventos e processos. O nível superior de SUMO é limitado a cerca de 1000 termos, a fim de mantê-la gerenciável e de fácil entendimento (PEASE, 2011).

[3] http://logic.stanford.edu/kif/dpans.html

Para exemplificar, na figura 13 temos uma parte da ontologia SUMO retirada a partir do termo *Organism*[4], essa consulta foi fornecida pela ferramenta *Sigma Knowledge Engineering Environment* [5]. *Sigma* é um sistema para desenvolver, visualizar e depurar teorias em lógica de primeira ordem, baseado em KIF desenvolvido para a SUMO.

Nessa consulta (figura 13) pode ser visto algumas superclasses e subclasses da classe *Organism* e suas definições.

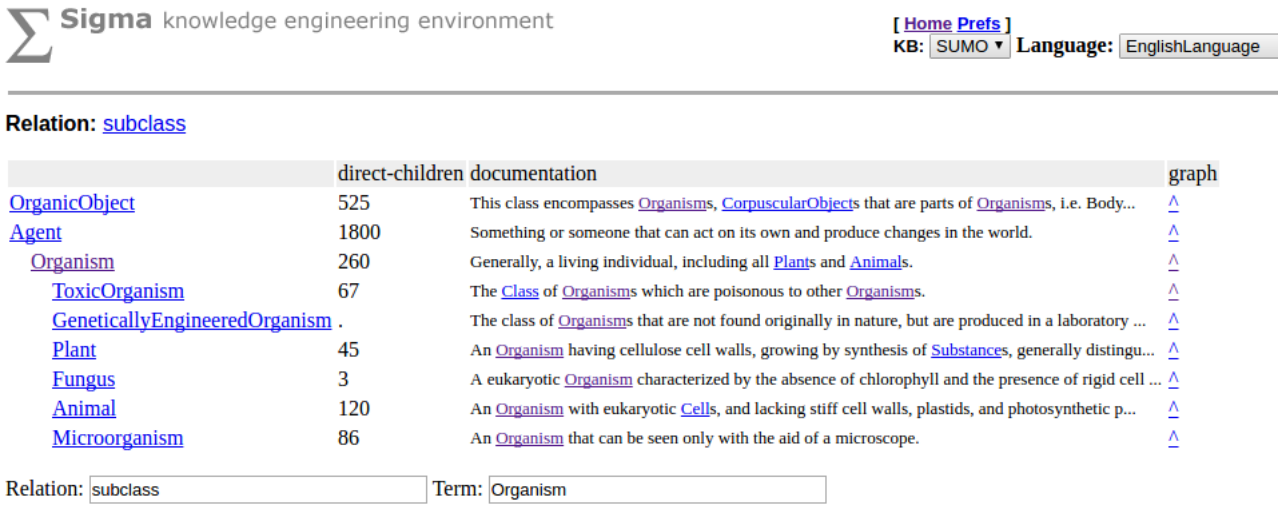

Figura 13 – Parte da ontologia SUMO a partir do termo *Organism*

3.5.2 CYC

A Base de Conhecimento do Cyc é uma representação formalizada de uma vasta quantidade de conhecimento humano fundamental: fatos, regras básicas e heurísticas para raciocinar sobre os objetos e eventos da vida cotidiana (CYCORP, 2017). Este conhecimento é representado em uma linguagem formal, CycL. A ontologia Cyc consiste em termos e afirmações que relacionam esses termos. Essas afirmações incluem tanto fatos simples (isto é, afirmações fundamentadas) quanto regra (LENAT, 1995)(CYCORP, 2017).

O projeto foi iniciado em 1984 como parte da Microelectronics e Computer Technology Corporation[6]. O objetivo era codificar, em forma utilizável por máquina, milhões de peças de conhecimento que compõem o senso comum humano. Cyc é um esquema de representação de conhecimento proprietário que utiliza relações de primeira ordem. Porém, a empresa detentora desta base de conhecimento fornece uma versão para uso livre pela comunida a OpenCyc (CYCORP, 2017).

A versão mais recente do OpenCyc inclui toda a ontologia Cyc contendo centenas de milhares de termos, juntamente com milhões de afirmações relacionando os termos entre

[4] http://sigma.ontologyportal.org:8080/sigma/Graph.jsp?kb=SUMO&lang=ArabicLanguage&term=Organism.
[5] http://sigmakee.sourceforge.net/
[6] http://www.cyc.com/

si. A ontologia Cyc, cujo domínio é toda a realidade de consenso humano, inclui em sua versão atual (CYCORP, 2017):

- 239.000 termos;
- 2.093.000 triplas;
- Instâncias de Classe (entre outras)

  place: 19,000

  organization: 26,000

  predicate: 22,000

  business related thing: 28,000

  person: 12,700

Na Figura 14 temos uma parte de código da ontologia Cyc a partir do termo *The Aorta*[7].

```
Hierarchical display radiating from (The Aorta)
      isa: Aorta, GolemRole, TheTerm
      Context: Union of all contexts
      Predicate: physicalParts
      Index: 1

-> (The WholeBody)
          (The CirculatorySystem) [VertebrateBodyStructureMt]
-> (The Person)
                (The Aorta)
                [GolemTestDataMt -> ((The Person))]
                [VertebrateBodyStructureMt -> ((The CirculatorySystem))]
                      (The AorticArch) [HumanBodyStructureMt] <-
                      (The AscendingAorta) [VertebrateBodyStructureMt]
                            (The Mediastinum) [HumanBodyStructureMt] ...
                      (The DescendingAorta) [VertebrateBodyStructureMt]
                            (The AbdominalAorta) [VertebrateBodyStructureMt] <-
                            (The ThoracicAorta) [VertebrateBodyStructureMt] <-
                      (The ThoracicAorta) [VertebrateBodyStructureMt] <-
```

Figura 14 – Parte da ontologia Cyc a partir do termo *The Aorta*.

3.5.3 UFO - *Unified Foundational Ontology*

A *Unified Foundational Ontology* (UFO) é uma ontologia de fundamentação, implementada em liguagem OWL, baseada em várias teorias de ontologia Formal, Lógica filosófica, Filosofia da Linguagem, Linguística e Psicologia Cognitiva, e desenvolvida com o intuito de prover uma fundamentação ontológica para linguagens gerais de modelagem conceitual (ZAMBORLINI, 2011).

Segundo (GUIZZARDI, 2005) UFO, é organizada em três camadas incrementais de conjuntos de conformidade:

[7] *The Aorta* se refere-se a **Aorta**, é o nome dado à maior e mais importante artéria do sistema circulatório do corpo humano.

1. UFO-A: é essencialmente a ontologia fundacional. Composta pelos *Endurants*, que estão no tempo, no sentido de que, se dissermos que, em uma circunstância $c1$, um *Endurant* tem uma propriedade $P1$ e em uma circunstância $c2$ tem a propriedade $P2$ (possivelmente incompatível com $P1$), estamos nos referindo ao mesmo *Endurant*. Exemplos de *Endurants* são uma casa, uma pessoa, a lua, um buraco, uma quantidade de areia. Por exemplo, podemos dizer que um indivíduo *John* pesa 80kg em $c1$, mas 68kg em $c2$, no entanto, estamos nos referindo ao mesmo *John*.

2. UFO-B: um incremento para UFO-A. Define os termos relacionados com *Perdurants*, também conhecidos como ocorrências, acidentes ou acontecimentos, são aquelas entidades para as quais apenas existe uma parte se as observarmos em um instante de tempo. Por exemplo, "correr". Se congelarmos o tempo, só vemos uma parte da execução, sem qualquer conhecimento prévio, nem sequer poderia determinar o processo real como sendo um processo de execução. Outros exemplos incluem uma ativação, um beijo ou um procedimento.

3. UFO-C: um incremento para UFO-B. Define os termos explicitamente relacionados com as esferas de coisas intencionais e sociais, incluindo as coisas linguísticas, exemplo comunicar-se.

Esta divisão reflete uma certa estratificação do nosso mundo do ponto de vista ontológico. Também reflete diferentes graus de consenso científico, onde há mais consenso sobre a ontologia dos *Endurants* do que sobre a ontologia dos *Perdurants* e há mais consenso sobre a ontologia dos *Perdurants* do que sobre a ontologia das coisas intencionais e sociais. (GUIZZARDI, 2005)

Na Figura 15 temos uma visualização da ontologia UFO-A a partir do termo *Object*. Nesta consulta é possível verificar as subclasses e a definição da classe *Object*.

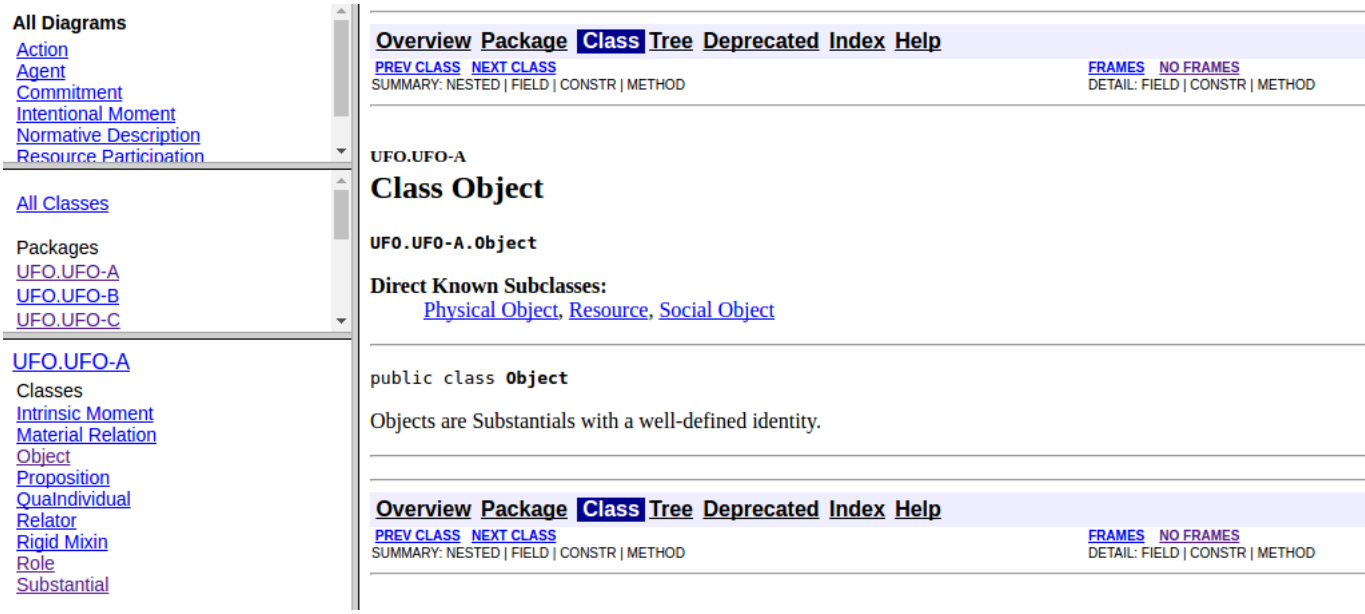

Figura 15 – Parte da ontologia UFO-A a partir do termo *Object*.

3.6 DISCUSSÕES

O processo de representação do conhecimento é uma tarefa bastante pesquisada e formalmente bem definida como foi possível ver nas seções anteriores.

Ontologias de topo permitem a representação não somente de coisas físicas, mas também de eventos de ocorrem no tempo. Termos em uma ontologia de topo são representados no mais alto nível de abstração e fornecem axiomas que podem ser reutilizados por subclasses favorecendo a identificação de inconsistências conceituais.

O processo de construção de ontologias ainda é uma atividade onerosa, visto que representar formalmente um conhecimento é uma tarefa que ainda exige conhecimento tecnológico, por exemplo entender OWL.

Arandu se posiciona como uma ferramenta que permite que especialistas de domínio com pouco conhecimento sobre ontologias de topo possam se favorecer das vantagens de reutilizar uma ontologia de topo.

O próximo capítulo tratará de Processamento de Linguagem Natural, uma atividade computacional fundamental para alcançar alguns dos objetivos desta proposta de Tese.

4 PROCESSAMENTO DE LINGUAGEM NATURAL

Processamento de Linguagem Natural (PLN) é uma área da Ciência da Computação, dentro do campo da Inteligência Artificial, que estuda os problemas da geração e compreensão automática de línguas humanas naturais (LIDDY, 2001).

Alguns desafios de PLN envolvem a compreensão e geração de textos em linguagem natural conectando o computador e o ser humano, uma tarefa considerada computacionalmente difícil.

O processamento de linguagem natural permite que desenvolvedores criem, por exemplo, sistemas de diálogo, resumos automáticos, sistemas de tradução de textos, reconhecimento de entidades, extração de relacionamentos, análise de sentimentos, reconhecimento de fala e segmentação de tópicos, entre outros.

O trabalho de processamento de linguagem natural é um processo de análise, interpretação e geração da linguagem e pode ser dividido em vários estágios visualizados na Figura 16 (VIEIRA; LIMA, 2001), e serão melhor detalhados nas seções seguintes.

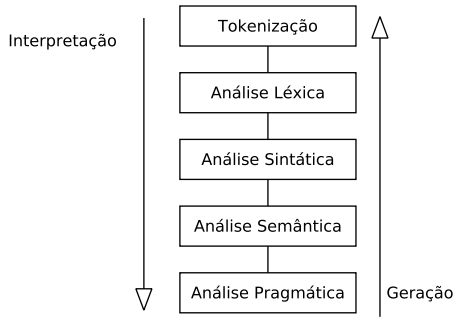

Figura 16 – Estágios do Processamento de Linguagem Natural

4.1 ESTÁGIOS DA ANÁLISE DA LINGUAGEM BASEADA EM REGRAS

Dado qualquer sistema de compreensão de linguagem natural, uma tarefa comum a eles, independente da técnica, é traduzir a sentença original em uma representação interna do seu significado. Duas etapas previamente utilizadas são as fases de *Tokenização* e análise léxica (VIEIRA; LIMA, 2001).

O processo de **Tokenização** é o processo de identificação dos *tokens* em uma sentença, que formarão as palavras . Nem todas as línguas fornecem as palavras em forma de texto separadas por espaços, por exemplo a língua Chinesa.

A **Análise Léxica** em processamento de linguagem natural refere-se aos mecanismos para análise textual a nível de palavra. Um processo conhecido da análise léxica é o processo de *lemmatização*, que consiste em reduzir a palavra ao seu radical utilizando

recursos de análise linguística. Por exemplo a palavra *pedalando* possui *lemma* ou radical, *pedal*.

Normalmente o estágio de compreensão de linguagem natural segue o fluxograma proposto por (LUGER, 2004), mostrado na figura 17.

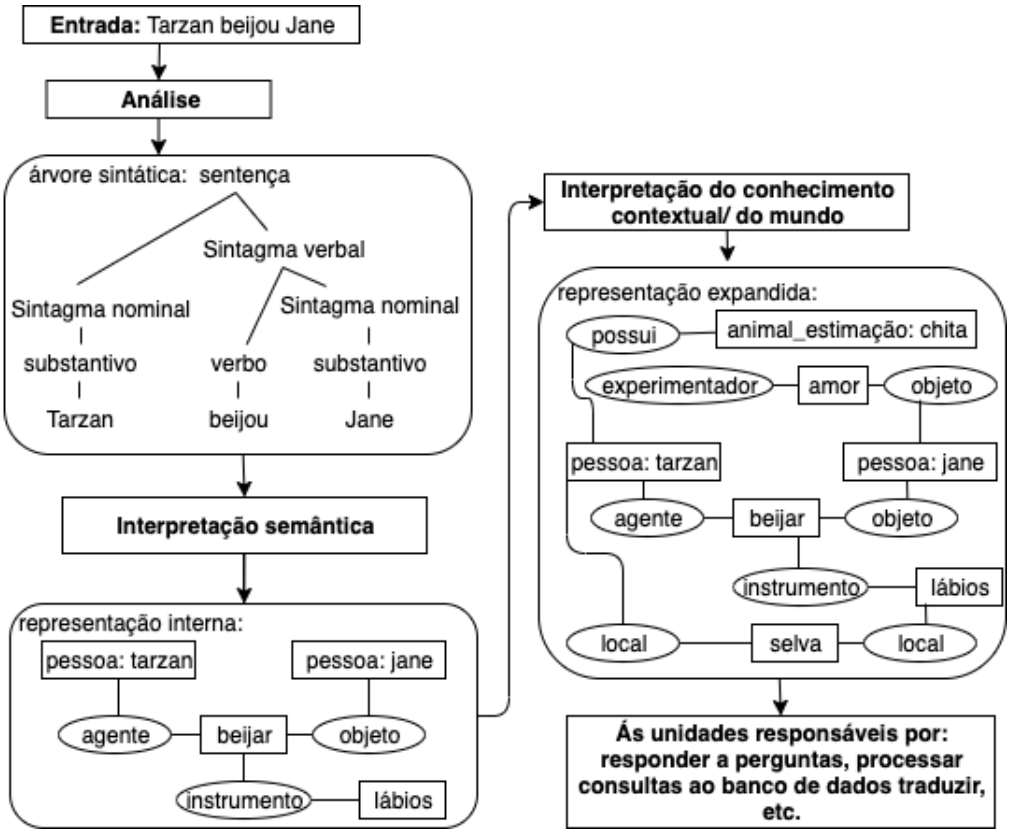

Figura 17 – Estágios de compreensão de linguagem natural (LUGER, 2004).

4.1.1 Sintaxe para análise baseada em regras

Uma das técnicas para realizar **análise sintática** é através do uso de **gramáticas livres de contexto**. Estas gramáticas são formadas por regras que formam um dicionário de palavras e definem um léxico da linguagem. A seguir temos um exemplo de regras que definem uma gramática para a sentença "O homem acaricia o cão"(as separações por vírgula à direita da regra indicam que são necessários os dois componentes para que ela seja válida):

1. sentença ↔ sintagma-nominal, sintagma-verbal
2. sintagma-nominal ↔ substantivo
3. sintagma-nominal ↔ artigo, substantivo
4. sintagma-verbal ↔ verbo

5. sintagma-verbal ↔ verbo, sintagma-nominal
6. artigo ↔ um
7. artigo ↔ o
8. substantivo ↔ homem
9. substantivo ↔ cão
10. verbo ↔ acaricia
11. verbo ↔ morde

Regras como esta permitem que se criem derivações que formem sentenças válidas. Existem variações do tipo *top-down*, que começa da sentença até os símbolos terminais, ou o inverso, do tipo *bottom-up*.

Nesse processo, os símbolos terminais ou *tokens* são etiquetados com suas respectivas funções na sentença. Por exemplo sintagma verbal, sintagma nominal, substantivo, verbo, adjetivo, advérbio, etc. Ferramentas de processamento de linguagem natural comumente utilizam abreviações para identificar essas etiquetas, esses símbolos podem ser vistos na tabela 8 (VIEIRA; LIMA, 2001).

Tabela 8 – Tabela com as abreviações das etiquetas comumente utilizadas por ferramentas de PLN.

Etiqueta	Função(inglês)	Função(português)
NN	noun, singular or mass	substantivo
NNP	proper noun, singular	substantivo próprio singular
NNPS	proper noun, plural	substantivo próprio plural
JJ	adjective	adjetivo
RB	adverb	advérbio
IN	preposition	preposição
VB	verb, base form	verbo no infinitivo
VBD	verb, past tense	verbo no passado
VBN	verb, past participle	verbo no particípio passado
VBP	verb, non-3rd person singular present	verbo na terceira pessoa

Uma derivação pode ser representada como uma estrutura de árvore, também conhecida como **árvore sintática**. Esta análise gera uma árvore hierárquica permitindo uma futura análise semântica, exemplo na Figura 18.

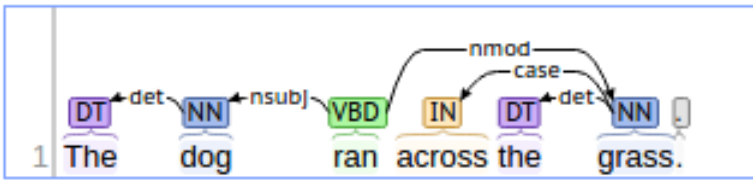

Figura 18 – Exemplo de árvore Sintática.

4.2 LINGUAGEM NATURAL CONTROLADA

Linguagem Natural Controlada (LNC) é o nome dado a subconjuntos de linguagem natural que têm como objetivo restringir ou reduzir uma gramática ou vocabulário (WYNER et al., 2010). LNCs são comumente utilizadas para melhorar legibilidade de leitores humanos ou para permitir análise semântica automática por softwares (SCHWITTER, 2010).

LNCs são efetivas na representação de conhecimento, pois contribuem para eliminar ambiguidades da linguagem natural preservando a coerência e expressividade (GAO, 2019). LNCs, semânticamente bem definidas, podem ser traduzidas deterministicamente para linguagens formais como Prolog, DL, entre outras (GAO, 2019).

Segundo (KUHN, 2014), uma LNC pode ser definida como uma linguagem construída a partir de uma linguagem natural, sendo mais restritiva lexicalmente, semanticamente e sintaticamente, porém preservando suas propriedades.

LNCs podem ser agrupadas em duas categorias (SCHWITTER, 2010):

- *Human-oriented* - possui foco na melhoria da comunicação entre humanos, provendo legibilidade e compreensão de textos técnicos;
- *Machine-oriented* - possui foco no processo de aquisição, representação e processamento de conhecimento realizado por máquinas.

4.2.1 LNCs de Propósito Geral

Nesta seção apresentaremos algumas LNCs *machine-oriented* que não foram desenvolvidas para um cenário em específico, que são comumente utilizadas como linguagens de representação do conhecimento.

Attempto Controlled English (ACE) (FUCHS; KALJURAND; KUHN, 2008) é uma LNC formada por um subconjunto do inglês que permite, por meio de uma estrutura de representação, conversão para lógica de primeira ordem e raciocínio automático. ACE consiste em um pequeno conjunto de regras de construção que descrevem sua sintaxe para construção de frases que podem ser ambíguas, usando o inglês completo.

O vocabulário de ACE é formado por sintagmas predefinidos, conjunções, pronomes, verbos, substantivos, adjetivos, advérbios, entre outros. ACE pode ser integrado a al-

guma ferramenta e contribui para que usuários construam frases com o apoio de avisos e mensagens de erro.

Processable English (PENG) (WHITE; SCHWITTER, 2009), é uma LNC semelhante ao ACE, porém mais leve, que abrange um conjunto menor do idioma inglês. PENG foi uma das primeiras LNCs usadas por editores preditivos. PENG fornece suporte à escrita e exime o usuário de aprender ou lembrar as regras de sua LNC.

PENG impõe dinamicamente restrições gramaticais de sua LNC enquanto o texto é construído. Para cada palavra inserida no editor, uma lista de opções é gerada incrementalmente, que dita como a estrutura da sentença atual pode ser continuada. As restrições sintáticas de PENG fazem com que o texto siga suas regras LNC e permita conversão para lógica de primeira ordem e processamento por um provador de teoremas.

Computer Processable Language (CPL) (CLARK et al., 2010), é uma linguagem controlada desenvolvida pela *Boeing Reserch an Technology*[1]. Diferente de ACE, que aplica um pequeno conjunto de regras de construção, e diferente de PENG, que usa um editor preditivo, CPL utiliza regras heurísticas para resolver ambiguidades no significado das palavras, provê rotulação semântica, interpretação de substantivos compostos, resolução de metonímia e outras atividades de processamento de linguagem.

CPL é usada principalmente para codificar conhecimento de senso comum e específico de domínio, e permitir que engenheiros de conhecimento façam consultas de maneira compreensível.

Nessa seção foram apresentadas alguns exemplos de LNCs que contribuem para que textos sejam construídos de forma que haja processamento computacional. Nessa tese não definimos formalmente uma LNC e não utilizamos outra LNC, porém utilizamos regras baseadas em LNCs para que as sentenças possam ser processadas.

As regras utilizadas por Arandu definem apenas o uso de sujeito simples, com frases completas que podem ter predicado simples ou composto, preferencialmente em voz ativa.

Arandu verifica por meio de um raciocinador se o axioma foi construído corretamente a partir da frase informada, esta atividade é dependente do processador de linguagem natural utilizado. Caso ele não consiga realizar a tradução, Arandu solicita que o usuário redigite a sentença e sugere um formato de frase que ele consiga traduzir.

A atividade de reuso de uma LNC ou a definição formal de uma LNC é uma tarefa importante para esta tese, e está prevista como os trabalhos futuros.

4.3 CHATBOTS

Os *Chatbots*, também conhecidos como *Chatterbots*, são sistemas de diálogo, onde o usuário humano introduz um texto e o *Chatbot* traz uma resposta (ALMANSOR; HUSSAIN, 2020).

[1] https://www.boeing.com/innovation/

Uma das motivações para os desenvolvedores de *Chatbots* é tentar enganar o usuário em pensar que eles estão conversando com outro ser humano. Inclusive, foram propostos desafios como o conhecido "Teste de Turing", proposto por Alan Turing. O teste tem como desafio que uma máquina seja capaz de dialogar com um humano a ponto de que este não possa distinguir se está falando com uma máquina ou com um ser humano (WEIZENBAUM, 1966).

Anualmente é realizado um concurso chamado Loebner Prize[2], lançado em 1991 pelo Dr. Hugh Loebner. O desafio já teve vários *chatbots* que conseguiram passar no teste, obtendo notas melhores que humanos.

Chatbots são programas de computador que trabalham com linguagens de programação orientadas para a manipulação da linguagem natural humana, para tanto utilizam técnicas de PLN (BERNARDINI; SôNEGO; POZZEBON, 2018). Um *chatbot* basicamente recebe sentenças em linguagem natural, que podem ser perguntas, e retorna uma resposta formulada a partir de sua base dados, associada a essa questão.

A maioria das conversas com *Chatbots* são baseadas em texto, embora alguns mais recentes façam uso da fala para entrada e saída, como por exemplo os Assistentes Virtuais[3]. Alguns incluem avatares ou cabeças falantes para dotar o *Chatbot* com característica mais humana. Geralmente, a interação com *Chatbots* toma a forma de conversa em que o *Chatbot* responde à entrada do usuário, ao invés de tomar a iniciativa na conversa, embora em alguns casos o *Chatbot* faça perguntas para manter a conversa (MCTEAR; CALLEJAS; GRIOL, 2016).

Podemos categorizar os *chatbots* em dois grupos. Os que conseguem interpretar mensagens do usuários e aprender a realizar tarefas, e os que realizam tarefas objetivas, com um vocabulário limitado ao que foi pré-definido. Ludicamente, a figura 19 ilustra um diálogo de um *chatbot* explicando esta divisão.

Podemos categorizar os *chatbots* também, de acordo com os avanços tecnológicos, que ditam as suas gerações. Nas seções a seguir apresentamos estas categorias.

4.3.1 Chatbots baseados em casamento de padrões

Chatbots baseados em casamento de padrões são desenvolvidos em uma estrutura conhecida como árvore de diálogo, com respostas objetivas e diretas. As respostas então são programadas a partir de regras específicas. Embora estes *chatbots* não utilizem técnicas aprimoradas de inteligência artificial, eles são bastante utilizados e cumprem seu objetivo em tarefas com propósitos bem definidos, como por exemplo atendimento a clientes, pedidos, etc.

[2] O Loebner Prize de Inteligência Artificial (IA) é a primeira instanciação formal de um Teste de Turing. http://www.loebner.net/Prizef/loebner-prize.html

[3] Asssistentes Virtuais são aplicativos de *smartphones* que auxiliam em tarefas como controle de agenda, previsão do tempo, informações de trânsito, etc. Os aplicativos mais conhecidos são SIRI da Apple, Google Now da Google, entre outros.

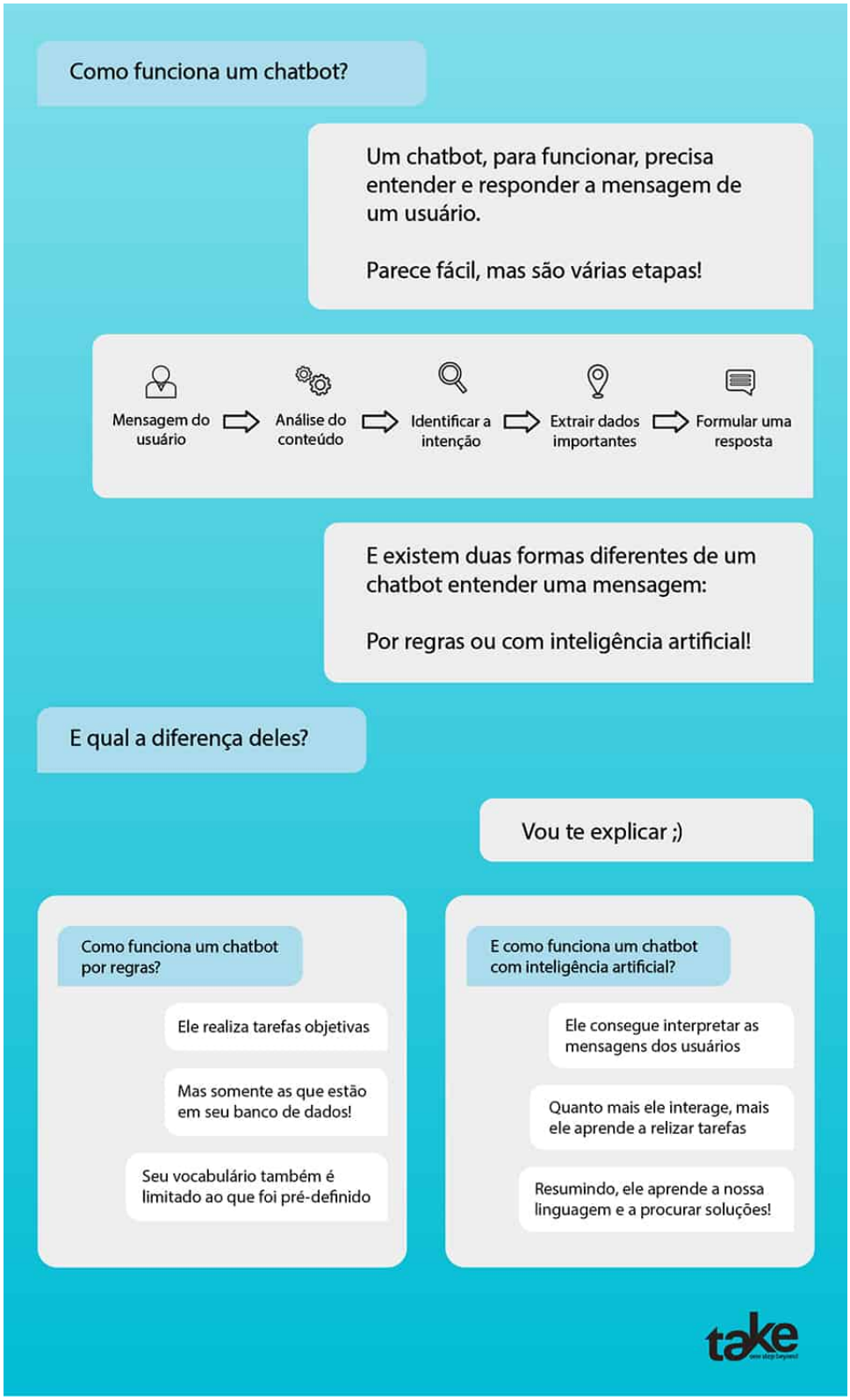

Figura 19 – Ilustração de um diálogo de um *chatbot* explicando as categorias de *chatbot*. Créditos: https://take.net/blog/chatbots/chatbot/

Uma das técnicas utilizadas neste tipo de *chatbot* é o **casamento de padrões**. Nesse tipo de técnica é especificado um dicionário com respostas padronizadas para algumas

palavras chave. O *chatbot* ELIZA desenvolvido por (WEIZENBAUM, 1966), que simula um psicoterapeuta Rogeriano, muitas vezes de forma convincente e inspirou muitas gerações de autores de *chatbots*, foi desenvolvido utilizando esta técnica.

Apesar do sucesso de alguns desses *chatbots* dessa geração, para serem efecientes necessitavam de grandes bases de dados para tentar cobrir o máximo de respostas possível (BARROS FLAVIA A.; TEDESCO, 2016).

4.3.2 Chatbots com IA

Uma falha dos *chatbots* iniciais era o não tratamento de fenômenos discursivos. A partir do uso de técnicas de IA e de PLN, foi possível o tratamento de referências pronominais e de frases repetidas. Este avanços permitiram maior realismo aos diálogos (BARROS FLAVIA A.; TEDESCO, 2016).

Chatbots baseados em Inteligência Artificial utilizam desde recursos de PLN, até técnicas de aprendizagem de máquina. Sistemas que utilizam essa abordagem poderão interpretar o que o usuário digita, porém para que ele seja eficiente é necessário ter uma base de treinamento.

Uma das etapas utilizadas nesse tipo de *chatbot* é o **pré-processamento**. Nessa etapa a sentença digitada pelo usuário é pré-processada, antes de ser enviada ao processador de linguagem natural. Estas tarefas podem ser as mais diversas como, reconhecimento de entidades, *lemmatização, tokenização*, etc (WILKS et al., 2011).

Segundo (NEVES; BARROS; HODGES, 2006), uma característica dessa geração de *chatbots* é a necessidade de tratamento do fluxo global da conversa, por exemplo, início, desenvolvimento e fechamento do diálogo. Outra característica desse grupo é oferecer tratamento adequado a perguntas ou intervenções em que o *chatbot* não possua resposta.

Uma outra técnica é a criação de linguagens autorais para *chatbots*, como o **Cleverscript** desenvolvido por Rollo Carpenter. Cleverscript[4] utiliza como conceito tabelas, compostas por palavras e frases que possam ser reconhecidas.

Outra linguagem autoral baseada em regras é o **Chatscript**[5]. Baseada em casamento de padrões, possui como diferencial um técnica chamada de conjunto de conceitos, que analisa a semântica da sentença dada como entrada pelo usuário (WILCOX; WILCOX, 2014).

Uma das mais conhecidas linguagens baseada em regras para *chatbots* é o **AIML, (Artificial Intelligence Markup Language)**, uma linguagem baseada em XML desenvolvida para criar diálogos semelhante a linguagem natural por meio de *softwares*, simulando assim inteligência humana (WALLACE, 2017).

AIML possui algumas classes de objetos definidos em uma linguagem de marcação que permitem adicionar comportamentos a *chatbots*. As *tags* mais importantes desta linguagem são (WALLACE, 2017):

[4] Detalhes sobre a linguagem Cleverscript podem ser acessadas em: http://www.cleverscript.com
[5] Detalhes sobre a linguagem Chatscript podem ser acessadas em : http://chatscript.sourceforge.net/

- <aiml> : descreve o início e o fim do documento AIML.
- <category> : esta *tag* define uma unidade de conhecimento na base de conhecimento.
- <pattern> : nesta *tag* é definida um padrão que corresponda à sentença digitada pelo usuário.
- <template> : esta *tag* contém a resposta que será emitida ao usuário.

Na Figura 20 podemos ver um exemplo de código AIML e o resultado que uma interação de um usuário com esse *Chatbot* pode gerar.

```
<?xml version = "1.0" encoding = "UTF-8"?>
<aiml version = "1.0.1" encoding = "UTF-8"?>

   <category>
      <pattern>I LIKE *</pattern>
      <template>
         I too like <star/>.
      </template>
   </category>

   <category>
      <pattern>A * IS A *</pattern>
      <template>
         How <star index = "1"/> can not be a <star index = "2"/>?
      </template>
   </category>

</aiml>

Resultado:

Human: I like mango
Robot: I too like mango.
Human: A mango is a fruit
Robot: How mango can not be a fruit?
```

Figura 20 – Exemplo de código AIML e resultado gerado por este código.

Vários avanços na Linguagem AIML já foram propostos, entre eles podemos destacar a inserção de um mecanismo de tratamento de intenções em um diálogo chamado iAIML (NEVES; BARROS; HODGES, 2006).

4.3.3 Avaliação de *Chatbots*

Segundo (MASCHE; LE, 2017), na literatura é possível classificar a avaliação de *chatbots* em quatro classes:

- A primeira, e a mais aplicada, é a avaliação qualitativa utilizada através de entrevistas ou questionários.
- A segunda, também largamente utilizada, é a avaliação quantitativa. A avaliação quantitativa é realizada por meio de protocolos de diálogo gerados por conversas entre usuários e o sistema.

- A terceira classe de avaliação é através de pré e pós testes. Esta avaliação é comumente utilizada em avaliações de agentes pedagógicos, para medir a efetividade da aprendizagem.

- A quarta classe de avaliação é através de competições, como por exemplo o Loebner Prize. Muitos dos *chatbots* citados neste texto já ganharam esse prêmio em algum ano.

4.3.4 Plataformas comerciais de *Chatbots*

A tecnologia de *chatbots* tem se popularizado e as grandes empresas como Facebook Messenger, Skype, Telegram , Twitter , Slack, Discord, Amazon Alexa, Google Home/Google Assistant, diponibilizam *Applications Programming Interfaces* (APIs) para desenvolvedores criarem *chatbots* para os mais diversos segmentos (BRASIL, 2020).

As plataformas de PLN disponibilizadas por essas empresas são as maiores aliadas da comunidade para o desenvolvimento de *chatbots*. Essas empresas disponibilizam em suas plataformas ferramentas para extração de intenções e entidades. As mais conhecidas são *WIT.ai, DialogFlow (antigo api.ai), Rasa NLU (open source), IBM Watson*, etc (BRASIL, 2020).

Existem também as plataformas específicas para o desenvolvimento de *chatbots*. Também utilizam ferramentas de PLN, porém disponibilizam um conjunto de ferramentas agregadas que permite o desenvolvimento e a integração com os mais diversos canais. As plataformas mais conhecidas são *Chatfuel, ManyChat, BLiP(brasileira), FlowXO, Motion.ai,* etc (BRASIL, 2020).

Dentre as muitas utilidades de *chatbots*, podemos subdividi-los em categorias (BRASIL, 2020):

- ***Analytics***, com o objetivo de mensurar e gerar estatísticas sobre audiência;

- **Marketing**, voltados para analisar dados;

- **Monetização**, para realizar vendas;

- **Prototipação**, para desenhar fluxo de conversação do *chatbot*.

Esta categorização de *chatbots* demonstra que a adoção do uso de *chatbots* acontece em diferentes domínios de conhecimento, o que favorece a formação de um ecossistema de colaboração sobre o tema. As informações dessa seção foram retiradas do site *https://medium.com/botsbrasil* (BRASIL, 2020), uma das maiores comunidades brasileiras de *chatbots*. A comunidade disponibiliza material, tendências e novidades sobre a tecnologia de *chatbots*. A comunidade disponibiliza também um *landscape* com os *chatbots* mais famosos do Brasil e suas categorias, que podem ser vistos nas figuras 21 e 22.

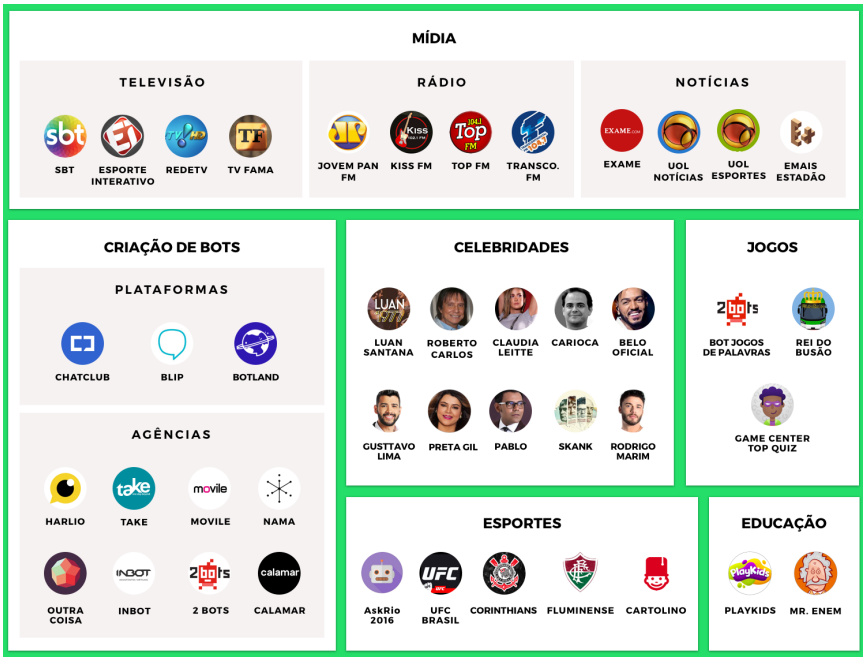

Figura 21 – *Chatbots* mais famosos do Brasil e suas categorias, disponível na comunidade *https://medium.com/botsbrasil.*(parte 1)

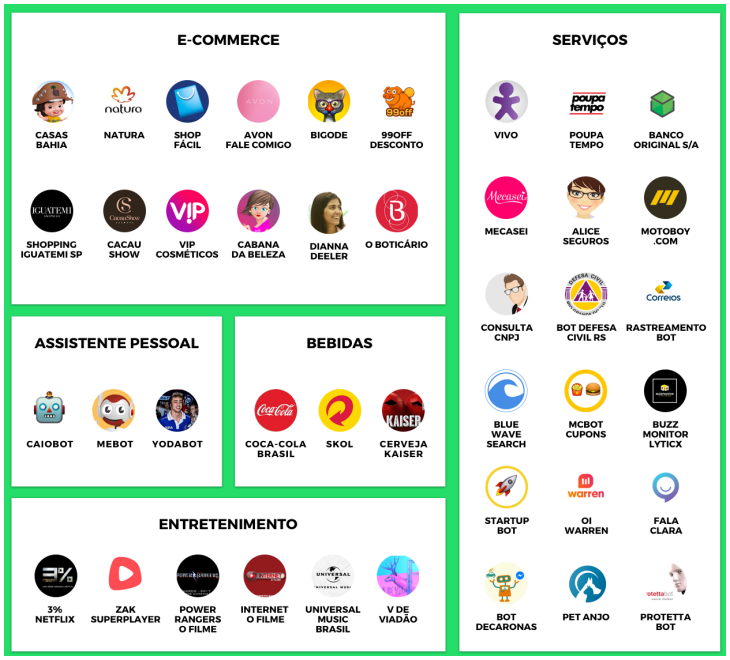

Figura 22 – *Chatbots* mais famosos do Brasil e suas categorias, disponível na comunidade *https://medium.com/botsbrasil.*(parte 2)

4.4 *RENAN* - UM SISTEMA DE DIÁLOGO INTELIGENTE BASEADO EM LÓGICA DE DESCRIÇÕES

Renan é um *Chatbot* que utiliza a linguagem AIML, uma linguagem baseada em regras, porém *Renan* aprende com as interações.

Através de uma abordagem baseada em Aprendizagem de Ontologias, *Renan* converte as sentenças em linguagem natural para uma ontologia e realiza inferências lógicas e detecta inconsistências (AZEVEDO et al., 2014a).

As atividades que *Renan* realiza são (AZEVEDO et al., 2014b):

(1) Traduz frases em linguagem natural para lógica de descrições e aprende a partir de interações em linguagem natural controlada com usuários.

(2) Raciocínio Automático, a abordagem permite que novos fatos sejam deduzidos a partir de outros, através de raciocínio de subsunção, checagem de inconsistências. *Renan*, provê uma solução adequada e efetiva na construção automática de Ontologias expressivas (expressividade máxima $\mathcal{ALC}$), a partir de raciocínio automático e interações em linguagem natural com seres humanos.

Renan é o principal trabalho relacionado à esta tese e algumas das propostas aqui desenvolvidas foram baseadas em sua arquitetura. *Renan* passou por modificações e melhorias, principalmente no seu sistema de conversação baseado em AIML feitas no trabalho de (LIMA, 2017).

As principais melhorias propostas por (LIMA, 2017) foram adaptação para uma ferramenta *online*, tratamento de plurais e melhorias no diálogo. O trabalho de (AZEVEDO et al., 2014a) ainda passou por melhorias propostas por (SILVA, 2019).

No trabalho de (SILVA, 2019), foram propostas melhorias na expressividade e na abordagem utilizada para conversão de linguagem natural para axiomas lógicos. Na tabela 9 é apresentado um comparativo entre os trabalhos de (AZEVEDO et al., 2014a) e (SILVA, 2019).

Tabela 9 – Tabela com o comparativo entre os trabalhos de (AZEVEDO et al., 2014a) e (SILVA, 2019)

	Renan (AZEVEDO et al., 2014a)	Pupo (SILVA, 2019)
Manipulação de ontologias	Criação e Edição	Criação e Edição
Método	Semiautomático	Semiautomático
Abordagem	Linguística	Linguística, Aprendizado de Máquina, SUMO, WordNet
Raciocínio Automático	SIM	SIM
Identificação e Tratamento de Inconsistências	SIM	SIM
Ferramenta de PLN	Stanford Parser	CoreNLP (Stanford Parser)
Linguagem Gerada	OWL	OWL
Expressividade DL	$\mathcal{ALC}$	$\mathcal{ALCHOQ}$
Taxa de sucesso	0.75	0.70

Os trabalhos citados nessa seção foram desenvolvidos pelo grupo de pesquisa ao qual pertence este trabalho. Arandu difere dos trabalhos citados por reutilizar conhecimento de uma ontologia de topo, e não ser somente uma ferramenta que converte sentenças em linguagem natural para axiomas lógicos. Arandu guia o usuário no processo de construção da ontologia e contribui com a quantificação de axiomas.

Arandu, além de uma ferramenta que converte um diálogo para uma ontologia é uma metodologia para construção de uma ontologia apoiada por software. Na próxima seção trataremos dos trabalhos relacionados que contribuíram tecnicamente para a construção do *chatbot* e dos conjuntos teóricos que embasaram a metodologia proposta.

4.5 TRABALHOS RELACIONADOS

Para elencarmos os trabalhos relacionados, os dividiremos em dois grupos que chamaremos de estado da técnica e quadro teórico.

Estado da técnica será formado por trabalhos encontrados na literatura relacionados ao desenvolvimento do artefato (Chatbot). Será composto por técnicas, princípios de design e artefatos relacionados que se propõem a converter linguagem natural para axiomas lógicos.

Quadro teórico será formado por trabalhos encontrados na literatura relacionados a metodologias para construção de ontologias.

4.5.1 Estado da técnica

Métodos caracterizados como Aprendizado de Ontologias, que criam ontologias de forma automática ou semi-automática, possuem bastante relação com este trabalho. No trabalho de (VÖLKER, 2009), os autores propõem uma ferramenta chamada LExO, que através de regras gramaticais utilizando um processador de linguagem natural chamado MINIPAR, convertem texto para uma linguagem formal de ontologias.

No trabalho de (SABOU et al., 2005), os autores utilizam uma estratégia semelhante a (VÖLKER, 2009) com o uso de Web Services, porém para construção de ontologias de domínio e realizam um experimento para o domínio de bioinformática. Os autores de (SABOU et al., 2005) concluem o trabalho citando que a participação de um especialista de domínio é fundamental e a ferramenta provê um ótimo ponto de partida para a construção de uma ontologia.

Aprendizado de Ontologia tem sido utilizado, segundo a literatura, como ferramentas iniciais para a construção de uma ontologia, como por exemplo (MAEDCHE; STAAB, 2004) e (NAVIGLI; VELARDI, 2004). Algumas dessas ferramentas podem ser integradas a outros softwares, como o (BUITELAAR; OLEJNIK; SINTEK, 2004) que pode ser integrado ao software Protégé (MUSEN, 2015), um dos softwares mais conhecido e utilizado para construir ontologias.

Contudo, esse trabalho propõe o uso de uma interface de conversação para construção de uma ontologia, convertendo texto para DL, com interação com um usuário. Arandu tem como principal trabalho relacionado (AZEVEDO et al., 2014a)(descrito com maiores detalhes na seção 4.4), porém diferente por conseguir reutilizar conhecimento de outra ontologia e se apresentar também como uma proposta metodológica.

4.5.2 Quadro teórico

Na literatura é possível encontrar várias metodologias para construção de ontologias, muitas inspiradas na engenharia de software com a participação de seres humanos e muitas vezes apoiadas por software. Algumas metodologias bastante conhecidas são METHONTOLOGY (FERNANDEZ-LOPEZ; GOMEZ-PEREZ; JURISTO, 1997), OTK (AKKERMANS et al., 2002), Methodology 101 (NOY; MCGUINNESS, 2001), RapidOWL (AUER; HERRE, 2007), etc.

Na seção 3.2 algumas destas metodologias foram discutidas com detalhes. Dentre elas a que possui maior relação com este trabalho é a RapidOWL (AUER; HERRE, 2007), por se tratar de uma metodologia com uma proposta ágil com foco no engenheiro de ontologias.

Arandu possui uma proposta de metodologia apoiada por software bem definida e facilmente verificável durante o processo de construção. A incorporação do reuso e quantificação de relações durante a conversão dos axiomas é uma das principais diferenças para as outras metodologias, fases que muitas vezes podem não ocorrer por ser uma tarefa opcional ao engenheiro.

4.6 *DESIGN SCIENCE RESEARCH* - DSR

Design Science Research é um paradigma de ciência que vem se popularizando na área de Sistemas de Informação (PIMENTEL; FILIPPO; SANTORO, 2019). Foi proposto por Herbert Simon, autor do livro "As Ciências do Artificial"*(The Sciences of the Artificial)* (SIMON, 1996).

DSR tem sido utilizado em diversos campos, é um paradigma de pesquisa muito utilizado em disciplinas aplicadas principalmente em arquitetura, engenharia, educação, psicologia e artes plásticas. No campo da ciência da computação esse paradigma tem sido a cada dia mais aplicado, em sistemas de informação compostos por hardware, softwares e interfaces mutáveis e adaptáveis, DSR tem contribuído para validar pesquisas que exigem ideias novas e criativas (HEVNER; CHATTERJEE, 2010).

A abordagem DSR provê rigor científico e relevância adequada para o desenvolvimento de artefatos tecnológicos (SIMON, 1996).

Em DSR o foco científico é voltado para o artefato. Artefatos não se restringem a objetos físicos, artefatos são projetados, por isso, podem ser abstrações ou qualquer coisa projetada para alcançar um objetivo (PEFFERS et al., 2007).

Na tabela 10 descrevemos alguns tipos de artefatos e suas descrições consideradas por (PIMENTEL; FILIPPO; SANTORO, 2019).

Tabela 10 – Tabela com os tipos de artefatos e suas descrições considerados por (PIMENTEL; FILIPPO; SANTORO, 2019)

Tipo de artefato	Descrição
Constructo	Vocabulário conceitual de um domínio
Modelo	Proposições que expressam relacionamentos entre os constructos
Framework	Guia, conceitual ou real, que serve como suporte ou guia
Arquitetura	Sistemas de estrutura de alto nível
Princípio de projeto	Princípios chave e conceitos para guiar o projeto
Método	Passos para executar tarefas
Instanciação	Implementações em ambientes que operacionalizam constructos, modelos, métodos e outros artefatos abstratos
Teorias de projeto	Conjunto prescritivo de instruções sobre como fazer algo para alcançar determinado objetivo. Uma teoria geralmente inclui outros artefatos abstratos, tais como constructos, modelos, frameworks, arquiteturas, princípios de design e métodos.

DSR é um paradigma metodológico epistemológico de pesquisa que tem como principal objetivo resolver um problema prático num contexto específico por meio de um artefato e gerar novo conhecimento científico (PIMENTEL; FILIPPO; SANTORO, 2019).

É possível identificar dois ciclos em pesquisas DSR, e o ciclo de engenharia e ciclo empírico. No ciclo de engenharia, o artefato é projetado para solucionar um problema real em determinado contexto. No ciclo empírico, são elaboradas as conjecturas teóricas relacionadas ao comportamento humano ou organizacional (PIMENTEL; FILIPPO; SANTORO, 2019).

O ciclo de engenharia pode ser dividido, algumas fases observadas são: investigação do problema, design da solução, validação da solução, implementação da solução, avaliação da solução. Na figura 23 temos uma representação destas fases, proposta por (PIMENTEL; FILIPPO; SANTORO, 2019) adaptada de (WIERINGA, 2014).

O ciclo empírico ou ciclo de conhecimento também pode ser dividido em algumas fases, que são: contexto da pesquisa, análise do problema de pesquisa, pesquisa e inferência do design, validação, execução da pesquisa, análise de dados, contribuição da pesquisa. Na figura 24 temos uma representação dessas fases, proposta por (PIMENTEL; FILIPPO; SANTORO, 2019) adaptada de (WIERINGA, 2014).

Em uma pesquisa do tipo DSR, normalmente realizam-se três avaliações. A primeira verifica se o artefato satisfaz aos requisitos. A segunda, se as conjecturas teóricas são válidas, e a terceira avaliação se o artefato é satisfatório (DRESCH ALINE; LACERDA, 2015).

Figura 23 – Fases que compõem o ciclo de engenharia em uma pesquisa DSR. (proposta por (PIMENTEL; FILIPPO; SANTORO, 2019) adaptada de (WIERINGA, 2014))

Figura 24 – Fases que compõem o ciclo empírico em uma pesquisa DSR. (proposta por (PIMENTEL; FILIPPO; SANTORO, 2019) adaptada de (WIERINGA, 2014))

Os autores (PIMENTEL; FILIPPO; SANTORO, 2019) propõem um mapa de elementos que podem guiar o pesquisador a reconhecer e planejar o que se espera em uma pesquisa na abordagem DSR. O mapa não é um método de pesquisa, porém serve como instrumento de apoio ao pesquisador. Na figura 25 temos uma representação do mapa de elementos esperados em uma pesquisa DSR.

No próximo capítulo, apresentaremos o artefato proposto.

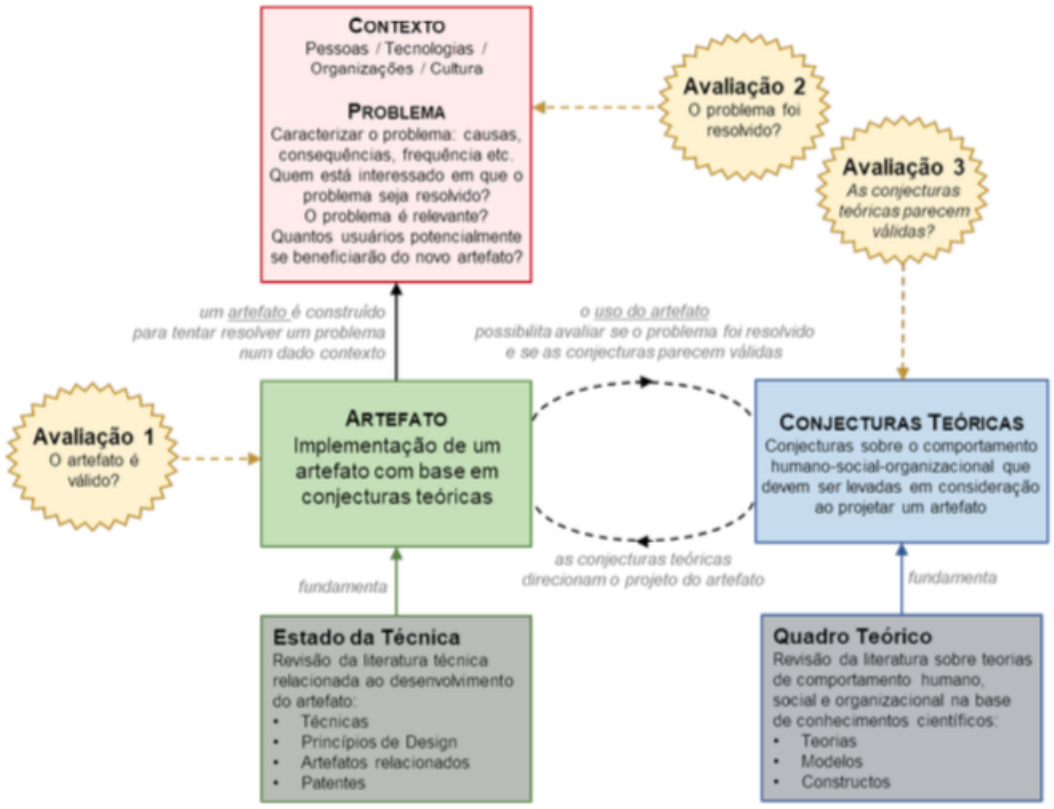

Figura 25 – Mapa de elementos esperados em uma pesquisa DSR, proposta por (PIMENTEL; FILIPPO; SANTORO, 2019)

5 ARANDU CHATBOT

Nesta tese, desenvolvemos um *chatbot* que é capaz de converter um diálogo em linguagem natural para uma ontologia. Baseado no *chatbot Renan*, Arandu consegue alcançar a mesma expressividade lógica, mas é diferente por ter sido implementado para a *web* e por reutilizar uma ontologia de topo.

Arandu possui a capacidade de:

- Converter linguagem natural para axiomas lógicos em DL;
- Reutilizar conhecimento de uma ontologia de topo;
- Quantificar os axiomas com restrições existencial ou universal;
- Raciocinar sobre novos fatos;
- Checar inconsistências lógicas;
- Guiar o usuário no processo de construção da ontologia.

Arandu pode também ser definido como uma metodologia apoiada por software para construção de uma ontologia. Por ser um *chatbot* com um objetivo bem definido, que é construir uma ontologia, Arandu solicita que o usuário informe por exemplo a URI e alguma definição do propósito da ontologia, uma das fases iniciais propostas na maioria das metodologias.

Como foi levantado nas seções anteriores, construir uma ontologia não é uma tarefa trivial. Mesmo que utilizando uma técnica automática, o papel do especialista de domínio é fundamental para validar a ontologia. A necessidade de conhecer uma linguagem formal para representar conhecimento pode ser então um empecilho para a criação de ontologias.

Além da dificuldade técnica de construir uma ontologia, construir uma boa ontologia também é fundamental. Boas ontologias fazem reuso de outras ontologias, isso permite correspondência entre ontologias, interoperabilidade entre sistemas que utilizam ontologias, clareza do conhecimento representado para possível compartilhamento.

Uma das maiores dificuldades de se reutilizar uma ontologia é conhecer a ontologia que se pretende reutilizar ou ainda conhecer uma ontologia para se reutilizar. Arandu através de um *web service*, facilita essa etapa no processo de construção de uma ontologia.

Vejamos o seguinte exemplo. Um especialista de domínio em um processo de construção de uma ontologia informa a classe "*Cell*". Ele possivelmente, dando continuidade no processo de construção, irá informar outras superclasses para esta classe ou poderia procurar por superclasses para a classe "*Cell*"em uma ontologia como a SUMO, exigindo que ele a conheça.

Utilizando o Arandu para construir uma ontologia, quando o especialista informar a classe "*Cell*", Arandu perguntará : *"The definition of Cell is : The fundamental structural and functional unit of living Organisms. ? YES/NO"*, para um especialista de domínio será fácil concordar ou não com esta definição.

Se o especialista de domínio concordar com a definição, Arandu adicionará na ontologia as superclasses *"Physical, BodyPart, OrganicObject"*, entre outras. Talvez o especialista possa não ter previsto a adição destas superclasses, isto acarretaria:

- Uma ontologia com poucas classes, incapaz de representar o modelo esperado;
- Uma ontologia com termos descritos de maneira diferente da área de conhecimento;
- Uma ontologia que não possui correspondência com outra ontologia e não faz reuso.

Porém, se o especialista de domínio utiliza Arandu, teria como vantagens:

- Uma representação mais robusta, com um modelo maior do que o esperado;
- Classes descritas de forma padronizada, possibilitando maior entendimento do domínio que foi representado;
- Correspondência com uma ontologia de topo.

Aliado a essas vantagens, o especialista se eximiu de entender a ontologia SUMO, bastando apenas conhecer o domínio que está sendo modelado.

Reutilizar uma ontologia de topo, por exemplo, permitiria ampliar o reuso da classe *Physical*, trazendo mais características que SUMO disponibiliza para ela, como disjunções ou outras superclasses.

O uso de um *chatbot* como o Arandu contribuiria muito com a modelagem do conhecimento. No exemplo utilizado, não seria possível realizar esse reuso de forma automática, sem a interação com o especialista de domínio, dado que a classe *Cell* pode ter um significado diferente em outro contexto, sendo até subclasse de *Physical*, mas não sendo subclasse de *BodyPart*. Na metodologia apoiada por software que propomos com o uso do Arandu, chamamos essa fase de desambiguação.

Nas seções seguintes, apresentaremos os módulos que compõem o Chatbot Arandu e a metodologia apoiada por software que guia o usuário no processo de construção de uma ontologia reutilizando uma ontologia de topo.

5.1 DESIGN DO ARTEFATO

Arandu, por meio de um diálogo em Linguagem Natural Controlada - LNC, guia o usuário no processo de construção de uma ontologia, recomendando classes de uma ontologia superior, fazendo inferências lógicas por meio de um raciocinador e verificando inconsistências lógicas.

Para apoiar o desenvolvimento do Chatbot Arandu elaboramos um mapa apresentado na figura 26, baseado no mapa de elementos esperados de uma pesquisa DSR proposta em (PIMENTEL; FILIPPO; SANTORO, 2019).

Este mapeamento contribuiu para direcionar o desenvolvimento do artefato tecnológico que guiará o usuário no processo de construção da ontologia, desenvolvido por meio de conjecturas teóricas fundamentadas por um quadro teórico estabelecido por uma análise do estado da arte.

Diante deste contexto, serão realizadas três avaliações. A primeira avalia se os requisitos do artefato foram atendidos, a segunda se este artefato contribui para a solução do problema observado nas conjecturas teóricas, e a terceira se os usuários validam o artefato como solução para as conjecturas teóricas.

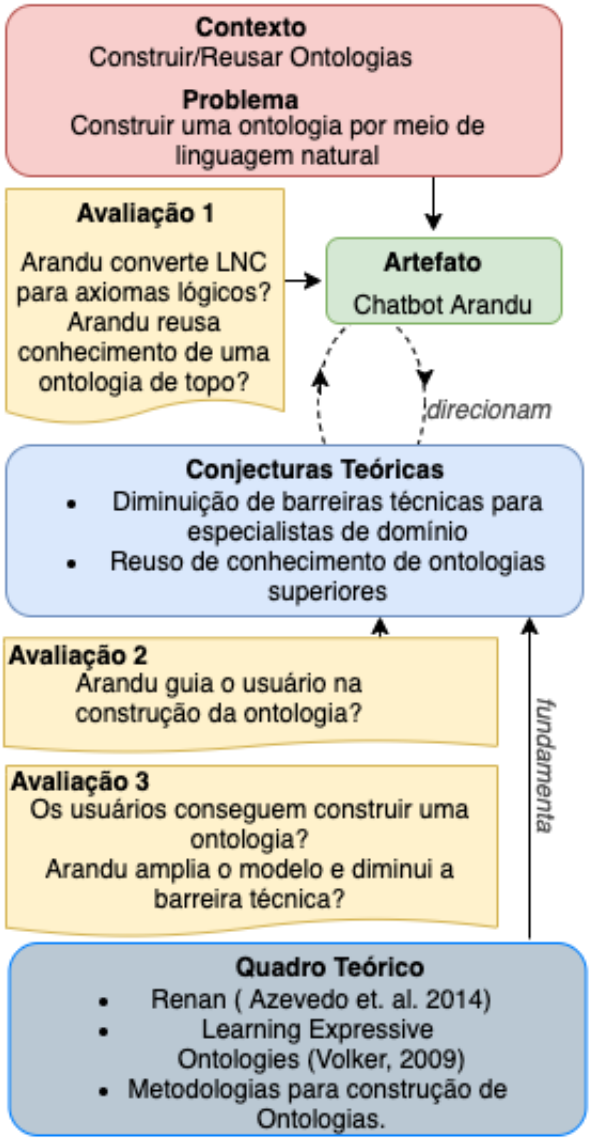

Figura 26 – Mapeamento dos elementos DSR do Chatbot Arandu, baseado em (PIMENTEL; FILIPPO; SANTORO, 2019)

.

5.2 PADRÃO DE PROJETO

Nesta seção, é apresentada a arquitetura utilizada para a construção do Chatbot Arandu. Esta arquitetura foi projetada com um acoplamento mínimo, que permite que módulos sejam substituídos ou adicionados.

O padrão de projeto escolhido foi baseado na arquitetura *RESTful Web API (Representational State Transfer (REST))*. Este padrão permite que sejam criados *web services* acessíveis a outros sistemas por meio de URIs (*Uniform Resource Identifier*), garantindo interoperabilidade entre sistemas de computadores na internet (RICHARDSON; AMUNDSEN; RUBY, 2013).

Os recursos em aplicações *RESTful* são acessados através de chamadas HTTP, que podem ser GET, HEAD, POST, PUT, PATCH, DELETE, CONNECT, OPTIONS e TRACE. Usando um protocolo sem estado e operações padrão, os sistemas *RESTful* buscam desempenho, confiabilidade e capacidade de crescimento rápido, reutilizando componentes que podem ser gerenciados e atualizados sem afetar o sistema como um todo, mesmo enquanto estiver em execução. Essas são ótimas características para um sistema que utilizará processamento de linguagem natural, conversão para linguagem OWL e deverá emitir respostas rápidas ao usuário que interage com o *chatbot*.

Arandu, por adotar este tipo de arquitetura, pode ser dividido em *back-end* e *front-end*. O *back-end* é responsável por fornecer os dados para a camada de *front-end*, nele são realizadas tarefas como PLN, conversão de axiomas, tratamento de resposta, armazenamento de dados, etc. Já o *front-end* é responsável pela interface com o usuário, que são os componentes gráficos, *buttons*, *dialogs*, etc. Na figura 27 temos uma representação visual desta divisão e das tecnologias que compõem cada subdivisão.

5.2.1 *Back-end*

Para o desenvolvimento do *Chatbot* Arandu, foram utilizadas algumas tecnologias que permitiram alcançar o seu propósito. Como pode ser visto na figura 27, temos uma subdivisão de módulos no *back-end*. A seguir temos uma descrição de cada módulo e as tecnologias que os compõem.

5.2.1.1 Módulo de Conversação

O módulo de conversação é responsável por gerar o diálogo. A partir de uma entrada do usuário o algoritmo analisará por meio de regras, qual a melhor resposta para aquele padrão de entrada. Este módulo está programado em linguagem de programação Java.

Como visto em seções anteriores um *chatbot* que possua estratégias de conversação adequadas pode promover uma melhor interação com o usuário. Arandu, inicialmente, possui um estratégia bem definida, que é a construção de uma ontologia. Por isso, não é possível um diálogo fora desse objetivo.

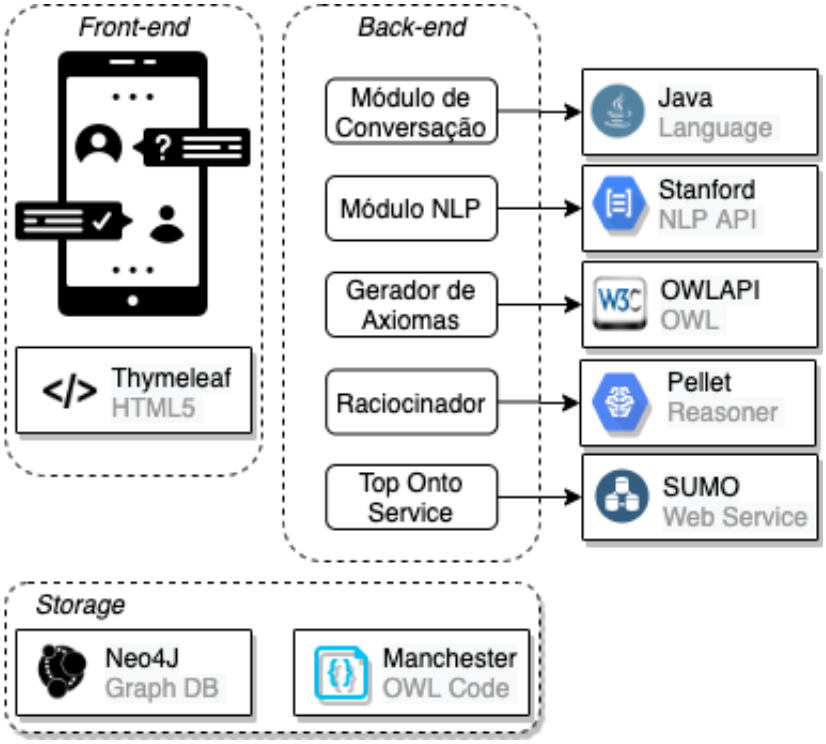

Figura 27 – Anatomia do Chatbot Arandu.

O módulo de conversação recebe a entrada e envia uma resposta que às vezes pode ser a entrada para outro módulo.

5.2.1.2 Módulo NLP

No módulo NLP, são realizadas a tarefas relacionadas a PLN para extração de classes, relações e axiomas das sentenças informadas pelo usuário durante a interação com o Chatbot Arandu. Para a extração dessas relações, são realizadas atividades de análise sintática e semântica das classes.

O processo de análise sintática é uma atividade de PLN que tem como objetivo analisar as palavras que compõem o discurso, seguindo regras gramaticais.

Para esta atividade o Arandu utiliza a API do *Stanford NLP Group* (KLEIN; MANNING, 2003).

Este módulo tem como principal função identificar os *tokens* da frase, identificar os sintagmas nominais que pertence cada *token* e o seu papel na árvore de dependência.

Após terem sido identificados os sintagmas a que pertencem os *tokens* que compõem o diálogo, bem como identificada a árvore de dependência que fornece a estrutura semântica do diálogo, são removidas as palavras conhecidas como *stop words*, palavras em conteúdo semântico, estas são palavras como por exemplo, *a, an, the, this, etc.*

A próxima atividade é a análise semântica das classes, que consiste na identificação dos *tokens* candidatos à classes na ontologia. Os sintagmas são identificados pelas tags **NN,**

NNP, NNS que representam os substantivos, **JJ** representa os adjetivos, **VB, VBZ, VBP, VBN, VPG** representam os verbos e a tag **NP** identifica o sujeito da frase.

Para que um *token* seja candidato a uma classe ele deve se encaixar em uma das seguintes regras:

- Se o sintagma a que pertence o *token* for um **NN, NNP ou NNS**.
- Ser composição de um sintagma **NP**

Dada uma sentença como exemplo: S1= *"The orange is a fruit of the citrus species."*, os *tokens "orange"* e *"fruit"* atendem a essas regras e então se tornam candidatos a classes na ontologia. Porém analisando o restante da sentença, identificamos os seguintes *tokens "citrus"* e *"species"*. Ambos atendem as regras definidas, porém pertencem ao mesmo sintagma **NP** e, analisando o contexto da sentença, é notável que o objetivo é que estes dois *tokens* sejam concatenados e formem um *token* único *"citrusSpecies"* e esse sim seja um candidato a classe na ontologia. Isso foi tratado com a adição da seguinte regra:

- Se um *token* com sintagma **NN, NNP ou NNS** for precedido de um *token* com sintagma **JJ, NN, NNP ou NNS**, pertencente ao mesmo sintagma **NP** eles devem ser concatenados.

Após realizada a etapa de identificação das classes que irão compor a ontologia, é necessário identificar as relações entre essas classes. As relações são os verbos, e ocorrem em sintagmas **VB, VBZ, VBP, VBN, VPG**.

Poderão ocorrer dois tipos de relações, relação de **subsunção** e a relação de **equivalência**.

A relação de subsunção é a relação em que uma classe se torna subclasse de outra classe. Em linguagem natural, essa relação é dada pelo verbo *to be*.

As sentenças, após serem processadas, são armazenadas no banco de dados em forma de grafo.

5.2.1.3 Gerador de Axiomas

O módulo Gerador de Axiomas é responsável por criar os axiomas na ontologia em linguagem OWL, bem como carregar a ontologia em memória, salvar as alterações e manipulá-las. Estas atividades são realizadas com suporte da OWLAPI[1], uma API *open source* para manipulação de ontologias (HORRIDGE; BECHHOFER, 2011).

Este módulo recebe como parâmetros um vetor de classes e relações que serão persistidas na ontologia.

[1] https://github.com/owlcs/owlapi

5.2.1.4 Raciocinador

Deduzir novos fatos em uma base de conhecimento é umas das mais importantes tarefas de IA. O módulo raciocinador é responsável por realizar inferências lógicas no código OWL que esta sendo gerado.

O raciocinador é um componente essencial para trabalhar com ontologias OWL. De fato, praticamente todas as consultas de uma ontologia OWL (e seu fechamento de importações) devem ser feitas usando um raciocinador. Isso ocorre porque o conhecimento em uma ontologia pode não ser explícito e é necessário um raciocínio para deduzir o conhecimento implícito, bem como inconsistências lógicas (HORRIDGE; BECHHOFER, 2011).

Para esta tarefa Arandu utiliza o raciocinador *Pellet*, um raciocinador implementado em Java integrado a OWL API. *Pellet* incorpora otimizações para nominais, resposta de consulta a conjunções e raciocínio incremental (SIRIN et al., 2007).

5.2.1.5 Top Onto Service

Top Onto Service é o módulo responsável por realizar consultas em uma ontologia de topo (sobre ontologias de topo, vide Seção 3.5). Essas consultas têm como objetivo extrair informações que já foram formalmente definidas na ontologia de topo e possam ser reutilizadas.

Ontologias de topo possuem muitos axiomas e muitas vezes exigem maior capacidade computacional para que seja possível realizar consultas a elas. Normalmente os desenvolvedores de ontologias de topo fornecem *softwares* que permitem que seja possível realizar consultas a essas ontologias sem a necessidade de realizar o carregamento desta ontologia localmente (como por exemplo o Sigma da ontologia Sumo (vide Seção 3.5.1)).

O módulo *Top Onto Service* é um *Web Service* que realiza consultas em *softwares* para manipulação de ontologias. Essa modelagem permite que seja possível configurar qual ontologia de topo se deseja utilizar e exige menor custo computacional por parte do Arandu, visto que carregar uma ontologia de topo para manipulá-la é uma tarefa computacionalmente custosa.

Como visto nas seções anteriores, após uma sentença ter sido processada, será invocado então o módulo *Top Onto Service* e esse módulo realizará uma requisição à ontologia de topo com os termos que foram criados como "classe"na ontologia local. Caso esse termo exista na ontologia de topo, ela retornará informações sobre o termo, como exemplo subclasses, superclasses, classes disjuntas ou equivalentes em código OWL, que serão adicionadas à ontologia local.

Por exemplo, se durante o processo de construção de uma ontologia surgir a classe *Plant*, o módulo *Top Onto Service* fará uma requisição à plataforma *Sigma*, que retornará um código OWL. Esse código trará uma definição criada por SUMO sobre a classe *Plant* e informações de superclasses que poderão ser reutilizadas. Na figura 28 podemos ver uma

parte do código OWL disponibilizado pelo *Sigma* para a classe *Plant*.

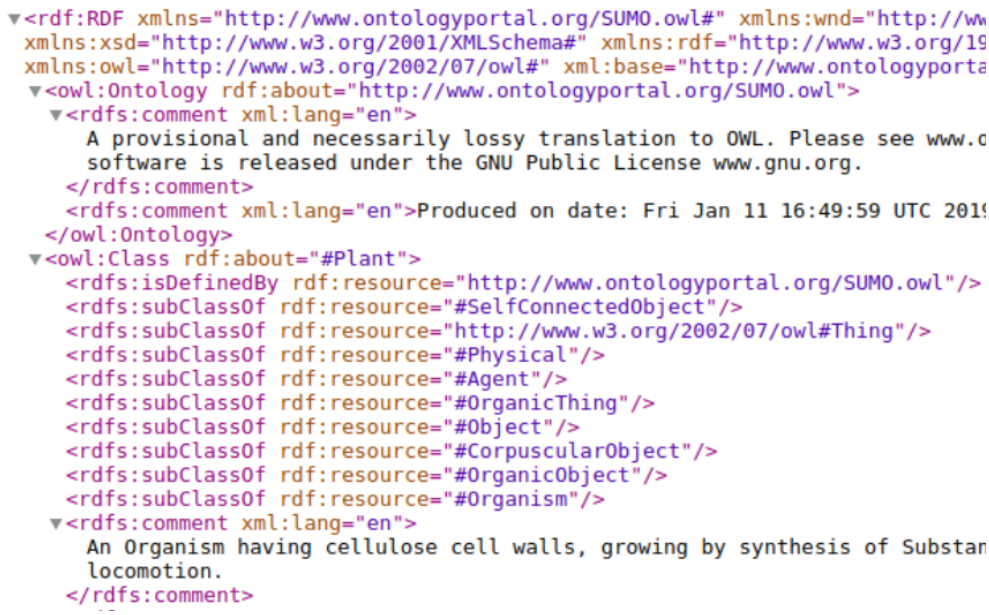

Figura 28 – Parte do código OWL para a classe *Plant* disponibilizada pelo *Sigma*.

5.2.2 *Storage*

Para o armazenamento dos dados produzidos, foi escolhido o banco de dados Neo4J[2], um banco de dados *open source* orientado a grafos. Bancos de dados (BDs) orientados a grafos são estruturas em grafos compostos por nós, arestas e propriedades que representarão os dados (ROBINSON; WEBBER; EIFREM, 2013), bastante adequado para armazenar uma ontologia, que é uma representação em forma de grafo.

BDs orientados a grafos pertencem à classe dos BDs NoSQL, criados para suprir limitações do bancos de dados relacionais. Diferentes de outros BDs seu foco está nos relacionamentos. Isto permite a recuperação rápida de estruturas hierárquicas complexas, difíceis de modelar em outras estratégias de armazenamento. Por exemplo, não é necessário fazer junção de tabelas para realizar consultas que envolvem relacionamento, como em um BD relacional, uma tarefa custosa computacionalmente para BDs (ROBINSON; WEBBER; EIFREM, 2013).

As consultas no BD Neo4j são realizadas através da linguagem *Cypher*, uma linguagem declarativa baseada nos conceitos e necessidades de um BD que usa os conceitos da teoria do grafos (FRANCIS et al., 2018). Uma consulta em um grafo no Neo4j pode ser feita a partir de um nó ou de um relacionamento. É possível adicionar índices aos nós para performar consultas, personalizar uma consulta com o grau de distância de um nó a outro, ou utilizar algoritmos clássicos da teoria dos grafos, por exemplo o de busca de menor caminho em um grafo.

[2] https://neo4j.com/

Por se tratar de uma estrutura flexível, BDs orientados a grafos permitem que sejam adicionados novos tipos de nós sem comprometer a estrutura funcional. O metamodelo de grafo usado por Arandu é composto por nove tipos de nós que possuem suas inter-relações. Na figura 29 é apresentada uma representação desse metamodelo e a seguir as definições de cada tipo de nó.

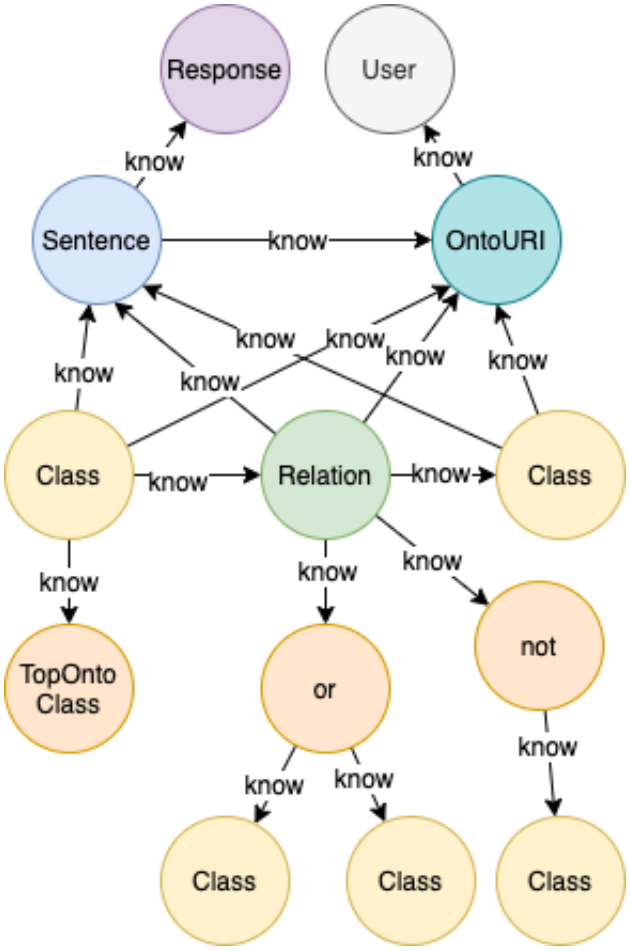

Figura 29 – Metamodelo de dados do Chatbot Arandu.

- **User** - Guarda as informações do usuário, por exemplo nome e email.
- **OntoUri** - Guarda a URI da ontologia que está sendo criada e sua descrição. Se relaciona com **User**.
- **Sentence** - Guarda as sentenças informadas pelo usuário. Se relaciona com **Response**.
- **Response** - São nós com respostas predefinidas para o usuário.
- **Class** - São os nós que foram gerados após o processamento do módulo NLP (vide seção 5.2.1.2). Se relaciona com o nó **Relation**, com o nó **Sentence** (que foi a

sentença que o gerou), **TopOntoClass**, e com o nó **OntoURI** (que é a ontologia a que pertence o nó).

- **Relation** - Representa o tipo de relação entre duas classes, a classe que incide nele e a classe em que ele incide formarão um axioma na ontologia. Se relaciona com os nós **Sentence**, **OntoURI**, **Class**, **not** e **or**.
- **not** - Nó gerado a partir do módulo NLP, serve para negar a relação em um axioma. Se relaciona com nó do tipo **Class**.
- **or** - Nó gerado a partir do módulo NLP, serve para criar um axioma de intersecção.
- **TopOntoClass** - Nó criado a partir das informações obtidas do módulo Top Onto Service (vide seção 5.2.1.5).

Através de consultas *Cypher* é possível personalizar consultas neste metamodelo e gerar uma ontologia completa ou parte dela. Esta característica abre espaço para trabalhos futuros na geração de módulos personalizáveis da ontologia criada.

O armazenamento de ontologias é um campo bastante ativo na Web Semântica. Os mais conhecidos sistemas de armazenamento de ontologias, chamados de *triple stores*, são o Jena[3], o RDF4J[4] (antigo Sesame) e o Virtuoso[5].

Arandu está modularizado e contém seu próprio módulo de raciocínio e geração de código OWL, além de ser um *chatbot* que salva informações dos usuários e *logs* de atividade. Essas características de *software* justificam a escolha desse BD. Neo4j possui as propriedades ACID (atomicidade, consistência, integridade e durabilidade), um conjunto de propriedades de bancos de dados transacionais que garantem validade dos dados em ocorrência de erros. Assim, Neo4j é uma boa opção para armazenamento de dados adequado ao *back-end* proposto. Na próxima seção, trataremos do *front-end*.

5.2.3 *Front-end*

Os canais de interação entre o usuário e o Chatbot Arandu podem ser os mais diversos, desde que permitam integração *Web*. Exemplo, páginas *Web, mobile apps*, etc.

Devido ao avanço dos dispositivos eletrônicos, é possível que muitos aparelhos que sejam utilizados para interagir com o Chatbot possuam nativamente recursos de interação por voz, vídeo ou teclados específicos para portadores de alguma necessidade especial, que convertam a entrada do usuário para texto. Estes recursos ampliam a experiência do usuário com o Arandu.

[3] https://jena.apache.org/
[4] http://www.rdf4j.org/
[5] https://virtuoso.openlinksw.com/

Arandu utiliza a tecnologia *Thymeleaf*[6], um *template* Java *server-side*. Arandu utiliza também as tecnologias Java Script, HTML5, CSS, e Bootstrap[7], com um *front-end* baseado em componentes Arandu promove uma experiência dinâmica com troca de componentes e apresentação de caixas de diálogo que respondem à interação com o *chatbot*.

As tecnologias até aqui apresentadas permitiram o desenvolvimento de um artefato tecnológico. Esse artefato é fundamental para a proposição da metodologia apoiada por software desenvolvida nesta tese. A seguir é apresentada a metodologia Arandu para desenvolvimento de uma ontologia com reuso.

5.3 METODOLOGIA ARANDU

Arandu, por meio do módulo de conversação, propõe uma metodologia interativa apoiada por software. Arandu combina elementos de outras metodologias já existentes com uma proposta ágil e com foco na facilidade do usuário estruturar conhecimento.

A metodologia Arandu se enquadra no manifesto ágil pelo fato de se basear em valores, princípios e práticas.

Valores estão implicitamente contidos na proposta de construção através de linguagem natural, que garante simplicidade. Com foco no usuário, o Chatbot interage com o usuário e o encoraja a informar definições, exigindo o mínimo de conhecimento de uma linguagem formal para construção de ontologias.

Príncipios estão no seu processo axiomático de construção. O processo é incremental e convergente. Através da indução de definição de classes já extraídas a construção é observável e coerente, com menor risco de falta de definições que compõem um axioma inicial. O feedback é rápido, tanto na avaliação quanto na inferência de novos axiomas por meio do raciocinador.

Práticas podem ser notadas na possibilidade de *short releases* da ontologia, que evolui a cada interação garantindo consistência desde a primeira iteração. O reuso de termos de uma ontologia de topo com suporte de anotações permite que não seja necessário usuários *experts* para conseguir reutilizar conhecimento de outra ontologia.

Uma representação visual desta metodologia pode ser vista na figura 30, e nas próximas seções serão detalhadas as fases que a compõe.

Esta metodologia foi construída para ser utilizada por um usuário humano e um Chatbot através de um diálogo em linguagem natural controlada, essa característica permite que algumas fases possam ser quebradas durante o díalogo. Porém, o módulo de conversação descrito anteriormente possui estratégias para tratar qualquer quebra de diálogo e manter o usuário em um fluxo.

Algumas fases podem não ocorrer, por exemplo, caso não haja conhecimento para ser reutilizado da ontologia de topo.

[6] https://www.thymeleaf.org
[7] https://getbootstrap.com/

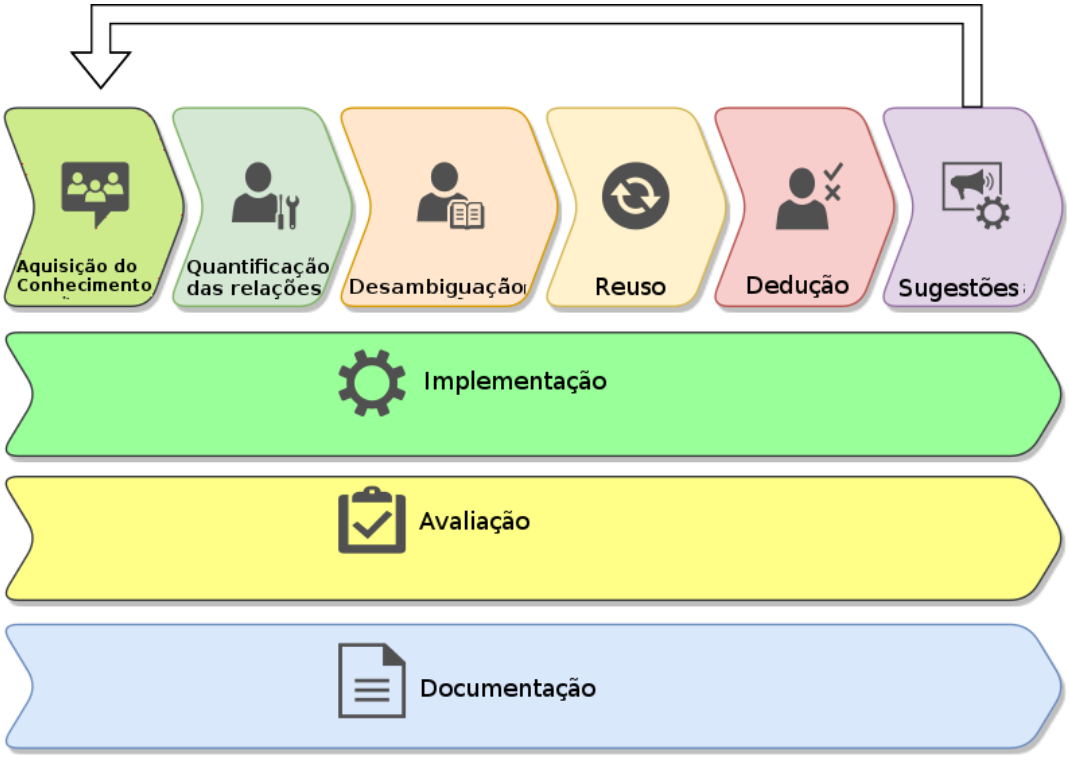

Figura 30 – Metodologia de construção de Ontologias Arandu.

5.4 CONSTRUÇÃO DA ONTOLOGIA

O Arandu Chatbot possui um módulo de conversação configurado para guiar o usuário no processo de construção de uma ontologia, não sendo possível nessa versão ter um diálogo que não tenha esse objetivo.

Inicialmente, o Arandu dá boas vindas e solicita o e-mail do usuário para identificá-lo. Na figura 31 é possível ver um exemplo desta interação.

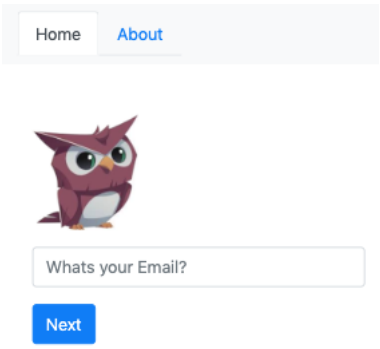

Figura 31 – Exemplo de interação para identificação do usuário.

Em seguida, o próximo passo para a construção da ontologia é definir a URI (*Uniform Resource Identifier*). A URI é uma sequência de caracteres utilizada para identificar um recurso, que pode ser um local na web ou um nome, ou ambos.

Arandu, cria uma URI baseado em sua URL, no nome da ontologia e no nome do usuário. Arandu solicita também um texto sobre a ontologia que será adicionado como um comentário na documentação da ontologia.

Um exemplo de diálogo para esse passo pode ser visto na figura 32.

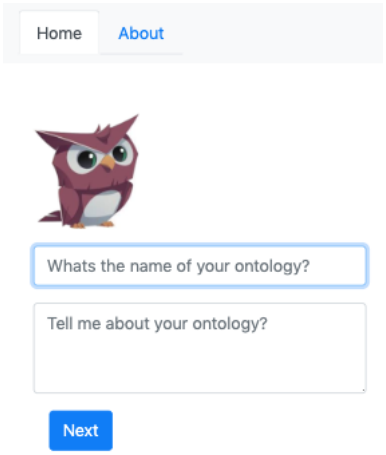

Figura 32 – Exemplo de interação para aquisição do nome da ontologia e uma descrição do seu propósito.

O processo de **aquisição do conhecimento** é realizado através da extração de classes e relações das sentenças informadas pelo usuário em linguagem natural, que serão convertidas para axiomas em DL.

Considere a sentença: S1 = *Cerrado is an ecosystem and terrestrial.*

A partir de S1 podemos extrair o seguintes termos como candidatos a classes:

$Classes = [Cerrado, Ecosystem, Terrestrial]$

S1 pode gerar o seguinte axioma lógico:

$$Cerrado\ SubclassOf: Ecosystem\ and\ Terrestrial$$

Na figura 33 temos um exemplo de interação com o Arandu informando S1.

Após a criação deste axioma, o Arandu irá interagir com o usuário trazendo definições que tenha encontrado na ontologia de topo. As definições adquiridas de SUMO não passam por nenhum tratamento e são mostradas da forma que foram definidas em SUMO. O usuário fará então a **desambiguação** do termo clicando em "sim"ou "não", se o termo possuir relação com o propósito da ontologia.

Para a classe *Ecosystem*, a ontologia de topo SUMO fornece o código OWL que pode ser visto na figura 34. Ao clicar em sim o Arandu fará **reuso** do conhecimento disponível em SUMO e adicionará como superclasses de Ecosystem as classes *Physical, Region, Object, GeographicArea.*

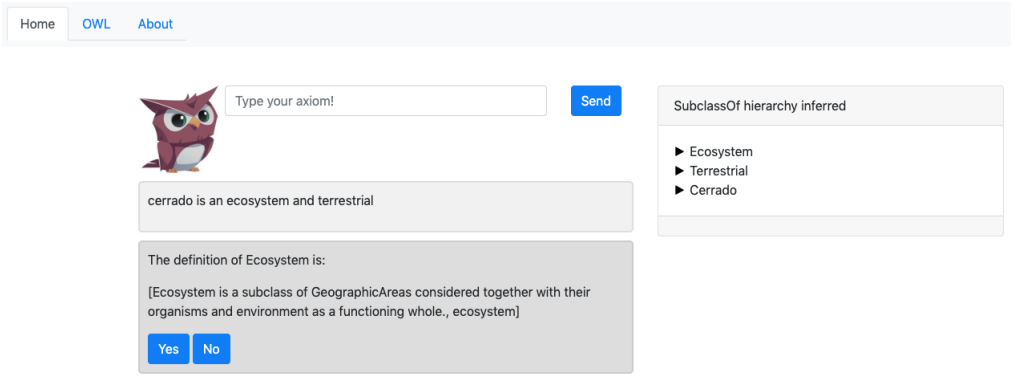

Figura 33 – Exemplo de interação informando a sentença: S1 = *Cerrado is an ecosystem and terrestrial* para ser processada.

```
▼<owl:Class rdf:about="#Ecosystem">
   <rdfs:isDefinedBy rdf:resource="http://www.ontologyportal.org/SUMO.owl"/>
   <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
   <rdfs:subClassOf rdf:resource="#Physical"/>
   <rdfs:subClassOf rdf:resource="#Region"/>
   <rdfs:subClassOf rdf:resource="#Object"/>
   <rdfs:subClassOf rdf:resource="#GeographicArea"/>
  ▼<rdfs:comment xml:lang="en">
     Ecosystem is a subclass of GeographicAreas considered together with their organisms and environment as a functioning whole.
   </rdfs:comment>
  ▼<externalImage rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">
     http://upload.wikimedia.org/wikipedia/commons/8/86/Daintree_Rainforest.JPG
   </externalImage>
  ▼<externalImage rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">
     http://upload.wikimedia.org/wikipedia/commons/b/b3/Male_lion_on_savanna.jpg
   </externalImage>
```

Figura 34 – Parte do código OWL fornecido por SUMO para a classe *Ecosystem*.

É possível analisar na figura 33 que, após o processamento da sentença S1, o Arandu informa a hierarquia de subclasses que foi formada.

Arandu também constrói axiomas com relações do tipo *Object Property*. Dada a sentença S2 = *Cerrado has plants and animals.*, temos os seguintes candidatos a classes:

$Classes = [Cerrado, Plant, Animal]$

S2 pode gerar o seguinte axioma lógico:

$$Cerrado\ SubclassOf:$$
$$(has\ some\ Plant)$$
$$and\ (has\ some\ Animal)$$

Na figura 35 temos um exemplo de extração de uma relação do tipo *Object Property* com a sentença S2.

Na interação da sentença S2, Arandu traz definições das classes *Plant* e *Animal*. Caso o usuário confirme essas definições, Arandu fará **reuso** do conhecimento disponível na ontologia de topo.

Quando forem encontradas relações no axioma que possam ser quantificadas, o Arandu Chatbot questionará o usuário sobre qual o quantificador que ele deseja inserir. Para a sen-

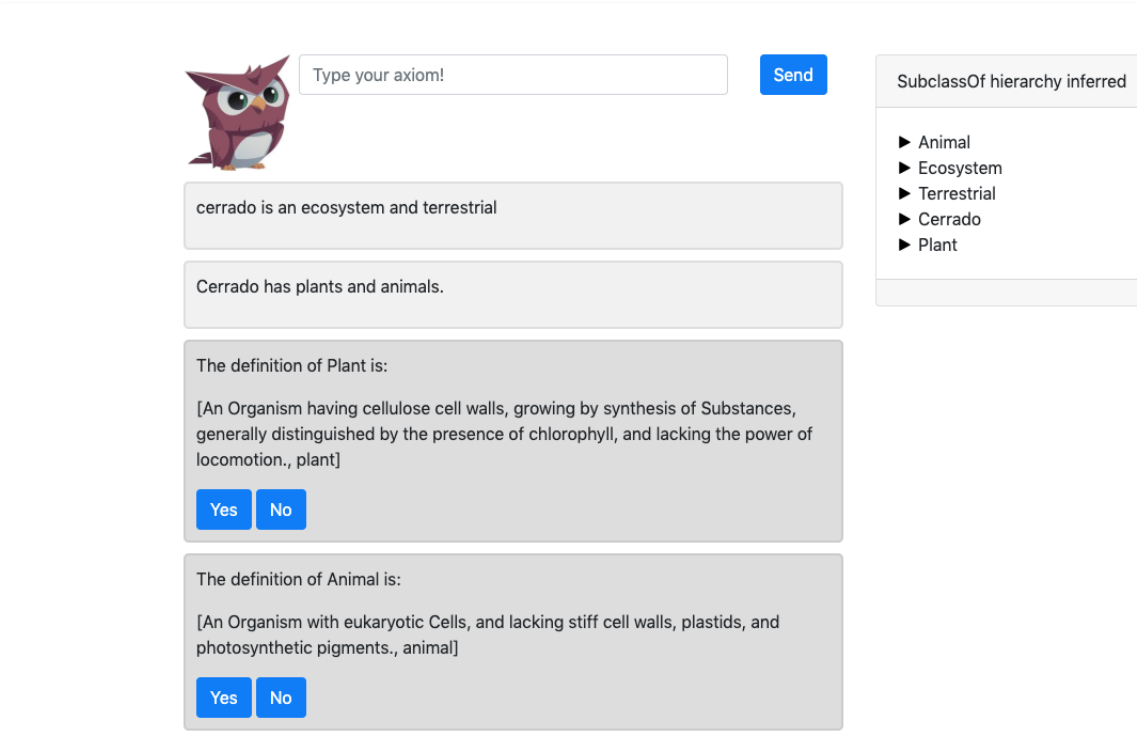

Figura 35 – Exemplo de interação onde é possível extrair uma relação do tipo *Object Property(have)*

tença S2, o usuário poderá optar pelo quantificador existencial *some* ou pelo quantificador universal *only*. Um exemplo desta interação pode ser visto na figura 36.

Ao escolher um quantificador, Arandu, questionará o usuário se ele deseja salvar esse axioma como uma condição **Necessária e suficiente**. Na figura 37 temos um exemplo desta interação.

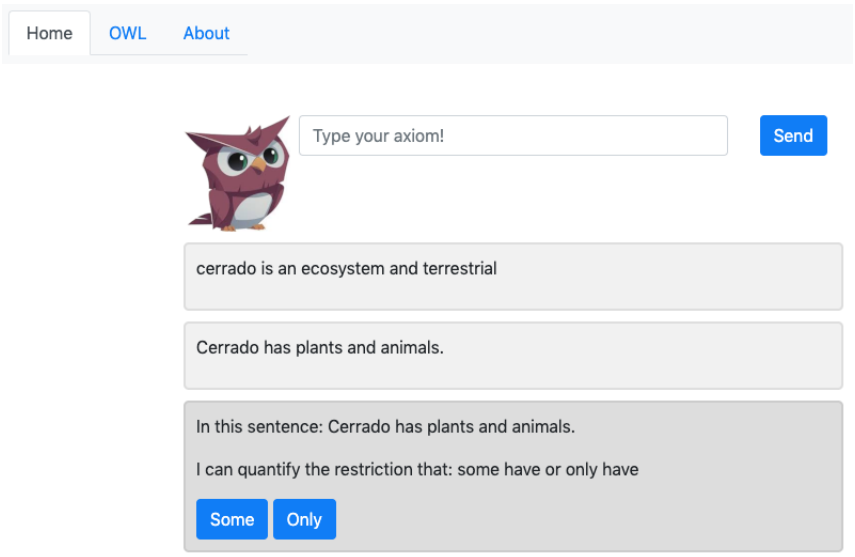

Figura 36 – Exemplo de interação para quantificação de uma relação.

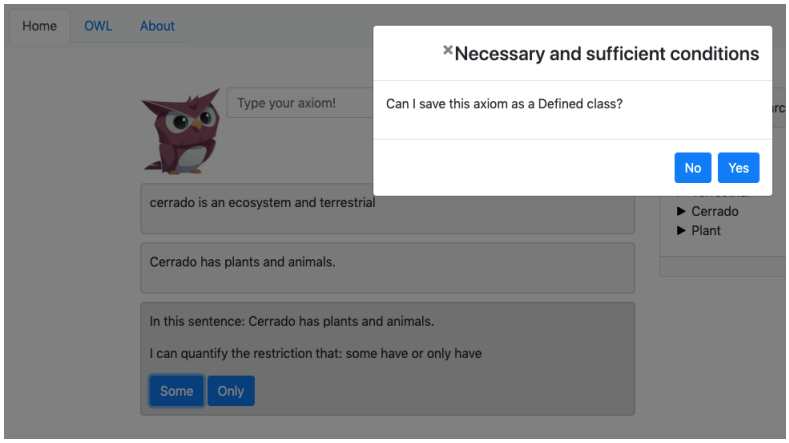

Figura 37 – Exemplo de interação para criação de definição necessária e suficiente para um axioma.

Arandu também permite a construção de axiomas de disjunção, através da negação de uma relação. Dada a sentença S3 = *Cerrado has no Sea*, temos os seguintes candidatos a classes:

$Classes = [Cerrado, Sea]$

S3 pode gerar o seguinte axioma lógico:

$$Cerrado\ SubclassOf:$$
$$not\ (has\ some\ Sea)$$

Na figura 38 temos a interação com Arandu com a sentença S3.

Dada uma sentença S4 = *Pequizeiro is a tree that has Pequi.*, temos um exemplo onde o processo de **desambiguação** foi fundamental, pois a definição do termo *Tree* em SUMO não pertence ao contexto da ontologia que está sendo criada.

Em S4 temos os seguintes candidatos a classes:

$Classes = [Pequizeiro, Tree, Pequi]$

S4 pode gerar o seguinte axioma lógico:

$$Pequizeiro\ SubclassOf:$$
$$Tree$$
$$and\ (has\ some\ Pequi)$$

Na figura 39 temos a interação com Arandu com a sentença S4.

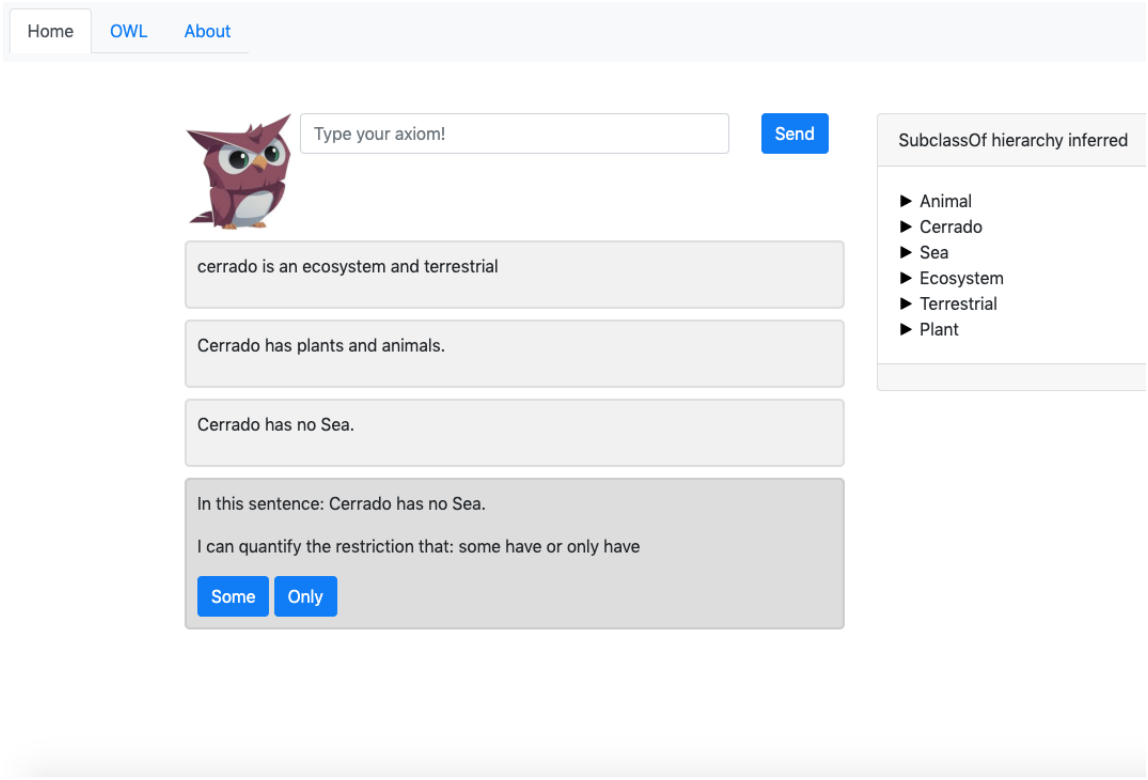

Figura 38 – Exemplo de interação onde é possível construir uma relação de disjunção com uma negação.

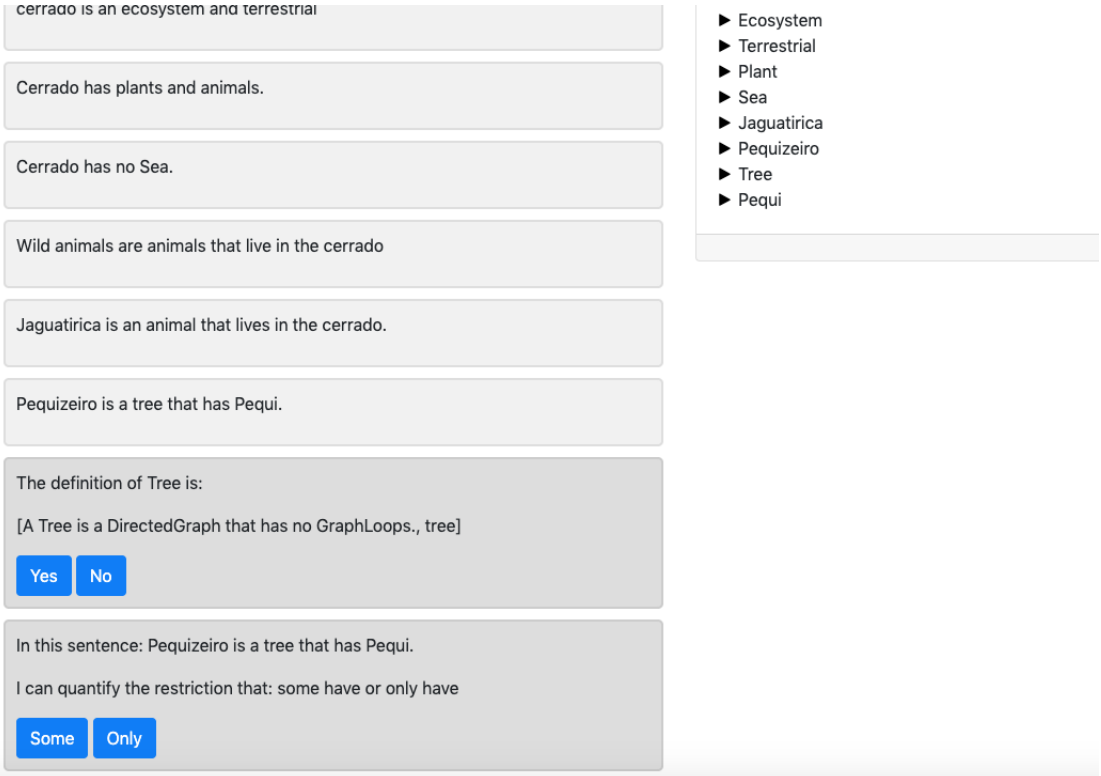

Figura 39 – Exemplo de interação onde foi descartado uma sugestão de SUMO

5.4.1 Dedução

Por meio de um raciocinador em DL, o Arandu Chatbot poderá inferir algum conhecimento que esteja implícito e adicionar esse axioma na ontologia.

Dada a sentença S5 = *Wild animals are animals that live in the cerrado*, teremos os seguintes candidatos a classes:

$Classes = [WildAnimal, Animal, Cerrado]$

S5 pode gerar o seguinte axioma lógico:

$$WildAnimal\ EquivalentTo:$$
$$Animal$$
$$and\ (live\ some\ Cerrado)$$

Na figura 40 temos um exemplo de interação com a sentença S5 salva como uma condição **necessária e suficiente**.

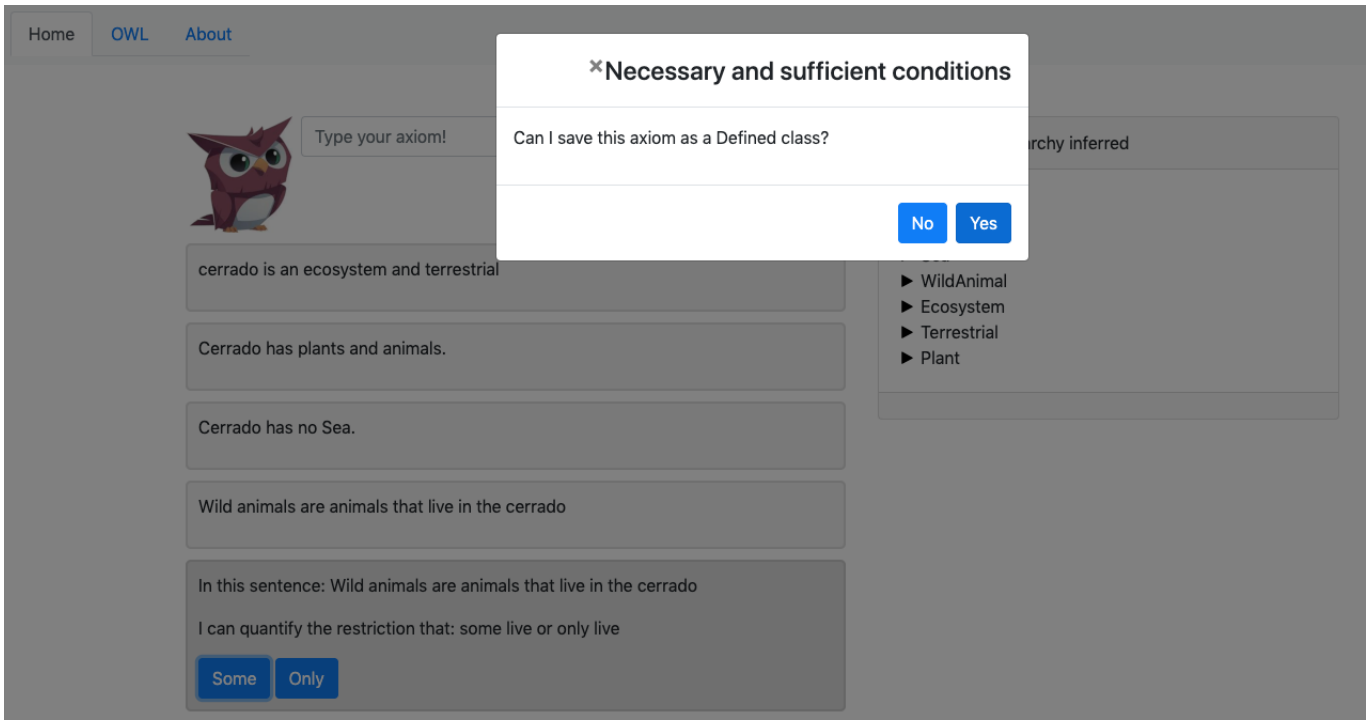

Figura 40 – Exemplo de interação com a sentença S5 = *Wild animals are animals that live in the cerrado*, salva como necessária e suficiente.

Adicionando agora à ontologia a sentença S6 = *Jaguatirica is an animal that lives in the cerrado.*, que gerará os seguintes candidatos a classes e axioma lógico respectivamente.

$Classes = [Jaguatirica, Animal, Cerrado]$

$Jaguatirica\ SubclassOf$:

$$Animal$$
$$and\ (live\ some\ Cerrado)$$

Salvando a sentença S6 com o quantificador *some* e como não sendo uma condição **necessária e suficiente**, Arandu irá **inferir** que *Jaguatirica* é subclasse de *WildAnimal*. Na figura 41 podemos visualizar este resultado.

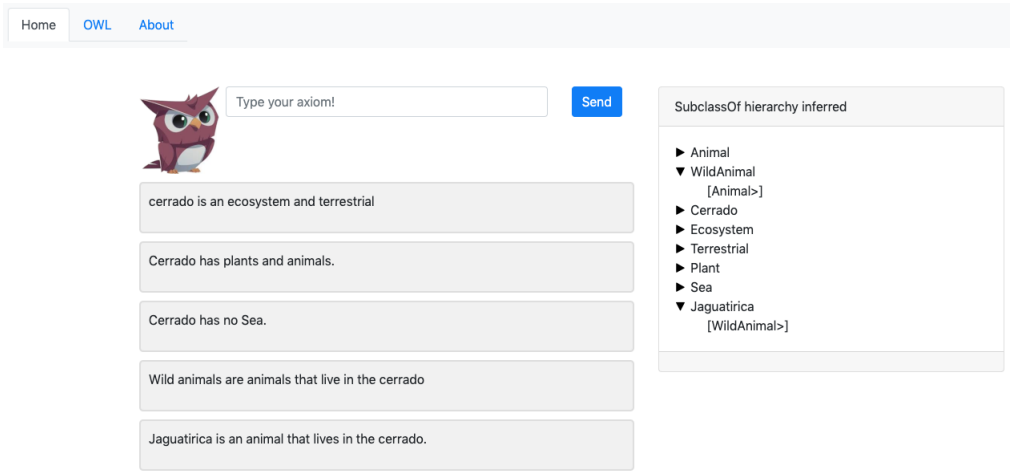

Figura 41 – Exemplo de interação que mostra a dedução da classe *Jaguatirica* como *subclassOf* de *WildAnimal*

5.4.2 Checagem de inconsistência

Arandu, possui um módulo de raciocínio que permite, além de inferências lógicas para descoberta de novos axiomas, checagem de consistência. Dizemos que uma classe é inconsistente quando não é possível que ela possua instâncias.

Dada a seguinte sentença S7 = *Jaguatirica eats no fruit.*

S7 gera o seguinte axioma lógico:

$Jaguatirica\ SubclassOf$:

$$not\ (eat\ some\ Fruit)$$

Dada a seguinte sentença S8 = *Pequi is a fruit.*

S8 gera o seguinte axioma lógico:

$Pequi\ SubclassOf$:

$$Fruit$$

Informando ao Arandu as sentenças S7 e S8, e se inserirmos uma nova sentença S9 = *Jaguatirica eats Pequi.*, teremos uma inconsistência lógica.

$Jaguatirica \sqsubseteq \neg\exists eat.Fruit$

$Pequi \sqsubseteq Fruit$

$\models$

$Jaguatirica \sqsubseteq \neg\exists eat.Pequi$

Ao detectar uma inconsistência, Arandu informa ao usuário quais as classes que estão inconsistentes e apresenta um botão de edição de axioma. Na figura 42 podemos ver um exemplo desta interação.

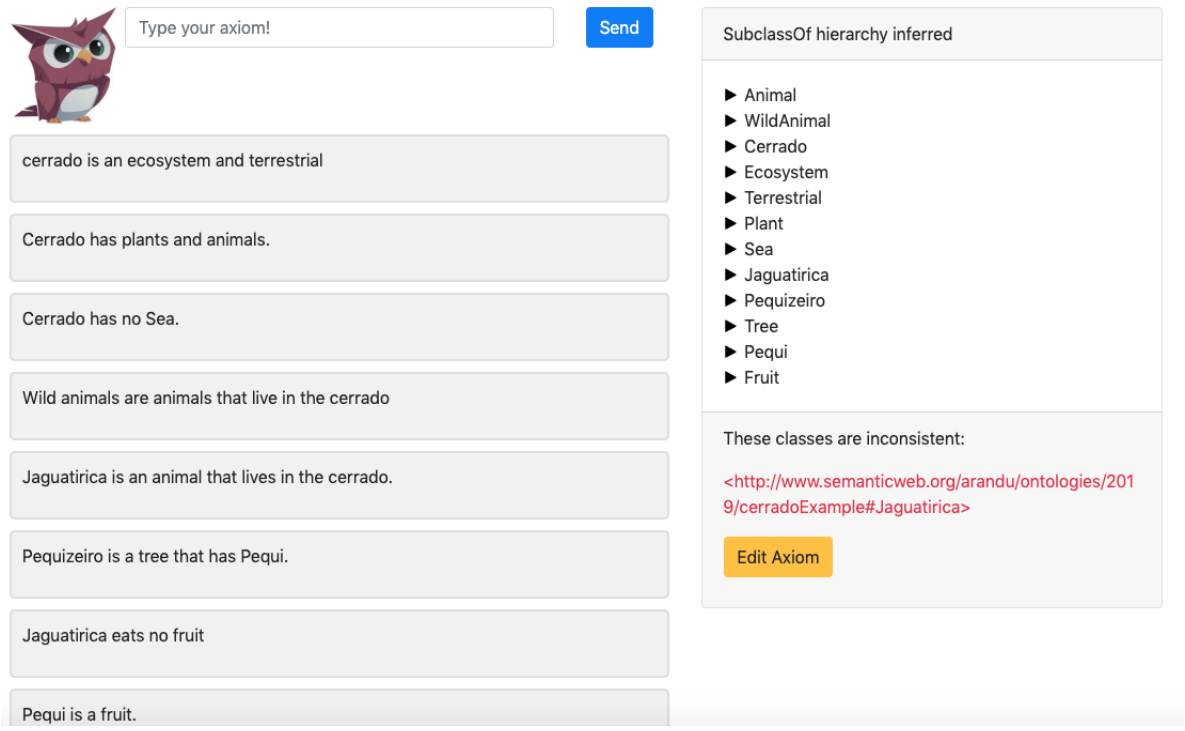

Figura 42 – Exemplo de interação com classes inconsistentes na ontologia.

Ao clicar no botão de edição de axiomas, o usuário poderá deletar as sentenças que geraram a inconsistência e novamente quantificar as relações. Na figura 43 temos um exemplo desta interação.

5.4.3 Geração de código OWL

O processo de geração de código OWL corresponde à fase de implementação, vista na metodologia Arandu, e é a codificação desses axiomas em uma linguagem formal. O Chatbot

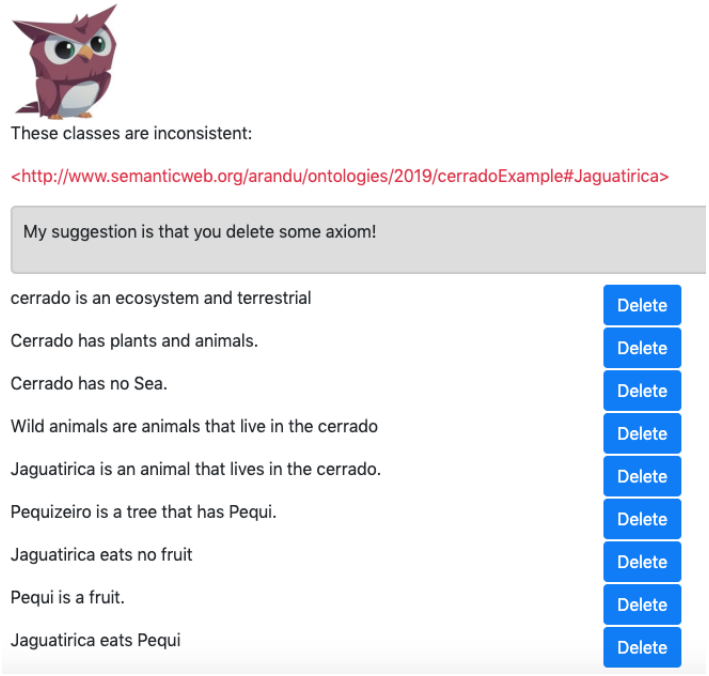

Figura 43 – Exemplo de interação para edição de axiomas, Arandu permite que o usuário delete a sentença que causou inconsistência.

Arandu converte os axiomas para linguagem OWL 2 garantindo expressividade $\mathcal{ALC}$.

Arandu possui uma aba chamada OWL que permite a visualização do código da ontologia em linguagem OWL na sintaxe de Manchester. Na figura 44 podemos ver parte do código gerado por Arandu.

5.4.4 Documentação

A documentação é realizada durante todo o processo de construção da ontologia, através da análise dos *logs* da interação, gerando um documento com as definições informadas, classes e relações extraídas, axiomas gerados, e axiomas reutilizados.

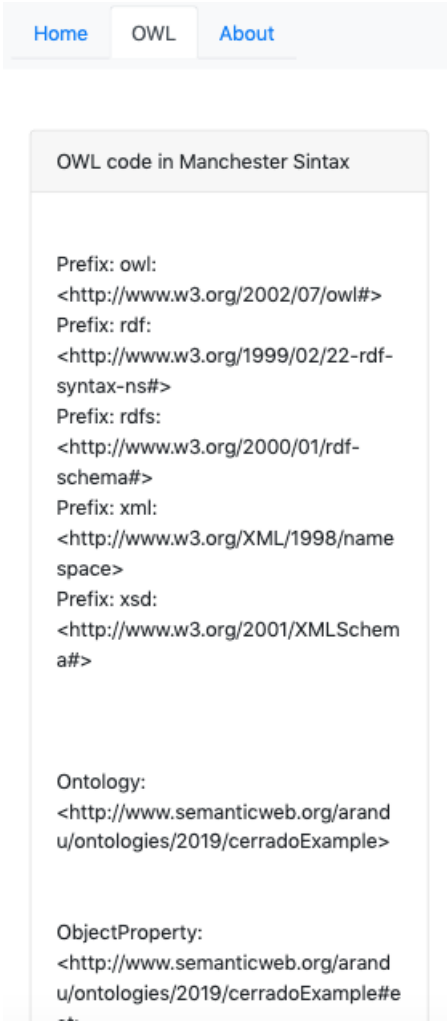

Figura 44 – Parte do código OWL em sintaxe Manchester da ontologia gerada a partir das sentenças S1-S8 informadas nas seções anteriores.

6 TESTES E RESULTADOS

Para testar o Arandu propomos uma ontologia de domínio para representar conhecimento sobre um ecossistema conhecido como Cerrado[1]. Escolhemos este domínio por ser de simples representação e por permitir correspondência com as ontologias de topo SUMO e Ontobio.

Os testes foram divididos em duas etapas. Na primeira etapa testamos Arandu quanto aos requisitos de software. A segunda etapa consistiu na utilização do Arandu por usuários, para que eles avaliassem o artefato.

6.1 REQUISITOS DE SOFTWARE

Baseados no paradigma epistemológico metodológico DSR, testamos os requisitos de software do Chatbot Arandu que foram propostos a partir das conjecturas teóricas propostas.

Para executar este teste, replicamos os procedimentos descritos na seção 5.4 com as sentenças descritas na tabela 11. Estas sentenças foram criadas pelo próprio autor em conjunto com especialistas da área.

Tabela 11 – Sentenças criadas para gerar a ontologia de domínio Cerrado Ontology.

S1	Cerrado is an ecosystem and terrestrial.
S2	Cerrado has plants and animals.
S3	Cerrado has no Sea.
S4	Pequizeiro is a tree that has Pequi.
S5	Wild animals are animals that live in the cerrado.
S6	Jaguatirica is an animal that lives in the cerrado.
S7	Jaguatirica eats no fruit.
S8	Pequi is a fruit.

Arandu foi implementado em um servidor web do tipo *container service* com configuração escalável. O *deploy* foi feito a partir de um arquivo .jar em um servidor de aplicação *Spring Boot*[2]. Arandu pode ser acessado pela URL *http://aranduweb.jelastic.saveincloud.net*.

Arandu converteu todas a sentenças da tabela 11, e apresentou navegabilidade como proposto na seção 5.4.

Foram executados testes sem o reuso de uma ontologia de topo, e reutilizando classes de SUMO e de Ontobio separadamente. Na tabela 12 temos as métricas[3] obtidas para a ontologia de domínio proposta.

[1] https://en.wikipedia.org/wiki/Cerrado
[2] https://spring.io/projects/spring-boot
[3] As métricas foram obtidas através do software Protégé

Tabela 12 – Métricas da ontologia de domínio Cerrado Ontology.

Métricas	**sem ontologia de topo**	**SUMO**	**Ontobio**
Axiom	28	91	32
Logical axiom count	13	57	15
Class count	12	31	14
SubClassOf axiom count	12	56	14
EquivalentClasses axioms count	1	1	1
Object Property count	3	3	3
DL expressivity	$\mathcal{ALC}$	$\mathcal{ALC}$	$\mathcal{ALC}$

As métricas obtidas apontam que Arandu amplia consideravelmente a quantidade de classes com o reuso de uma ontologia de topo. Esse resultado é positivo, pois essa alta correspondência garante qualidade a ontologia que está sendo criada porque estes termos já foram formalmente definidos em SUMO.

6.2 APLICAÇÃO DO CHATBOT

Com o intuito de validar o artefato proposto, convidamos um grupo de 11(onze) pessoas para testar o Chatbot replicando a ontologia proposta na seção 5.4 e responder um *survey* após o teste.

A escolha das pessoas foi feita pelo autor com candidatos que se dispuseram a testar o *chatbot*. No grupo havia pessoas com formação em Agronomia (uma pessoa), Letras (duas pessoas) e Computação (oito pessoas).

O teste foi realizado online. Informamos as sentenças necessárias para replicar a ontologia proposta, e os usuários acessaram o Arandu a partir de computadores e celulares conectados à Internet. Todos os usuários convidados conseguiram replicar a ontologia, ou seja alcançaram os mesmo resultados da tabela 12.

O *survey* foi dividido em seções com o objetivo de caracterizar o grupo quanto ao conhecimento sobre ontologias, conhecimento técnico de programação de computadores, conhecimento sobre o domínio proposto e avaliação do Chatbot Arandu.

Nas próximas seções são apresentadas as perguntas e respostas obtidas no *survey*.

6.2.1 Perfil do grupo

Para traçar o perfil de cada usuário e caracterizar o grupo, foram feitas 4 perguntas:

P 1 : *De 1 a 5, o quanto você entende de programação?* : O objetivo desta pergunta é entender o quanto o usuário conhece de formalização de código computacional. Essa informação é importante para descobrir se o grupo entende da dificuldade e do processo formal necessário para se criar um código que seja legível por máquina.

As respostas obtidas para esta pergunta, apontam que temos um grupo heterogêneo. Com pessoas que sabem programar e com pessoas que não sabem. As resposta obtidas são mostradas na tabela 13.

Tabela 13 – Tabela para identificação do conhecimento do grupo sobre programação de computadores.

De 1 a 5 o quanto você entende de programação?	Respostas
1- Não tenho conhecimento sobre programação.	3
2- Já fiz algum algoritmo, sei o que é uma variável ou laço de repetição.	0
3- Sei programar, porém, estou sem praticar há algum tempo.	0
4- Sei programar, lido com pelo menos uma linguagem de programação frequentemente.	2
5- Sei programar, conheço várias linguagens de programação.	6

P2 : *Antes de testar o Arandu, você sabia o que era uma ontologia?* : O objetivo desta pergunta é saber se o usuário já teve algum contato com ontologias. Essa informação não é suficiente para definir se o usuário é um especialista em ontologias, mas reflete a importância que o grupo deu ao tema de estudo.

As respostas obtidas para **P2** apontam que a maioria do grupo já havia tinha contato com o tema ontologias em algum momento. As respostas obtidas são mostradas na tabela 14.

Tabela 14 – Tabela para identificação do conhecimento do grupo sobre ontologias.

Antes de testar o Arandu, você sabia o que era uma ontologia?	Respostas
Sim	8
Já tinha ouvido falar	2
Não	1

P3 : *Você já construiu uma ontologia ? (pode ser algum exemplo)* : As informações obtidas nas perguntas **P1** e **P2**, apontam para um grupo heterogêneo em conhecimento prático de programação, contudo com conhecimento sobre ontologias. A pergunta **P3** teve o objetivo de identificar no grupo quantas pessoas já haviam modelado alguma ontologia.

As respostas obtidas para **P3**, apontam que a maioria do grupo não tinha conhecimento prático sobre construção de ontologias, apenas três pessoas já haviam construído alguma ontologia. As respostas obtidas são mostradas na tabela 15.

P4 : *Você conhece alguma ferramenta de construção de ontologias ?(exemplo Protégé)* : O objetivo desta pergunta foi identificar o quanto o grupo conhecia sobre ferramentas de construção de ontologias. Como visto nas respostas obtidas, a maioria do grupo conhece sobre ontologias, a maioria não implementou alguma ontologia, mas as

Tabela 15 – Tabela para identificação do conhecimento prático do grupo sobre construção de ontologias.

Você já construiu uma ontologia (pode ser algum exemplo)	Respostas
Sim	3
Não	8

respostas de **P4** podem identificar se o grupo mesmo sem ter tido experiência prática, sabe quais as ferramentas utilizadas para se construir uma ontologia.

As respostas obtidas para **P4**, apontam que a maioria do grupo não conhece nenhuma ferramenta para construção de ontologias. Apenas duas pessoas conhecem o software Protégé, e apenas uma pessoa conhece mais de uma ferramenta que pode ser utilizada para se construir uma ontologia. As respostas obtidas são mostradas na tabela 16

Tabela 16 – Tabela para identificação do conhecimento do grupo acerca de ferramentas de construção de ontologias.

Você conhece alguma ferramenta de construção de ontologias ?(exemplo Protégé)	Respostas
Não conheço nenhuma.	8
Somente o Protégé.	2
Conheço o Protégé e outras.	1

As perguntas relacionadas ao perfil do grupo permitiram identificar um grupo heterogêneo quanto ao conhecimento sobre programação, que tinha um conhecimento prévio sobre ontologias, porém com pouca experiência prática em geral.

6.2.2 Sobre Chatbots ou interfaces de conversação

Uma ameaça à validade da aplicação do Chatbot Arandu seria a resistência à tecnologia de Chatbots. Como foi fundamentado teoricamente nas seções anteriores, o uso de Chatbots ou interfaces de conversação para as mais diversas aplicações é promissor e já vem sendo estudado há algum tempo.

Com o objetivo de descobrir qual a opinião dos usuários sobre o uso de Chatbots, inclusive após o teste do Arandu, fizemos a seguinte pergunta **P5:** ***Como você avalia a possibilidade de obter alguma informação ou construir algum conhecimento interagindo com um robô de conversação (Chatbot)?***.

As respostas foram unânimes, todos avaliaram como positivo o uso de Chatbots para obter ou construir conhecimento. As resposta obtidas são apresentadas na tabela 17.

Tabela 17 – Tabela com a opinião dos usuários sobre o uso de Chatbots.

Como você avalia a possibilidade de obter alguma informação ou construir algum conhecimento interagindo com um robô de conversação (Chatbot)?	Respostas
Positiva	11
Negativa	0
Neutra	0

6.2.3 Domínio da ontologia

O teste do Chatbot Arandu foi executado *online* acessando uma URL[4] e replicando a ontologia proposta na seção 5.4. A escolha desta ontologia foi feita por ser uma área de conhecimento comum e na qual há grande número de classes que poderiam ser reutilizados da ontologia de topo configurada, a SUMO.

Questionamos os usuários sobre o domínio escolhido com o propósito de validar se eles estavam entendendo o conhecimento que estava sendo gerado. As sentenças para a construção da ontologia foram definidas, porém os usuários seriam responsáveis por desambiguar as definições advindas da ontologia de topo e concordar se estavam ou não no contexto da Ontologia.

Para fazer esta análise realizamos a seguinte pergunta **P6: *O quanto você conhece sobre o domínio de conhecimento escolhido para o exemplo?***.

As respostas obtidas demonstram que o grupo conseguiu entender o domínio escolhido e dois usuários teriam conhecimento maior, sendo capazes de criar mais axiomas para esta ontologia. As respostas obtidas são apresentadas na tabela 18.

Tabela 18 – Tabela com as respostas dos usuários sobre o entendimento do domínio de ontologia escolhido.

O quanto você conhece sobre o domínio de conhecimento escolhido para o exemplo?	Respostas
Desconheço, não consegui entender o assunto.	0
Conheço, consegui entender o domínio do assunto	9
Conheço bem, seria capaz de construir algum conhecimento sobre o assunto	2
Conheço muito, sou especialista ou tenho afinidade com o assunto.	0

6.2.4 Sobre o Arandu

Para avaliar o uso do Arandu quanto à facilidade de uso, elaboramos quatro perguntas:

P7: *Você se sentiu guiado pelo Arandu durante a construção da ontologia?*. O objetivo desta pergunta foi analisar o quanto o Arandu direcionou o usuário para

[4] aranduweb.jelastic.saveincloud.net

o processo de construção da ontologia. A metodologia proposta pelo Arandu, mantém o usuário em um fluxo coerente para a construção da ontologia, as etapas deste processo podem ser revisitadas na figura 30.

As respostas obtidas para essa pergunta apontam que a maioria dos usuários se sentiu guiado pelo Arandu durante a construção da ontologia. As respostas obtidas podem ser vistas na tabela 19.

Tabela 19 – Tabela com a opinião dos usuários sobre a participação do Arandu nas etapas de construção da ontologia.

Você se sentiu guiado pelo Arandu durante a construção da ontologia?	Respostas
Sim	10
Não	1

P8: ***Arandu demonstrou estar participando do processo de construção da ontologia? (sugerindo conhecimento)***. O objetivo desta pergunta foi descobrir se o usuário teve o Arandu como um sujeito ativo no processo de construção da ontologia.

As repostas obtidas para esta pergunta demonstram que a maioria dos usuários perceberam o Arandu como uma ferramenta que participa do processo de construção da ontologia. As respostas obtidas podem sem vistas na tabela 20.

Tabela 20 – Tabela com as respostas dos usuários sobre a participação do Arandu no processo de construção da ontologia.

Arandu demonstrou estar participando do processo de construção da ontologia? (sugerindo conhecimento)	Respostas
Sim	10
Não	1

P9: ***Arandu, com suas sugestões, ampliou o domínio esperado de ontologia?***. O objetivo desta pergunta foi analisar se os usuários consideraram que a ontologia foi ampliada com o uso do Arandu, ou seja o número de classes foi maior que o esperado.

As respostas obtidas para esta pergunta demonstram que todos os usuários concordam que Arandu amplia o modelo de ontologia esperado. Através do uso de uma ontologia de topo, superclasses foram reutilizadas e novos axiomas foram gerados a partir de raciocínio automático. As respostas obtidas podem ser vistas na tabela 21.

Tabela 21 – Tabela com a avaliação dos usuários sobre a ampliação do domínio da ontologia que foi criada com o Arandu.

Arandu, com suas sugestões, ampliou o domínio esperado de ontologia?	Respostas
Sim	11
Não	0

P10: ***Com o apoio de Arandu, você considera ter criado uma ontologia sem erros? Por que?***. O objetivo desta pergunta foi questionar aos usuários se eles consideram que o Arandu contribuiu para construir uma ontologia consistente, sem erros lógicos.

As respostas para esta pergunta apontam que a maioria dos usuários considera a ontologia criada consistente e sem erros. Ao replicar a ontologia que foi proposta, os usuários tiveram a oportunidade de verificar como o Arandu iria se comportar ao identificar um axioma inconsistente. Algumas[5] das respostas obtidas e considerações feitas pelos usuários podem ser vistas na tabela 22.

Tabela 22 – Tabela com respostas e opinião dos usuários sobre a contribuição do Arandu para criar uma ontologia consistente, sem erros lógicos.

Com o apoio de Arandu, você considera ter criado uma ontologia sem erros? Por que?
Sim.
Porque as inconsistências foram corrigidas
Sim. Por que ele detecta inconsistências e ajuda na relação entre as inferências
Sim. Porque a ferramenta auxilia na construção da ontologia e quando insere uma afirmação que dá erro na ontologia a ferramenta avisa e bloqueia o uso da afirmação.
Existe um tratamento baseado na coerência das respostas retornadas.
Foi intuitivo e me retornou um resultado de acordo com minhas respostas
Sim, porque ele encontrou a inconsistência no momento em que foi adicionada pelo último axioma.
Consegui sem erros, o Arandu mostra ao usuário inconsistências portanto foi possível fazer sem erros.
Sim. O programa se apresentou didático e bem explicativo.
Não. Pois, algumas definições sobre as ontologias eu não concordei.

P11: ***Classifique o nível de dificuldade encontrado por você para construir a ontologia nessa metodologia proposta?(1 - difícil, 5 - fácil)***. O objetivo desta pergunta foi que os usuários mensurassem o nível de dificuldade encontrado por eles para utilizar o Arandu.

As respostas obtidas apontam que a maioria dos usuários considerou de médio à fácil utilizar o Arandu. As respostas obtidas são mostradas na tabela 23.

P12: ***Classifique o nível de dificuldade que você supõe que uma pessoa com pouco conhecimento técnico teria para construir uma ontologia utilizando o Arandu (favor desconsiderar o idioma)***. O objetivo desta pergunta foi que os usuários tentassem mensurar, a partir de sua experiência, o quanto uma pessoa que seja

[5] Um usuário não respondeu esta questão.

Tabela 23 – Tabela com os dados sobre a dificuldade encontrada pelos usuários ao utilizar o Arandu.

Classifique o nível de dificuldade encontrado por você para construir a ontologia nessa metodologia proposta?(1 - difícil, 5 - fácil)	Respostas
1	0
2	1
3	4
4	2
5	4

especialista de um domínio e sem conhecimentos técnicos sobre ontologias, consiga utilizar o Arandu sem dificuldades.

As respostas obtidas para esta pergunta apontam que a expectativa de que especialistas de domínio com pouco conhecimento técnico sobre ontologias tenham dificuldade ao utilizar o Arandu seja média ou próxima a fácil, porém dois usuários acreditam que pessoas com pouco conhecimento técnico teriam dificuldade para utilizar o Arandu. As respostas obtidas são mostradas na tabela 24.

Tabela 24 – Tabela com as respostas sobre a expectativa de dificuldade que os usuários que testaram acreditam que usuários com pouco conhecimento técnico teriam para utilizar Arandu.

Classifique o nível de dificuldade que você supõe que uma pessoa com pouco conhecimento técnico teria para construir uma ontologia utilizando o Arandu (favor desconsiderar o idioma)(1 - difícil, 5 - fácil)	Respostas
1	2
2	0
3	4
4	5
5	0

6.3 DISCUSSÕES

De acordo com as avaliações propostas no mapeamento DSR proposto na figura 26, observamos os seguintes resultados:

Avaliação 1: ***Arandu converte LNC para axiomas lógicos? Arandu reusa conhecimento de uma ontologia de topo?***: analisando a tabela 12, temos as métricas obtidas para a conversão das sentenças propostas na tabela 11 para linguagem OWL. Arandu satisfaz os requisitos de software propostos convertendo LNC para axiomas lógicos, e demonstrou capacidade de reuso de termos de duas ontologias de topo. Foram

realizados três testes, um sem o uso de uma ontologia de topo, um utilizando a ontologia SUMO e outro teste utilizando a ontologia Ontobio.

Avaliação 2: ***Arandu guia o usuário na construção da ontologia?***: a partir do cenário proposto na seção 5.4, Arandu guia o usuário no processo de construção de uma ontologia de acordo com a metodologia proposta na figura 30.

Avaliação 3: ***Os usuários conseguem construir uma ontologia? Arandu amplia o modelo e diminui a barreira técnica?***: analisando as respostas do usuários obtidas com o teste realizado, podemos afirmar que é possível que um usuário não especialista em ontologias consiga construir uma ontologia utilizando o Arandu. Reutilizar classes de uma outra ontologia exige conhecimento da outra ontologia, com Arandu basta conhecimento sobre o domínio para que isso seja possível.

7 CONCLUSÕES

Os resultados alcançados demonstram a capacidade do Arandu em reutilizar conhecimento de uma ontologia de topo. Arandu torna o processo de reuso mais fácil e intuitivo.

O reuso de termos é um fator importante no processo de construção de uma ontologia, porém exige que o especialista do domínio conheça a ontologia a ser reutilizada. Arandu permite que apenas com conhecimento do domínio a ser construído seja possível reutilizar conhecimento de uma ontologia de topo.

A maioria das metodologias para construção de ontologias propõem o reuso de ontologias, porém como uma etapa isolada que pode não ser realizada. Através de um diálogo Arandu torna o processo de reuso parte do diálogo, com o usuário apenas concordando ou discordando das definições advindas da ontologia de topo.

Os axiomas que são adicionados à ontologia que está sendo criada, poderiam não estar previstos pelo especialista de domínio. Isso torna a ontologia melhor, pois amplia o seu modelo esperado e possibilidade de ser também reutilizada.

Com um *chatbot* guiando o especialista de domínio na construção da ontologia, garantimos que nessa metodologia todas as fases são compreendidas, desde a definição da URI, documentação, adição de quantificadores e principalmente reuso são realizadas. Por meio de um racionador, o conhecimento implícito é identificado e a verificação de inconsistências na ontologia é realizada em tempo de construção.

A metodologia proposta, apoiada por um *chatbot*, permite que especialistas de domínio com menor conhecimento técnico construam ontologias.

7.1 CONTRIBUIÇÕES

Essa tese contribui para o avanço da área de engenharia de ontologias. A criação de ferramentas que permitam que especialistas de domínio criem ontologias com qualidade, contribui para a adoção de ontologias e a implantação da *Web* Semântica.

Com o apoio de um *chatbot* a metodologia proposta guia o usuário no processo de construção da ontologia, torna o processo de reuso mais fácil e democratiza a possibilidade de que alguém com pouco conhecimento técnico construa uma ontologia.

Esse processo gamificado torna a construção da ontologia menos onerosa, axiomatizada e verificável. Cada passo da construção é realizado de forma clara e intuitiva.

Podemos caracterizar as contribuições dessa tese de forma horizontal e vertical.

De forma vertical podemos dizer o processo de construção de uma ontologia se torna mais fácil e produtivo, com uma ferramenta que potencializa o conhecimento do especialista de domínio com o reuso de ontologias já validadas.

De forma horizontal com o avanço para a área de ontologias e representação do conhecimento com mais ontologias que poderão ser construídas de forma mais fácil, incluindo especialistas de domínio que possuem menor conhecimento técnico. Contribui também com ontologias com maior qualidade, que fazem reuso e possuem correspondência com ontologias validadas.

Ainda como contribuição, apresentamos a seguir na tabela 25 uma comparação entre alguns aspectos que foram melhorados do trabalho de (AZEVEDO et al., 2014a).

Tabela 25 – Tabela com o comparativo entre os trabalhos de (AZEVEDO et al., 2014a) e Arandu.

	Renan (AZEVEDO et al., 2014a)	Arandu
Manipulação de ontologias	Criação e Edição	Criação e Edição
Método	Semiautomático	Semiautomático
Abordagem	Linguística	Linguística
Raciocínio Automático	SIM	SIM
Identificação e Tratamento de Inconsistências	SIM	SIM
Ferramenta de PLN	Stanford Parser	CoreNLP (Stanford Parser)
Linguagem Gerada	OWL	OWL
Expressividade DL	$\mathcal{ALC}$	$\mathcal{ALC}$
Metodologia para construir ontologias	NÃO	SIM
Reusa ontologias de topo	NÃO	SIM
Desambiguação de termos	NÃO	SIM
Quantificação das relações	SIM	SIM
API	NÃO	SIM

Na tabela 25 não apresentamos nenhuma comparação relacionada a performance, visto que as características de software entre *Renan (desktop)* e Arandu (*web*) não permite esta comparação. Também não foi realizada nenhuma comparação relacionada a taxa de acertos na conversão de sentenças como foi feito no trabalho de (SILVA, 2019) apresentado na tabela 9.

7.2 LIMITAÇÕES

Durante o processo de construção do Arandu e após realizados testes com usuários, pudemos identificar as seguintes limitações:

Diálogo - uma limitação comum a todos os *chatbots* é se perder durante o diálogo com um usuário, fazendo que ele perca o interesse. Arandu inicialmente foi desenvolvido como um *chatbot* que dialogava com o usuário sobre assuntos que não eram específicos sobre

o desenvolvimento de uma ontologia. Com uma estratégia de chaveamento, tentávamos identificar quando o usuário tinha interesse em dialogar sobre assuntos gerais ou não. Para mitigar essa limitação, delimitamos o seu escopo de diálogo.

PLN - uma limitação que pode comprometer o processo de construção da ontologia é uma possível falha no processamento de linguagem natural, por exemplo, a não identificação correta dos sintagmas que compõem a sentença pode comprometer a geração do axioma. Essa limitação é dependente da ferramenta de PLN utilizada. Para mitigar esse problema estamos realizando testes com outras ferramentas de PLN e identificando no processo falhas que possam ser corrigidas pelo usuário em tempo de construção.

Reuso - como apresentado nos resultados, a configuração das ontologias interfere na quantidade de reuso de termos e pode interferir na qualidade. Para mitigar essa limitação adicionamos como trabalhos futuros a capacidade de alterar ou configurar a ontologia de topo a ser reutilizada, ou utilizar mais de uma ontologia de topo simultaneamente.

Arandu, por ter sido desenvolvido de forma modularizada, favorece a correção das limitações com a troca ou adição de novos módulos.

7.3 TRABALHOS FUTUROS

Os resultados alcançados com este trabalho vislumbram um horizonte de oportunidades no campo de ontologias. Dentre estas possibilidades, podemos elencar quais as próximas pesquisas que serão realizadas:

- Modelagem conceitual orientada pela OntoUML.

Reutilizar ontologias é uma tarefa importante no processo de construção de ontologias. Trabalhos recentes de (VERDONCK et al., 2019) apontam que construir uma ontologia guiada por um modelo conceitual amplia a qualidade da ontologia que está sendo criada. É possível adicionar mais esta etapa à metodologia proposta nesta tese.

- Permitir reuso simultâneo ou configurável de ontologias de topo.

Foram executados testes com duas ontologias de topo. Em trabalhos futuros, implementaremos a possibilidade de se utilizar duas ou mais ontologias simultaneamente e o usuário poder escolher de qual ontologia deseja reutilizar conhecimento.

- Reaproveitar a API Arandu para outras aplicações.

Arandu foi desenvolvido no conceito de *RESTful API*. Isto permite o seu reuso para a construção, por exemplo, de uma interface CLI (*Command Line Interface*) que pode ser integrada a um outro software. É possível também a integração a um *Serious Game* que pode ampliar os conceitos de gamificação já inseridos nesta tese, ou também a redes sociais com propósitos definidos.

Também são propostas futuras adição de novas *features* e melhorias no artefato desenvolvido, bem como aplicação em outros cenários.

7.4 PUBLICAÇÕES

Durante o desenvolvimento desta tese foram realizadas submissões em congressos e periódicos científicos.

Publicados:

1 - FERRAZ, A., FREITAS, F. **Uma metodologia gamificada para construção de ontologias, apoiada por um Chatbot**. *in* SBC - Proceedings of SBGames 2019 - ISSN: 2179-2259 Education Track – Full Papers.

2 - FERRAZ, A.; FREITAS, F. **Arandu, um Chatbot para construção de Ontologias guiado por uma Ontologia de Topo**. *in*: Ontobras - Workshop de Teses e Dissertações em Ontologias, 2019, Porto Alegre-RS. nais do Workshop de Teses e Dissertacoes em Ontologias, 2019.

REFERÊNCIAS

AKKERMANS, H.; BROEKSTRA, J.; DAVIES, J.; YING; DING; DUKE, A.; ENGELS, R.; FENSEL, D.; HORROCKS, I.; IOSIF, V.; KAMPMAN, A.; KIRYAKOV, A.; KLEIN, M.; LAU, T.; OGNYANOV, D.; REIMER, U.; SIMOV, K. I.; RUDI; STUDER, D.; MEER, J. van der; HARMELEN, F. van. On-to-knowledge : Semantic web enabled knowledge management. In: . [S.l.: s.n.], 2002.

ALMANSOR, E.; HUSSAIN, F. Survey on intelligent chatbots: State-of-the-art and future research directions. In: _____. [S.l.: s.n.], 2020. p. 534–543. ISBN 978-3-030-22353-3.

AUER, S.; HERRE, H. Rapidowl — an agile knowledge engineering methodology. In: _____. *Perspectives of Systems Informatics: 6th International Andrei Ershov Memorial Conference, PSI 2006, Novosibirsk, Russia, June 27-30, 2006. Revised Papers*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007. p. 424–430. ISBN 978-3-540-70881-0. Disponível em: <https://doi.org/10.1007/978-3-540-70881-0_36>.

AZEVEDO, R. R. D.; FREITAS, F.; ROCHA, R. G. C.; MENEZES, J. A. A. D.; RODRIGUES, C. M. D. O.; SILVA, G. D. F. P. e. An approach for learning and construction of expressive ontology from text in natural language. In: *Proceedings of the 2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT) - Volume 01*. Washington, DC, USA: IEEE Computer Society, 2014. (WI-IAT '14), p. 149–156. ISBN 978-1-4799-4143-8. Disponível em: <http://dx.doi.org/10.1109/WI-IAT.2014.28>.

AZEVEDO, R. R. de; FREITAS, F.; ROCHA, R. G. C.; MENEZES, J. A. A. de; RODRIGUES, C. M. O.; GOMES, M. C. Towards a framework for ontology learning from interactions in natural language and reasoning. In: *Proceedings of 24th Annual International Conference on Computer Science and Software Engineering*. Riverton, NJ, USA: IBM Corp., 2014. (CASCON '14), p. 120–132. Disponível em: <http://dl.acm.org/citation.cfm?id=2735522.2735536>.

BAADER, F.; CALVANESE, D.; MCGUINNESS, D. L.; NARDI, D.; PATEL-SCHNEIDER, P. F. (Ed.). *The Description Logic Handbook: Theory, Implementation, and Applications*. New York, NY, USA: Cambridge University Press, 2003. ISBN 0-521-78176-0.

BARROS FLAVIA A.; TEDESCO, P. C. A. R. Agentes inteligentes conversacionais: Conceitos básicos e desenvolvimento. In: (ORG.), J. C. M. J. V. F. M. E. D. S. M. (Ed.). *35º JAI - Jornada de Atualização em Informática*. [S.l.]: Sociedade Brasileira de Computação, 2016. v 1, p. 169–218.

BERNARDINI, A.; SôNEGO, A.; POZZEBON, E. Chatbots: An analysis of the state of art of literature. In: . [S.l.: s.n.], 2018. p. 1.

BERNERS-LEE, T.; HENDLER, J.; LASSILA, O. The semantic web. *Scientific American*, v. 284, n. 5, p. 34–43, maio 2001. Disponível em: <http://www.sciam.com/article.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21>.

BLOMQVIST, E.; HAMMAR, K.; PRESUTTI, V. Engineering ontologies with patterns - the extreme design methodology. In: *Ontology Engineering with Ontology Design Patterns :*. [S.l.: s.n.], 2016, (Studies on the Semantic Web, 25). p. 23–50. ISBN 978-1-61499-675-0.

BRACHMAN, R.; LEVESQUE, H. *Knowledge Representation and Reasoning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2004. ISBN 1558609326.

BRASIL, B. *Bots Brasil - Conteúdos sobre Bots, Inteligência Artificial e Interfaces Conversacionais.* 2020. <https://medium.com/botsbrasil>. Acessed: 2020-02-12.

BUITELAAR, P.; OLEJNIK, D.; SINTEK, M. A protégé plug-in for ontology extraction from text based on linguistic analysis. In: SPRINGER. *European Semantic Web Symposium*. [S.l.], 2004. p. 31–44.

CIMIANO, P. Ontology learning and population from text - algorithms, evaluation and applications. In: . [S.l.: s.n.], 2006.

CIMIANO, P.; VÖLKER, J.; BUITELAAR, P. Ontology construction. In: INDURKHYA, N.; DAMERAU, F. J. (Ed.). *Handbook of Natural Language Processing, Second Edition*. Chapman and Hall/CRC, 2010. p. 577–604. ISBN 978-1-4200-8592-1. Disponível em: <http://www.crcnetbase.com/doi/abs/10.1201/9781420085938-c24>.

CLARK, P.; MURRAY, W. R.; HARRISON, P.; THOMPSON, J. Naturalness vs. predictability: A key debate in controlled languages. In: FUCHS, N. E. (Ed.). *Controlled Natural Language*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010. p. 65–81. ISBN 978-3-642-14418-9.

CYCORP. *Knowledge Base Cycorp*. 2017. <http://www.cyc.com/kb/>. Accessed: 2017-02-09.

DRESCH ALINE; LACERDA, D. P. A. J. J. A. V. *Design Science Research: método de pesquisa para avanço da ciência e tecnologia.* Porto Alegre: Bookman, 2015.

FERNANDEZ-LOPEZ, M.; GOMEZ-PEREZ, A.; JURISTO, N. Methontology: from ontological art towards ontological engineering. In: *Proceedings of the AAAI97 Spring Symposium*. Stanford, USA: [s.n.], 1997. p. 33–40.

FERNáNDEZ-LóPEZ, M.; POVEDA-VILLALóN, M.; SUáREZ-FIGUEROA, M. C.; GóMEZ-PéREZ, A. Why are ontologies not reused across the same domain? *Journal of Web Semantics*, v. 57, p. 100492, 2019. ISSN 1570-8268. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1570826818300726>.

FØLSTAD, A.; BRANDTZÆG, P. B. Chatbots and the new world of hci. *interactions*, ACM, New York, NY, USA, v. 24, n. 4, p. 38–42, jun. 2017. ISSN 1072-5520. Disponível em: <http://doi.acm.org/10.1145/3085558>.

FOX, M.; GRÜNINGER, M. Enterprise modelling. *Ai Mag*, v. 19, 12 1997.

FRANCIS, N.; GREEN, A.; GUAGLIARDO, P.; LIBKIN, L.; LINDAAKER, T.; MARSAULT, V.; PLANTIKOW, S.; RYDBERG, M.; SELMER, P.; TAYLOR, A. Cypher: An evolving query language for property graphs. In: *Proceedings of the 2018 International Conference on Management of Data*. New York, NY, USA: Association

for Computing Machinery, 2018. (SIGMOD '18), p. 1433–1445. ISBN 9781450347037. Disponível em: <https://doi.org/10.1145/3183713.3190657>.

FREITAS, F.; OTTEN, J. A connection calculus for the description logic *ALC*. In: *Canadian Conference on AI*. [S.l.]: Springer, 2016. (Lecture Notes in Computer Science, v. 9673), p. 243–256.

FUCHS, N. E.; KALJURAND, K.; KUHN, T. Attempto controlled english for knowledge representation. In: _____. *Reasoning Web: 4th International Summer School 2008, Venice, Italy, September 7-11, 2008, Tutorial Lectures*. Berlin, Heidelberg: Springer-Verlag, 2008. p. 104–124. ISBN 9783540856566. Disponível em: <https://doi.org/10.1007/978-3-540-85658-0_3>.

GAO, T. *Controlled Natural Languages and Default Reasoning*. 2019.

GRUBER, T. R. Toward principles for the design of ontologies used for knowledge sharing. *Int. J. Hum.-Comput. Stud.*, Academic Press, Inc., Duluth, MN, USA, v. 43, n. 5-6, p. 907–928, dez. 1995. ISSN 1071-5819. Disponível em: <http://dx.doi.org/10.1006/ijhc.1995.1081>.

GUARINO, N. *Formal Ontology in Information Systems: Proceedings of the 1st International Conference June 6-8, 1998, Trento, Italy*. 1st. ed. Amsterdam, The Netherlands, The Netherlands: IOS Press, 1998. ISBN 9051993994.

GUARINO, N.; OBERLE, D.; STAAB, S. What is an ontology? In: STAAB, S.; STUDER, R. (Ed.). *Handbook on Ontologies*. Second. [S.l.]: Springer, 2009.

GUARINO, N.; WELTY, C. Evaluating ontological decisions with ontoclean. *Commun. ACM*, ACM, New York, NY, USA, v. 45, n. 2, p. 61–65, fev. 2002. ISSN 0001-0782. Disponível em: <http://doi.acm.org/10.1145/503124.503150>.

GUIZZARDI, G. *Ontological foundations for structural conceptual models*. Tese (Doutorado), Enschede, 2005. Disponível em: <http://doc.utwente.nl/50826/>.

HAASE, P.; VÖLKER, J. Ontology learning and reasoning - dealing with uncertainty and inconsistency. In: *ISWC-URSW*. [S.l.: s.n.], 2005.

HEARST, M. A. Automatic acquisition of hyponyms from large text corpora. In: *COLING 1992 Volume 2: The 15th International Conference on Computational Linguistics*. [s.n.], 1992. Disponível em: <https://www.aclweb.org/anthology/C92-2082>.

HEIJST, G. van; SCHREIBER, A.; WIELINGA, B. Using explicit ontologies in kbs development. *International Journal of Human-Computer Studies*, v. 46, n. 2, p. 183 – 292, 1997. ISSN 1071-5819. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1071581996900907>.

HEVNER, A.; CHATTERJEE, S. Design science research in information systems. In: _____. *Design Research in Information Systems: Theory and Practice*. Boston, MA: Springer US, 2010. p. 9–22. ISBN 978-1-4419-5653-8. Disponível em: <https://doi.org/10.1007/978-1-4419-5653-8_2>.

HORRIDGE, M.; BECHHOFER, S. The owl api: A java api for owl ontologies. *Semant. Web*, IOS Press, NLD, v. 2, n. 1, p. 11–21, jan. 2011. ISSN 1570-0844.

ISOTANI, S.; BITTENCOURT, I. I. *Dados Abertos Conectados*. São Paulo, SP: Novatec, 2015. ISBN 978-85-7522-449-6.

KIRYAKOV, A.; OGNYANOV, D.; MANOV, D. Owlim – a pragmatic semantic repository for owl. In: DEAN, M.; GUO, Y.; JUN, W.; KASCHEK, R.; KRISHNASWAMY, S.; PAN, Z.; SHENG, Q. Z. (Ed.). *Web Information Systems Engineering – WISE 2005 Workshops*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005. p. 182–192. ISBN 978-3-540-32287-0.

KLEIN, D.; MANNING, C. D. Accurate unlexicalized parsing. In: *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2003. (ACL '03), p. 423–430. Disponível em: <http://dx.doi.org/10.3115/1075096.1075150>.

KRÖTZSCH, M. OWL 2 profiles: An introduction to lightweight ontology languages. In: *Reasoning Web. Semantic Technologies for Advanced Query Answering - 8th International Summer School 2012, Vienna, Austria, September 3-8, 2012. Proceedings*. [s.n.], 2012. p. 112–183. Disponível em: <https://doi.org/10.1007/978-3-642-33158-9_4>.

KUHN, T. A survey and classification of controlled natural languages. *Computational Linguistics*, v. 40, n. 1, p. 121–170, mar. 2014. Disponível em: <https://www.aclweb.org/anthology/J14-1005>.

LENAT, D. B. Cyc: A large-scale investment in knowledge infrastructure. *Commun. ACM*, ACM, New York, NY, USA, v. 38, n. 11, p. 33–38, nov. 1995. ISSN 0001-0782. Disponível em: <http://doi.acm.org/10.1145/219717.219745>.

LIDDY, E. D. Natural Language Processing. In: *Encyclopedia of Library and Information Science*. 2. ed. NY: Marcel Decker, Inc., 2001. ISBN 9783662459232.

LIMA, C. E. T. *Um Chatterbot Para Criação E Desenvolvimento De Ontologias Com Lógica De Descrição*. Dissertação (Mestrado) — Cin/UFPE, Recife-PE, 2017.

LIMA, R.; ESPINASSE, B.; FREITAS, F. Ontoilper: an ontology- and inductive logic programming-based system to extract entities and relations from text. *Knowl. Inf. Syst.*, v. 56, n. 1, p. 223–255, 2018.

LUGER, G. *Inteligência Artificial - 4.ED.: Estruturas e estratégias para a solução de problemas complexos*. Bookman, 2004. ISBN 9788577801206. Disponível em: <https://books.google.com.br/books?id=h2GJLP6jQ4kC>.

MAEDCHE, A.; STAAB, S. Ontology learning. In: *Handbook on ontologies*. [S.l.]: Springer, 2004. p. 173–190.

MANNING, C. D.; SURDEANU, M.; BAUER, J.; FINKEL, J.; BETHARD, S. J.; MCCLOSKY, D. The Stanford CoreNLP natural language processing toolkit. In: *Association for Computational Linguistics (ACL) System Demonstrations*. [s.n.], 2014. p. 55–60. Disponível em: <http://www.aclweb.org/anthology/P/P14/P14-5010>.

MARAN, V.; MACHADO, G. M.; MACHADO, A.; AUGUSTIN, I.; LIMA, J. a. C. D.; OLIVEIRA, J. P. M. d. Database ontology-supported query for ubiquitous environments. In: *Proceedings of the 23rd Brazillian Symposium on Multimedia and the Web*. New York, NY, USA: ACM, 2017. (WebMedia '17), p. 185–188. ISBN 978-1-4503-5096-9. Disponível em: <http://doi.acm.org/10.1145/3126858.3131575>.

MASCHE, J.; LE, N.-T. A review of technologies for conversational systems. In: *ICCSAMA*. [S.l.: s.n.], 2017.

MCTEAR, M.; CALLEJAS, Z.; GRIOL, D. *The Conversational Interface: Talking to Smart Devices*. Cham: Springer, 2016. ISBN 978-3-319-32965-9.

MUSEN, M. A. The protégé project: a look back and a look forward. *AI Matters*, v. 1, n. 4, p. 4–12, 2015. Disponível em: <https://doi.org/10.1145/2757001.2757003>.

NAVIGLI, R.; VELARDI, P. Learning domain ontologies from document warehouses and dedicated web sites. *Comput. Linguist.*, MIT Press, Cambridge, MA, USA, v. 30, n. 2, p. 151–179, jun. 2004. ISSN 0891-2017. Disponível em: <https://doi.org/10.1162/089120104323093276>.

NEVES, A. M. M.; BARROS, F. A.; HODGES, C. iaiml: a mechanism to treat intentionality in aiml chatterbots. In: *ICTAI*. IEEE Computer Society, 2006. p. 225–231. ISBN 0-7695-2728-0. Disponível em: <http://dblp.uni-trier.de/db/conf/ictai/ictai2006.html#NevesBH06>.

NOY, N. F.; MCGUINNESS, D. L. *Ontology Development 101: A Guide to Creating Your First Ontology*. [S.l.], 2001.

PAVEL, S.; EUZENAT, J. Ontology matching: State of the art and future challenges. *IEEE Trans. on Knowl. and Data Eng.*, IEEE Educational Activities Department, Piscataway, NJ, USA, v. 25, n. 1, p. 158–176, jan. 2013. ISSN 1041-4347. Disponível em: <http://dx.doi.org/10.1109/TKDE.2011.253>.

PEASE, A. *Ontology: A Practical Guide*. Angwin, CA: Articulate Software Press, 2011.

PEFFERS, K.; TUUNANEN, T.; ROTHENBERGER, M.; CHATTERJEE, S. A design science research methodology for information systems research. *Journal of Management Information Systems*, v. 24, p. 45–77, 01 2007.

PIMENTEL, M.; FILIPPO, D.; SANTORO, F. M. Metodologia de pesquisa em informática na educação: Concepção da pesquisa. In: _____. *Metodologia de Pesquisa em Informática na Educação: Concepção da Pesquisa.* Porto Alegre: SBC, 2019. cap. Design Science Research: fazendo pesquisas científicas rigorosas atreladas ao desenvolvimento de artefatos computacionais projetados para a educação.

PINTO, H. S.; STAAB, S.; TEMPICH, C. D. Diligent: Towards a fine-grained methodology for distributed, loosely-controlled and evolving engineering of ontologies. In: . [S.l.: s.n.], 2004. p. 393–397.

RICHARDSON, L.; AMUNDSEN, M.; RUBY, S. *RESTful Web APIs: Services for a Changing World*. O'Reilly Media, 2013. ISBN 9781449359737. Disponível em: <https://books.google.com.br/books?id=ZXDGAAAAQBAJ>.

ROBINSON, I.; WEBBER, J.; EIFREM, E. *Graph Databases*. [S.l.]: O'Reilly Media, Inc., 2013. ISBN 1449356265.

RODRIGUES, C. M. de O.; FREITAS, F. L. G. de; BARREIROS, E. F. S.; AZEVEDO, R. R. de; FILHO, A. T. de A. Legal ontologies over time: A systematic mapping study. *Expert Systems with Applications*, v. 130, p. 12 – 30, 2019. ISSN 0957-4174. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0957417419302398>.

SABOU, M.; WROE, C.; GOBLE, C.; MISHNE, G. Learning domain ontologies for web service descriptions: An experiment in bioinformatics. In: *Proceedings of the 14th International Conference on World Wide Web*. New York, NY, USA: Association for Computing Machinery, 2005. (WWW '05), p. 190–198. ISBN 1595930469. Disponível em: <https://doi.org/10.1145/1060745.1060776>.

SCHWITTER, R. Controlled natural languages for knowledge representation. In: *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*. USA: Association for Computational Linguistics, 2010. (COLING '10), p. 1113–1121.

SILVA, I. F. de L. A. *Chatterbot para Criaçaão e Refinamento de Ontologias em Lógica de Descrições*. Dissertação (Mestrado) — Cin/UFPE, Recife-PE, 2019.

SIMON, H. A. *The Sciences of the Artificial (3rd Ed.)*. Cambridge, MA, USA: MIT Press, 1996. ISBN 0-262-69191-4.

SIRIN, E.; PARSIA, B.; GRAU, B. C.; KALYANPUR, A.; KATZ, Y. Pellet: A practical owl-dl reasoner. *Web Semantics*, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 5, n. 2, p. 51–53, 2007. ISSN 1570-8268.

SUáREZ-FIGUEROA, M. C.; GOMEZ-PEREZ, A.; FERNáNDEZ-LóPEZ, M. The neon methodology for ontology engineering. In: _____. [S.l.: s.n.], 2012. p. 9–34. ISBN 9783642247934.

USCHOLD, M. Building Ontologies: Towards a Unified Methodology. In: *Proceedings of Expert Systems '96, the 16th Annual Conference of the British Computer Society Specialist Group on Expert Systems*. Cambridge, UK: [s.n.], 1996.

USCHOLD, M.; KING, M. Towards a methodology for building ontologies. In: *In Workshop on Basic Ontological Issues in Knowledge Sharing, held in conjunction with IJCAI-95*. [S.l.: s.n.], 1995.

VERDONCK, M.; GAILLY, F.; PERGL, R.; GUIZZARDI, G.; MARTINS, B.; PASTOR, O. Comparing traditional conceptual modeling with ontology-driven conceptual modeling: An empirical study. *Information Systems*, v. 81, p. 92 – 103, 2019. ISSN 0306-4379. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0306437918303727>.

VIEIRA, R.; LIMA, V. L. S. Linguistica computacional: princípios e aplicações. In: *Mini-curso das Jornadas de Atualização em Inteligência Artificial. In: Anais do Encontro Nacional de Inteligência Artificial*. [S.l.: s.n.], 2001.

VÖLKER, J. *Learning Expressive Ontologies: Volume 2 Studies on the Semantic Web*. Amsterdam, The Netherlands, The Netherlands: IOS Press, 2009. ISBN 1607500337, 9781607500339.

VÖLKER, J.; VRANDECIC, D.; SURE, Y.; HOTHO, A. Learning disjointness. In: *The Semantic Web: Research and Applications*. [S.l.]: Springer Berlin / Heidelberg, 2007. (LNCS), p. 175–189. ISBN 978-3-540-72666-1.

WALLACE, R. S. *ALICE A. I. Foundation*. 2017. <http://www.alicebot.org/>. Accessed: 2017-04-02.

WEIZENBAUM, J. Eliza a computer program for the study of natural language communication between man and machine. *Commun. ACM*, ACM, New York, NY, USA, v. 9, n. 1, p. 36–45, jan. 1966. ISSN 0001-0782. Disponível em: <http://doi.acm.org/10.1145/365153.365168>.

WHITE, C.; SCHWITTER, R. An update on peng light. *Proceedings of ALTA 2009*, 01 2009.

WIERINGA, R. *Design science methodology for information systems and software engineering*. [S.l.]: Springer, 2014. 10.1007/978-3-662-43839-8. ISBN 978-3-662-43838-1.

WILCOX, B.; WILCOX, S. Making it real: Loebner-winning chatbot design. *Arbor*, v. 189, p. a086, 12 2014.

WILKS, Y.; CATIZONE, R.; WORGAN, S.; DINGLI, A.; MOORE, R.; FIELD, D.; CHENG, W. A prototype for a conversational companion for reminiscing about images. *Comput. Speech Lang.*, Academic Press Ltd., GBR, v. 25, n. 2, p. 140–157, abr. 2011. ISSN 0885-2308. Disponível em: <https://doi.org/10.1016/j.csl.2010.04.002>.

WYNER, A.; ANGELOV, K.; BARZDINS, G.; DAMLJANOVIC, D.; DAVIS, B.; FUCHS, N.; HOEFLER, S.; JONES, K.; KALJURAND, K.; KUHN, T.; LUTS, M.; POOL, J.; ROSNER, M.; SCHWITTER, R.; SOWA, J. On controlled natural languages: Properties and prospects. In: FUCHS, N. E. (Ed.). *Controlled Natural Language*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010. p. 281–289. ISBN 978-3-642-14418-9.

ZAMBORLINI, V. C. *Estudo de Alternativas de Mapeamento de Ontologias da Linguagem OntoUML Para OWL: Abordagens Para Representação de Informação Temporal*. Dissertação (Mestrado) — Federal University of Espírito Santo, 2011. Disponível em: <http://www.inf.ufes.br/{~}rguizzardi/OEO/dissertacao-Veruska.p>.