



**UNIVERSIDADE FEDERAL DE SANTA CATARINA
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA**

ALAN VINICIUS CEZAR ENSINA

**Estudo comparativo de tecnologias de processamento de
linguagem natural para avaliação de histórias de usuário**

Florianópolis
2022

ALAN VINICIUS CEZAR ENSINA

**Estudo comparativo de tecnologias de processamento de
linguagem natural para avaliação de histórias de usuário**

Trabalho de conclusão de curso submetido ao curso de Sistemas de Informação para a obtenção do grau de Bacharel em Sistemas de Informação pela Universidade Federal de Santa Catarina – UFSC

Orientadora: Prof^a Dr^a Fabiane Barreto Vavassori Benitti

Co-orientador: Prof^o Mattheus da Hora França

Florianópolis
2022

ALAN VINICIUS CEZAR ENSINA

Estudo comparativo de tecnologias de processamento de linguagem natural para avaliação de histórias de usuário

Este Trabalho de conclusão de curso foi julgado aprovado para a obtenção do Título de “Bacharel em Sistemas de Informação” e aprovado em sua forma final pelo curso de Sistemas de Informação.

Florianópolis, dezembro de 2022

Profº Drº. Álvaro Junio Pereira Franco
Coordenador do Curso

Banca examinadora:

Profº Drª Fabiane Barreto Vavassori Benitti
Orientadora

Profº Fabiane Barreto Vavassori Benitti
Co-orientador

Profº Dr Elder Rizzon Santos

RESUMO

Histórias de usuário são as representações das necessidades de um usuário e são utilizadas para facilitar o entendimento entre a equipe de negócios e a equipe de desenvolvimento para obter um maior acerto no desenvolvimento do produto com base na especificação. Porém, devido ao fato de serem escritas de maneira simples e curtas, diversas vezes podem causar dúvidas no momento da implementação. Sendo assim, é necessário encontrar uma forma de automatizar a avaliação dessas histórias afim de obter uma maior completude, uniformidade e consistência. Se tratando de automatização, o Processamento de Linguagem Natural (PLN) é uma subárea da inteligência artificial capaz de compreender automaticamente línguas humanas naturais capaz de automatizar diversos processos, porém devido ao alto número de tecnologias de PLN presente hoje no mercado, ainda é necessário compará-las para que seja possível aferir qual tecnologia possui, por exemplo, uma maior exatidão em seus processamentos, melhor performance e qual é a mais adequada no contexto de histórias de usuário. O presente trabalho pretende realizar uma análise comparativa entre soluções de PLN para a avaliação de histórias de usuário.

Palavras-chave: engenharia de software, histórias de usuário, processamento de linguagem natural, PLN

ABSTRACT

User Stories are representations of a user's needs and are used to help the understanding between the business team and the development team to achieve greater accuracy in product development based on the specification. However, due to the fact that they are written in a simple and short way, they can often cause doubts at the time of implementation. Therefore, it is necessary to find a way to automate the evaluation of these stories in order to obtain greater completeness, uniformity and consistency. When it comes to automation, Natural Language Processing (NLP) is a subarea of artificial intelligence capable of automatically understanding natural human languages capable of automating several processes, but due to the high number of NLP technologies present on the market, it is still necessary to compare them so that it is possible to assess which technology has, for example, greater accuracy in its processing, better performance and which is the most appropriate in the context of user stories. The present work intends to carry out a comparative analysis between NLP solutions for the evaluation of user stories.

Keywords: software engineering, user stories, natural language processing, NLP

LISTA DE FIGURAS

Figura 01 - Critérios de qualidade definidos por Heck e Zaidman	16
Figura 02 - Critérios de qualidade definidos por Lucassen, et. al. (2016)	17
Figura 03 - Etapas de análise em um processamento de linguagem natural (INDURKHYA; DAMERAU, 2010)	23
Figura 04 - Processo de execução da pesquisa e análise das tecnologias	32
Figura 05 - Tela inicial do da API do Swagger	40
Figura 06 - Endpoint responsável para avaliação de histórias de usuário utilizando o template de Cohn	40
Figura 07 - Exemplo de resposta na avaliação de história de usuário utilizando o template de Cohn.....	41
Figura 08 - Endpoint responsável para avaliação de histórias de usuário utilizando o template orientado a cenário.....	42
Figura 09 - Exemplo de resposta na avaliação de história de usuário utilizando o template orientado a cenário	43

LISTA DE TABELAS

Tabela 01 - Etapas e metodologias aplicadas	14
Tabela 02 - Template de história de usuário definido por Cohn(2009)	15
Tabela 03 - Critérios de qualidade definidos por autor	17
Tabela 04 - Tabela comparativa entre Requirement Smells e critérios de qualidade	18
Tabela 05 - Template de comportamento orientado a cenário	20
Tabela 06 - Exemplo de cenários (RODRIGUES, 2020)	21
Tabela 07 - Tipos de tokenização e exemplo	24
Tabela 08 - Lista de tecnologias que atendem aos critérios de inclusão	33
Tabela 09 - Lista das tecnologias com abordagem e etapas	35
Tabela 10 - Lista das tecnologias com suas características	36
Tabela 11 - Descrição dos atributos retornados após processamento da história de usuário	41
Tabela 12 - Cronograma das próximas etapas	43

LISTA DE ABREVIATURAS E SIGLAS

PLN - Processamento de Linguagem Natural	9
NLTK - Natural Language Toolkit	11
API - Application Programming Interface	11
XP - Extreme Programming	15
IEEE - Institute of Electrical and Electronics Engineers	16
BDD - Behavior-Driven Development	20
QP - Questão de pesquisa	30
RH - Requisito para histórias de usuário.....	X
RC - Requisito para cenários.....	X
JSON - JavaScript Object Notation	40

SUMÁRIO

1. INTRODUÇÃO	9
1.1 Problema	11
1.2 Solução proposta	12
1.3 Objetivos	13
1.3.1 Objetivo geral	13
1.3.2 Objetivos específicos	13
1.4 Metodologia	13
2. FUNDAMENTAÇÃO TEÓRICA	15
2.1 Histórias de usuário	15
2.2 Critérios de qualidade	16
2.3 Critérios de aceitação	20
2.4 Processamento de linguagem natural	21
2.4.1 Abordagem clássica	22
2.4.1.1 Pré-processamento de texto	23
2.4.1.1.1 Tokenização	24
2.4.1.1.2 Segmentação de frase	25
2.4.1.2 Análise léxica	25
2.4.1.3 Análise sintática	26
2.4.1.4 Análise semântica	26
2.4.2 Abordagem estatística	27
2.4.2.1 Part-of-Speech Tagging	29
3. ESTUDO COMPARATIVO	30
3.1 Método de pesquisa	30
3.1.1 Questões de pesquisa	30
3.1.2 Processo de busca	31
3.1.3 Critérios de inclusão e exclusão	31
3.2 Execução	32
3.3 Resultados	32
3.3.1 QP1: Quais são as tecnologias presentes no mercado?	33
3.3.2 QP2: Como a tecnologia é classificada dentro do contexto do PLN?	35
3.3.3 QP3: Quais são suas características?	36
3.4 Considerações finais	37
4. DESENVOLVIMENTO	38
4.1 Requisitos do sistema	38
4.2 Design da interface	39
4.3 Considerações finais	43
4.3.1 Cronograma	43
5. REFERÊNCIAS	45

1. INTRODUÇÃO

“Tempo é dinheiro” (FRANKLIN, 1748) famosa frase dita por Benjamin Franklin na metade do século 18 ainda ecoa na cabeça de muitos seres humanos. Em busca de mais tempo as pessoas procuram então otimizar suas tarefas. Uma forma de otimizar as tarefas é a criação de automações. As automações buscam por uma melhor produtividade, redução de custos e maior tempo livre para se concentrar em outras tarefas que não podem ser automatizadas. Silva (2019) define que “... automação é um dos processos mais utilizados para a facilitação de inserção dos recursos tecnológicos. Através dessa tecnologia, são utilizadas ferramentas para soluções tecnológicas com o objetivo de otimizar e tornar simples os processos internos, além de diminuir custos operacionais.”

Um grande exemplo disso são as assistentes virtuais, como por exemplo a Alexa da Amazon, a Siri da Apple e o Google Home do Google. Esses assistentes virtuais são capazes de realizar diversas tarefas através de um simples comando de voz. Essa interação entre seres humanos e máquinas está cada vez mais presente nos sistemas, mas para que isso seja possível, é utilizado o Processamento de Linguagem Natural - PLN (RACKSPACE TECHNOLOGY, 2020).

Johnson (2021) define o PLN sendo um ramo dentro da Inteligência Artificial responsável em fazer com que as máquinas possam compreender a linguagem dos seres humanos, ou seja, o PLN funciona como um tradutor, permitindo assim que as tecnologias possam entender seus usuários, mesmo eles utilizando a linguagem natural.

O PLN também está presente em outras plataformas além das assistentes virtuais. Por exemplo, ele auxilia em sites de busca realizando interpretações entre o que o usuário digita com conteúdos de sites que poderão ser exibidos. Também está presente no auto-completar em plataformas de busca, onde sugestões automáticas são exibidas na tela no momento em que o usuário está digitando. Chatbots, que são utilizados por empresas para se comunicar com seus clientes, também fazem uso do PLN realizando a “tradução” do que o cliente deseja com possíveis soluções das quais as empresas podem oferecer (TAKE BLIP, 2019).

Para que seja possível criar sistemas voltados a automações, é necessário levantar os requisitos que esse sistema irá possuir. Em engenharia de requisitos, a etapa responsável para o levantamento dessas informações é a elicitacão. Para Thayer (1997), a elicitacão de requisitos é o processo em que os clientes e usuários são questionados pela equipe de desenvolvimento a falarem o quê espera como funcionalidades no sistema que será desenvolvido. Nessa etapa de elicitacão serão definidas as exigências, os recursos, os objetivos e as utilidades que o sistema deve cumprir.

Segundo Sommerville(2011, pág. 57):

Os requisitos de um sistema são as descrições do que o sistema deve fazer, os serviços que oferece e as restrições de seu funcionamento. Esses requisitos refletem as necessidades dos clientes para um sistema que serve a uma funcionalidade determinada, como controlar um dispositivo, colocar um pedido ou encontrar informações. O processo de descobrir, analisar, especificar e verificar esses serviços e restrições é chamado engenharia de requisitos.

A especificacão de requisitos no desenvolvimento ágil pode ser feito por meio de histórias de usuário (User Stories). Através delas, o usuário utiliza de uma abordagem de escrever sobre os requisitos, tudo isso por meio de uma ou duas frases escritas na perspectiva de quem deseja o recurso/funcionalidade.

Para Cohn (2009, pág. 4), “uma história de usuário descreve a funcionalidade que será valiosa para um usuário ou comprador de um sistema ou software”. Já Rehkopf (2020) define histórias de usuário como “uma explicacão informal e geral sobre um recurso de software escrita a partir da perspectiva do usuário final. Seu objetivo é articular como um recurso de software pode gerar valor para o cliente.”

As técnicas de PLN também podem oferecer diversas vantagens para melhorar a qualidade das histórias de usuário. Segundo Raharjana, Siahaan e Fatichah (2021):

As técnicas de processamento de linguagem natural (PLN) oferecem vantagens potenciais para melhorar a qualidade das histórias de usuários. O PLN pode ser usado para analisar ou extrair os dados da história do usuário. Tem sido amplamente utilizado para ajudar no domínio da engenharia de software (por exemplo,

gerenciamento de requisitos de software, extração de atores e ações no documento de requisitos, teste de software, etc.).

1.1 Problema

Cohn (2009) comenta que ao definir os requisitos de software a comunicação pode ser uma adversidade, pois aqueles que desejam um novo software devem se comunicar com quem irá desenvolvê-lo.

Heck (2014) propõe critérios específicos para avaliar a qualidade em histórias de usuários: completude, uniformidade, consistência e correção. Porém, muitos desses critérios, no entanto, requerem informações complementares que não são capturadas em um texto de história do usuário. Femmer (2013) define o termo *Requirement Smell* como indicador de má qualidade na especificação de requisitos. Femmer (2014) subdivide o *Requirement Smell* em 9 tipos: ambiguidade de advérbios e adjetivos, pronomes vagos, linguagem subjetiva, comparações, superlatividade, afirmações negativas, termos não verificados, *loopholes* (*brechas*) e referências não verificadas.

Dentro do contexto de histórias de usuário, seria possível avaliá-las utilizando soluções de PLN levando em consideração os critérios de qualidade?

Atualmente existem inúmeras soluções utilizadas para o PLN. Parker(2019) em seu artigo cita 12 ferramentas open source em diversas linguagens de programação, como por exemplo Python, Node e Java. Dentre as soluções citadas por Parker (2019), destaca-se a Natural Language Toolkit (NLTK) em Python, por ser a solução com mais recursos disponíveis, capaz de implementar todas as etapas de PLN e oferece suporte a vários idiomas. Outra solução que se destaca é a OpenNLP em Java. É hospedada pela Apache Foundation, ou seja, é fácil integrá-la com outros serviços da Apache. Assim como a NLTK, oferece suporte a vários idiomas e cobre todas as etapas de PLN.

No entanto, ainda é necessário compará-las para que seja possível aferir qual tecnologia possui, por exemplo, uma maior exatidão em seus processamentos e qual possui a melhor performance. Sendo assim, levando

em consideração os critérios de qualidade (completude, uniformidade e consistência), qual a solução mais adequada para o PLN no contexto de histórias de usuário?

Neste sentido, o objetivo central desse trabalho é realizar um estudo comparativo entre pequenas soluções utilizando PLN para avaliar a qualidade de histórias de usuário nos idiomas português e inglês.

1.2 Solução proposta

Com base no cenário atual, no qual existem diversas tecnologias voltadas para PLN, se faz necessário uma análise comparativa entre essas tecnologias afim de definir qual ou quais tecnologias são mais adequadas para a avaliação de histórias de usuário.

Para que isso seja possível, serão selecionadas algumas tecnologias para que sejam previamente avaliadas em aspectos como por exemplo: documentação, linguagem de programação, conteúdo disponível na internet a respeito da tecnologia (sites, fóruns e *threads* em redes sociais) e o uso atual no mercado. Após feito esse levantamento de dados, as tecnologias que mais se destacarem serão selecionadas como objetos de estudo e será implementado uma API (Application Programming Interface) que será capaz de avaliar histórias de usuário utilizando as tecnologias de PLN selecionadas.

Essa API avaliará as histórias de usuário levando em consideração critérios de qualidade em requisitos de software, como por exemplo eficiência, acurácia e funcionalidades. Quanto as histórias de usuário, serão selecionados alguns critérios de qualidade, conforme literatura, para avaliação.

Para além dos critérios de qualidade dos requisitos, pretende-se também avaliar nas tecnologias utilizadas os aspectos relacionados a eficiência no processamento para os idiomas inglês e português e também a produtividade.

1.3 Objetivos

Nessa sessão, serão expostas o objetivo geral e os objetivos específicos deste trabalho.

1.3.1 Objetivo geral

Desenvolver um estudo comparativo entre soluções de PLN com o propósito de avaliar qual ou quais tecnologias são mais adequadas para analisar critérios de qualidade em requisitos de software descritos como história de usuário.

1.3.2 Objetivos específicos

- Analisar e avaliar soluções atuais no mercado, comparando-as dentro dos critérios preestabelecidos;
- Implementar uma API voltada para a avaliação de histórias de usuário seguindo o template de Cohn(2009) e histórias de usuário seguindo o template de Gherkin orientado a cenários, utilizando as duas soluções de PLN mais bem avaliadas;
- Avaliar os as tecnologias utilizadas

1.4 Metodologia

Metodologia é a estrutura filosófica dentro da qual a pesquisa é conduzida ou a base sobre a qual a pesquisa se baseia (BROWN, 2006). Já O'Leary (2004) descreve a metodologia como a estrutura que está associada a um conjunto particular de suposições paradigmáticas usadas para conduzir a pesquisa.

Sendo assim, dado o contexto de metodologia, o estudo seguirá o modelo científico em camadas (Research Onion) de Saunders (2007), seguindo a forma transversal, indutiva e interpretativa. Seguirá um modelo multimétodo, com procedimento de pesquisa bibliográfica (GIL, 2010), estudo comparativo das tecnologias (FACHIN, 2001), design e prototipação

(SOMMERVILLE, 2011) e Goal Question Metric (GQM) (BASILI, CALDIERA, ROMBACH, 1994).

Etapas	Atividades	Métodos	Resultados
Etapas 1 - Síntese da fundamentação teórica	- Sintetizar contexto histórico de processamento de linguagem natural, de histórias de usuário e de critérios de qualidade	Pesquisa bibliográfica (GIL, 2010)	Fundamentação teórica
Etapas 2 - Estudo comparativo	- Pesquisar as tecnologias mais utilizadas - Análise das tecnologias conforme os requisitos preestabelecidos - Definir as duas tecnologias mais promissoras	Estudo comparativo (FACHIN, 2001)	Análise comparativa de potenciais soluções
Etapas 3 - Prototipação	- Implementar protótipos da API com as soluções A e B em português e inglês	- Design e prototipação (SOMMERVILLE, 2011)	Protótipos da API das tecnologias selecionadas
Etapas 4 - Avaliação comparativa	- Avaliar e comparar os resultados obtidos das tecnologias A e B	- GQM (BASILI et al., 1994)	Avaliação das tecnologias e tabela comparativa

Tabela 01 - Etapas e metodologias aplicadas

2. FUNDAMENTAÇÃO TEÓRICA

Com o objetivo de contextualizar os temas abordados neste estudo, neste capítulo são introduzidos um embasamento teórico sobre histórias de usuário, critérios de qualidade e processamento de linguagem natural.

2.1 Histórias de usuário

Segundo Francino (2017), o conceito de histórias de usuário foi introduzido pela primeira vez em 1998 na XP (Extreme Programming) comparando-as com Casos de Uso. Com o aumento da popularidade da XP e do Scrum, as histórias de usuário se tornaram uma abordagem muito conhecida para a definição de requisitos.

Uma história de usuário pode ser descrita como uma frase curta e semiestruturada capaz de ilustrar os requisitos de um software na perspectiva do usuário, ou seja, pode ser usada para identificar o desejo do usuário com relação ao produto (RAHARJANA, HARRIS, JUSTITIA. 2020).

Wautelet, et al. (2017) definem que uma história de usuário consiste de quatro elementos:

- **Papel:** comportamento esperado do ator no contexto do problema
- **Objetivo:** condição desejada pelas partes interessadas
- **Tarefa:** obrigação específica que devem ser realizadas a fim de atingir os objetivos
- **Capacidade:** a habilidade dos atores em atingir as metas com base em certas condições ou eventos

Já para Cohn(2009), para facilitar a escrita de histórias de usuário, ele sugere o seguinte template:

Template	<i>“Como <tipo de usuário>, quero <algum objetivo> para que <algum motivo>”</i>
Exemplo	<i>“Como cliente, quero utilizar a forma de pagamento por pix para que eu possa pagar minha compra.”</i>

Tabela 02: Template de história de usuário definido por Cohn (2009)

Cohn (2009) também afirma que mais importante do que escrever histórias de usuários é a discussão a respeito dela, sendo assim, sugere que elas devem ser escritas em pequenos papéis ou até mesmo em notas adesivas, para que sejam facilmente expostas em paredes ou murais para facilitar o planejamento e a discussão.

2.2 Critérios de qualidade

Ao realizar as especificações de requisitos de software, o IEEE recomenda 9 características bases para qualidade de requisitos: necessária, apropriada, não ambígua, completa, singular, praticável, verificável, correta e conforme (IEEE Computer Society, 2018).

Se tratando de critérios de qualidade voltados a requisitos de software, Heck e Zaidman (2014) desenvolveram um framework voltado a verificação de requisitos ágeis, neste framework, foram definidos três critérios de qualidade para verificação de alto nível:

Completeness: Todos os elementos necessários na Área de Negócios (*Business Area*) devem estar presentes.

Uniformidade: O estilo dos elementos da Área de Negócios deve ser padronizado.

Consistência e correção. Todos os elementos devem estar em conformidade com a propriedade objeto da certificação.

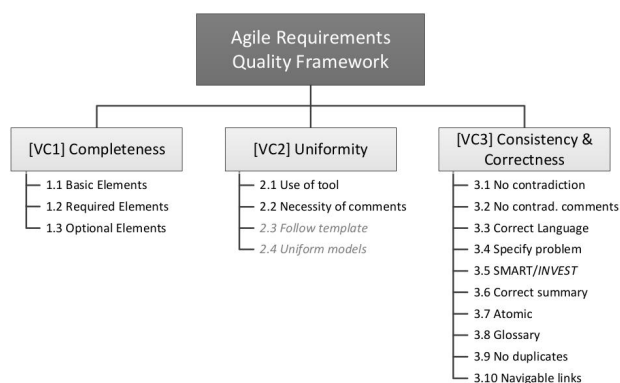


Figura 01: Critérios de qualidade definidos por Heck e Zaidman (2014)

Na Figura 01, é possível observar como cada critério de qualidade é subdividido, os itens em itálico são aplicados apenas para histórias de

usuário, já o restante, são aplicados tanto para histórias de usuário quanto para solicitações de requisitos de software.

Lucassen, et. al. (2016) em seu estudo desenvolve um framework voltado a avaliação de histórias de usuário utilizando processamento de linguagem natural. Neste framework são utilizados 13 critérios de qualidade a serem validadas, sub-divididas em 3 grupos: sintáticas, semânticas e pragmáticas.



Figura 02: Critérios de qualidade definidos por Lucassen, et. al. (2016)

Se tratando de critérios de qualidade definidos pela IEEE (IEEE Computer Society, 2018), Heck e Zaidman (2014) e Lucassen, et. al. (2016), nota-se que há muitos critérios com o mesmo significado, porém com nomenclaturas diferentes, sendo assim, segue abaixo uma tabela comparativa destacando os critérios definidos por cada autor juntamente com os critérios estabelecidos pela IEEE (IEEE Computer Society, 2018).

IEEE Computer Society (2018)	LUCASSEN et. al. (2016)	Heck e Zaidman (2014)
Necessária		
Apropriada	Sem conflito	
Não ambígua	Não ambígua	
Completa	Completa	Compleitude
Singular	- Única	

	- Independente - Atômica	
Praticável	Estimável	
Verificável		
Correta	Bem formada	Consistência e correção
Conforme	Uniforme	Uniformidade
	Mínima	
	Orientada ao problema	
	Sentença completa	
	Conceito sólido	

Tabela 03 - Critérios de qualidade definidos por autor

Por outro lado, atualmente existem diversos estudos voltados a indicadores de má qualidade na especificação de requisitos, sendo assim, Nascimento et. al (2018) apresenta em seu estudo um mapeamento sistemático de literaturas que investigam a existência de indicadores de má qualidade que podem prejudicar negativamente a compreensão, manutenção e qualidade dos artefatos. Estes indicadores são descritos pelo autor como *Requirement Smells*.

No mapeamento sistemático apresentado por Nascimento et. al. (2018), 41 estudos são analisados desde 2013 onde 9 tipos de *Requirement Smells* são citados. Segue abaixo uma tabela comparativa dos *Requirement Smells* citados e o critério de qualidade oposto definido pela IEEE (IEEE Computer Society, 2018):

Requirement Smells (Nascimento et. al. (2018))	Descrição do Requirement Smell (Nascimento et. al. (2018))	Critérios de qualidade (IEEE Computer Society, 2018)
Advérbios e Adjetivos Ambíguos	Adjetivos e advérbios que causam ambiguidade na compreensão dos requisitos. Exemplo: Se a qualidade for muito baixa , uma falha deve ser gravada na memória de erros.	Não ambígua

Pronomes vagos	São pronomes com relações pouco claras. Exemplo: O software deve implementar serviços para aplicativos, que devem se comunicar com os aplicativos do controlador implantados em outros controladores.	Completa
Linguagem subjetiva	São palavras cuja semântica não é objetiva. Exemplos: amigável, fácil de usar, econômico.	Apropriada
Comparações específicas	São advérbios e adjetivos, onde os requisitos expressam uma relação do sistema com outros sistemas específicos. Exemplo: melhor que, maior qualidade.	Singular
Advérbios e adjetivos superlativos	São advérbios e adjetivos, onde os requisitos expressam uma relação do sistema com todos os outros sistemas. Exemplo: melhor desempenho, menor tempo de resposta.	Apropriada
Afirmações negativas	São palavras usadas em funcionalidades que o sistema não deve fornecer, pois podem levar a falta de explicação sobre o comportamento do sistema em tais casos. Exemplo: o sistema não deve aceitar cartões de crédito VISA.	Apropriada
Termos não verificáveis	São palavras difíceis de verificar por oferecer várias possibilidades de execução do sistema. Exemplo: O sistema só pode ser ativado se todos os sensores necessários (...) trabalharem com precisão de medição suficiente.	Verificável
Loopholes	São palavras que possibilitam os stakeholders ignorar as especificações. Exemplos: se possível, conforme apropriado, conforme aplicável.	Conforme
Referências incompletas	São referências que o leitores não conseguem encontrar	Completa

Tabela 04 - Tabela comparativa entre Requirement Smells e critérios de qualidade

2.3 Critérios de aceitação

Após definido os critérios de qualidade para se avaliar uma história de usuário, deve-se também definir os critérios de aceitação para validar se a história escrita atende ao objetivo proposto.

Segundo Rahate (2021):

O conceito de Critérios de Aceitação vem junto com as Histórias de Usuários da Extreme Programming. Normalmente, uma história de usuário é uma breve descrição de 3 linhas que expressa um requisito da perspectiva do cliente. Para quaisquer detalhes adicionais, os desenvolvedores colaborariam com a empresa (cliente) e entenderiam mais. Durante essa conversa, eles (desenvolvedores e clientes) concordam com o comportamento aceitável do requisito. Essa confirmação acordada do comportamento dos requisitos é chamada de Critérios de Aceitação.

Em engenharia de software o BDD (Behavior-Driven Development), ou, Desenvolvimento Orientado a Comportamento, é um processo de desenvolvimento de software ágil que incentiva a colaboração entre desenvolvedores e testadores na garantia de qualidade em um projeto de software (NORTH, 2006).

Oliveira (2017) define que o BDD é um conjunto de práticas que reúnem analistas de negócios, desenvolvedores e testadores para definir de forma colaborativa e através de cenários quais os requisitos serão executados.

Estes cenários são expressos em um formato conhecido como Gherkin, projetado para ser facilmente compreensível pelas partes interessadas. Cada cenário é composto por uma série de etapas onde cada etapa começa com uma palavra-chave. A ordem do cenário é definida quando é **dado** uma pré-condição, **quando** a ação será executada e **então** o resultado que é esperado (OLIVEIRA, 2017).

Template	Dado <pré-condição>, quando <ação que será executada>, então <resultado esperado>
Exemplo	Dado que esteja logado no sistema, quando eu clicar em um produto, então a página será redirecionada aos detalhes do produto.

Tabela 05: Template de comportamento orientado a cenário

Rodrigues (2020) na Tabela 06 descreve dois cenários utilizando como exemplo a troca de luzes de um semáforo:

Funcionalidade: Alteração automática de luzes de um semáforo
Como um pedestre
Quero que as luzes do semáforo fiquem vermelhas
Assim que eu me aproximar da faixa de pedestre
Cenário 1: Alteração das luzes de verde para vermelho Dado que a luz do semáforo esteja verde Quando uma pessoa se aproximar da faixa de pedestre Então a luz do semáforo deve ficar vermelha
Cenário 2: Alteração das luzes de vermelho para verde Dado que a luz do semáforo esteja vermelho Quando não houver pessoas na faixa de pedestres E não houver pessoas se aproximando da faixa de pedestre Então a luz do semáforo deve ficar verde Mas somente até pessoas se aproximem da faixa de pedestre

Tabela 06: Exemplo de cenários (RODRIGUES, 2020)

2.4 Processamento de linguagem natural

Após a segunda guerra mundial, as pessoas notaram a necessidade em traduzir informações de um idioma para outro, sendo assim, esperavam automatizar esse processo através de uma máquina capaz de realizar essas traduções automaticamente, foi então o início dos estudos na área de processamento de linguagem natural (ROBERTS, 2004).

O processamento de linguagem natural (PLN) é uma disciplina que combina linguística, ciência da computação e inteligência artificial para estudar as interações entre sistemas de computador e linguagem natural humana (FERRARIO et. al., 2020).

O uso de técnicas de PLN está presente em uma variedade de aplicações do mundo real em vários campos, incluindo pesquisa médica, mecanismos de pesquisa e inteligência de negócios (LUTKEVICH, 2021). O

PLN pode ser dividido em duas abordagens: clássica e estatística. Porém, mesmo com essa divisão não significa necessariamente que uma é superior a outra. Um exemplo disso é que na abordagem estatística requer uma grande quantidade de dados rotulados no idioma desejado, ou seja, é mais viável utilizar quando o conjunto de dados é vasto. Já na abordagem clássica, não há uma necessidade de estar presa a um só idioma, pois há uma sequência de passos pré-definidos e se faz necessário apenas o conhecimento sobre a estrutura do idioma utilizado (BAASCH, 2021) (INDURKHYA; DAMERAU, 2010).

2.4.1 Abordagem clássica

Tradicionalmente, o PLN na abordagem clássica tende a ser um processo decomposto em etapas, sendo elas espelhadas em distinções linguísticas teóricas entre sintáticas, semânticas e pragmáticas (INDURKHYA; DAMERAU, 2010).

Durante esse processo o texto é dividido em sentenças onde a sintaxe dos termos são analisadas, buscando produzir uma estrutura mais amigável à análise semântica. Por fim, uma análise pragmática é realizada com o propósito de avaliar se a palavra ou sentença possuem sentido dentro do contexto aplicado (BAASCH, 2021) (INDURKHYA; DAMERAU, 2010).

Com o conhecimento disponível hoje, a abordagem clássica foi refinada e decomposta em: tokenização, análise léxica, análise sintática, análise semântica e análise pragmática (BAASCH, 2021) (INDURKHYA; DAMERAU, 2010). Segue abaixo na Figura 03, uma ilustração das etapas do PLN na abordagem clássica, onde o texto a ser processado é recebido, passa etapa de tokenização, análise léxica, análise sintática, análise semântica e análise pragmática, para que no fim a máquina tenha o entendimento do texto recebido no início do processo.

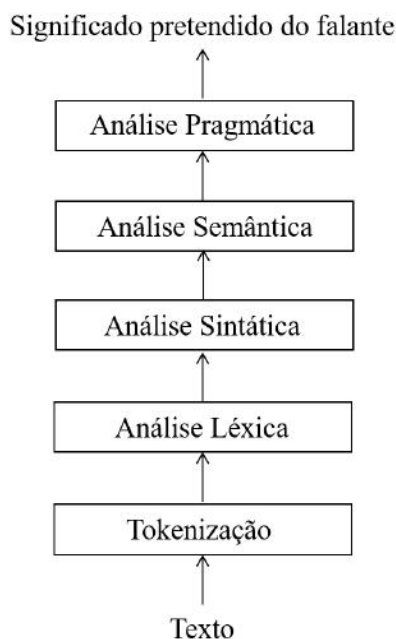


Figura 03: Etapas de análise em um processamento de linguagem natural (INDURKHYA; DAMERAU, 2010)

2.4.1.1 Pré-processamento de texto

Antes de inicializar o processamento do texto, esse texto recebido como entrada deve ser tratado, afim de identificar erros que prejudiquem as análises. O pré-processamento de texto é uma etapa essencial no processamento de linguagem natural, pois as palavras e sentenças serão utilizadas como base das etapas subsequentes (BAASCH, 2021) (INDURKHYA; DAMERAU, 2010).

Para que a análise seja eficiente, é preciso definir quais serão os caracteres utilizados pelo texto, juntamente das palavras e sentenças. Nesta etapa, deve-se realizar um processo de limpeza que consiste em identificar a codificação do texto e convertê-la para a codificação que será utilizada, também deve-se remover as imagens presentes no texto, links, tags HTML ou qualquer outro elemento que não traga valor ao texto que será processado (BAASCH, 2021) (INDURKHYA; DAMERAU, 2010).

2.4.1.1.1 Tokenização

Tokenização é um processo fundamental dentro do PLN, separando o texto em pedaços menores, o então chamados *tokens*. Os *tokens* podem ser classificados em três tipos: tokenização de palavras, tokenização de caracteres e tokenização de subpalavras (PAI, 2020).

A maneira mais comum de criar *tokens* é baseada em espaços em branco. Por exemplo, na frase: “Nunca desista”, assumindo o espaço como um delimitador, cada palavra será um *token*, logo são identificados dois *tokens* para esta frase (PAI, 2020).

Exemplo: Deep Learning		
Tokenização por palavra	Tokenização por caracter	Tokenização por subpalavra
Deep, Learning (2 <i>tokens</i>)	D,e,e,p,L,e,a,r,n,i,n,g (12 <i>tokens</i>)	Deep, Learn, ing (3 <i>tokens</i>)

Tabela 07 - Tipos de tokenização e exemplo

Pode-se observar na Tabela 07 como os três tipos de tokenização se comportam. Como exemplo foi dado o texto “*Deep Learning*”, onde na tokenização por palavra utilizando o espaço como delimitador, sendo assim foram gerados dois *tokens*. No segundo exemplo, na tokenização por caracter, cada caracter de cada palavra gera um novo *token*, portanto, doze *tokens* foram gerados. No terceiro exemplo, na tokenização por subpalavra, foram gerados três *tokens*, pois ‘*ing*’ juntamente de um verbo no idioma inglês determina o gerúndio de uma ação.

O processo de tokenização pode sofrer algumas dificuldades, pois nem sempre a delimitação das palavras é feita apenas com espaços, a delimitação pode ocorrer também com sinais de pontuação, por exemplo: pontos, vírgulas, aspas, hífens e outras marcações. Essas pontuações podem gerar ambiguidade no momento da criação dos *tokens*, pois podem delegar funções diferentes em uma frase. Sendo assim, o tokenizador utilizado deve estar preparado para receber sinais de pontuação e determinar

quando um sinal faz parte do token ou quando é apenas um símbolo (BAASCH, 2021) (INDURKHYA; DAMERAU, 2010).

2.4.1.1.2 Segmentação de frase

Na maioria dos idiomas, sinais de pontuação delimitam o fim de uma frase, porém essa regra nem sempre é bem delimitada, fazendo com que o segmentador de frases possa ter um mal entendimento em quando uma frase foi finalizada ou não. A complexidade da execução desta etapa está totalmente dependente do idioma utilizado pelo segmentador (BAASCH, 2021) (INDURKHYA; DAMERAU, 2010).

Sendo assim, tende a ser arbitrária a definição do que constitui uma frase, ficando em muitos casos a cargo do desenvolvedor que está implementando o sistema definir quais serão as regras a serem utilizadas para limitar uma frase. Entretanto, os sistemas que utilizam PLN, em sua grande maioria, utiliza um método que consiste em verificar espaços seguidos por uma palavra iniciada com letra maiúscula seguido até um ponto final, interrogação, exclamação entre outros pontos, para delimitar o início e fim de uma frase (BAASCH, 2021) (INDURKHYA; DAMERAU, 2010).

2.4.1.2 Análise léxica

O processo de decompor um texto em palavras, frases e outros elementos significativos é também definido como análise léxica. Nesta análise é baseada ao nível de palavra, ou seja, o foco é no significado das palavras, frases, e outros elementos, como os símbolos. Em alguns momentos, a análise léxica é vagamente descrita como um processo de tokenização (THANAKI, 2017).

Na análise léxica, duas etapas são muito comuns: *lemming* e *stemming*:
- *Lemming*: nesta etapa são relacionadas diferentes ocorrências morfológicas de uma determinada palavra em uma única forma, ou seja, a forma mais básica de uma palavra, em outros termos, seu radical: *lema*. Exemplo: comido, comeu, comendo, são do mesmo *lema* comer (BAASCH, 2021) (INDURKHYA; DAMERAU, 2010).

- *Stemming*: é uma etapa mais bruta, onde o final de uma palavra é removido com o objetivo de reduzi-la a forma mais básica para encontrar outras ocorrências dessa palavra em diferentes formatos ao longo do texto (STANFORD, 2008).

Ambas as etapas compartilham do mesmo objetivo que é reduzir a palavra para sua forma mais básica. Porém, no *lemming*, geralmente necessita de ferramentas adicionais que lidem somente com essa tarefa, onde requer mais processamento, fazendo com que esse processo seja mais útil em sistemas mais robustos. No caso de aplicações mais simples, é mais aconselhável a utilização do *stemming* por ser mais rápido, porém o seu uso pode causar perda de informação dependendo do que for removido de uma palavra, visto que é um método mais bruto (BAASCH, 2021).

2.4.1.3 Análise sintática

Na análise sintática, é feita uma análise gramatical no texto sobre uma sequência de palavras fornecidas, de modo comum sendo um frase, onde é gerado uma estrutura de acordo com a gramática escolhida, esta estrutura é utilizada a fim de atribuir um significado. As palavras fornecidas normalmente serão processadas nas etapas de tokenização e análise léxica. Nesta etapa também é atribuído *tags*. As *tags* facilitam quando uma informação pertinente precisa ser extraída do texto (BAASCH, 2021) (INDURKHYA; DAMERAU, 2010).

2.4.1.4 Análise semântica

Encontrar significado para o texto é o papel da análise semântica. Nesta etapa, o computador tem o poder de entender e interpretar frases, parágrafos ou documentos inteiros, analisando sua gramática e identificando as relações entre as palavras de uma frase em um contexto específico. Sendo assim, o objetivo principal da análise semântica é extrair do texto o significado exato ou o significado do dicionário de palavras (GOYAL, 2021).

Nesta etapa, como o texto já foi tratado pelas etapas anteriores, tem como objetivo compreender o significado do texto analisado. Conforme a

implementação realizada, pode-se também realizar a extração de certas informações com o objetivo de adquirir algum conteúdo que possa ser relevante ao usuário, sendo assim poupando-o de que ele tenha que realizar a leitura completa ou compreender todo o conteúdo que o texto possa oferecer. Também pode-se utilizar a extração de informação para a realização de resumos automáticos, mineração de dados e tradução automática. Esta etapa também pode ser considerada uma análise pragmática, pois busca compreender o significado da sentença fornecida (BAASCH, 2021) (INDURKHYA; DAMERAU, 2010).

Conforme a análise for mais robusta e refinada, mais simples se torna a compreensão dos dados de entrada fornecidos pelo usuário, tornando assim mais eficiente a Interação Humano-Computador (BAASCH, 2021) (INDURKHYA; DAMERAU, 2010).

2.4.2 Abordagem estatística

A abordagem estatística para o PLN utiliza de técnicas de aprendizado de máquina, onde para desenvolver um sistema capaz de processar com linguagem natural são utilizados conjuntos de dados com um enorme número de registros, esse conjunto de dados é fornecido a um algoritmo que busca padrões dentre os dados. Ao encontrar padrões, eles passam a fazer parte de um modelo que possui a capacidade de compreender os dados de entrada fornecidos para o processamento de linguagem natural (BAASCH, 2021) (INDURKHYA; DAMERAU, 2010).

O domínio estatístico muitas vezes leva o PLN a ser descrito como Processamento Estatístico de Linguagem Natural, para que não haja confusão ao associá-lo aos métodos da abordagem clássica (BROWNIEE, 2017).

A popularidade da abordagem estatística tem aumentado nos últimos anos por não necessitar de conhecimentos tão especializados, pois não exige uma análise muito aprofundada, sendo necessário apenas possuir uma quantidade de dados e uma classificação correta dos dados utilizados no treinamento (BAASCH, 2021).

2.4.2.1 Part-of-Speech Tagging (POS Tagging)

Seja na abordagem clássica ou na abordagem estatística, o uso de *tags* para associar uma palavra em uma sentença à sua classe gramatical é indispensável (FERNANDES, 2022).

Uma das principais etapas no PLN é a marcação da parte da fala (POS), onde, normalmente é uma abordagem baseada em frases e conforme uma frase é formada por uma sequência de palavras, a marcação POS tenta criar rótulos para cada palavra com sua parte correta do discurso (também chamada como categoria de palavras, classe de palavras ou categoria lexical) (INDURKHYA; DAMERAU, 2010).

Este processo também pode ser considerado com uma forma simplificada, ou um subprocesso, de análise morfológica. Enquanto na análise morfológica, busca-se encontrar a estrutura interna de uma palavra (forma de raiz, affixes, etc.), a marcação POS lida com a atribuição de uma etiqueta POS à palavra dada. Isso é mais comum em línguas indo-europeias, que são as línguas mais estudadas na literatura. Outras línguas, como as urálicas ou turcas, podem necessitar de uma análise mais refinada para a marcação POS devido às suas estruturas morfológicas mais complexas (INDURKHYA; DAMERAU, 2010).

3. ESTUDO COMPARATIVO

Tendo em vista a falta de estudos relacionados a avaliação de histórias de usuário utilizando PLN, será realizado um estudo comparativo entre algumas tecnologias. Para identificar as tecnologias a serem comparadas, são utilizadas algumas diretrizes de estudos sistemáticos.

O objetivo da revisão sistemática consiste em encontrar tecnologias que atendam a avaliação de histórias de usuário. Contudo, ressalta-se que essa pesquisa não realizará uma revisão sistemática por completa, mas sim utilizará de algumas diretrizes e práticas para auxiliar o estudo comparativo das tecnologias.

3.1 Método de pesquisa

Um estudo sistemático difere de um tradicional uma vez que procura superar vieses seguindo um método preestabelecido na busca, seleção e avaliação das pesquisas; e na coleta, síntese e interpretação dos dados oriundos das pesquisas (GALVÃO; SAWADA; TREVIZAN, 2004).

O objetivo principal dessa pesquisa é levantar as tecnologias mais citadas no mercado de PLN que atendam as questões relacionadas a pesquisa.

3.1.1 Questões de pesquisa

Foram definidas 3 questões de pesquisa que auxiliará no processo de definição de tecnologias de PLN.

QP1: *Quais são as tecnologias presentes no mercado?*

Tem como objetivo verificar quais as tecnologias mais citadas, empresa responsável pelo desenvolvimento da tecnologia e link para download.

QP2: *Como a tecnologia é classificada dentro do contexto do PLN?*

Tem como objetivo verificar qual a abordagem a tecnologia se aplica (clássica ou estatística) e quais etapas são utilizadas.

QP3: Quais são suas características?

Tem como objetivo verificar as características que a tecnologia possui, como linguagem de programação utilizada, idiomas disponíveis (inglês/português), empresas que utilizam, documentação e tipos de licença.

3.1.2 Processo de busca

Por se tratar de um processo de busca de tecnologias e não uma busca de artigos científicos, foi realizado uma busca no Google na data de 16 de junho de 2022. Foi construída e utilizada a seguinte *string* de busca:

("Tool" OR "Tools" OR "Ferramenta" OR "Ferramentas") AND ("PLN" OR "Processamento de Linguagem Natural" OR "NLP" OR "Natural Language Processing") AND ("free" OR "gratuita" OR "open source" OR "código aberto")

A justificativa para a estrutura da *string* de busca se deve ao fato de possuir muitas soluções/ferramentas disponíveis que não são gratuitas, sendo assim, ficou limitado a busca de ferramentas *open source* ou que não haja custo no desenvolvimento da pesquisa.

3.1.3 Critérios de inclusão e exclusão

Para facilitar a pesquisa, foram definidos alguns filtros que ajudam a eliminar resultados irrelevantes e fora do escopo das questões, sendo assim, foram adotados alguns critérios de inclusão e exclusão.

Critérios de inclusão:

- A tecnologia processa textos nos idiomas português ou inglês
- A tecnologia não possui nenhum custo associado ao uso
- A tecnologia implementa totalmente ou parcialmente as etapas das abordagens clássica ou estatística

Critérios de exclusão:

- A tecnologia possui custo associado
- A tecnologia não processa textos nos idiomas português ou inglês

3.2 Execução

Como se trata de um levantamento de tecnologias presentes no mercado, a quantidade de dados retornados na busca é muito grande, sendo assim, o levantamento foi feito considerando os 30 primeiros sites retornados, ou seja, os sites com maior relevância segundo o Google.

Após pesquisa, foram aplicados os 3 critérios de inclusão e descartadas as tecnologias que não se adequavam aos critérios preestabelecidos. Segue abaixo a Figura 04 exemplificando o método de pesquisa e aplicação dos critérios:

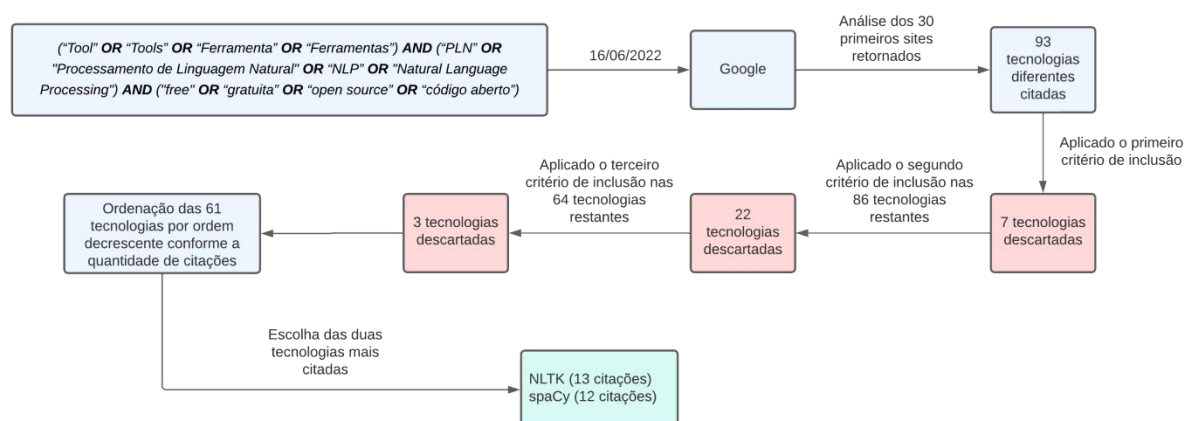


Figura 04: Processo de execução da pesquisa e análise das tecnologias

3.3 Resultados

A filtragem das tecnologias, de acordo com os critérios de inclusão e exclusão, aconteceu de acordo com três etapas conforme mostrado na Figura 04. Das 93 tecnologias encontradas, 61 se enquadram dentro dos critérios preestabelecidos. Como critério de desempate, as tecnologias foram ordenadas e contadas a quantidade de citações dentro dos 30 sites analisados, e por fim, as duas tecnologias mais citadas foram selecionadas: NLTK e spaCy.

A seguir as tecnologias encontradas serão analisadas conforme as questões de pesquisa.

3.3.1 QP1: Quais são as tecnologias presentes no mercado?

Por se tratar de um levantamento de tecnologias de PLN presentes no mercado, encontrar todas as tecnologias presentes no mercado pode ser uma tarefa morosa e impraticável, sendo assim, foi definido que após realizado a busca no Google, apenas as tecnologias citadas nos primeiros 30 sites encontrados seriam analisadas, ou seja, as de maior relevância.

Segue abaixo a Tabela 08 com as 61 tecnologias encontradas que se enquadram nos critérios de inclusão, juntamente com a quantidade de citações de cada tecnologia.

Colocação	Tecnologias	Citações
1	NLTK	13
2	SpaCy	12
3	StanfordNLP	11
4	GenSim	8
5	TextBlob	8
6	OpenNLP	7
7	AllenNLP	7
8	IBM Watson	5
9	Google Cloud Natural Language	4
10	Berkeley Neural Parser	3
11	Apache Mahout	2
12	CogCompNLP	2
13	Natural	2
14	Nlp.js	2
15	PyTorch-NLP	2
16	Retext	2
17	TensorFlow	2
18	Textacy	2
19	Aika	1

20	Amazon Comprehend	1
21	Apache Stanbol	1
22	Apache SystemML	1
23	Apache UIMA	1
24	BERT	1
25	BERTimbau	1
26	BLLIP Parser	1
27	Caffe	1
28	Carrot2	1
29	Coh-Metrix	1
30	CRF++	1
31	Datumbox	1
32	Deeplearning4j	1
33	Distributed Machine Learning Toolkit	1
34	Enelvo	1
35	fastHan	1
36	Flair	1
37	GATE- General Architecture for Text Engineering	1
38	Intel NLP Architect	1
39	KH Coder	1
40	KNIME Text Processing	1
41	LibShortText	1
42	LPU	1
43	MeTA	1
44	MITIE: MIT Information Extraction	1
45	Moses	1
46	Mycroft	1
47	NILC embeddings	1
48	OpenCog	1
49	Opinando	1
50	Pattern	1
51	QDA Miner Lite	1
52	S-EM	1
53	Spark NLP	1
54	TAMS	1

55	text2vec	1
56	Textable	1
57	TiMBL	1
58	TL;DR	1
59	tm - Text Mining Package	1
60	Torch	1
61	VisualText	1

Tabela 08: Lista de tecnologias que atendem aos critérios de inclusão

3.3.2 QP2: Como a tecnologia é classificada dentro do contexto do PLN?

Como visto anteriormente, as tecnologias de PLN podem ser classificadas conforme sua abordagem: clássica ou estatística, tendo cada uma delas etapas que são utilizadas durante o processamento.

Tendo em vista o alto número de tecnologias encontradas, nas questões de pesquisa 2 e 3 foram analisadas apenas as 10 tecnologias melhores colocadas no ranking.

Segue abaixo a Tabela 09 com as 10 primeiras tecnologias juntamente da sua abordagem e etapas disponíveis:

Colocação	Tecnologias	Abordagem	Etapas
1	NLTK	Clássica	Tokenização Conversão de minúsculas Derivação Lematização Geração de árvore de análise ou árvore de sintaxe Tagging
2	SpaCy	Clássica	Tokenização Tagging Parser Detecção e criação de labels Lematização Categorização Customização
3	StanfordNLP	Clássica	Tokenização Lematização Tagging Parser
4	GenSim	Clássica	Tokenização Parser Tagging

5	TextBlob	Clássica	Tokenização Tagging Parser Lematização Geração de árvore de análise
6	OpenNLP	Clássica	Tokenização Tagging Lematização Parser Geração de árvore de análise
7	AllenNLP	Clássica	Tokenização Tagging Parser Detecção e criação de labels Lematização Categorização Customização
8	IBM Watson	Clássica	Tokenização Tagging Parser Lematização Geração de árvore de análise
9	Google Cloud Natural Language	Clássica	Tokenização Lematização Tagging Parser Geração de árvore de análise
10	Berkeley Neural Parser	Clássica	Parser Geração de árvore de análise

Tabela 09: Lista das tecnologias com abordagem e etapas

3.3.3 QP3: Quais são suas características?

Além da abordagem e das etapas contempladas, cada tecnologia possui características específicas, dentre elas a linguagem de programação, idiomas processados, empresas que utilizam, se possui documentação e qual o tipo de licença para utilização.

Sendo assim, segue abaixo a Tabela 10 com as características descritas acima para as 10 tecnologias que mais citadas.

Colocação	Tecnologia	Linguagem	Processa Inglês e Português/BR?	Empresas	Possui documentação?	Licença
1	NLTK	Python	Sim	https://shelf.io/ https://botanalytics.co https://autonom8.com	Sim	Apache License 2.0
2	SpaCy	Python e Cython	Sim	https://www.jpamor-ganchase.com/ https://maximus.com/ https://aiven.io/	Sim	MIT License

3	StanfordNLP	Java	Não, apenas inglês	https://www.mygwork.com/en/ https://www.oneviant.com/ https://www.issgovernance.com/	Sim	GNU General Public License
4	GenSim	Python e Cython	Não, apenas inglês	Não encontrado	Sim	GNU LGPLv2.1
5	TextBlob	Python	Sim	https://www.ucla.edu/	Sim	Não encontrado
6	OpenNLP	Java	Sim	https://www.staples.com/ https://www.xfinity.com/	Sim	Apache License 2.0
7	AllenNLP	Python	Não, apenas inglês	Não encontrado	Sim	Apache License 2.0
8	IBM Watson	Java, C++ e Prolog	Sim	https://www.1800flow.com/ https://www.staples.com/ https://www.chevrolet.com/	Sim	Não encontrado
9	Google Cloud Natural Language	Javascript, Python, Java e Go	Sim	https://global.rakuten.com/ https://medtourea.com/ http://enexusglobal.com/	Sim	Não encontrado
10	Berkeley Neural Parser	Python	Sim	Não encontrado	Sim	Não encontrado

Tabela 10: Lista das tecnologias com suas características

3.4 Considerações finais

Após análise das tecnologias encontradas, foi decidido dar continuidade a pesquisa com a implementação de dois protótipos, sendo um utilizando NLTK e o outro utilizando spaCy.

A escolha dessas tecnologias deve-se ao fato de serem as tecnologias mais consolidadas no mercado de PLN e por atenderem todos os requisitos buscados nas questões da pesquisa.

4. DESENVOLVIMENTO

Neste trabalho será desenvolvido uma API capaz de avaliar histórias de usuário. A API será desenvolvida utilizando as duas tecnologias que mais se destacaram no estudo comparativo realizado anteriormente: NLTK e spaCy. Também será utilizado o Swagger UI que é um framework open source e gratuito que permitirá visualizar e interagir com a API desenvolvida.

Lucassen, et. al. (2016) em seu estudo expõe 13 critérios de qualidade para avaliação de histórias de usuário divididos em 3 grupos: sintáticas, semânticas e pragmáticas. Em ambos os protótipos que serão desenvolvidos, serão avaliados apenas os três critérios correspondentes ao grupo de critérios de qualidades sintáticas, ou seja:

- Bem-formada: a história de usuário possui apenas uma funcionalidade com um propósito
- Atômica: a história de usuário representa um requisito para exatamente um recurso
- Mínima: a história de usuário contem nada mais que uma função, um meio e um fim.

4.1 Requisitos do sistema

Serão definidos os requisitos para as histórias de usuário que seguem o template de Cohn (2009) e para as histórias que serão escritas seguindo o critério de aceitação orientado a cenário que foi exposto na fundamentação teórica.

4.1.1 Requisitos para as histórias de usuário

RH01 - A API deve ser capaz de identificar o ator da história de usuário

RH02 - A API deve ser capaz de identificar a ação da história de usuário

RH03 - A API deve ser capaz de identificar a finalidade da história de usuário

RH04 - A API deve ser capaz de processar uma história seguindo o template de História de Usuário definido por Cohn (2009)

RH05 - A API deve ser capaz de analisar se a história de usuário segue os critérios de qualidades sintáticas (LUCASSEN, 2016)

RH06 - A API deve ser capaz de processar uma ou mais histórias ao mesmo tempo

RH07 - A API deve ser capaz de cronometrar e exibir o tempo de processamento ao analisar uma história de usuário

RH08 - A API deve ser capaz de processar histórias de usuário nos idiomas português/BR e inglês.


RH09 - A API deve ser capaz de processar a mesma história de usuário em ambas tecnologias de PLN

4.1.2 Requisitos para as histórias de usuário orientado a cenário

RC01 - A API deve ser capaz de identificar a pré-condição do cenário

RC02 - A API deve ser capaz de identificar a ação do cenário

RC03 - A API deve ser capaz de identificar a finalidade do cenário

RC04 - A API deve ser capaz de processar uma história seguindo o template definido nos critérios de aceitação **utilizando o template orientado a cenários** 

RC05 - A API deve ser capaz de processar um ou mais cenários ao mesmo tempo

RC06 - A API deve ser capaz de analisar se o cenário segue os critérios de qualidades sintáticas (LUCASSEN, 2016)

RC07 - A API deve ser capaz de cronometrar e exibir o tempo de processamento ao analisar um cenário

RC08 - A API deve ser capaz de processar cenários nos idiomas português/BR e inglês.

RC09 - A API deve ser capaz de processar o mesmo cenário em ambas tecnologias de PLN

4.2 Design da interface

Para facilitar o entendimento de como a API irá se comportar, foram desenvolvidos algumas imagens da interface do Swagger UI. Na Figura 05 é

possível observar a tela inicial da API, onde será possível encontrar os dois *endpoints* responsáveis para a avaliação das histórias de usuário.

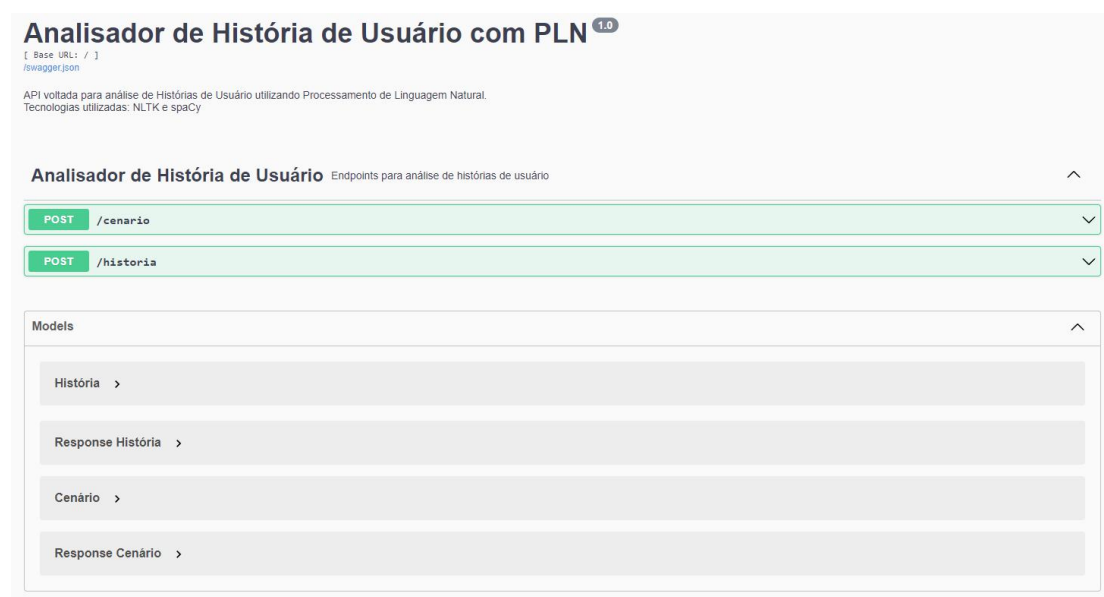


Figura 05: Tela inicial do da API do Swagger

Ao clicar no primeiro *endpoint*, uma área será expandida onde será possível inserir os dados de entrada para a avaliação da história de usuário utilizando o template padrão de Cohn (2009). Será possível inserir um JSON (JavaScript Object Notation), com uma ou mais histórias de usuário para serem avaliadas em seu respectivo idioma. Segue abaixo Figura 06 como exemplo:

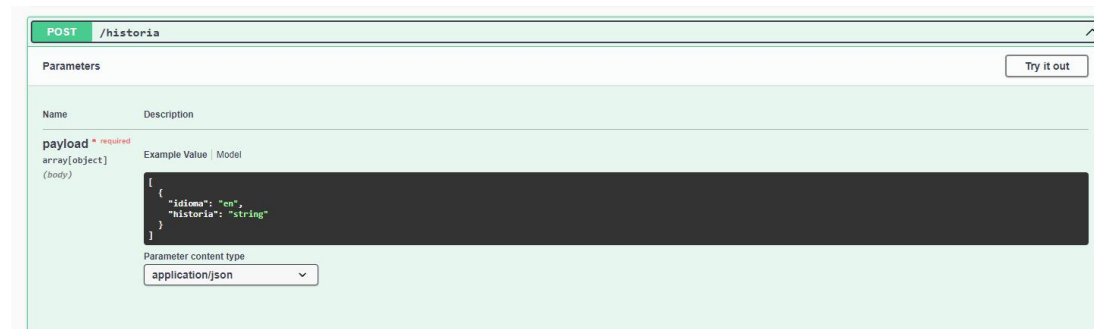


Figura 06: Endpoint responsável para avaliação de histórias de usuário utilizando o template de Cohn

Em seguida, podemos observar na Figura 07, um modelo de resposta que a API irá retornar após o processamento. Será disponibilizado um JSON contendo uma lista das histórias que foram processadas com as seguintes

informações. Também será possível identificar o sucesso ou erro conforme os códigos HTTP retornados na requisição, onde 200 será de sucesso e 400 de erro. Segue abaixo a Tabela 11 com a descrição de cada propriedade retornada.

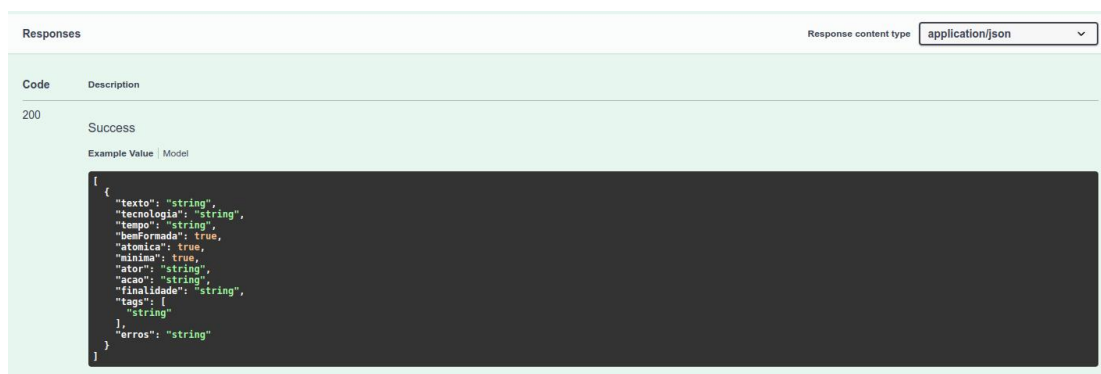


Figura 07: Exemplo de resposta na avaliação de história de usuário utilizando o template de Cohn

Propriedade	Tipo	Descrição
texto	String (texto)	História que foi analisada
tecnologia	String (texto)	Tecnologia de PLN utilizada no processamento (NLTK ou spaCy)
tempo	String (texto)	Tempo de processamento
bemFormada	boolean	Critério de qualidade “Bem Formada”. Caso o critério for aceito o valor será true , caso contrário false
atomica	boolean	Critério de qualidade “Atômica”. Caso o critério for aceito o valor será true , caso contrário false
minima	boolean	Critério de qualidade “Mínima”. Caso o critério for aceito o valor será true , caso contrário false
ator	String (texto)	Ator que desempenha a ação na história de usuário
acao	String (texto)	Ação desempenhada na história de usuário
finalidade	String (texto)	Finalidade pela qual a ação está sendo desempenhada. A finalidade é opcional dentro do template de Cohn.
tags	List<String>	Lista de tags gerada após o processamento de linguagem natural

erros	String (texto)	Erros que possam surgir durante o processamento da história de usuário.
-------	----------------	---

Tabela 11: Descrição dos atributos retornados após processamento da história de usuário

Também será disponibilizado um *endpoint* específico para avaliação de histórias de usuário utilizando o template orientado a cenários.

Este *endpoint* segue a mesma estrutura do anterior onde será possível inserir um ou mais cenários a serem avaliados. Segue Figura 08 como exemplo.



Figura 08: Endpoint responsável para avaliação de histórias de usuário utilizando o template orientado a cenário

A resposta da requisição processada pelo endpoint da Figura 09 segue a mesma estrutura do endpoint de histórias de usuário que utiliza o template de Cohn, a única diferença é o nome da propriedade cenário, que traz o cenário que foi processado, o restante das propriedades tem a mesma definição descritas na Tabela 11, a única diferença é que não há ator, mas sim uma pré-condição.

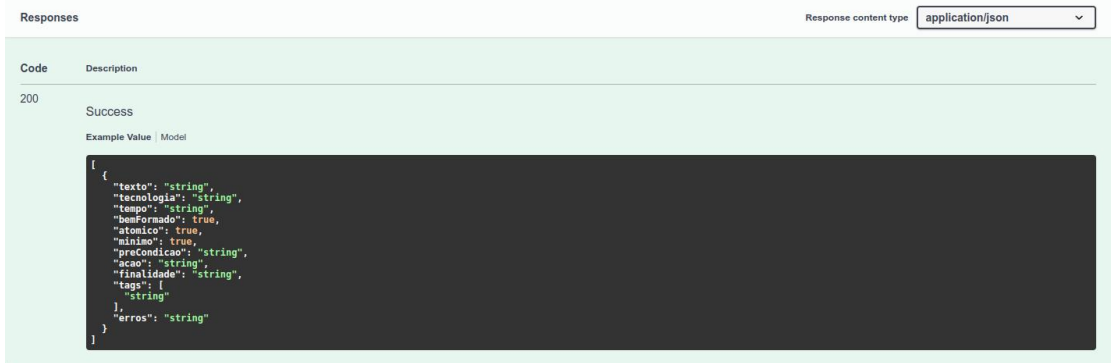


Figura 09: Exemplo de resposta na avaliação de história de usuário utilizando o template orientado a cenário

4.3 Implementação

Nesta seção, são descritas todas as etapas realizadas na prototipação da API, desde o pré-processamento das histórias até as regras de validação.

4.3.1 Pré-processamento

Conforme visto na seção 2.4.1.1 Pré-processamento de texto, o texto deve ser tratado, afim de identificar erros que prejudiquem as análises. Tendo em vista que a análise será feita apenas sintaticamente e não semanticamente, o texto a ser processado deverá ser tratado antes da requisição ser enviada para a API.

Tendo como base o estudo realizado por Lucassen (2016), para que as histórias e cenários sejam processados, é necessário seguir um padrão de palavras chave estruturando as frases antes do seu processamento, pois a mesma será subdividida em sentenças pela qual será possível identificar o ator/pré-condição, ação e a finalidade.

Exemplo de história de usuário no template de Cohn (2009) em português	<i>“Eu como vendedor gostaria de cadastrar meus produtos para que eu possa listá-los posteriormente”</i>
Exemplo de história de usuário no template de Cohn (2009) em inglês	<i>“I as a seller I would like to register my products so I can list them later.”</i>



Exemplo de história de usuário orientado a cenário no template de Gherkin (HAMILTON, 2009) em português	<i>“Dado que o cliente deseja abrir uma conta, informou o CPF, informou o RG e informou o endereço, quando entrar com essas informações no cadastro, então uma nova conta deve ser criada.”</i>
Exemplo de história de usuário orientado a cenário no template de Gherkin (HAMILTON, 2009) em inglês	<i>“Given a customer wants to open an account, and the ID was informed, and the address was informed, when all those information was typed, then a new account must be created.”</i>

Tabela 12: Exemplos de templates com as palavras-chave em negrito

Esse tipo de tratamento é primordial para que seja possível validar os critérios de qualidade e de aceitação, pois determinados critérios levam em consideração a ordem das sentenças, ou seja, no exemplo acima, ao validar qual o ator da história de usuário, espera-se que ele seja encontrado na primeira sentença da história: “Eu como vendedor”.

4.3.2 Normalização entre idiomas

Para que seja possível validar as histórias de usuário nos idiomas português e inglês, foi necessário normalizar as classes gramaticais entre os idiomas, pois o idioma português possui diversos tipos de pronomes (pronome pessoal, demonstrativos, interrogativos, possessivos, relativos e indefinidos), já em inglês não há essa distinção entre pronomes.

Portanto, todas as classes gramaticais que possuem mais que um tipo foram agrupadas da maneira mais abrangente possível, ou seja, uma *tag* gerada através de PLN que seja um pronome pessoal, será considerado apenas como um pronome. Por exemplo, ao processar a frase: “Eu como vendedor gostaria de cadastrar meus produtos para que eu possa listá-los posteriormente.”, cada palavra da frase recebe uma *tag*:

"tags": [
"Eu --> PROESS --> PRONOME",
"como --> PREP --> PREPOSIÇÃO",
"vendedor --> N --> SUBSTANTIVO",
"gostaria --> V --> VERBO",
"de --> PREP --> PREPOSIÇÃO",
"cadastrar --> V --> VERBO",
"meus --> PROADJ --> PRONOME",
"produtos --> N --> SUBSTANTIVO",
"para --> PREP --> PREPOSIÇÃO",
"que --> PROSUB --> PRONOME",
"eu --> PROESS --> PRONOME",
"possa --> V --> VERBO",
"listá-los --> N --> SUBSTANTIVO",
"posteriormente --> ADV --> ADVÉRBIO",
". --> . --> INVÁLIDO"
]

Nota-se que a palavra “eu” é um pronome pessoal e como forma de normalizar na sua forma mais simples, foi definido apenas como pronome.

4.3.3 Fluxo de processamento e validação

Para que se tenha uma imparcialidade ao avaliar as tecnologias, ambas seguem o mesmo fluxo de processamento e validação:

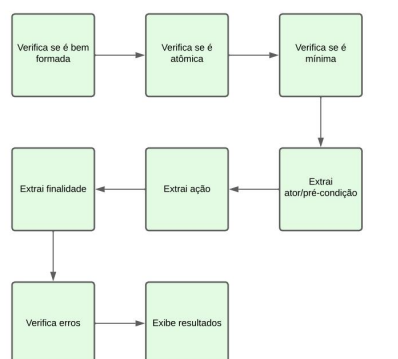


Figura 10: Fluxo de processamento e validação

Na primeira linha, as três primeiras etapas são responsáveis em aferir os critérios de qualidade da história de usuário. Na segunda linha, as próximas 3 etapas são responsáveis na extração do ator/pré-condição, ação e finalidade, e por fim, na última linha, as duas últimas etapas são responsáveis em verificar se houve algum erro e exibir os resultados.

4.3.4 Regras de validação

Como foi dito anteriormente, os critérios de qualidade avaliados levam em consideração a parte sintática dos textos, sendo assim, para que seja possível aferir esses critérios, serão levados em consideração as classes gramaticais de cada palavra em todas as sentenças processadas.

Para que seja possível descobrir qual a classe gramatical de cada palavra, será utilizado o *POS-Tagging* gerado por cada tecnologia em seu processamento.

“Eu como vendedor, gostaria de cadastrar meus produtos, para que eu possa listá-los posteriormente.”	
Eu -> PRONOME	
como -> PREPOSIÇÃO	
vendedor -> SUBSTANTIVO	
gostaria -> VERBO	
de -> PREPOSIÇÃO	
cadastrar -> VERBO	
meus -> PRONOME	
produtos -> SUBSTANTIVO	
para -> PREPOSIÇÃO	
que -> PRONOME	
eu -> PRONOME	
possa -> VERBO	
listá-los -> VERBO	
posteriormente -> ADVÉRBIO	



Tabela 13: Exemplo de derivação gramatical das sentenças

4.3.4.1 Bem formada

Para validar o primeiro critério de qualidade, foi levado em consideração a estrutura definida por Lucassen, et. al. (2016), no qual uma história é bem formada quando há o seguinte formato: *Quem realizará a tarefa + objetivo da tarefa + finalidade para a realização da tarefa (opcional)*.

Lucassen, et. al. (2016) em seu estudo informa que existem diversos templates possíveis para essa validação, porém, este trabalho segue o mesmo template que Lucassen, et. al. (2016) utilizou:

Sujeito + Adjetivo (opcional) + Verbo + Objeto indireto (opcional) + Objeto direto

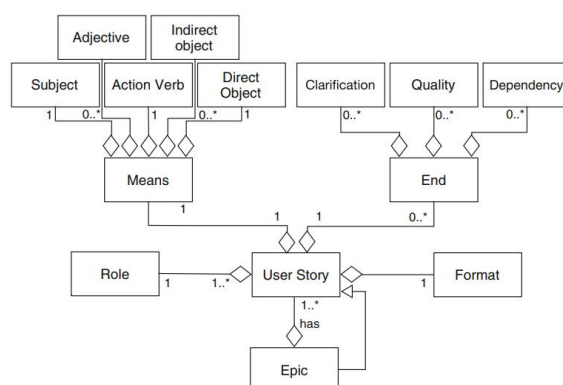


Figura 11: Modelo conceitual de história de usuário utilizado por Lucassen, et. al. (2016)

Sendo assim, para validar se a história é bem formada, deve-se validar o sujeito (ator/pré-condição), o verbo (ação) e o objeto direto (finalidade).

4.3.4.1.1 Validação do ator

No caso de histórias de usuário que seguem o template de Cohn (2009), o ator deverá ser identificado na primeira sentença e essa sentença deverá possuir as seguintes classes gramaticais:

substantivo + (pronome OU preposição OU artigo)

Caso a primeira sentença não possua um substantivo somado a um pronome, preposição e artigo, a história não será bem formada pois haverá uma inconsistência ao encontrar o ator.

4.3.4.1.2 Validação da pré-condição

No caso de cenários, não será identificado o ator, mas sim uma pré-condição na primeira sentença. Para isso, será utilizado a mesma estrutura utilizada em histórias de usuário somados da presença da palavra Dado/Given:

Dado/Given + substantivo + (pronome OU preposição OU artigo)

Caso a primeira sentença não possua Dado/Given somados de um substantivo e um pronome, preposição ou artigo, o cenário não será bem formado pois haverá uma inconsistência ao encontrar a pré-condição.

4.3.4.1.3 Validação da ação

A validação da ação é o segundo critério a ser avaliado ao definir se uma história de usuário é bem formada. Para isso, ela segue duas estruturas diferentes: uma para o template de Cohn (2009) e outra para a sintaxe de Gherkin (HAMILTON, 2022).

Para o template de Cohn (2009), a ação deverá ser encontrada na segunda sentença e é composta conforme a seguinte estrutura:

verbo + substantivo + pronome + (preposição OU advérbio)

No caso de cenários, a estrutura é semelhante, porém com alguns ajustes:

Pré-condição antes da ação + Quando/When + verbo + substantivo + (pronome OU preposição OU advérbio)

Neste caso, é validado se possui as palavras chaves da pré-condição (Dado/Given) antes da palavra chave da ação (Quando/When), se possui a palavra chave da ação (Quando/When), somados a um verbo, um substantivo e um pronome, preposição ou advérbio.

Caso não possua essa estrutura, a história não será bem formada pois haverá uma inconsistência ao encontrar a ação.

4.3.4.1.4 Validação da finalidade

No caso de histórias de usuários seguindo o template de Cohn (2009), a finalidade é opcional, ou seja, poderá ou não estar presente na frase. Caso esteja presente, ela deverá ser encontrada na terceira sentença e seguirá a seguinte estrutura:

verbo + (pronome OU preposição OU substantivo OU advérbio)

Já no caso de cenários, a finalidade é obrigatória e deverá ser encontrada na sentença que se inicia com a palavra chave Então/Then.

Para validar a finalidade em cenários, a estrutura é a mesma do template de Cohn (2009), porém é validado também a ordem das palavras chaves de todas as sentenças: Dado -> Quando -> Então:

Ordem correta das palavras chave + verbo + (pronome OU preposição OU substantivo OU advérbio)

Caso o cenário não possua essa estrutura, ele não será bem formado pois haverá uma inconsistência ao encontrar a finalidade.

4.3.4.2 Atômica

Segundo Lucassen, et. al. (2016), uma história de usuário é atômica quando há apenas um objetivo (ação) na tarefa. Sendo assim, para validar esse segundo critério de qualidade, primeiramente é identificado qual a sentença de ação da história/cenário.

Após identificado qual a sentença de ação, no caso de histórias de usuário que seguem o template de Cohn (2009), é verificado se há alguma conjunção utilizando os conectivos **e**, **ou**, **and** e **or**. Ou seja, caso a história de usuário possua uma dessas conjunções será considerado como mais que uma ação, sendo assim viola a atomicidade da história de usuário.

No caso de cenários que seguem o template de Gherkin (HAMILTON, 2022), a ação pode ser somada a condições, ou seja, a validação por meio de conectivos como **e**, **ou**, **and** e **or** não é válida. Sendo assim, para validar cenários, é verificado se a palavra-chave **quando/when** é utilizada mais que uma vez. Portanto, se no cenário for identificado mais que uma palavra-chave **quando/when** na sentença de ação, logo viola a atomicidade do cenário.

4.3.4.3 Mínima

O terceiro critério de qualidade avaliado é se a história de usuário é mínima. Uma história/cenário é mínima quando ela é bem formada (primeiro critério de qualidade) e não há informações extras, como comentários e notas adicionais (Lucassen, et. al., 2016).

Para validar o terceiro critério de qualidade, é verificado primeiramente se a história/cenário é bem formada e em seguida é verificado em todas as sentenças processadas se há algum caracter inválido, como por exemplo: *, [,], (,), {, }, _, :

Sendo assim, leva-se a entender que caso um desses caracteres esteja presente, significa que seja alguma nota adicional ao texto, portanto viola o critério de qualidade de minimalidade da história/cenário.



5. REFERÊNCIAS

BAASCH. A. V. S, “**Aplicação de processamento de linguagem natural para análise de texto e indicação de notícias similares: uma ferramenta de apoio para a identificação de fake news**”, 2021

BARKER. D. “**12 open source tools for natural language processing**”, 2019. Disponível em: <https://opensource.com/article/19/3/natural-language-processing-tools> Acesso em 11 dez. 2021

BASILI, V. R. CALDIERA, G.; ROMBACH, H. D. “**Goal Question Metric Paradigm**”. In: MARCINIAK Encyclopedia of Software Engineering. [S.l.]: John Wiley & Sons, 1994.

BROWN R. B, “**Doing Your Dissertation in Business and Management: The Reality of Research and Writing**”, 2006. Sage Publications

BROWNIEE. J., “**What Is Natural Language Processing?**”, 2017. Disponível em: <https://machinelearningmastery.com/natural-language-processing/> Acesso em 16 abr. 2022

BUDGEN, D., TURNER, M., BRERETIB, P., KITCHENHAM, B.: “**Using mapping studies in software engineering**”. In: Proceedings of PPIG. vol. 8, pp. 195–204. Lancaster University (2008)

COHN, M. “**User stories applied for agile software development**”, 13. ed. Crawfordsville, Indiana. 2009. 263 p.

FEMMER, H. “**Reviewing Natural Language Requirements with Requirements Smells—A Research Proposal**”. Proceedings of IDoESE, 2013

FEMMER, H., FERNÁNDEZ, D.M., JUERGENS, E., KLOSE, M., ZIMMER, I., ZIMMER, J.: “**Rapid requirements checks with requirements smells: two**

case studies". In: Proceedings of the 1st International Workshop on Rapid Continuous Software Engineering. pp.10–19. ACM, 2014

FERNANDES. R, "**POS Tagging — da teoria à implementação**", 2022.

Disponível em: <https://medium.com/turing-talks/pos-tagging-da-teoria-%C3%A0-implementa%C3%A7%C3%A3o-eafa59c9d115> Acesso em 04 mai 2022

FERRARIO. A, NÄGELIN. M, "**The Art of Natural Language Processing: Classical, Modern and Contemporary Approaches to Text Document Classification**", 2020

FRANCINO. Y. "**The essential guide to user story creation for agile leaders**", TechBeacon, 2017. Disponível em:

<https://techbeacon.com/app-dev-testing/essential-guide-user-story-creation-agile-leaders> Acesso em 17 mar. 2022

FRANKLIN. B. "**Advice to a Young Tradesman**", 1748. Disponível em:

<https://founders.archives.gov/documents/Franklin/01-03-02-0130> Acesso em 10 dez. 2021

GALVÃO, C. M.; SAWADA, N. O.; TREVIZAN, M. A. **Revisão sistemática: recurso que proporciona a incorporação das evidências na prática da enfermagem**. Revista Latino Americana de Enfermagem, Ribeirão Preto, v. 12, n. 3, p. 549-556, 2004. PMid:15303213.

<http://dx.doi.org/10.1590/S0104-11692004000300014>

GIL, A. C. "**Como elaborar projetos de pesquisa**". São Paulo: Atlas, 2010. ISBN 5ª edição.

GLINZ, M. 2000, "**Improving the quality of requirements with scenarios**". In: Proceedings of the World Congress on Software Quality (WCSQ), pp 55–60

GOYAL, C. **“Part 9: Step by Step Guide to Master NLP – Semantic Analysis”**, 2021, Disponível em: <https://www.analyticsvidhya.com/blog/2021/06/part-9-step-by-step-guide-to-master-nlp-semantic-analysis/> Acesso em 14 abr. 2022

HAMILTON, T. **“Gherkin Language: Format, Syntax & Gherkin Test in Cucumber”**, 2022, Disponível em: <https://www.guru99.com/gherkin-test-cucumber.html> Acesso em 24 out. 2002

HECK, P. KLABBERS, M. VAN EEKELEN, M. C. J. D. **“A software product certification model,”** Software Quality Journal, vol. 18, no. 1, pp. 37–55, 2010.

HECK, P. ZAIDMAN A., **“A quality framework for agile requirements: a practitioner’s perspective”**, 2014

HEATH, F. **The trouble with user stories**. 2020, DZone. Disponível em: <https://dzone.com/articles/the-trouble-with-user-stories-1> . Acesso em 04 dez. 2021

IEEE Computer Society (2018), **“Systems and software engineering — Life cycle processes — Requirements engineering”**, Second Edition 2018-11 ISO/IEC/IEEE 29148

INDURKHYA, N.; DAMERAU, F. J. **“Handbook of natural language processing.”** [S.l.]: CRC Press, 2010. v. 2.

LUCASSEN, G. DALPIAZ, F. WERF, J. M. V. D, BRINKKEMPER. S. **“Improving agile requirements: the Quality User Story framework and tool”**, 2016

LUTKEVICH, B. **“Natural language processing (NLP)”**, 2021. Disponível em: <https://www.techtarget.com/searchenterpriseai/definition/natural-language-processing-NLP> Acesso em 09 abr 2022.

NASCIMENTO, R. ARANHA, E. KULESZA. U, LUCENA. M, **“Requirements Smells como indicadores de má qualidade na especificação de requisitos: Um Mapeamento Sistemático da Literatura.”**

10.17771/PUCRio.wer.inf2018-40, 2018

NIVRE, Joakim. **“On Statistical Methods in Natural Language Processing”**, 2002

NORTH. D. **“Introducing BDD”**, 2006 . Disponível em:

<https://dannorth.net/introducing-bdd/> Acesso em 16 jul. 2022

O’LEARY Z. **“The essential guide to doing research”**, 2004. Sage.

<https://www.bestbuy.com/site/cyberpowerpc-gamer-xtreme-gaming-desktop-intel-core-5-12600kf-16gb-memory-nvidia-geforce-rtx-3050-500gb-ssd-black/6500510.p?skuld=6500510>

OLIVEIRA. G, MARCZAK, S, **“On the Empirical Evaluation of BDD Scenarios Quality: Preliminary Findings of an Empirical Study”**, 2017

PAI. A. **“What is Tokenization in NLP? Here’s all you need to know”**, 2020. Disponível em: <https://www.analyticsvidhya.com/blog/2020/05/what-is-tokenization-nlp/> Acesso em 11 abr. 2022

RAHARJANA, I. K, HARRIS. F, JUSTITIA. A, **“Tool for generating behavior-driven development test-cases,”** J. Inf. Syst. Eng. Bus. Intell., vol. 6, no. 1, p. 27, Apr. 2020, doi: 10.20473/jisebi.6.1.27-36.

RAHARJANA, I. K, SIAHAAN. D e FATICHAH. C, **“User Stories and Natural Language Processing: A Systematic Literature Review,”** in IEEE Access, vol. 9, pp. 53811-53826, 2021, doi: 10.1109/ACCESS.2021.3070606.

RAHATE. P. **"Definition of Done vs Acceptance Criteria"**, 2021 Disponível em: <https://agilemania.com/definition-of-done-vs-acceptance-criteria/> Acesso em 16 jul. 2022

RACKSPACE TECHNOLOGY, **"Dos chatbots à Alexa: a evolução do Processamento de Linguagem Natural"**, 2020. Disponível em: <https://www.rackspace.com/pt/solve/evolution-nlp> Acesso em 31 jan 2022.

REHKOPF. M. **"Histórias de usuários com exemplos e um template"**, 2020. Disponível em: <https://www.atlassian.com/br/agile/project-management/user-stories> . Acesso em 09 dez. 2021

ROBERTS, E. **"Natural Language Processing: History"**, 2004. Disponível em: https://cs.stanford.edu/people/eroberts/courses/soco/projects/2004-05/nlp/overview_history.html Acesso em 11 abr. 2022

RODRIGUES. F. **"Explicando BDD para iniciantes"**, 2020. Disponível em: <https://www.linkedin.com/pulse/explicando-bdd-para-iniciantes-fernando-r-de-sousa/?originalSubdomain=pt> Acesso em 16 jul. 2022

SAUNDERS, M., LEWIS, P., & THORNHILL, A. **"Research Methods for Business Students"**, 2007, (6th ed.) London: Pearson.

SILVA, P. **"O que é automação e para que serve? Conversando com o CTO"**, 2019. Disponível em: <https://gobacklog.com/blog/o-que-e-automacao-e-para-que-serve/> . Acesso em 6 jan. 2022.

SOMMERVILLE, I. . **"Engenharia de software"**, 9. ed. Pearson. 2011. 529 p.

STANFORD. **"Stemming and lemmatization"**, 2008. Disponível em: <https://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html> Acesso em: 14 abr. 2022

TAKE BLIP, “**Tudo sobre NLP: o que é processamento de linguagem natural e seus desafios na Inteligência Artificial**”, 2019. Disponível em: <https://www.take.net/blog/tecnologia/nlp-processamento-linguagem-natural/> . Acesso em 01 dez. 2021.

THANAKI. J. “Python Natural Language Processing”, 2017 Disponível em: <https://www.oreilly.com/library/view/python-natural-language/9781787121423/f7f54f6d-8257-4904-9c8e-88d4ac491b94.xhtml> . Acesso em 14 abr. 2022

THAYER, R. H. e DORFMAN, M.; “**Introduction to Tutorial Software Requirements Enginnering**” in Software Requirements Engineering, IEEE-CS Press, Second Edition, 1997, p.p. 1-2.

WAKE, B. “**INVEST in Good Stories, and SMART Tasks**”, 2003. Disponível em: <https://xp123.com/articles/invest-in-good-stories-and-smart-tasks/> Acesso em 11 jan 2022.

WAUTELET. Y, HENG. S, KIV. S, KOLP. M, “**User-story driven development of multi-agent systems: A process fragment for agile methods,**” Comput. Lang., Syst. Struct., vol. 50, pp. 159–176, Dec. 2017, doi: 10.1016/j.cl.2017.06.007.

HAMILTON, T. “**Gherkin Language: Format, Syntax & Gherkin Test in Cucumber**”, 2022. Disponível em: <https://www.guru99.com/gherkin-test-cucumber.html>. Acesso em 14 set 2022