Programa de Pós-Graduação em

Computação Aplicada

Mestrado Acadêmico

Robson Kerschner de Lima

Nhatos: Um Modelo para Recomendação de Requisitos Baseado em Históricos de Contextos

São Leopoldo, 2020

Robson Kerschner de Lima

NHATOS: UM MODELO PARA RECOMENDAÇÃO DE REQUISITOS BASEADO EM HISTÓRICOS DE CONTEXTOS

Dissertação apresentada como requisito para a obtenção do título de Mestre pelo Programa de Pós-Graduação em Computação Aplicada da Universidade do Vale do Rio dos Sinos — UNISINOS

Orientador:
Prof. Dr. Jorge Luis Victória Barbosa

São Leopoldo
2020

L732n Lima, Robson Kerschner de.
Nhatos : um modelo para recomendação de requisitos baseado em históricos de contextos / por Robson Kerschner de Lima. – 2020.
97 f. : il. ; 30 cm.

Dissertação (mestrado) — Universidade do Vale do Rio dos Sinos, Programa de Pós-Graduação em Computação Aplicada, São Leopoldo, RS, 2020.
"Orientador: Dr. Jorge Luis Victória Barbosa".

1. Engenharia de requisitos. 2. Sistema de recomendação em engenharia de software (SRES). 3. Históricos de contextos. 4. Processamento de linguagem natural. 5. Sistemas multi-agentes. I. Título.

CDU: 004.414.38

Catalogação na Publicação (CIP):
Bibliotecário Alessandro Dietrich - CRB 10/2338

(Esta folha serve somente para guardar o lugar da verdadeira folha de aprovação, que é obtida após a defesa do trabalho. Este item é obrigatório, exceto no caso de TCCs.)

Dedico este trabalho ao meu sábio pai Nerí Nunes de Lima,
minha querida mãe Edite K. Lima e minha amada esposa Marciane F. Lima.

AGRADECIMENTOS

Gostaria de agradecer ao professor Jorge Barbosa, por sua consultoria e sabedoria durante todo este difícil projeto. Agradeço ao professor Alexsandro Filippetto, por seu apoio junto aos artigos desenvolvidos. Sou grato pela ajuda do professor Kleinner Farias, que contribuiu com seu vasto conhecimento em engenharia de software. Também agradeço aos colegas Bruno Martini e Jorge Aranda, com os quais tive a oportunidade de desenvolver os trabalhos propostos durante as disciplinas do curso. Agradeço também ao colega Henrique Damasceno, por me auxiliar em momentos de dúvidas. Agradeço aos colegas de trabalho Eliano Almança, Cristiano Dobrowski e Hugo Dominiak que contribuíram com o estudo de caso e dispuseram seu tempo para contribuir com esta pesquisa.

Learning never exhausts the mind.
— LEONARDO DA VINCI

RESUMO

O ambiente de gerenciamento de projetos de software possui inúmeras incertezas. Durante o ciclo de vida dos projetos, novos requisitos e solicitações de mudanças ocorrem a todo instante, tornando o seu gerenciamento estratégico. As falhas que ocorrem durante o gerenciamento de requisitos prejudicam as chances de sucesso dos projetos. Porém, o gerenciamento adequado dos requisitos, através da cooperação das equipes e utilização dos históricos de projetos, reduz o risco quanto aos desvios do planejamento em relação ao tempo, custo e qualidade do projeto. Uma pesquisa de campo foi conduzida, envolvendo 56 profissionais de projetos, para coletar sua percepção com relação às ferramentas de apoio em gestão de projetos utilizadas em seu ambiente de trabalho. Nesta pesquisa, a gestão de escopo apresentou-se como a área mais crítica para o sucesso dos projetos. Sobre os problemas enfrentados, esta mesma área também destacou-se nas respostas obtidas. Quando questionados sobre quais áreas deveriam receber uma ferramenta de apoio proativo, novamente a gestão do escopo foi citada. Neste sentido, essa dissertação apresenta um modelo intitulado Nhatos. O Nhatos contribui para a redução da probabilidade de falha dos projetos através da recomendação de requisitos. Seus diferenciais acadêmicos seguem conceitos estabelecidos pela computação ubíqua. O modelo recomenda requisitos através da análise dos históricos de contextos de projetos, considerando os avanços nos seus ciclos de vida. E, utiliza os benefícios da colaboração entre os envolvidos no projeto. O Nhatos foi exposto a um ambiente da indústria e avaliado através de uma base histórica contendo 183 projetos. Após analisar diferentes cenários, o Nhatos alcançou um percentual de acerto de 83,04% das recomendações realizadas.

Palavras-chave: Engenharia de Requisitos. SRES. Históricos de Contextos. Processamento de Linguagem Natural. Sistemas Multi-Agentes.

ABSTRACT

The software project management environment has numerous uncertainties. During the life cycle of projects, new requirements, and requests for changes occur at all times, making their management strategic. Failures that occur during requirements management impair the chances of successful projects. However, proper requirements management reduces the risk of deviations from planning about the time, cost, and quality of the project. Field research was conducted involving project professionals to collect the teams' perception of the application of a proactive computational model to support software teams during their decision making. In this sense, this qualification presents a model called Nhatos. Nhatos contributes to reducing the probability of failure of projects through the recommendation of requirements. Nhatos follows ubiquitous computing concepts, since it recommends requirements through the analysis of historical project contexts, considering the advances in their life cycles. The model was exposed to an industry environment and evaluated using a historical basis containing 183 projects. After analyzing different scenarios, Nhatos achieved a correct percentage of 83.04% of the recommendations made.

Keywords: Requirements Engineering. RSSE. Contexts History. Natural Language Processing. Multi-Agents System.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

ER	Engenharia de Requisitos
GE	Gestão de Escopo
GCT	Google Cloud Translation
GNL	Google Natural Language
GP	Gerenciamento de Projetos
GR	Gestão de Requisitos
JSON	JavaScript Object Notation
PLN	Processamento de Linguagem Natural
SRES	Sistema de Recomendação em Engenharia de Software

SUMÁRIO

1 INTRODUÇÃO

1.1 Motivação

Atualmente, as organizações estão inseridas em um mundo globalizado e dinâmico, sujeito a rápidas mudanças. Nesse ambiente, novas demandas surgem diariamente, impulsionadas pela inovação e posicionamento no mercado. Essas demandas são traduzidas em projetos, onde o gerenciamento é estratégico para a obtenção de resultados satisfatórios que atendam os parâmetros estabelecidos de tempo, custo e qualidade (PMI, 2017a).

O ambiente de projetos é rodeado por incertezas. Estas incertezas são inevitáveis, considerando que cada projeto é único e temporário. Deste modo, novos requisitos e solicitações de mudanças podem ocorrer a qualquer momento durante o ciclo de vida dos projetos. Além disso, o gerenciamento de projetos ocorre de maneira incremental e integrada, fazendo com que ações realizadas durante um processo se reflitam nos demais, principalmente no cenário de desenvolvimento de softwares, o que torna o gerenciamento de requisitos fundamental durante a sua execução (PMI, 2017b; ELIZABETH H. JEREMY D., 2017).

Os requisitos representam a base de um projeto, definindo o que os *stakeholders* – clientes, desenvolvedores e gestores – necessitam em um novo sistema, e o que este sistema deve realizar para satisfazer estas necessidades (ELIZABETH H. JEREMY D., 2017). No cenário de desenvolvimento de software, o processo de descobrir, analisar, documentar e monitorar estas necessidades é chamado de Engenharia de Requisitos (SOMMERVILLE, 2016).

A incorreta gestão de requisitos gera problemas ao projeto com o passar do tempo. Deste modo, o desempenho é afetado, ocasionando a redução das chances de cumprimento do escopo definido pelos envolvidos (PMI, 2017a). Identificar os requisitos corretos para os clientes é o aspecto mais importante sob o ponto de vista gerencial (PMI, 2017b), enquanto que o uso de informações históricas como más definições ou mudanças de requisitos dos projetos passados podem auxiliar na gestão dos projetos futuros (FILIPPETTO; LIMA; BARBOSA, 2011).

O uso de conceitos de computação ubíqua (BARBOSA, 2015) apresenta-se como alternativa para apoiar os times de projetos durante o gerenciamento dos requisitos. De acordo com WEISER (1999) e Satyanarayanan (2001), um sistema de computação ubíqua é ciente de seu contexto, ou seja, conhece as informações relativas ao usuário e seu ambiente. Assim, este sistema tem a capacidade de adaptar-se às necessidades dos usuários (DEY; ABOWD; SALBER, 2001).

Estudos têm se voltado para a recomendação de contexto (ROSA; BARBOSA; RIBEIRO, 2016), onde a partir de contextos armazenados em uma base histórica, aplicam-se métodos para a recomendação de um novo contexto. Assim, ocorre a recomendação de contextos, tornando o sistema proativo (AMEYED; MIRAOUI; TADJ, 2015; PEJOVIC; MUSOLESI, 2013; VANSYCKEL; BECKER, 2014). O sistema passa a compreender as intenções dos usuários, prevendo como será seu contexto no futuro, viabilizando análises sobre novos cenários (BURBEY; L. MARTIN, 2012). Deste modo, a análise de similaridade destes contextos possibilita

a aplicação de estratégias tanto de recomendação de contextos, como de necessidades dos envolvidos que utilizam o sistema (WIEDMANN et al., 2016; DUPONT; BARBOSA; ALVES, 2019).

Através da aplicação de Sistemas Multi-Agentes (WOOLDRIDGE, 2009) é possível capturar os históricos de contextos dos projetos ao longo do seu ciclo de vida por meio das informações inseridas pelas equipes de projetos. A aplicação de Processamento de Linguagem Natural (LUGER, 2016) permite agrupar projetos semelhantes e analisar também a similaridade dos seus requisitos. Portanto, este trabalho explora estes conceitos com o objetivo de recomendar requisitos aos times de projeto e avaliar a sua aceitação, permitindo assim que os gestores possam tomar as devidas ações para alcançar os objetivos do projeto.

1.2 Definição do Problema de Pesquisa

Nos últimos anos, constantes mudanças ocorreram na sociedade e nos negócios. O uso contínuo de novas tecnologias resulta em uma Transformação Digital, que traz consigo mudanças disruptivas em todos os domínios (REIS et al., 2018). As técnicas que até agora se mostraram cruciais para elicitar requisitos estão perdendo força diante das mudanças de paradigma ocorridas. Portanto, Villela et al. (2019) argumentam que a Engenharia de Requisitos (ER) terá que evoluir em várias dimensões e, assim, tornar-se ubíqua.

Os softwares se tornaram presentes na ampla maioria das corporações, sendo raras as empresas que não possuem algum nível de automação. As corporações lidam atualmente com sistemas cada vez mais diversificados, complexos e interconectados. Enquanto isso, a demanda por rápidas inovações exige ciclos cada vez mais curtos para a obtenção de *feedback* (VILLELA et al., 2019). A propagação dos softwares em ambientes *business-to-consumer* e *business-to-business* torna difícil envolver o crescente número de partes interessadas. Técnicas tradicionais de elicitação de requisitos, como entrevistas ou grupos de foco, apresentam problemas de escalabilidade e limitação quando necessitam ocorrer continuamente envolvendo o crescente número de partes interessadas (Villela et al., 2018).

A Engenharia de Requisitos destaca-se como uma das áreas de maior criticidade para o resultado dos projetos de software (MULLER; FORNO, 2017a). Fatores como definição de objetivos, planejamento do projeto, envolvimento e identificação das necessidades dos usuários são aspectos determinantes para o sucesso dos projetos (HASTIE; WOJEWODA, 2015a; MULLER; FORNO, 2017b). A incorreta aplicação da Engenharia de Requisitos é um dos principais motivos para falhas em projetos, aumentando o tempo e consequentemente o custo de desenvolvimento (ELIZABETH H. JEREMY D., 2017). A gestão apropriada dos requisitos aumenta as chances de êxito dos projetos. Pesquisas indicam que 30% dos fatores de sucesso dos projetos estão relacionados diretamente com os processos da Engenharia de Requisitos (HASTIE; WOJEWODA, 2015b).

Uma opção para apoiar os engenheiros de requisitos é o reuso de software. Sistemas de

Recomendação para Engenharia de Software (SRESs) ajudam as equipes a selecionar informações e tomar decisões quando estas não possuem experiência, ou não podem considerar todos os dados disponíveis (ROBILLARD et al., 2014). Pois, analisar informações históricas despende um tempo significativo quando realizada manualmente, o que consequentemente torna-se um método pouco utilizado pelas equipes (PORTUGAL et al., 2017; WHITEHEAD, 2007). No entanto, estabelecer contexto é um desafio geral para os sistemas de recomendação (ROBILLARD et al., 2014).

Dessa forma, este trabalho utiliza conceitos estabelecidos pela Computação Ubíqua (Satyanarayanan, 2001), fazendo uso dos históricos de contextos (DEY; ABOWD; SALBER, 2001; BARBOSA et al., 2018) dos projetos como meio para auxiliar as equipes durante os processos da Engenharia de Requisitos. O estudo, através da identificação de características dos projetos e análise de similaridade dos seus históricos de contextos, apresenta a recomendação de novos requisitos de projeto como uma ferramenta inteligente de apoio aos envolvidos durante o gerenciamento de projetos de software.

O trabalho apresenta um modelo computacional de recomendação intitulado Nhatos. O trabalho se difere da literatura anterior ao explorar a formação de históricos de contextos (BARBOSA et al., 2018) e análise de similaridade entre projetos para auxiliar os processos da ER através da recomendação de contextos futuros. Assim, novos requisitos são recomendados, tanto nas fases iniciais quanto ao longo do ciclo de vida dos projetos, cobrindo todos os processos da Engenharia de Requisitos.

A pesquisa possui as seguintes questões de pesquisa (QP):

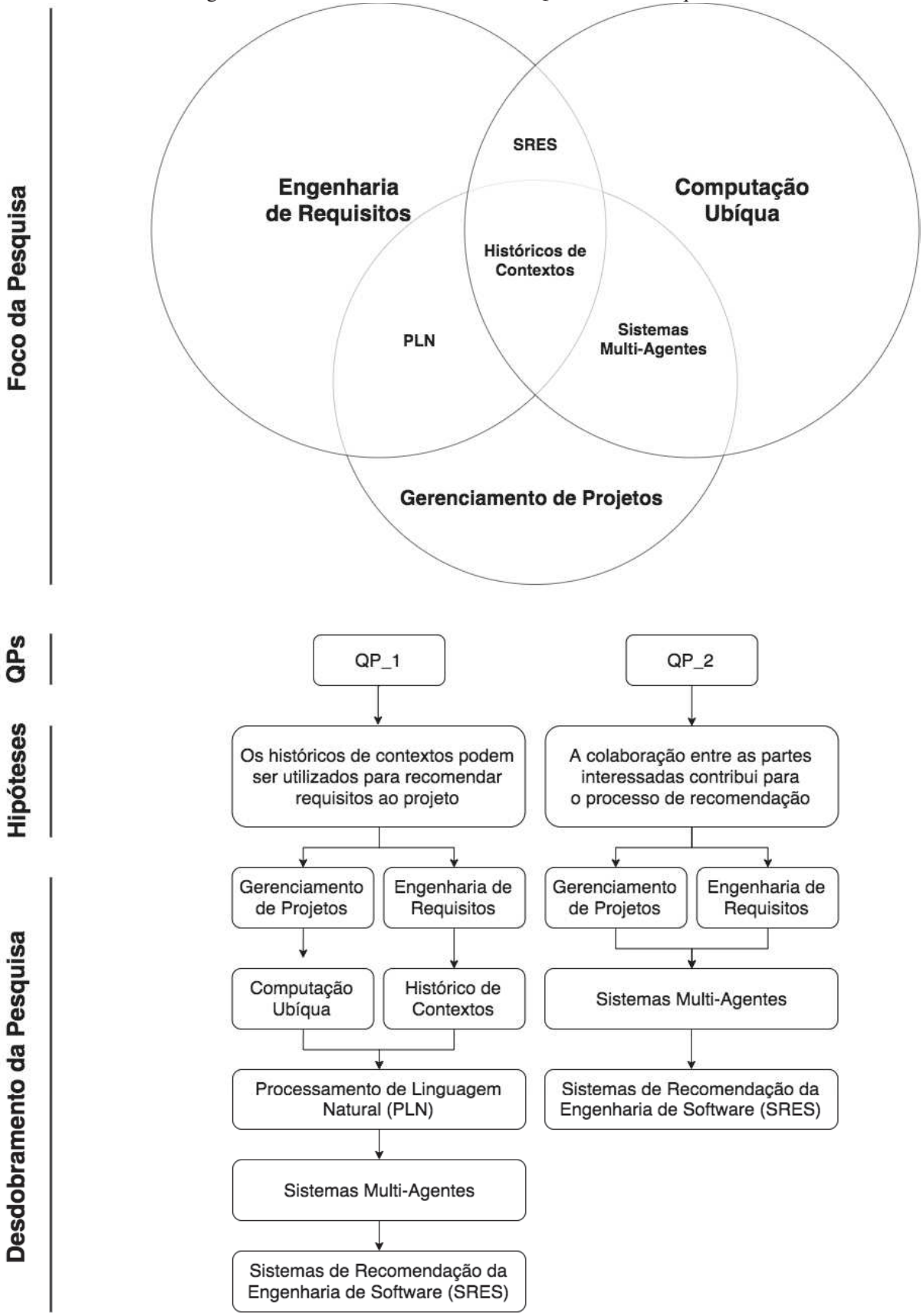
- QP 1. “É possível utilizar os históricos de contextos dos projetos para inferir requisitos, ao longo dos processos da Engenharia de Requisitos, considerando as características e similaridade dos projetos?”;

- QP 2. "A colaboração entre as partes interessadas contribui para o processo de recomendação?".

O uso dos históricos de contextos dos projetos pode fornecer informações importantes sobre projetos executados no passado, além de permitir uma análise sobre as lições aprendidas. Assim, o modelo proposto tem como objetivo identificar requisitos entre projetos similares, permitindo a recomendação de requisitos. Assim, os gestores podem tomar suas devidas ações, para alcançar os objetivos do projeto, estando de posse de informações contextualizadas.

A Figura 1 apresenta a concentração do foco de pesquisa, tais como Engenharia de Requisitos, Gerenciamento de Projetos, Computação Ubíqua, Históricos de Contextos, Sistemas Multi-Agentes, Sistemas de Recomendação para Engenharia de Software e Processamento de Linguagem Natural. Estes conceitos, através de suas intersecções, dão origem ao Modelo Nhatos. A figura também apresenta o desdobramento das questões de pesquisa elaboradas (QP 1 e QP 2) e as hipóteses definidas para cada questão ao longo da pesquisa que envolve este trabalho.

Figura 1 – Foco e Desdobramento das Questões de Pesquisa



Fonte: Elaborado pelo autor

1.3 Objetivos

O objetivo deste trabalho é a criação de um modelo computacional para possibilitar a recomendação de requisitos durante o gerenciamento de projetos de software. Este modelo, intitulado Nhatos, baseia-se nos conceitos propostos pela área da Computação Ubíqua. Onde, o contexto de ambiente é analisado para prover melhores tomadas de decisões, tanto aos gerentes aos projetos como às demais partes interessadas. Assim sendo, o modelo visa fornecer informações úteis aos engenheiros de requisitos, durante os processos da área de Engenharia de Requisitos.

O modelo Nhatos leva em consideração uma detalhada revisão da literatura. Para tanto, um mapeamento sistemático dos principais trabalhos relacionados aos temas de estudo desta pesquisa foi reproduzido. Este estudo reuniu os trabalhos presentes nas principais bases de dados acadêmicas, o que permitiu analisar diversos aspectos destes trabalhos, assim como identificar *insights* e lacunas de pesquisas que contribuíram para a composição dos objetivos do presente estudo.

O trabalho se concentra nos seguintes objetivos específicos: (1) Modelar e Implementar o Modelo Nhatos; (2) Desenvolver um protótipo que atenda aos requisitos e elementos definidos pelo modelo implementado; (3) Elaborar os cenários de validação do modelo; (4) Validar o modelo através do uso de um protótipo em um ambiente real de projetos na indústria.

1.4 Metodologia

Visando atingir os objetivos propostos, algumas etapas precisaram ser executadas no decorrer de uma extensa pesquisa para dar origem ao trabalho. Portanto, cinco etapas foram definidas: (1) Estudo de mapeamento sistemático; (2) Pesquisa com profissionais da área de gerenciamento de projetos (3) Definição e detalhamento do modelo; (4) Desenvolvimento de um protótipo para atender os requisitos originados pelo modelo; (5) Avaliação do modelo em um ambiente real de gerenciamento de projetos de software.

A primeira etapa envolveu a realização de um estudo de mapeamento sistemático sobre o tema de "Recomendação em Engenharia de Requisitos". O estudo determinou a forma como os trabalhos relacionados oferecem suporte à recomendação, bem como as tecnologias e informações utilizadas pelos autores. Outro aspecto observado foi a aplicação das avaliações e seus respectivos ambientes, assim como as informações inseridas que originaram as recomendações.

A segunda etapa consistiu em coletar a visão dos profissionais de gerenciamento de projetos sobre a proposta geral deste estudo. Procurou-se observar se existe na indústria uma aceitabilidade ou necessidade para aplicação de um sistema inteligente que forneça recomendações no decorrer dos processos da Engenharia de Requisitos.

A terceira etapa foi composta pelo detalhamento do modelo Nhatos. Nesta etapa foram especificados os componentes do modelo, seus relacionamentos e de que forma contribuem

para que o objetivo do Nhatos seja alcançado.

Uma vez especificados os componentes, foi detalhada a construção de um protótipo. Assim, a quarta etapa detalha o desenvolvimento de um protótipo que contenha as funcionalidades necessárias para possibilitar, em um ambiente de projetos, a recomendação de requisitos em projetos de software. Para tanto, necessitou-se desenvolver os recursos básicos para gerenciamento dos projetos, requisitos, atividades e alocação de recursos. O protótipo tem por principal objetivo possibilitar a validação do modelo. Através da utilização do protótipo, os envolvidos registram as informações relativas aos seus projetos, como: (a) projetos e características; (b) requisitos; (c) atividades; (d) recursos; (e) envolvidos. Esta interação ocorre através do uso de dispositivos móveis e computadores convencionais, sendo estes utilizados no ambiente onde ocorrem os projetos.

A quinta etapa consiste na exposição do protótipo e sua devida observação em funcionamento. Portanto, as recomendações geradas são analisadas em diferentes cenários. A aceitação das saídas originadas pelo modelo possibilita estimar a avaliação funcional do protótipo durante a recomendação de requisitos.

1.5 Organização do Texto

O presente trabalho está organizado da seguinte forma: o capítulo 2 aborda a fundamentação teórica, contendo os conceitos principais do tema deste estudo. O capítulo 3 apresenta um estudo de mapeamento sistemático, o qual possibilitou a identificação dos trabalhos relacionados ao tema e a comparação destes trabalhos com o presente estudo. No capítulo 4 é apresentado o modelo Nhatos. O capítulo 5 apresenta a avaliação do modelo e os respectivos resultados obtidos. Por fim, o capítulo 6 apresenta a conclusão deste estudo.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo aborda os principais conceitos relacionados ao estudo. A Engenharia de Requisitos é o tema de aplicação do trabalho. Gerenciamento de Projetos, Computação Ubíqua e Históricos de Contextos são explorados para possibilitar o entendimento, classificação e similaridade do domínio dos projetos. Processamento de Linguagem Natural é a tecnologia adotada para explorar os conteúdos armazenados durante o estudo através de Sistemas Multi-Agentes. Por fim, o conceito de Softwares de Recomendação da Engenharia de Requisitos é aplicado durante o uso das interfaces de aplicações pelos usuários.

2.1 Engenharia de requisitos

O Gerenciamento do Escopo do projeto inclui os processos necessários para assegurar que o projeto contenha o trabalho necessário para alcançar seu sucesso (PMI, 2017a). Além de planejar e definir o escopo do projeto, o Gerenciamento do Escopo também abriga o Gerenciamento de Requisitos (PMI, 2017b). Na engenharia de software, o processo de descoberta, análise, documentação e monitoração das necessidades de um sistema é denominado Engenharia de Requisitos (ER) (SOMMERVILLE, 2016).

Sommerville (2016) estabelece quatro atividades principais que compreendem os processos de ER, sendo estas:

- Elicitação: Relaciona-se aos métodos empregados para que se obtenha conhecimento sobre o domínio e os requisitos do projeto, como fontes e técnicas. As várias fontes de conhecimento do domínio incluem clientes, manuais de negócios, o software existente do mesmo tipo, padrões e outras partes interessadas do projeto. As técnicas usadas para obter requisitos são entrevistas, *brainstorming*, análise de tarefas, prototipagem, etc;

- Especificação: Esta atividade é usada para produzir modelos formais de requisitos de software. Todos os requisitos, incluindo os requisitos funcionais e não funcionais, assim como suas restrições, são especificados por estes modelos. Os modelos usados neste estágio incluem diagramas de entidade e relacionamento, diagramas de fluxo de dados, diagramas de decomposição de funções, dicionários de dados, entre outros;

- Validação: Refere-se a um conjunto diferente de tarefas que garantem que o software que foi construído seja rastreável aos requisitos do cliente. Se os requisitos não forem validados, os erros nas definições de requisitos serão propagados para outros estágios, possivelmente resultando em modificações e retrabalhos;

- Monitoramento ou Gerenciamento: Define-se pelo processo de analisar, documentar, acompanhar, priorizar e autorizar requisitos, controlando a comunicação com as partes interessadas relevantes. Este estágio trata da variabilidade dos requisitos. É necessário

> garantir que a especificação de requisitos seja a mais maleável possível, de modo a incorporar mudanças nos requisitos especificados pelos usuários finais em estágios posteriores. Ser capaz de modificar o software, de acordo com os requisitos de maneira sistemática e controlada, é uma parte extremamente importante do processo de Engenharia de Requisitos.

Independente da utilização de metodologia (ágil ou tradicional) empregada durante o GP, os requisitos são normalmente passados dos clientes para os desenvolvedores em formato de linguagem natural. Pesquisas indicam que cerca de 79% dos requisitos são escritos em linguagem natural (LUISA; MARIANGELA; PIERLUIGI, 2004). Por isso, este trabalho utiliza Processamento de Linguagem Natural para a classificação deste tipo de conteúdo não estruturado.

2.2 Processamento de Linguagem Natural

A linguagem é um fenômeno que envolve processos variados como o reconhecimento de sons ou textos impressos, análise sintática, inferências semânticas, entre outros. Existem diferentes níveis de análise que podem ser aplicadas à linguagem natural. Entre eles: morfologia (estudo dos componentes que compõem as palavras), sintaxe (estuda as regras para combinar palavras em sentenças válidas) e semântica (considera o significado das palavras e sentenças) (LUGER, 2016).

O processamento da Linguagem Natural (PLN) é uma das subáreas da Inteligência Artificial (IA). Este tópico tornou-se um dos campos ativos da ciência da computação, onde observam-se diversos estudos relacionados nos últimos anos (Reshma; Remya, 2017; TRENDS, 2019a). O PLN é caracterizado pelo desenvolvimento de modelos computacionais para a realização de tarefas que dependem de informações expressas em alguma língua natural. Por exemplo, tradução e interpretação de textos, busca de informações em documentos e interface homem-máquina (LUGER, 2016).

Os algoritmos de computação de PLN possuem capacidade de compreensão da linguagem em diversos idiomas. Softwares baseados nessa tecnologia possuem mecanismos para compreender frases em linguagem natural e, assim, executar comandos ou classificar informações, possibilitando a análise de conteúdos textuais não estruturados (LUGER, 2016).

O PLN é uma tecnologia que permite aos computadores entender as linguagens humanas a partir da análise gramatical e semântica. A análise gramatical e semântica de nível profundo geralmente usa as palavras como unidade básica e a segmentação de palavras é geralmente a principal tarefa do PLN para agrupar ou classificar conteúdos (Wang; Su; Yu, 2020). Como conteúdos do tipo não estruturado são os mais utilizados pelas equipes de projetos durante os processos da Engenharia de Requisitos (LUISA; MARIANGELA; PIERLUIGI, 2004), esta tecnologia apresenta-se como promissora para efetuar análises deste tipo de conteúdo.

2.3 Sistemas Multi-Agentes

Um sistema de computador que está situado em algum ambiente, e que é capaz de agir de maneira autônoma neste ambiente, a fim de cumprir os objetivos para os quais foi desenvolvido, é considerado um agente de software (PADGHAM; WINIKOFF, 2004). Um agente para ser considerado inteligente deve ser reativo, proativo e social. Portanto, um agente inteligente deve responder em tempo hábil às mudanças em seu ambiente, perseguir persistentemente suas metas e interagir com outros agentes (WOOLDRIDGE, 2009).

Os agentes são aplicáveis em situações em que o ambiente é dinâmico ou imprevisível, em que ações autônomas se fazem necessárias de acordo com mudanças impostas pelo ambiente. A proatividade e reatividade tornam os agentes mais humanos na maneira em como lidam com os problemas. Isto levou a um número de aplicações em que os agentes de software são usados como substitutos para humanos em certos domínios limitados (PADGHAM; WINIKOFF, 2004).

2.4 Sistemas de Recomendação para Engenharia de Software

Os aplicativos que utilizam os conceitos de Sistemas de Recomendação visam apoiar os usuários em suas tomadas de decisão. Estes aplicativos interagem com grandes bases de dados. Eles recomendam itens de interesse para os usuários, com base nas preferências que eles expressaram, explícita ou implicitamente (ROBILLARD et al., 2010). Um sistema de recomendação para engenharia de software (SRES) é um aplicativo de software que fornece itens de informação estimados como valiosos para uma tarefa de engenharia de software em um determinado contexto (GASPARIC; JANES, 2016).

De acordo com Gasparic e Janes (2016), um sistema de informação útil processa alguma forma de entrada e gera uma determinada saída. Segundo Venkatesh et al. (2003), o tipo de entrada necessária e o tipo de saída produzida influenciam a aceitação por parte dos usuários deste sistema. Portanto, SRESs apoiam os engenheiros de software em seus processos, relacionando sua saída aos interesses destes usuários. Os envolvidos podem expressar seus interesses explicitamente através de uma consulta direta ou implicitamente através de ações. O SRES considera estes dados durante realização das suas recomendações. A seleção destas informações destaca um desafio geral para os sistemas de recomendação. Pois, um SRES necessita estabelecer contexto, incluindo todas as informações relevantes sobre o usuário, seu ambiente de trabalho e o status do projeto ou tarefa no momento da recomendação (ROBILLARD et al., 2010).

Além de informações de contexto, um SRES pode analisar e trabalhar com dados adicionais para realizar as suas recomendações. Pode ser considerado pelo aplicativo incluir o código-fonte do projeto, o histórico completo de alterações no sistema, artefatos como *e-mails* publicados em listas de discussão e relatórios de *bugs*, dados de interação acumulados em muitas sessões de programação, relatórios de cobertura de teste e bases de código externas ao projeto

e informações obtidas durante o gerenciamento do projeto. (ROBILLARD et al., 2010).

2.5 Gerenciamento de Projetos, Computação Ubíqua e Históricos de Contextos

Ainda que um requisito sempre esteja vinculado a um projeto, seu ciclo de vida muitas vezes é iniciado antes mesmo da criação do projeto. O requisito pode originar-se na camada de portfólio, na gestão estratégica ou durante a execução de um outro projeto. O requisito pode ser atribuído futuramente ao projeto, para assim compor um produto ou serviço (PMI, 2017b).

O ambiente de projetos é um ambiente altamente variável (PMI, 2017a). Sendo assim, entender o domínio do projeto e seu contexto é crucial para a Engenharia de Requisitos. O contexto pode ser determinado como qualquer informação relevante para a interação entre o usuário e o ambiente no qual o mesmo se encontra (DEY; ABOWD; SALBER, 2001). Informações de contexto são dados suscetíveis às mudanças de estado (Satyanarayanan, 2001). No ambiente de projetos de software, diversas informações apresentam mudanças de estado, como quantidade de envolvidos em um projeto, percentual de evolução, prazos ou tamanho da equipe. Deste modo, estas informações podem ser consideradas informações de contexto do projeto ao longo do seu ciclo de vida.

Contexto pode ser considerado como qualquer informação relevante para a interação entre o usuário e o ambiente em que ele se encontra (DEY; ABOWD; SALBER, 2001; NARENDRA et al., 2005). Considerando que um sistema de computação ubíqua deve ser minimamente intrusivo, este sistema deve estar ciente de seu contexto. Deve ainda, estar ciente do estado dos usuários e do ambiente em que estes se encontram, adaptando-se ou provendo informações relevantes de acordo com essas informações (Satyanarayanan, 2001). O contexto de um usuário, projeto ou requisito pode conter um grande número de informações, como por exemplo, atributos do local físico, histórico pessoal, padrões, hábitos de comportamento, entre outros. Em suma, características que mudam conforme a passagem do tempo e interferência do ambiente (BARBOSA et al., 2018).

A questão chave para a ciência de contextos é a obtenção de informações diversas a respeito dos usuários, ambiente, hardware e software para que o sistema trabalhe de modo sensível ao contexto. Em determinados casos, as informações desejadas já podem fazer parte dos dados do projeto ou equipes (BARBOSA, 2015). Algumas técnicas utilizam dados históricos para prever novos contextos. Além do contexto presente, são armazenadas informações sobre os contextos passados (AMEYED; MIRAOUI; TADJ, 2015; BURBEY; L. MARTIN, 2012). A estratégia de recomendação de contextos tem se destacado através da aplicação de análise de similaridade (ROSA; BARBOSA; RIBEIRO, 2016; WIEDEMANN TIAGO; BARBOSA, 2014). O modelo Nhatos se projeta neste mesmo cenário.

Informações de contexto são normalmente classificadas em três subconjuntos, chamados domínios de contexto (FOURNIER et al., 2006): domínio de usuário, domínio de sistema e domínio físico.

- o Domínio de Usuário: provê o conhecimento para que as aplicações possam se adaptar de acordo com o comportamento dos usuários;
- o Domínio de Sistema: descreve os recursos de software e hardware disponíveis aos usuários. As aplicações se utilizam das descrições e características dos dispositivos e recursos para adaptar o comportamento do sistema, alterar seu conteúdo ou modificar as interfaces para os usuários;
- por fim, o Domínio Físico ou ambiente, trata da descrição do local e das condições do ambiente físico. A mobilidade do usuário ou uma variação natural das condições físicas podem alterar o comportamento de uma aplicação dentro de um ambiente ciente de contexto.

No cenário de gerenciamento dos projetos, o paradigma da computação ubíqua passa a dar suporte à gestão dos projetos, auxiliando para uma melhor condução dos mesmos. Ao longo dos últimos anos, diversos estudos exploram este tema (TRENDS, 2019b). Os conceitos da computação ubíqua podem auxiliar em áreas específicas, como a gestão de tempo, auxiliando a montagem de cronogramas (LUZ et al., 2009), gestão de riscos (Filippetto et al., 2016) ou monitoramento e controle (KASPERBAUER et al., 2013), por exemplo. Sistemas baseados na computação ubíqua também podem atuar em mais de uma área, auxiliando os gestores de projetos em diferentes necessidades, como apresentado por Batista et al. (2011) e FILIPPETTO, LIMA e BARBOSA (2011), onde modelos que contemplam o ciclo de vida de um projeto são apresentados. Diferentes técnicas relacionadas à computação ubíqua podem ser utilizadas para auxiliar os times na gestão de seus projetos, como por exemplo, o conceito de sensibilidade ao contexto, onde um cronograma ou *Dashboard* do projeto podem ser adaptados de acordo com a localização ou perfil dos envolvidos no projeto. Além disso, o uso do histórico de contextos dos projetos pode identificar lições aprendidas em projetos passados, a fim de auxiliar durante a condução de novos projetos. O uso da recomendação de contextos visa identificar situações que possam ocorrer durante a execução do projeto e assim auxiliar as equipes durante o ciclo de vida dos projetos.

2.6 Considerações sobre o Capítulo

Este capítulo apresentou os principais conceitos que envolvem esta pesquisa. Abordou-se os temas: Engenharia de requisitos, Processamento de Linguagem Natural, Histórico de Contextos, Computação Ubíqua em Gerenciamento de Projetos, Sistemas Multi-Agentes e Softwares de Recomendação para Engenharia de Software. O próximo capítulo apresenta um estudo de mapeamento sistemático que envolve trabalhos relacionados aos temas de estudo apresentados.

3 TRABALHOS RELACIONADOS

Durante a seleção dos trabalhos relacionados, realizou-se um estudo de mapeamento sistemático sobre as pesquisas que envolvem a área de Engenharia de Requisitos, com foco em recomendação de requisitos de software. O mapeamento sistemático é uma forma estruturada para a identificação do estado da arte em uma determinada área, utilizada para compreensão dos estudos realizados, assim como tendências sobre o tema pesquisado. Neste mapeamento será apresentada uma visão geral das tecnologias ou modelos pesquisados na literatura classificando os trabalhos desenvolvidos quanto à área de gestão de requisitos.

Deste modo, o estudo de mapeamento sistemático realizado foi organizado em três etapas: (1) identificação de questões de pesquisa; (2) elaboração da estratégia de busca; (3) definição dos critérios para a filtragem dos resultados.

3.1 Mapeamento Sistemático

O mapeamento sistemático reduz o viés quando comparado às referências únicas, obtendo assim resultados mais confiáveis (PETERSEN; VAKKALANKA; KUZNIARZ, 2015). A metodologia do estudo consiste na execução dos seguintes passos:

- Elaboração das questões de pesquisa;
- Composição do processo de busca;
- Definição dos critérios para filtrar os resultados.

3.1.1 Questões de Pesquisa

As questões de pesquisa foram separadas em três categorias principais: questões gerais (QG), questões de foco (QF) e questões estatísticas (QE). Para cada categoria, duas questões foram elaboradas.

A primeira questão geral procura responder em quais processos da área de Engenharia de Requisitos os estudos estão concentrados, sendo estes: (a) Elicitação; (b) Especificação; (c) Verificação e validação; e (d) Gerenciamento. A segunda questão geral tem por objetivo identificar o método empregado para realizar as recomendações. Por exemplo: uso do histórico de um sistema de domínio específico, preferência dos usuários ou colaboração entre os envolvidos.

A primeira questão foco visa responder quais foram os recursos computacionais aplicados nos estudos para tornar possível a recomendação. Por exemplo, processamento de linguagem natural, ciência de dados, técnicas de análise de conteúdos não estruturados, etc. A segunda questão foco procura descobrir como os autores avaliaram seus estudos, seja utilizando um ambiente acadêmico ou da indústria.

Tabela 1 – Questões de Pesquisa

Questões Gerais	
QG 1	Em que atividades de ER os estudos se concentram?
QG 2	Quais dados são utilizados para apoiar a recomendação?
Questões Foco	
QF 1	Quais recursos tecnológicos foram utilizados para durante a recomendação em ER?
QF 2	Como os sistemas de recomendação foram avaliados?
Questões Estatísticas	
QE 1	Onde os artigos foram publicados?
QE 2	Qual a quantidade anual de publicações?

Fonte: Elaborado pelo autor

As duas questões estatísticas identificam em quais periódicos os estudos foram publicados e a quantidade anual de publicações. O período pesquisado foi um período de doze anos. As questões de pesquisa estão listadas na Tabela 1.

3.1.2 Estratégia de Busca

A estratégia de busca foi definida em três etapas: especificação da *string* de busca, escolha das bases de dados e coleta dos resultados. A estratégia de busca, proposta por Petersen et al. (2008), foi adotada nesta fase do estudo. A primeira etapa identifica os principais termos e seus sinônimos mais relevantes. Foram selecionados *Engenharia de Requisitos* e *Recomendação* como os termos principais deste trabalho. Os sinônimos para Engenharia de Requisitos baseiam-se nos três termos principais que se referem à área de Engenharia de Requisitos proposta por Elizabeth H. Jeremy D. (2017): (a) Análise de requisitos; (b) Gerenciamento de requisitos; e (c) Avaliação de requisitos. Para o termo Recomendação, realizou-se uma busca por sinônimos nos dicionários online Thesaurus (2009) e Cambridge (2017). A Tabela 2 mostra os termos e sinônimos resultantes no final desta etapa.

Tabela 2 – *Strings* de Pesquisa

Termos da *string* de pesquisa	
Termos principais	Sinônimos
Requirements Engineering	“requirement engineering” OR “requirement analysis” OR “requirement management” OR “requirement assessment”
Recommendation	prediction OR prevision OR recommendation OR prognosis OR forecast OR anticipation

Fonte: Elaborado pelo autor

A partir da definição dos termos e sinônimos, a *string* de busca foi elaborada. A seguir a

composição da *string* que foi utilizada como parâmetro de pesquisa nas bases de dados selecionadas neste estudo. A Tabela 3 mostra a sequência de busca gerada a partir da definição de termos e sinônimos para cada base de dados.

Tabela 3 – *Strings* de Pesquisa e Bases de Dados

Base de Dados	***String***
Science Direct, IEEE, Springer	("requirement engineering"OR "requirement analysis"OR "requirement management"OR "requirement assessment") AND ("prediction"OR "prevision"OR "recommendation" OR "prognosis"OR "forecast"OR "anticipation")
ACM	+("requirement engineering"OR "requirement analysis"OR "requirement management"OR "requirement assessment") +("prediction"OR "prevision"OR "recommendation"OR "prognosis"OR "forecast"OR "anticipation")
Google Scholar	allintext: ("requirement engineering"OR "requirement analysis" OR "requirement management"OR "requirement assessment") AND (prediction OR prevision OR recommendation OR prognosis OR forecast OR anticipation)
Scopus	(ALL ("requirement engineering"OR "requirement analysis" OR "requirement management"OR "requirement assessment") AND ALL ("prediction"OR "prevision"OR "recommendation" OR "prognosis"OR "forecast"OR "anticipation"))

Fonte: Elaborado pelo autor

A seguir é apresentada a *string* de pesquisa: ("requirement engineering" OR "requirement analysis" OR "requirement management" OR "requirement assessment") AND ("prediction" OR "prevision" OR "recommendation" OR "prognosis" OR "forecast" OR "anticipation")

Os parâmetros de pesquisa aplicados às bases de dados foram determinados a partir da *string* apresentada. Portanto, o passo seguinte foi selecionar as bases de dados. Selecionou-se bases de dados que continham artigos da área de computação. Seis bases de dados de pesquisa foram utilizadas durante o estudo: *ACM Digital Library*, *IEEE Xplore*, *Science Direct*, *Springer Link*, *Google Scholar* e *Scopus Elsevier*. Estas são bases de dados renomadas na academia, que contém artigos do campo de estudo de ciência da computação.

A busca na *ACM Digital Library*, no *IEEE Xplore*, *Springer*, *Scopus Elsevier* e *Google Scholar* exigiu o uso de recursos de pesquisa avançados. A cadeia de pesquisa foi inserida no campo de edição avançada. Em todas as bases de dados, os campos de resumo e título e conteúdo dos artigos foram considerados.

O processo de pesquisa no repositório do *Science Direct* envolveu a aplicação da *string* de consulta ao título, resumo e palavras-chave sem o uso de pesquisa avançada. Por fim, na Biblioteca *Springer*, foram removidos documentos categorizados como *Preview Only*. Em seguida, o filtro de pesquisa intitulado *Computer Science* foi selecionado para obter resultados compatíveis com a área do presente estudo.

3.1.3 Filtragem dos Resultados

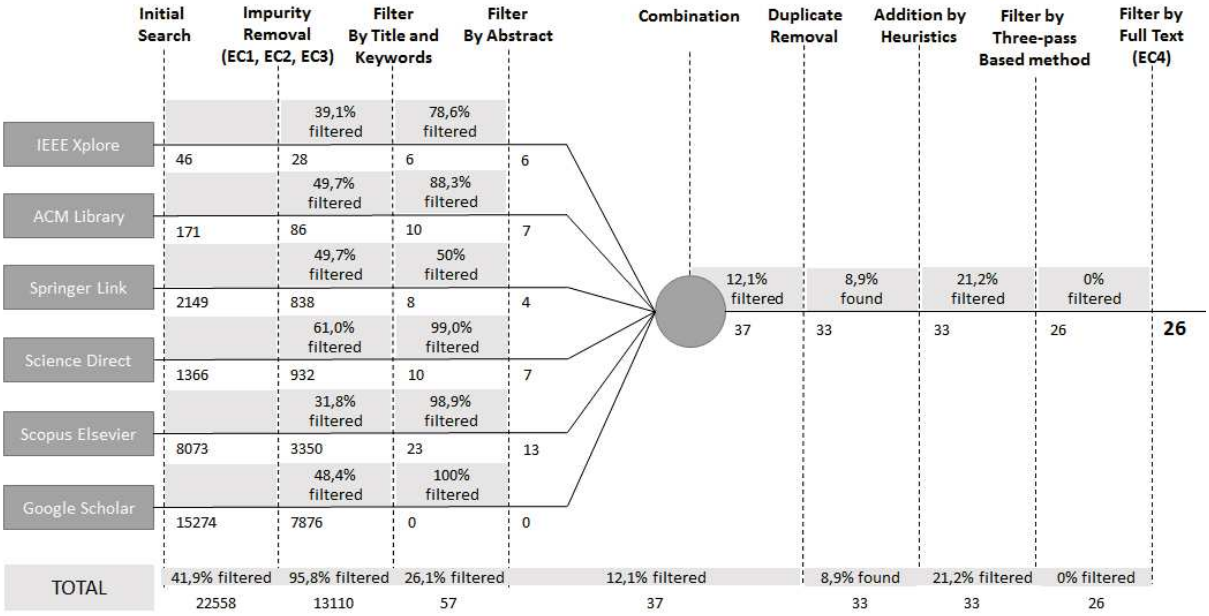
Os estudos foram filtrados com o objetivo de selecionar os artigos que apresentassem maior foco no tema de "Recomendação em Engenharia de Requisitos". Para tanto, alguns Critérios de Inclusão (CI) foram definidos para validar a aderência dos trabalhos quanto às suas respectivas contribuições e relevância. A seguir são listados os critérios de inclusão aplicados nesta fase do estudo:

- CI 1: O estudo deve ter sido publicado em uma conferência, *workshop* ou *journal*;
- CI 2: O estudo deve ser um artigo completo;
- CI 3: O estudo deve ser da área de ciência da computação;
- CI 4: O estudo deve abordar o tema de recomendação em Engenharia de Requisitos.

Os Critérios de Exclusão (CE) foram definidos, conforme segue:

- CE 1: Os estudos que foram publicados antes de 2017;
- CE 2: Os estudos que não foram escritos em inglês;
- CE 3: Os estudos que não possuem relação com as questões de pesquisa.

Figura 2 – Processo de Filtragem



Fonte: Elaborado pelo autor

A Figura 2 mostra o processo de filtragem. Os critérios de inclusão (CI) e critérios de exclusão (CE) foram aplicados em cada etapa deste estudo de mapeamento sistemático. Estes

critérios auxiliam durante a aplicação dos filtros para obter os estudos relacionados à pesquisa e eliminar eventuais desvios. Os estudos resultantes no processo de busca foram filtrados, retirando impurezas que não atendiam aos critérios de inclusão (CI1, CI2, CI3 e CI4). Os demais estudos foram armazenados no software *Mendeley Desktop* (MENDELEY, 2017), onde os trabalhos foram organizados em coleções distintas de acordo com cada base de dados pesquisada.

Posteriormente, os artigos foram filtrados por título e resumo. Então, os estudos duplicados foram removidos. Dois trabalhos foram adicionados por heurística, considerada a sua compatibilidade com o estudo, apesar de não terem sido encontrados no processo de busca (DUMITRU et al., 2011; HERRERA et al., 2008).

Após a aplicação das *strings* de pesquisa em cada base de dados, foi realizado o *download* dos resultados diretamente dos *sites* das bases de dados em formato CSV (*Comma Separated Value*). Os arquivos de cada base de dados foram agrupados em uma única planilha do software *Microsoft Excel*. Neste arquivo, apenas o título, autor, resumo e ano das colunas de publicação foram selecionados.

O conteúdo desta planilha foi analisado a partir da aplicação dos filtros de pesquisa. As palavras-chave de *string* de pesquisa foram parametrizadas nesses filtros. Desta forma, as impurezas foram removidas. Uma vez removidas as impurezas, foi possível realizar o próximo passo da análise. Esta etapa consiste em aplicar os critérios de inclusão e exclusão.

Em seguida, cada estudo foi filtrado com base nas duas primeiras etapas da abordagem chamada *"Three Pass Method"*, proposta por Keshav (2007). Este método consiste em uma rápida varredura de cada trabalho seguindo os passos: 1) Ler o título, resumo e introdução; 2) Ler apenas os títulos das seções e subseções, ignorando todo o restante do artigo; 3) Analisar brevemente o conteúdo matemático (se houver), determinando os fundamentos teóricos e metodologias utilizadas pelo autor; e 4) Ler as conclusões.

O próximo passo na leitura da filtragem foi analisar cuidadosamente os diagramas de figuras e outras ilustrações nos artigos. Nesta fase, a atenção foi focada especialmente nas representações gráficas presentes em cada trabalho. A filtragem final consistiu em analisar o texto completo de cada artigo, observando o terceiro critério de exclusão, analisando se os estudos possuíam relação às questões de pesquisa.

As bases de dados *IEEE Xplore* e *ACM Digital Library* trouxeram inicialmente um número menor de estudos relacionados à *string* de pesquisa. No entanto, menos trabalhos foram eliminados durante o processo de filtragem. Já as bases de dados *Scopus Elsevier* e *Google Scholar* tiveram mais de 99% dos resultados eliminados durante o processo de filtragem. Apesar de terem trazido mais resultados durante a aplicação das *strings* de pesquisa, quando comparados com as outras bases de dados, as buscas originaram alta quantidade de artigos não relacionados ao tema deste trabalho.

Após a aplicação da filtragem, a base de dados do *Google Scholar* não permaneceu com nenhum estudo que abordasse o tema de recomendação em Engenharia de Requisitos. No entanto, seus dados foram mantidos no estudo para registrar que esta base de dados foi considerada

durante o processo de filtragem.

Posteriormente, uma planilha do *Microsoft Excel* foi utilizada para extrair os dados dos artigos. As características observadas foram as questões de pesquisa, processos da Engenharia de Requisitos, autores, tecnologias aplicadas, métodos empregados, informações utilizadas pelos autores para subsidiar os trabalhos, ano de publicação, base de dados e objetivos. Nesta etapa, foi realizada a leitura completa de cada artigo uma ou mais vezes. Em cada leitura, as questões de pesquisa foram analisadas individualmente. Desta forma, foi possível reunir as anotações necessárias para compilar os gráficos e informações presentes nos resultados das seções seguintes. Os resultados finais da filtragem são mostrados na Tabela 4, onde o autor, o título e a finalidade de cada artigo são detalhados. Um diretório foi compartilhado para acesso a todos os dados deste estudo (DRIVE, 2019).

Tabela 4: Lista de Artigos por Ordem de Publicação

Referência	**Título**	**Objetivos**	**Ano**
Kim, Dey e Lee (2019)	Ontology-Driven Security Requirements Recommendation for APT Attack	Uma base de conhecimento de ontologia para recomendar requisitos de segurança com base nos casos de ataque e no conhecimento do domínio do sistema.	2019
Zhiying et al. (2019)	Crowdsourcing service requirement oriented requirement pattern elicitation method	Identificação de padrões textuais através da aplicação de lógica *fuzzy*.	2019
Liu et al. (2018)	Mining Android App Descriptions for Permission Requirements Recommendation	Recomendar requisitos de permissões de sistema, que explora possíveis explicações a partir das descrições de aplicativos semelhantes.	2018
Zhao e Zhang (2018)	Collaborative Filtering Service Recommendation Algorithm Based on Trusted User and Recommendation Evaluation	Algoritmo de recomendação de serviço de filtragem colaborativa baseado em usuários confiáveis e avaliação de recomendação.	2018

Continua na próxima página

Tabela 4 – continuando da página anterior

Referência	Título	Objetivos	Ano
Liu et al. (2017a)	A statistical analysis approach to predict user's changing requirements for software service evolution	Prever requisitos com base na descoberta dos objetivos das intenções do usuário, analisando suas ações e seu contexto de ambiente	2017
Wang et al. (2017a)	A Regression Model Based Approach for Identifying Security Requirements in Open Source Software Development	Prever requisitos não funcionais aplicando filtro de regressão de análise textual	2017
Wang et al. (2017b)	Automated Software Security Requirements Recommendation Based on FT-SR Model	Recomendar requisitos de segurança não funcionais com base nos atributos dos requisitos funcionais	2017
Bakar et al. (2016)	Extracting features from online software reviews to aid requirements reuse	Recomendar funcionalidades baseadas em revisões, requisitos, produtos e descrições através de processamento de texto, marcação e agrupamento	2017
Portugal et al. (2017)	GH4RE: Repository recommendation on Github for requirements elicitation reuse	Recomendar requisitos do histórico do projeto do Github (arquivos Readme)	2017
Williams et al. (2017)	Mining Twitter Feeds for Software User Requirements	Prever os requisitos do usuário a partir do resumo e coleta de dados da rede social (tweets)	2017

Continua na próxima página

Tabela 4 – continuando da página anterior

Referência	Título	Objetivos	Ano
Sagrado et al. (2018)	Stability prediction of the software requirements specification	Fornecer não conformidade de definições de requisitos a partir do uso de uma ferramenta comercial para gerenciamento de requisitos (InSCo Requisite) e um protótipo que usa rede Bayesiana	2017
Liu et al. (2017b)	Requirements cybernetics: Elicitation based on user behavioral data	Prever novos recursos com base no comportamento do usuário e nos dados de ambiente (logs)	2017
Garcia et al. (2016)	REQAnalytics: A recommender system for requirements maintenance	Recomendar requisitos a partir do mapeamento semi automático de telas do sistema, requisitos do e uso do software	2016
Hamza et al. (2015)	Recommending features and feature relationships from requirements documents for software product lines	Previsão de funcionalidades dos documentos de requisitos	2015
Blake et al. (2015)	Shared service recommendations from requirement specifications: A hybrid syntactic and semantic toolkit	Recomendar requisitos de software com base na similaridade de textos e descrições	2015
Nakatani et al. (2014)	Predicting requirements changes by focusing on social relations	Antecipar mudanças de requisitos com base nas relações sociais e no contexto do ambiente	2014
Shi et al. (2013)	Learning from evolution history to predict future requirement changes	Prever requisitos que são propensos a mudar com base na história evolutiva	2013

Continua na próxima página

Tabela 4 – continuando da página anterior

Referência	Título	Objetivos	Ano
Zhang et al. (2013)	Non-functional requirement analysis and recommendation for software services	Recomendar serviços de atributos selecionados de requisitos não funcionais	2013
Dumitru et al. (2011)	On-demand feature recommendations derived from mining public product descriptions	Recomendar funcionalidades baseadas na similaridade de produtos (descrições) aplicando Data Mining	2011
Casamayor et al. (2010)	GH4RE: Identification of non-functional requirements in textual specifications: A semi-supervised learning approach	Recomendar requisitos não funcionais da aplicação do processamento de linguagem natural	2010
Felfernig et al. (2010)	Recommendation and decision technologies for requirements engineering	Recomendar requisitos com base nas preferências do usuário	2010
Park et al. (2010)	Requirements attributes to predict requirements related defects	Previsão de defeitos em requisitos baseados na aplicação de Data Mining no repositório de software	2010
Herrera et al. (2009a)	A recommender system for requirements elicitation in large-scale software projects	Recomendar requisitos com base no uso de fórum colaborativo, identificando perfis de usuário	2009
Herrera et al. (2009b)	Enhancing stakeholder profiles to improve recommendations in online requirements elicitation	Recomendar fóruns de elicitação de requisitos com base nos interesses dos usuários	2009
Chen et al. (2008)	An in-process customer utility prediction system for product conceptualisation	Prever necessidades com base na avaliação do usuário e feedback sobre avaliação de uso	2008

Continua na próxima página

Tabela 4 – continuando da página anterior

Referência	Título	Objetivos	Ano
Herrera et al. (2008)	Using data mining and recommender systems to facilitate large-scale, open, and inclusive requirements elicitation processes	Recomendar requisitos através da colaboração de usuários e seus respectivos perfis em um fórum	2008

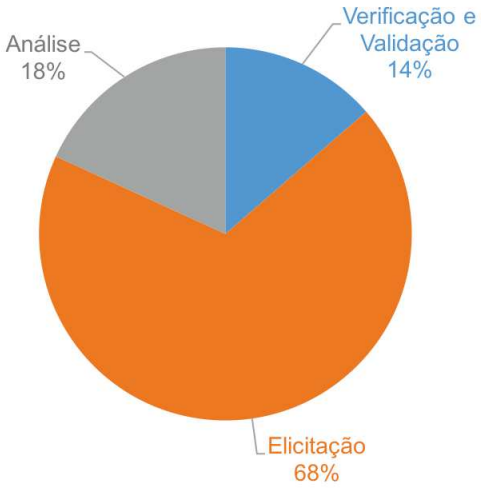
3.2 Resultados

Os resultados do estudo foram agrupados de acordo com as questões de pesquisa. Os resultados extraídos de cada questão de pesquisa são apresentados nas subseções seguintes.

3.2.1 QG 1: Em quais processos de ER os estudos estão concentrados?

A Figura 3 mostra a distribuição dos artigos de acordo com os processos de Engenharia de Requisitos. Os artigos foram classificados de acordo com cada processo em que seus estudos apresentavam concentração. A maioria dos trabalhos abordou o processo de elicitação ou levantamento de requisitos, apresentando um percentual de 68,18% de artigos. As demais atividades, Análise e Especificação, e Validação e Gerenciamento receberam um percentual de 18,18% e 13,63% respectivamente.

Figura 3 – Percentual de Estudos e Processos da Engenharia de Requisitos



Fonte: Elaborado pelo autor

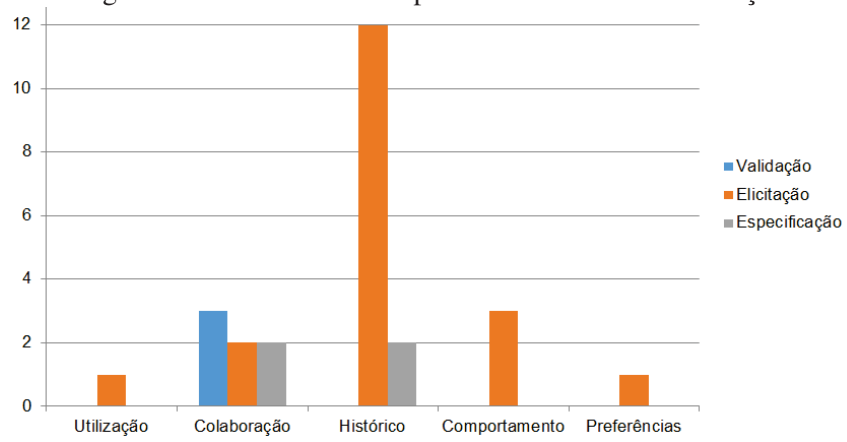
Os trabalhos que se concentraram no estágio de elicitação apresentaram estudos recomen-

dando novos requisitos ou novas funcionalidades. Os estudos que abordaram o processo de Análise apresentaram propostas de recomendação de más definições e defeitos nos requisitos. Os outros trabalhos, focados nos processos de Validação e Gerenciamento, apresentaram abordagens colaborativas como fóruns e análises de perfis de usuários e suas respectivas preferências por recomendações personalizadas.

3.2.2 QG 2: Quais dados são utilizados para apoiar a recomendação?

Os artigos foram agrupados de acordo com as informações utilizadas pelos autores para apoiar os a recomendação durante os processos de Engenharia de Requisitos. A maior parte dos trabalhos utilizou como principal subsídio um histórico com informações dos projetos, somando um percentual de 53,8% dos artigos. Em seguida, identificou-se o uso de colaboração entre os *stakeholders* para as recomendações, compondo um total de 26,9% dos artigos. Outras informações também foram identificadas, como identificação de preferência do usuário, *logs* de utilização do sistema e análise de comportamento dos usuários. A Figura 4 mostra os resultados desta etapa.

Figura 4 – Dados Utilizados para Subsidiar a Recomendação

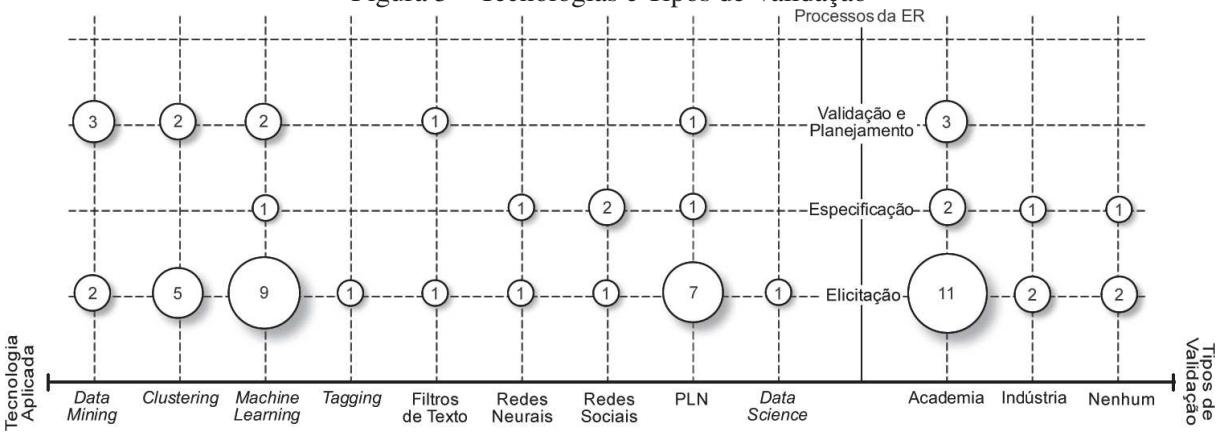


Fonte: Elaborado pelo autor

3.2.3 GF 1: Quais recursos tecnológicos foram utilizados para apoiar a recomendação?

A escrita de requisitos em formato de linguagem natural é normalmente utilizada de acordo com as boas práticas da área RE (ELIZABETH H. JEREMY D., 2017). De acordo com a análise realizada nos trabalhos filtrados, 40,9% utilizaram *Machine Learning*, 31,8% abordaram Processamento de Linguagem Natural e 22,7% utilizaram tecnologia de filtragem de textos ou *Clustering* para extração, categorização e sumarização de dados. A Figura 5 mostra um mapeamento das tecnologias presentes nos estudos e em quais processos/atividades estas foram aplicadas.

Figura 5 – Tecnologias e Tipos de Validação



Fonte: Elaborado pelo autor

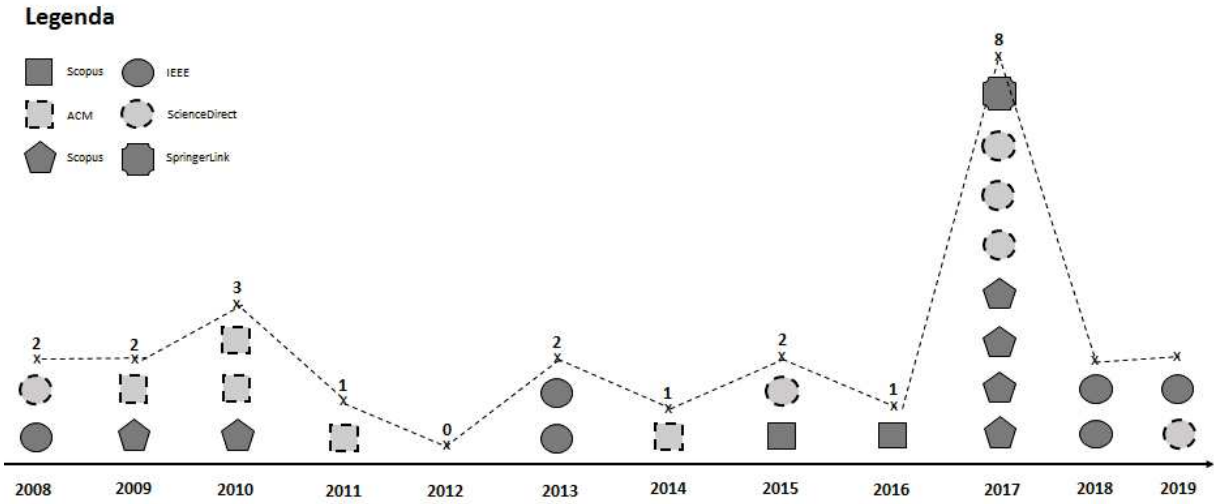
3.2.4 QF 2: Como os sistemas de recomendações estão sendo validados?

A Figura 5 também mostra os tipos de validação encontrados nos trabalhos filtrados. Foi identificado um percentual de 72,7% dos artigos validados em experimentos no meio acadêmico. Em relação aos modelos de avaliação, os estudos utilizaram cálculos de probabilidade e inferência estatística, como o Cálculo de Similaridade (SIM), por meio da aplicação do Aprendizado de Máquina. Esta abordagem foi usada em sete trabalhos Dumitru et al. (2011); Herrera et al. (2008), Herrera et al. (2009b), Chen et al. (2008); Wang et al. (2017a); Herrera et al. (2009a) e Blake et al. (2015). O uso do algoritmo *k-Nearest-Neighborhood* (KNN) esteve presente em três dos trabalhos analisados Dumitru et al. (2011), Herrera et al. (2009a) e Herrera et al. (2009b). O algoritmo *Naive Bayes* (NB) também foi utilizado em três estudos Casamayor et al. (2010), Williams et al. (2017) e Sagrado et al. (2018). Outras características foram usadas nos outros artigos, tais como: cálculo de precisão, Precision, Recall, Positive e F-Measure, (BAKAR et al., 2016; CASAMAYOR et al., 2010; SHI et al., 2013; WILLIAMS et al., 2017), TF-IDF (BAKAR et al., 2016; WILLIAMS et al., 2017), K-Means (PORTUGAL et al., 2017), cMAP (WANG et al., 2017b; CASAMAYOR et al., 2010), na maioria dos casos para classificação e agrupamento das informações dos requisitos.

3.2.5 QE 1: Onde os artigos foram publicados?

Os estudos foram classificados de acordo com suas respectivas bases de dados de publicação. A Figura 6 mostra a distribuição dos artigos. O banco de dados mais preciso para executar a string de consulta foi o *IEEE Xplore*. Os bancos de dados menos precisos foram a Biblioteca *Springer* e o *Google Scholar*. A análise de precisão considerou as bases com o menor percentual de trabalhos eliminados dos resultados iniciais, encontrados pela cadeia de busca após a aplicação dos critérios de inclusão e exclusão.

Figura 6 – Periódicos e Quantidade Anual de Publicações



Fonte: Elaborado pelo autor

3.2.6 QE 2: Qual a quantidade anual de publicações?

Os estudos foram classificados de acordo com o ano de sua respectiva publicação. A Figura 6 mostra a distribuição dos estudos. Em 2017, o número de publicações sobre recomendação em Engenharia de Requisitos aumentou em relação aos anos anteriores. Esse crescimento mostra o interesse de pesquisadores em melhorar o controle tanto dos processos quanto do gerenciamento de requisitos por meio da aplicação de novos métodos e novas tecnologias.

3.3 Ameaças à Validade do Estudo

Os estudos de mapeamento sistemático estão expostos a riscos que podem invalidá-los. Da mesma forma, este trabalho apresenta riscos que podem afetar os resultados encontrados. Esses riscos podem surgir de decisões tomadas durante a execução do mapeamento sistemático ou dos métodos selecionados para realizar a extração das informações.

Com o intuito de garantir os melhores resultados para esta pesquisa, seis bases de dados foram escolhidas. As bases de dados selecionadas são conhecidas no meio acadêmico por sua relevância na área de informática e afins. Ao selecionar bancos de dados renomadas e especializadas em uma área de conhecimento, seu impacto nos resultados obtidos no estudo foi mitigado.

Dois termos principais e seus sinônimos foram escolhidos para construir a *string* que fundamentou a pesquisa. Para o termo *Engenharia de Requisitos* os sinônimos utilizados foram selecionados na literatura especializada (ELIZABETH H. JEREMY D., 2017), visando maior assertividade para a busca e também para cobrir os trabalhos com abordagem ao tema correto.

A ambiguidade da linguagem dos termos *Previsão* e *Recomendação* pode ter afetado os resultados do estudo. Em alguns casos, os autores referem-se ao mesmo assunto usando pala-

vras diferentes. Portanto, para o termo *Recomendação*, foram selecionados sinônimos em dois dicionários on-line da língua inglesa: *Thesaurus* e *Cambridge*. Os sinônimos aperfeiçoaram a assertividade da busca e ampliaram a cobertura durante a obtenção dos resultados.

Alguns artigos relacionados ao tema da pesquisa podem ter sido removidos deste estudo devido ao processo de filtragem. Portanto, uma técnica de mapeamento sistemático amplamente adotada na literatura foi aplicada durante o processo de filtragem para minimizar este risco. Esta técnica foi desenvolvida por Petersen et al. (2008) e apresenta um guia para a elaboração e execução de estudos de mapeamento sistemáticos.

Durante todo o processo de filtragem, os estudos foram analisados e selecionados sem um segundo revisor. Os resultados obtidos podem ter sido afetados. Os procedimentos de revisão utilizados por outros autores foram usados para minimizar este risco (DIAS et al., 2018). Utilizou-se também um software para apoiar o processo de seleção, a ferramenta *Mendeley Desktop* (MENDELEY, 2017).

Este estudo não aprofundou os aspectos conceituais da área de Engenharia de Requisitos. A pesquisa manteve seu foco exclusivamente na aplicação da recomendação. Essa decisão contribuiu para a filtragem apresentada neste estudo, considerando um contexto específico dentro da grande área de Engenharia de Requisitos, que compreende um número substancial de artigos acadêmicos.

3.4 Discussão Sobre o Estudo

Este estudo focou no uso de recomendação na área de Engenharia de Requisitos. A pesquisa pode orientar trabalhos futuros sobre o uso de abordagens e tecnologias que apoiam este tema.

A análise de 26 artigos filtrados permitiu concluir que tecnologias como *Machine Learning*, Processamento de Linguagem Natural, *Clustering* e *Data Mining* são predominantemente utilizadas para a análise automatizada de dados não estruturados durante os processos de Engenharia de Requisitos.

O estudo apresentou as maneiras em que a tecnologia foi utilizada para apoiar os sistemas de recomendação na ER: (a) colaboração entre os envolvidos (HERRERA et al., 2008, 2009a,b; BAKAR et al., 2016; WILLIAMS et al., 2017; NAKATANI et al., 2014; SAGRADO et al., 2018); (b) comportamento das partes interessadas, utilizando as informações do ambiente (LIU et al., 2017a; GARCIA et al., 2016; LIU et al., 2017b); (c) dados históricos dos projetos (DUMITRU et al., 2011; WANG et al., 2017a,b; PORTUGAL et al., 2017; CASAMAYOR et al., 2010; SHI et al., 2013; ZHANG et al., 2013; HAMZA et al., 2015; PARK et al., 2010; BLAKE et al., 2015); (d) preferências dos usuários do sistema (CHEN et al., 2008; FELFERNIG et al., 2010).

A maioria dos estudos mediu a eficiência das recomendações através do uso de algoritmos estatísticos e de probabilidade, correspondendo a 68,1% dos estudos. Os estudos restantes foram avaliados empiricamente por meio do *feedback* de seus usuários ou não demonstraram

algum tipo de validação.

Durante a última década, a quantidade de trabalhos anuais de recomendação em Engenharia de Requisitos permaneceu estável, ou seja, entre 1 e 3 publicações anuais foram aprovadas nas bases pesquisadas. No entanto, houve um aumento considerável de publicações no ano de 2017, demonstrando o interesse dos pesquisadores em revisitar esta área, uma vez que as tecnologias para análise de textos não estruturados continuam evoluindo na área da computação.

Os processos análise e validação apresentaram menos artigos quando comparados com processos de elicitação. Isso demonstra uma oportunidade de pesquisa para recomendação de más definições ou problemas em requisitos.

Observou-se o uso de dados históricos relacionados aos requisitos. No entanto, os trabalhos não relacionam esses dados aos projetos. Nenhum artigo cita a análise de similaridade dos projetos como ponto de partida para a recomendação para projetos que são semelhantes. Esta análise é considerada importante durante a recomendação, uma vez que o contexto do projeto é importante para definir a similaridade entre os requisitos e seu reuso.

3.5 Análise e Comparativo dos Trabalhos Relacionados

Nesta etapa, foram selecionados os estudos resultantes do mapeamento sistemático que apresentassem relação com o desenvolvimento de modelos de recomendação para a ER. O critério adotado para a escolha dos trabalhos priorizou artigos que abordassem: (I) modelos ou sistemas de recomendação de requisitos; (II) análise de similaridade de projetos ou seus requisitos; (III) sistema de *feedback* sobre as recomendações de novos requisitos.

Xie et al. (2017) propõem uma metodologia que emprega *Conditional Random Fields* (CRF) para fornecer uma exploração quantitativa das interações entre usuários e sistemas, visando descobrir potenciais requisitos. Ao analisar os padrões de comportamento dos usuários em tempo de execução, especialistas de domínio realizaram previsões sobre como as intenções dos usuários mudam. Os autores propuseram melhorias e soluções do sistema para ajudar a atender as necessidades semelhantes identificadas.

Herrera et al. (2009b) introduzem uma estrutura para prover suporte à elicitação, utilizando técnicas em mineração de dados e aprendizado de máquina. O *framework* desenvolvido pelos autores agrupa ideias de partes interessadas postadas em fóruns e, usando tecnologias de recomendação, ajudam a promover esses fóruns entre partes potencialmente interessadas nos mesmos conteúdos. Bakar et al. (2016) apresentam uma abordagem semi-automatizada, conhecida como *Feature Extraction for Reuse of Natural Language Requirements* (FENL), para a extração de frases que possam representar recursos de software. Os autores visam extrair recursos de revisões *online* de produtos, permitindo assim o reuso de requisitos de software.

Portugal et al. (2017) propõem o uso de um repositório de versionamento de softwares (GitHub) como fonte de informação. Para lidar com grandes massas de dados e prover o acesso à fontes adequadas, os autores criaram perfis de projetos com atributos úteis para a ER. Após,

aplicaram *clustering* e Processamento de Linguagem Natural (PLN) para recomendar projetos através da identificação de palavras chaves semelhantes na sua descrição. Williams et al. (2017) utilizam a rede social *Twitter* como fonte de requisitos. Seu objetivo é permitir um processo de ER orientado por dados, interativo e adaptável. A análise é conduzida usando 4.000 *tweets* de 10 sistemas de software amostrados de variados domínios de aplicativos. Os resultados revelam que cerca de 50% dos *tweets* coletados contêm informações técnicas úteis e mostram que classificadores de texto como *Support Vector Machines* e *Naive Bayes* podem ser eficazes na captura e categorização de *tweets* tecnicamente informativos.

Shi et al. (2013) definem métricas para acompanhar o histórico de evolução dos requisitos. Os autores apresentam um método, que emprega aprendizado de máquina, para prever os requisitos que evoluem com base nestas métricas. Os autores focam no auxílio quanto à qualidade dos requisitos, prevendo más definições e mudanças ao longo do ciclo de vida dos requisitos. Nakatani et al. (2014) propõem em seu artigo a concentração nos fatores ambientais dos requisitos e seus atores intencionais. O objetivo é propor um método para prever mudanças de requisitos, concentrando-se nas relações sociais. Os autores apresentam um novo método e avaliam sua eficácia através de uma simulação de mudanças dos requisitos.

Dumitru et al. (2011) propõem um sistema para recomendação de funcionalidades. A abordagem explora descrições de produtos, aplicando *Data Mining*, aprendizado de máquina (*k-Nearest-Neighbor*) e *clustering* para descobrir necessidades específicas de domínio. Assim, gerando um modelo que representa semelhanças, variantes e artifícios de categoria cruzada para a recomendação de funcionalidades reutilizáveis entre diferentes produtos.

Garcia et al. (2016) apresentam um sistema de recomendação que coleta o histórico de uso de um serviço *Web*, relaciona essas informações aos requisitos e gera relatórios com recomendações que podem aumentar a qualidade deste serviço. A abordagem proposta visa fornecer relatórios analíticos em uma linguagem próxima do negócio. O sistema indica novos fluxos de trabalho, caminhos de navegação, identifica potenciais recursos que podem ser removidos e correlaciona os requisitos e as mudanças propostas, ajudando a manter a especificação dos requisitos de software atualizada.

Liu et al. (2018) conduzem um estudo que aborda o desenvolvimento ou manutenção de aplicações *Android*. Em seu artigo, os autores se concentram nos múltiplos desafios que os desenvolvedores enfrentam ao criar explicações sobre o uso de permissões. Eles propõem uma nova estrutura, que explora possíveis recomendações de requisitos de segurança a partir das descrições de aplicativos semelhantes. O trabalho utiliza técnicas de recuperação de informações e resumo de texto para encontrar usos frequentes de permissão.

Kim, Dey e Lee (2019) propõem uma base de conhecimento de ontologia e seu processo de design para recomendar requisitos de segurança com base em casos de ataque e no conhecimento do domínio do sistema. A base de conhecimento proposta é dividida em três partes; Ontologia APT, ontologia de conhecimento geral de segurança e ontologia de conhecimento específico de domínio. Cada ontologia pode ajudar a entender as preocupações de segurança

em seus conhecimentos. Ao integrar três ontologias à ontologia do domínio do problema, os requisitos de segurança apropriados podem ser derivados com o processo de recomendação de requisitos de segurança. A base de conhecimento e o processo propostos podem ajudar a derivar os requisitos de segurança, considerando ataques em sistemas reais.

A Tabela 5 mostra as características adotadas na comparação entre o modelo *Nhatos* e os demais trabalhos. O primeiro item (processos) informa quais dos processos da Engenharia de Requisitos os artigos abordam - elicitação (E), validação (V), especificação (S) ou gerenciamento (G). O segundo item (recomendação) mostra o tipo de item recomendado no estudo – más definições (M), fóruns de discussões (F), requisitos (R) ou projetos (P). O terceiro mostra o caminho estratégico utilizado pelos autores para as recomendações. Por fim, o quarto item se refere à colaboração das partes interessadas durante o processo de recomendação.

Tabela 5 – Comparação dos Trabalhos Relacionados

Autor	**Processos**	**Recomendação**	**Estratégia**	**Colaboração**
Kim, Dey e Lee (2019)	V[1]	R	Ontologias	✔
Liu et al. (2018)	V	R	Descrição	✘
Xie et al. (2017)	V	M[5]	Histórico	✔
Herrera et al. (2009b)	E[2]	F[6]	*Reviews*	✔
Bakar et al. (2016)	E	R[7]	Descrição	✘
Portugal et al. (2017)	E	P[8]	*Commits*	✘
Shi et al. (2013)	V	M	Histórico	✘
Williams et al. (2017)	E	R	*Reviews*	✘
Dumitru et al. (2011)	E	R	Descrição	✘
Nakatani et al. (2014)	V	M	Descrição	✘
Garcia et al. (2016)	E	R	*Logs*	✔
Modelo *Nhatos*	E, V, S[3], G[4]	R	Históricos de Contextos	✔

[1] Validação [2] Elicitação [3] Especificação [5] Gerenciamento
[4] Redefinições [6] Fóruns [7] Requisitos [8] Projetos

Fonte: Elaborado pelo autor

O modelo *Nhatos* difere-se dos outros estudos em quatro aspectos: (1) Aborda todos os processos da engenharia de requisitos de maneira colaborativa, permitindo que todos os envolvidos contribuam ao longo dos projetos; (2) Aborda recomendações de requisitos no início de um novo projeto, usando históricos de projetos já executados, através de características comuns entre projetos e ou requisitos; (3) Durante a implementação do projeto, a recomendação baseada na análise de similaridade dos históricos de contextos e no Processamento de Linguagem Natural (PLN) é usada para identificar requisitos semelhantes que podem ser recomendados; (4) Recomenda contextos futuros a partir da análise de similaridade aplicada aos históricos de contextos armazenados.

A captura de históricos de contextos dos projetos possibilita a recomendação de requisitos através da análise de similaridade dos seus respectivos históricos de contextos. Deste modo, o modelo recomenda requisitos que ocorreram em um contexto semelhante a partir da compa-

ração de contextos passados com contextos de projetos em execução. A colaboração entre os envolvidos ao projeto complementa o processo de gestão dos requisitos. Assim, as informações são capturadas a qualquer instante e com diferentes pontos de vista referente aos requisitos, ao longo de todo o ciclo de vida dos projetos, contemplando assim os processos da ER.

3.6 Considerações sobre o Capítulo

Considerando a crescente demanda das organizações por inovação e desenvolvimento de novos produtos, o gerenciamento de requisitos apresenta-se como uma área relevante de pesquisa no campo de gerenciamento de projetos. Conforme demonstrado nos artigos analisados neste estudo, a engenharia de requisitos apresenta um crescente número de estudos conforme o passar dos últimos anos.

O principal objetivo deste estudo de mapeamento sistemático foi obter *insights* sobre o que foi investigado na literatura sobre a recomendação na área de engenharia de requisitos. O estudo apresentou o estado da arte no uso da recomendação de ER. Os artigos estudados também mostraram as principais tecnologias utilizadas nas pesquisas, processos da ER às quais as pesquisas se concentram e as informações de entrada utilizadas atualmente na literatura para a geração de recomendações.

A pesquisa demonstrou uma maior concentração de artigos no processo de elicitação e menor quantidade quanto aos demais processos da ER, como gerenciamento e especificação por exemplo. Isso demonstra uma possível lacuna de pesquisa para a exploração de recomendações dos demais processos da área de ER.

A revisão de trabalhos publicados na última década forneceu *insights* iniciais para novos estudos baseados na recomendação de requisitos de projetos de software. A análise da similaridade entre os projetos não é explorada nos estudos analisados. Este fator adicionaria escalabilidade à recomendação de requisitos, permitindo a recomendação de requisitos entre projetos de diferentes áreas de conhecimento, mas que possuem objetivos equivalentes.

Um desafio para este estudo é aprofundar as pesquisas de análise de similaridade para o agrupamento de projetos semelhantes, fazendo recomendações aplicáveis a diferentes cenários. Explorar também outras atividades na área de engenharia de requisitos, além da elicitação, pode adicionar uma contribuição acadêmica relevante. Além disso, os históricos de contextos (ROSA et al., 2015) de projetos tendem a melhorar a identificação de requisitos por meio de estratégias analíticas.

Nesse sentido, a partir desta pesquisa conclui-se que a formalização de um contexto (DEY; ABOWD; SALBER, 2001) aplicado ao gerenciamento de projetos permitiria analisar a similaridade entre projetos através destes históricos. Deste modo, possibilitando a recomendações de requisitos com maior grau de aceitabilidade.

4 MODELO NHATOS

Uma pesquisa envolvendo profissionais de projetos foi conduzida para avaliar a aplicabilidade de uma ferramenta inteligente de apoio a profissionais de projetos durante suas atuações. Esta pesquisa visou responder se as equipes de projetos necessitam de uma ferramenta proativa para suporte às suas atividades durante os processos da Engenharia de Requisitos.

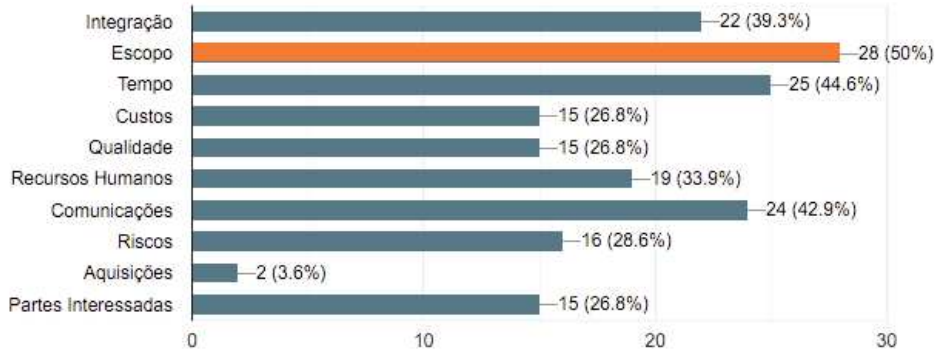
4.1 Pesquisa com Profissionais de Projetos

A pesquisa de campo foi realizada com equipes de projetos da área de tecnologia da informação. Esta pesquisa envolveu 56 profissionais atuantes na indústria, predominantemente dos ramos de automação bancária, ensino e tecnologia. Divulgou-se um questionário eletrônico aos participantes, com questões de múltipla escolha e transcritas (MOBILAB, 2018).

Cerca de 75% dos entrevistados possuíam mais de 5 anos de experiência em projetos. Mais de 70% dos entrevistados atuavam em empresas com mais de 1000 funcionários. O principal objetivo da pesquisa foi capturar a percepção dos profissionais com relação às ferramentas de apoio em gestão de projetos utilizadas atualmente em seu ambiente de trabalho. Esta pesquisa possibilitou a identificação de lacunas e compreensão de possíveis melhorias na área de Engenharia de Requisitos, viabilizando a aplicação do modelo Nhatos em ambientes semelhantes. A seguir, são apresentadas as principais questões e os seus respectivos resultados:

- *Quais áreas você considera mais críticas para o sucesso do projeto?* Quando questionados sobre as áreas mais críticas para o sucesso dos projetos, 50% dos entrevistados selecionaram o escopo como a área de maior criticidade. Também foram citadas as áreas de Tempo, Comunicações e Integração com 44,6% 42,9% e 39,3% das respostas respectivamente, conforme mostra a Figura 7;

Figura 7 – Áreas Críticas para o Sucesso dos Projetos

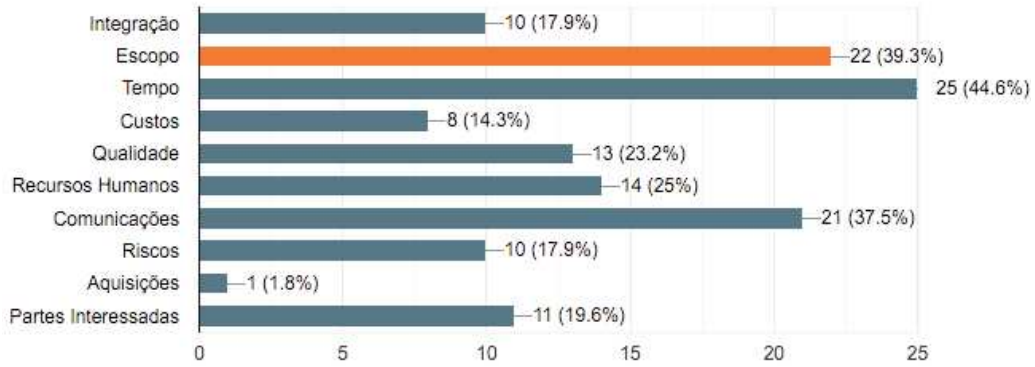


Fonte: Elaborado pelo autor

- *Nos projetos em que ocorreram problemas, quais foram as áreas em que os problemas foram identificados?* Os entrevistados foram indagados sobre quais seriam os projetos

em que mais ocorreram problemas e quais foram as áreas em que os problemas foram identificados. Um total de 35% dos participantes responderam que problemas na gestão dos projetos acontecem devido ao incorreto gerenciamento do escopo. Outras áreas de projeto como Gestão do Tempo e Comunicações obtiveram 25 (44,6%) e 21 (37,5%) respostas respectivamente, conforme mostra a Figura 8;

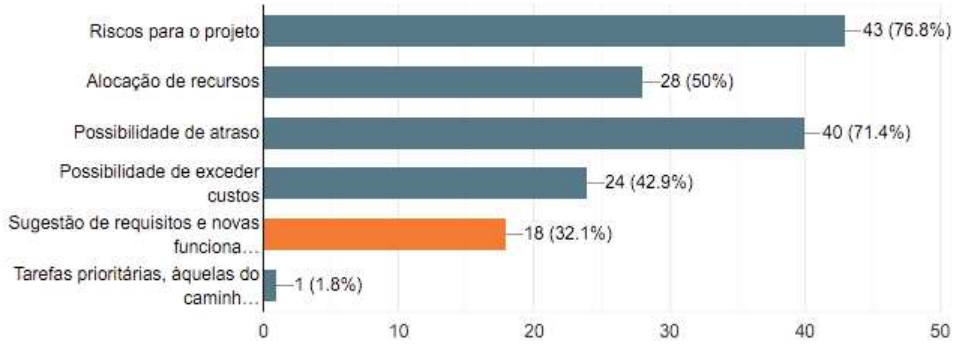
Figura 8 – Áreas Onde Mais Ocorrem Problemas



Fonte: Elaborado pelo autor

- *Quais tipos de sugestão você gostaria de receber de uma ferramenta proativa de gestão de projetos?* Os profissionais foram questionados sobre quais os tipos de sugestão gostariam de receber de uma ferramenta proativa durante a gestão de projetos. Segundo a percepção dos entrevistados, 32,1% respondeu que uma ferramenta deveria sugerir de novos requisitos aos projetos. A Figura 9 mostra os resultados coletados;

Figura 9 – Tipos de Sugestões Esperados pelas Equipes em uma Ferramenta Proativa

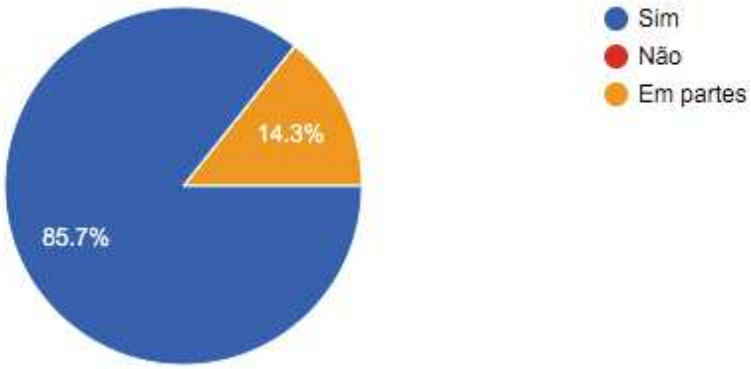


Fonte: Elaborado pelo autor

- *Você acredita que informações de outros projetos já concluídos poderiam auxiliar na gestão do projeto?* Os entrevistados foram questionados quanto à sua confiança em informações de projetos já concluídos para auxiliar na gestão dos projetos em andamento e, 85,7% dos participantes confirmaram que históricas contribuem na gestão dos novos projetos, conforme mostra a Figura 10.

Considerando a coleta da percepção das equipes realizada no ambiente de projetos, predominantemente com profissionais com mais de cinco anos de atuação com gerenciamento de projetos, observa-se que os times despertaram problemas pertinentes que afetam a área de escopo. Isto evidencia a necessidade da criação de ferramentas inteligentes para apoiar os engenheiros de requisitos e gestores de projetos durante as suas atividades. A oportunidade para a criação de ferramentas proativas para a gestão de requisitos justifica o desenvolvimento do modelo Nhatos, para auxiliar as equipes durante os processos de Engenharia de Requisitos.

Figura 10 – Confiança dos Times de Projeto em Informações Históricas



Fonte: Elaborado pelo autor

4.2 Visão Geral do Modelo Nhatos

De acordo com Robillard et al. (2010), um SRES necessita atender certos requisitos para que este seja considerado um sistema de recomendação, sendo estes: (a) Possuir um mecanismo de coleta de dados e artefatos do processo de desenvolvimento em um modelo de dados; (b) Conter um mecanismo de recomendação para analisar o modelo de dados e gerar recomendações; e (c) Apresentar uma interface do usuário para acionar o ciclo de recomendação e apresentar seus resultados.

O modelo Nhatos atende os três requisitos citados, pois: (a) Coleta dados através do seu sistema multi-agentes ao longo de todo o ciclo de vida dos projetos; (b) Realiza recomendações considerando o contexto atual do projeto e a sua semelhança com os demais; e (c) Possui uma interface em dispositivos móveis para apresentar os resultados aos usuários e coletar o seu *feedback*.

A Figura 11 mostra uma visão geral do modelo Nhatos. A partir das informações dos projetos e seus requisitos, o modelo tem como base a análise de similaridade, tanto das características dos projetos, quanto de seus históricos de contextos. Assim sendo, o modelo foi resumido em 6 partes: (1) Dados; (2) Similaridade por características; (3) Gestão de requisitos; (4) Contexto; (5) Similaridade por históricos de contextos; (6) Recomendação.

- **Dados**: As informações são inseridas no modelo por intermédio de uma interface de

Figura 11 – Visão Geral do Modelo Nhatos

Fonte: Elaborado pelo autor

sistema. Esta interface permite tanto incluir dados em massa (importando informações extraídas de um software de gerenciamento de projetos, como o *MSProject* por exemplo), como adicionar informações ao longo do projeto (utilizando uma interface de sistema);

- **Similaridade por características**: A partir de uma base de dados, realiza-se a análise de similaridade dos projetos. Esta análise agrupa os projetos de acordo com características em comum;

- **Gestão de requisitos**: O modelo apresenta recursos para a gestão de requisitos. As equipes detalham o escopo dos projetos, elicitam, especificam e gerenciam os seus requisitos, assim como suas atividades e alterações ocorridas ao longo do ciclo de vida de cada requisito;

- **Contexto**: O modelo captura e armazena históricos de contextos durante o gerenciamento dos projetos e, os utiliza futuramente para realizar análises de similaridade entre os projetos e seus requisitos;

- **Similaridade por históricos de contextos**: Os históricos capturados são comparados tanto no início, como ao longo do ciclo de vida dos projetos, considerando configurações

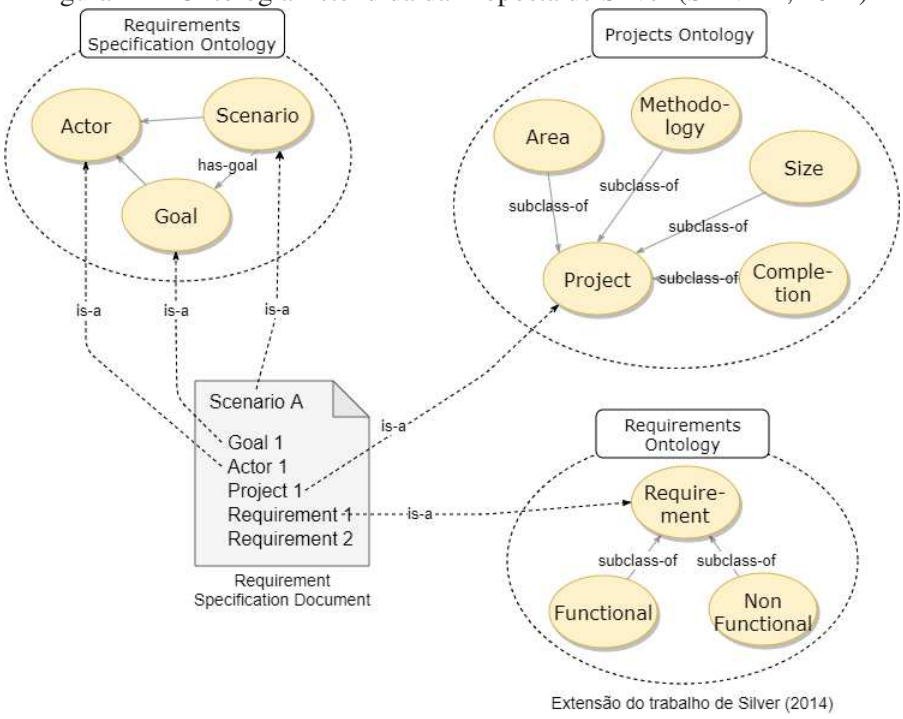
aplicadas por um especialista;

- **Recomendação**: As recomendações baseiam-se na análise de similaridade dos históricos armazenados. Os componentes interagem com os módulos de recomendação e gerenciamento de requisitos, onde as informações relativas ao contexto do projeto passam a ser consideradas em cada recomendação realizada. Deste modo, as equipes passam a colaborar de acordo com seus contextos. Portanto, através da análise de similaridade dos projetos e dos históricos de contextos, são identificadas possíveis recomendações de requisitos entre os projetos.

4.3 Representação de Domínio

A Figura 12 mostra a representação dos projetos, requisitos e sua respectiva especificação através de suas ontologias. Esta representação é uma extensão do trabalho de Silver (SILVER, 2014), com o acréscimo da ontologia de projetos *Projects Ontology*. O domínio é formado por três ontologias: a) *Requirements Specification Ontology* que compõe a especificação dos requisitos; b) *Projects Ontology* que é caracterizada pelo domínio do projeto e suas informações contextuais; e c) *Requirements Ontology* que representa os requisitos.

Figura 12 – Ontologia Estendida da Proposta de Silver (SILVER, 2014)



Fonte: Elaborado pelo autor

A ontologia de projetos possui a classe *Project*, que é composta pelas subclasses *Area*, *Completion Level*, *Methodology* e *Size*. Estas subclasses definem as informações contextuais dos

projetos. A ontologia que compreende a especificação de requisitos é composta pelas classes *Scenario*, *Goal* e *Actor*. Estas classes representam o processo de elicitação, identificando novos requisitos e relacionando-os aos projetos. A ontologia que determina os requisitos possui as a classe *Requirement* e suas subclasses *Functional Requirement* e *Non Functional Requirement*, que definem os tipos dos requisitos.

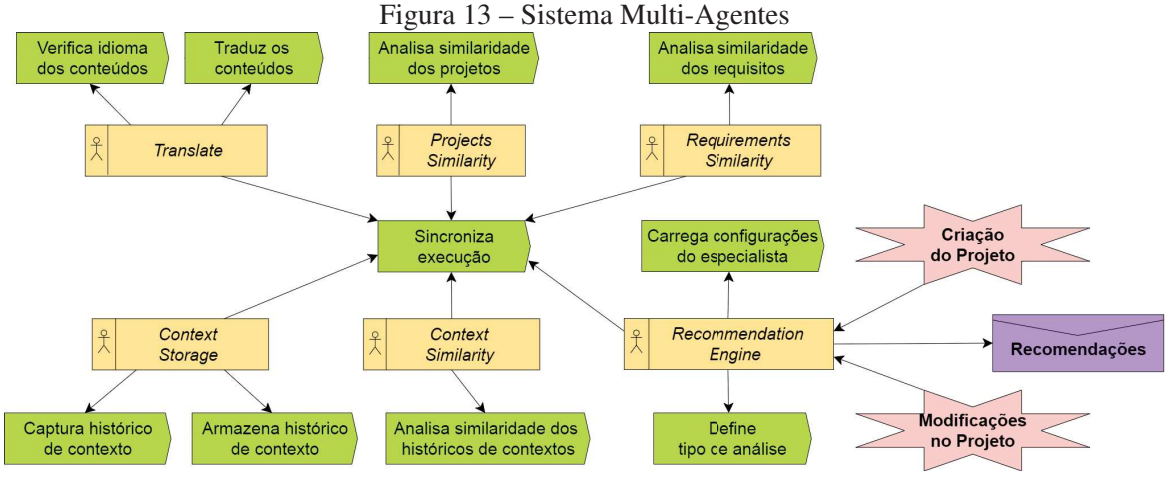
Para exemplificar como o requisito é instanciado ao utilizar as ontologias, um documento de especificação de exemplo foi adicionado (*Requirement Specification Document*). Cada documento possui um cenário, onde os atores, através dos objetivos definidos, identificam requisitos para um determinado projeto.

4.4 Análise de Similaridade

Ao longo dos últimos anos, muitas técnicas foram desenvolvidas para oferecer suporte ao reúso de software. Estas técnicas exploram o fato de os sistemas de mesmo domínio de aplicação, apresentarem semelhanças, possuindo assim potencial para reuso. Linhas de produto de software (arquitetura em comum que possa ser compartilhada entre clientes) e padrões de projeto (interações evolutivas semelhantes) apresentam-se como caminhos que facilitam a reutilização (SOMMERVILLE, 2016).

A análise de similaridade do modelo Nhatos ocorre em dois momentos: (1) análise de similaridade baseada nas características dos projetos; e (2) análise de similaridade baseada nos históricos de contextos dos projetos. A primeira análise ocorre a cada inserção de um novo projeto, analisando características compatíveis entre os projetos presentes na base de dados. A segunda análise acontece durante qualquer evolução no ciclo de vida dos projetos, e examina progressos que apresentem padrões parecidos. A Figura 13 mostra a representação do sistema multi-agentes e suas atividades, de acordo com cada tipo de análise.

Figura 13 – Sistema Multi-Agentes

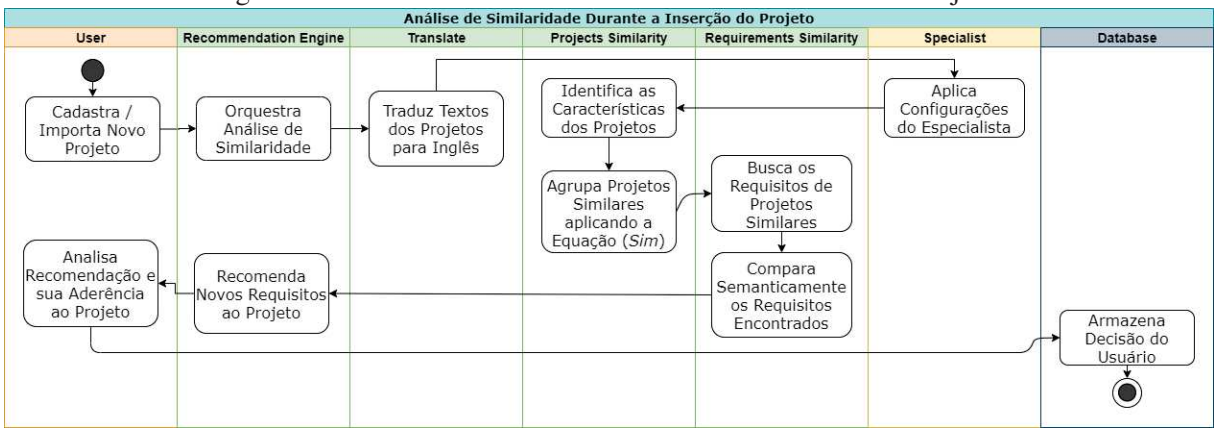


Fonte: Elaborado pelo autor

4.4.1 Análise de Similaridade das Características dos Projetos

Durante a criação de um novo projeto, suas características são capturadas. Neste momento, a execução do projeto ainda não foi executada, ou seja, o projeto está em fase de criação ou planejamento. O PMBOK (PMI, 2017a) apresenta características que são consideradas propriedades de um projeto. Estas características são determinadas nas fases iniciais do ciclo de vida, sendo estas: (a) Área de conhecimento (linhas do produto); (b) Metodologia de desenvolvimento (ágil, tradicional ou híbrida); (c) Nível de completude do cronograma (percentual); e (d) Tamanho (pequeno, médio ou grande porte). A Figura 14 mostra um diagrama do fluxo de recomendação desta etapa.

Figura 14 – Análise de Similaridade das Características dos Projetos



Fonte: Elaborado pelo autor

Uma vez capturadas as características iniciais do projeto, pesos são definidos para cada uma delas. Um especialista define a importância de cada característica com base no seu grau de relevância para a organização. As definições de peso variam entre 0,0 (irrelevante) e 1,0 (altamente relevante). Assim sendo, é possível medir a similaridade entre os projetos antes do início de sua execução.

A Equação 4.1 calcula a similaridade dos projetos (*Sim*). O sistema Multi-Agentes percorre os históricos armazenados e, compara as variáveis características de cada projeto do histórico com as mesmas variáveis do projeto original. O modelo considera as configurações pré-informadas pelo especialista – estas configurações compõem um sistema de pesos, que é aplicado durante o cálculo. Os pesos são atribuídos tanto às variáveis do projeto, quanto à similaridade mínima aceitável durante a comparação dos projetos.

Inicialmente é realizada uma consulta na base de dados, que obtém todos os projetos do histórico ($Ph_{0..n}$). Para cada projeto deste histórico (Ph) é verificada a sua similaridade ao projeto original (Po). Para tanto, considera-se cada variável do projeto, examinando se as mesmas são iguais às variáveis do projeto original (if $vPh_z = vPo_z$).

Posteriormente, todos os pesos das variáveis do projeto original (w_z) são somados, encon-

trando assim o valor máximo de peso possível a ser atingido. Depois, o resultado desta soma é multiplicado pelo nível de similaridade mínimo aceitável (w_g). Deste modo, define-se um ponto de corte para considerar projetos similares. Finalmente, é verificado se o resultado da soma das variáveis do projeto do histórico é maior ou igual ao ponto de corte resultante da soma realizada.

$$Sim = Ph_{0..n} \left(\sum_{1..z} (w_z,\ if\ vPh_z = vPo_z)\ \geq \sum_{1..z} (w_z) * w_g \right) \tag{4.1}$$

Fonte: Elaborado pelo autor

Por exemplo, considerando o projeto A como projeto original, e suas características, sendo estas: metodologia, tamanho, nível de conclusão e área. Os pesos de importância (para a organização) de cada característica serão configurados previamente por um especialista - sendo 0 (zero) para menos importante e 1 (um) para mais importante. Os valores configurados são metodologia: 0,5, tamanho: 1, nível de conclusão: 1 e área: 0,8. O especialista também informou ao modelo que aceita recomendações para projetos que sejam, no mínimo, 70% semelhantes.

O agente pesquisou os projetos no histórico e encontrou os projetos dispostos na tabela 6. O projeto B, quando comparado ao original, obteve soma de pesos de 1,8. Pois, se tratava de um projeto de mesmo tamanho (peso 1) e área (peso 0,8). Por sua vez, o projeto C obteve uma somatória de pesos de valor 2,3. Este projeto continha mesma metodologia (peso 0,5), nível de conclusão (peso 1) e área (peso 0,8). Por fim, o projeto C se assemelhou ao projeto original (A) em metodologia (peso 0,5), tamanho (peso 1) e nível de conclusão (peso 1). O projeto D também foi classificado como similar, uma vez que obteve a soma de pesos de 2,5.

Tabela 6 – Exemplo da Aplicação da Equação (Sim) no Histórico de Projetos

Proj.	**Instância**	**Metodologia** (Peso: 0,5)	**Tamanho** (Peso: 1)	**Conclusão** (Peso: 1)	**Área** (Peso: 0,8)	Soma de Pesos	70% Similar
A	Original	Ágil	Pequeno	Início	Finanças	3,3	
B	Histórico	Tradicional	Pequeno	Finalizando	Finanças	1,8	Não
C	Histórico	Ágil	Médio	Início	Finanças	2,3	Sim
D	Histórico	Ágil	Pequeno	Início	Empréstimos	2,5	Sim

Fonte: Elaborado pelo autor

A soma máxima de pesos a ser alcançada no cenário de exemplo é 3,3. Pois, este valor corresponde a um projeto com as mesmas características de metodologia, tamanho, nível de conclusão e área do projeto original. No entanto, o especialista configurou a aceitabilidade mínima de 70% de semelhança. Assim, projetos que obtenham uma somatória de pesos de 2,31 ou mais, serão considerados similares pelo modelo. Então, os projetos C e D obedecem este requisito.

Depois que os projetos semelhantes são agrupados, os históricos de contextos desses projetos são analisados para identificar requisitos reutilizáveis. Portanto, para projetos de mesmo grupo, o modelo calcula a aproximação da distância semântica entre os seus requisitos.

A distância semântica é definida pela análise da distância entre documentos de texto proposta por Kusner et al. (2015). Esta abordagem recorre aos resultados de Mikolov et al. (2013), cujo modelo *word2vec* gera combinações de palavras, em uma ontologia de escala natural, para conjuntos de dados muito grandes (por exemplo, utiliza-se neste modelo um treinamento de aproximadamente 100 bilhões de palavras). Deste modo, os requisitos são comparados a partir dos textos descritos em seus objetivos.

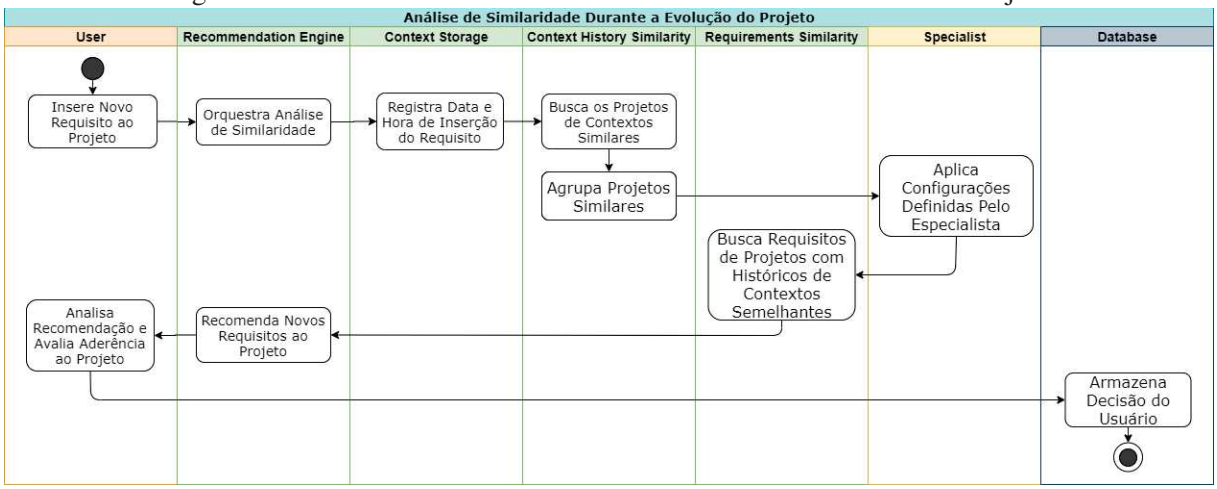
Os resultados da comparação semântica são armazenados no histórico, possibilitando a recomendação de requisitos que se referem a um mesmo tema (propósitos equivalentes) e projetos semelhantes. A recomendação de requisitos nas fases iniciais dos projetos visa trazer informações históricas às equipes de projetos, principalmente aos engenheiros de requisitos. Posteriormente, todos os envolvidos poderão aceitar ou rejeitar as recomendações, garantindo assim que nenhum requisito da base histórica seja desconsiderado durante a gestão do projeto.

4.4.2 Análise de Similaridade dos Históricos de Contextos

Ao longo do ciclo de vida dos projetos, os eventos evolutivos são armazenadas nos históricos de contextos. Neste ponto, aplica-se o conceito proposto por Satyanarayanan (2001), onde identificam-se informações suscetíveis à mudanças de estado ao longo do ciclo de vida dos projetos, sendo estas: (a) Objetivo dos requisitos; (b) Atores envolvidos.

Sempre que ocorre inserção de um novo requisito ao projeto, ou ao menos uma destas informações de contexto sofre modificação, inicia-se a análise de similaridade de projetos por históricos de contextos. O modelo Nhatos utiliza os históricos armazenados para complementar a análise de similaridade baseada nas características, atualizando suas recomendações com base nas novas informações. A Figura 15 mostra como ocorre o fluxo de recomendação nesta etapa.

Figura 15 – Análise de Similaridade dos Históricos de Contextos do Projeto



Fonte: Elaborado pelo autor

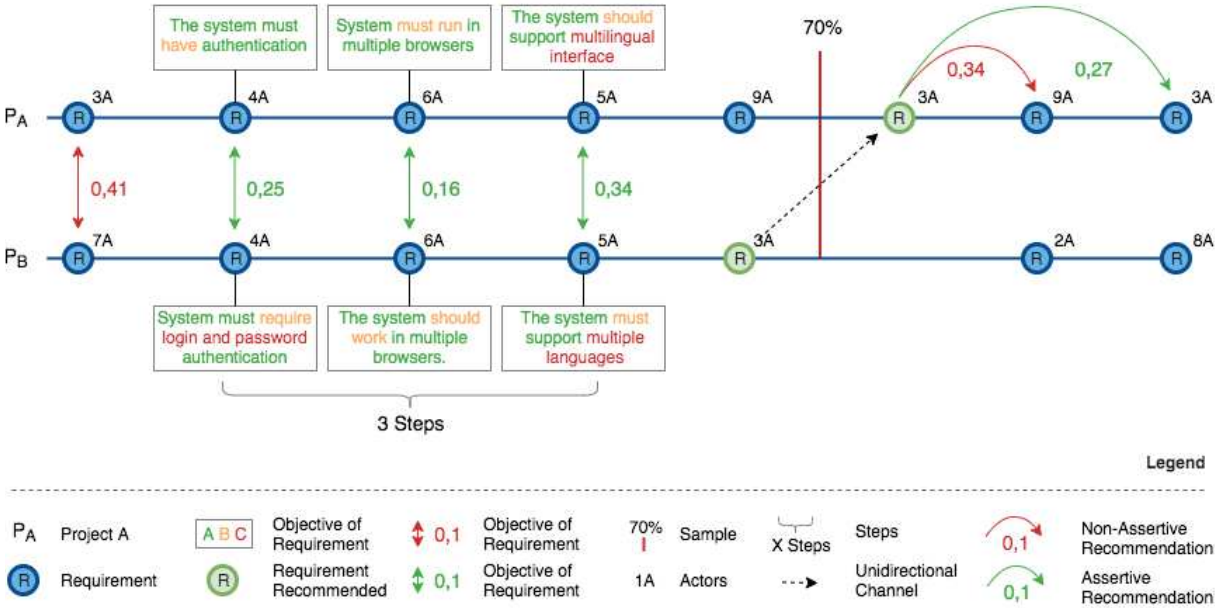
Este fluxo difere-se do primeiro apresentado ao analisar o histórico do contexto do projeto

e compará-lo com os outros contextos armazenados, buscando assim contextos semelhantes. Cada etapa apresentada na linha do tempo dos projetos representa as informações sobre ocorrências de eventos armazenadas em históricos de contexto (BARBOSA et al., 2018). Deste modo, posteriormente estas informações podem ser utilizadas pelo modelo para gerar uma nova recomendação. Sempre que um dos eventos definidos ocorre, o Nhatos cria um registro no histórico de contextos do projeto.

Durante a recomendação de um novo requisito, o modelo Nhatos compara cada contexto do projeto com o contexto de projetos semelhantes. O modelo utiliza a distância semântica entre os requisitos a partir dos históricos previamente armazenados. Assim, se a distância entre os requisitos é aceitável (de acordo com as configurações do especialista) e a quantidade de atores é igual entre os requisitos comparados, a próxima ocorrência do histórico de contexto é recomendada para o projeto em execução.

A Figura 16 mostra um exemplo de análise dos históricos de contextos de um projeto em execução com os projetos que foram identificados como semelhantes. Dessa forma, o contexto é comparado a outros projetos para recomendar o próximo requisito a ser adicionado ao projeto. Neste exemplo, todo o fluxo de recomendação é elucidado passo a passo.

Figura 16 – Análise de Similaridade por Históricos de Contextos



Fonte: Elaborado pelo autor

Dois projetos foram considerados como exemplo. Ambos os projetos (Pa e Pb) possuem 5 requisitos. O sistema Multi-Agentes compara os requisitos entre os projetos em uma ordem cronológica. Neste cenário o especialista configurou a distância semântica mínima de 0,3, amostragem de treinamento 70% e 3 passos semelhantes para gerar as recomendações. Sendo assim, quando comparados o primeiro requisito do Projeto A com o primeiro requisito do Projeto B, obteve-se uma distância semântica de 0,41 (distância não aceitável de acordo com os parâme-

tros configurados). Outro fator excludente é que estes requisitos possuem diferente número de atores. Deste modo, o passo 1 não foi considerado semelhante. Por outro lado, os passos 2, 3 e 4 obedecem tanto a distância semântica mínima aceitável, quanto ao mesmo número de atores envolvidos aos requisitos comparados. Tendo em vista que o especialista configurou recomendações que exigem pelo menos 3 passos semelhantes consecutivos, o quinto requisito do projeto B é recomendado ao projeto A.

Por fim, considerando que a amostragem de treinamento configurada seja de 70%, o modelo utiliza o percentual restante do projeto (30%) para verificar se ao menos um requisito de mesma distância semântica e mesmo número de atores, utilizando como base o requisito recomendado, de fato ocorreu ao longo do ciclo de vida deste projeto. Uma vez ocorrendo, a recomendação realizada é considerada assertiva.

A análise de mais de um contexto cronológico, ocorridas no decorrer do projeto, visa identificar projetos que tenham uma sequência de execução semelhante. Esta análise contribui para um maior grau de precisão nas recomendações realizadas. O número mais significativo de contextos consecutivos semelhantes indica a proximidade entre a implementação dos projetos. No entanto, quanto mais contextos são considerados (passos), menos projetos devem ser identificados como semelhantes e, portanto, menos recomendações feitas, uma vez que cada projeto é único (PMI, 2017a).

4.5 Arquitetura

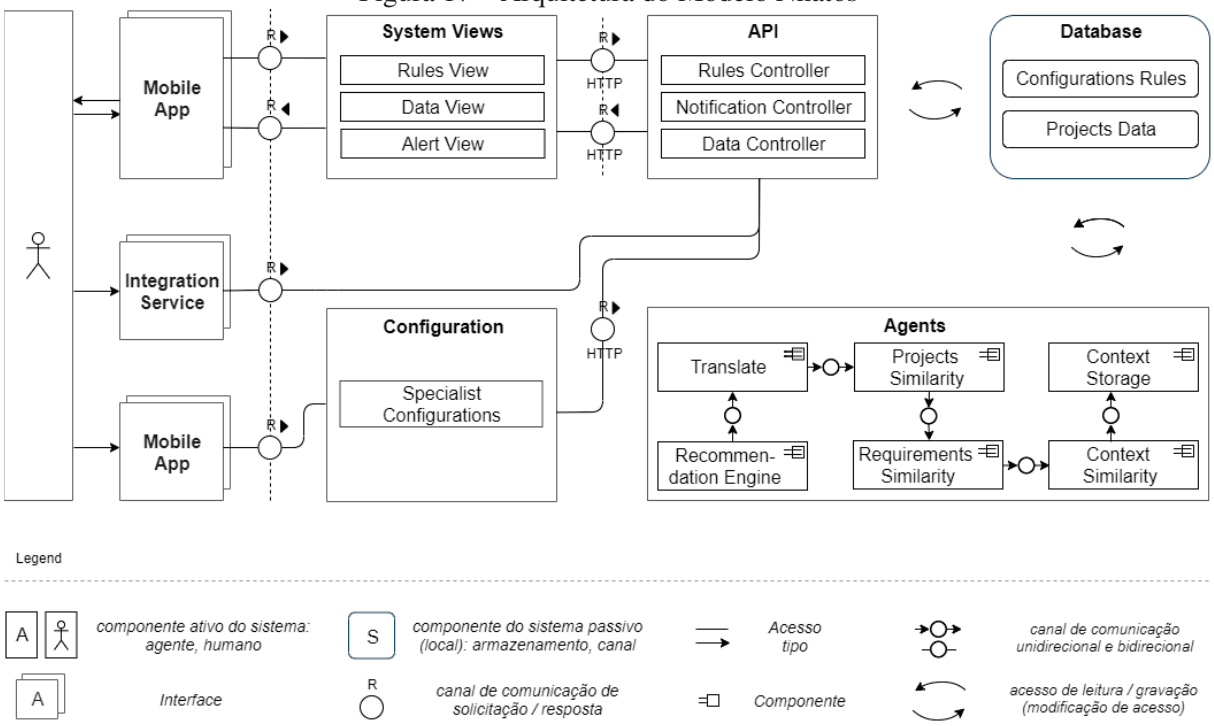
A arquitetura do modelo Nhatos foi projetada utilizando o conceito da modelagem TAM (*Technical Architecture Module*) (SAP, 2019). A Figura 17 mostra o modelo, dividido em 6 partes: *Mobile App*, *Integration Service*, *Configuration*, *System Views* e *API*, *Agents* e *Database*.

- ***Mobile App***: Aplicação em formato híbrido, desenvolvida em linguagem *Javascript*, que opera tanto em aplicações móveis (Android/IOS), como em navegadores *web*. Esta aplicação é utilizada durante as interações com os *stakeholders*. Estes *stakeholders* podem: (a) Adicionar e gerenciar os projetos e suas informações; (b) Adicionar requisitos e modificá-los; (c) Visualizar as recomendações geradas pelo modelo; (d) Prover *feedback* em relação às recomendações recebidas, aceitando-as ou rejeitando-as;

- ***Integration Service***: Possibilita a importação de dados em massa de projetos em andamento. Através do uso desta interface *web* de sistema, é possível importar projetos e requisitos exportados de ferramentas terceiras de GP, como o *MSProject*, por exemplo;

- ***Configuration***: As preferências das equipes de projeto são inseridas através de um agente assistente, acessado via dispositivos móveis. Cada variável do projeto recebe um peso (área de conhecimento, tamanho, metodologia, nível de completude do cronograma). O peso é configurado por um especialista e levado em consideração em cada recomendação realizada pelo modelo;

Figura 17 – Arquitetura do Modelo Nhatos



Fonte: Elaborado pelo autor

- ***System Views*** e ***API***: Estas aplicações operam de forma integrada em um ambiente servidor;
  - *System Views*: É caracterizada por controladores responsáveis pelas regras de negócio do modelo, obtendo informações já armazenadas na base de dados para prover informações aos envolvidos;
  - *API*: Uma interface de acesso a dados, que utiliza o protocolo *RESTFull* para interligar a aplicação *Mobile App* e *Database*.
- ***Agents***: Aplicação em formato multi-agentes (Figura 13), responsável pela captura dos eventos que caracterizam evolução ou modificação em um projeto. A adição de um novo requisito, finalização de uma atividade ou avanço no percentual de conclusão do projeto acionam esta captura. Sempre que um destes eventos ocorre, os agentes iniciam a geração da recomendação através de seis componentes:
  - *Recommendation Engine*: esta *engine* monitora permanentemente os eventos do projeto. Quando um destes eventos é identificado, a *engine* orquestra a execução dos demais agentes (*Agents*);
  - *Translate*: o PLN é realizado utilizando a língua inglesa como base, assim, quando necessário este agente traduz as informações, caso esta não seja a língua nativa do projeto;

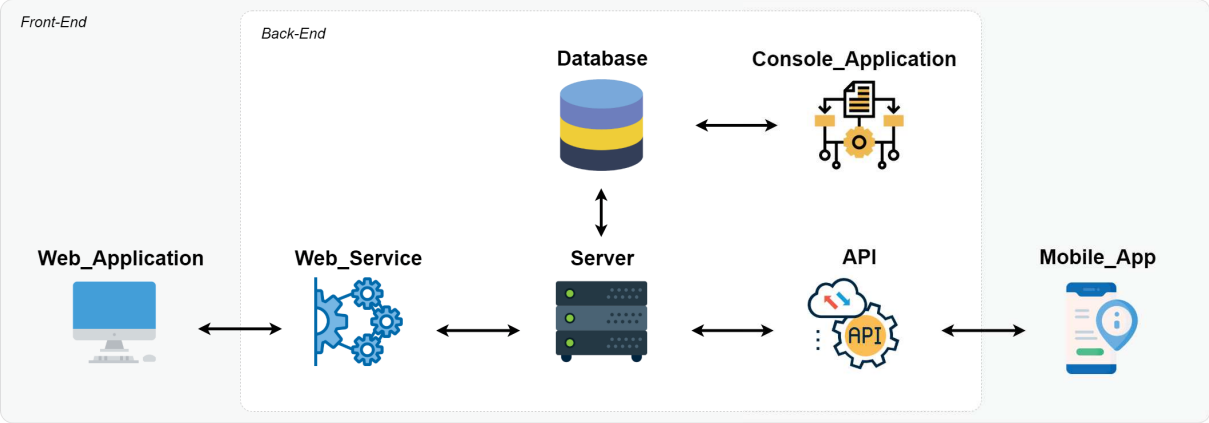
  - *Projects Similarity*: analisa a similaridade dos projetos no repositório. Através de técnicas de PLN, os projetos são agrupados de acordo com sua área de atuação, e as demais características, como tamanho do projeto, metodologia, etc.;
  - *Requirements Similarity*: este agente realiza a análise de similaridade semântica dos requisitos, utilizando sua descrição escrita em linguagem natural, verificando se os requisitos comparados possuem a mesma quantidade de atores;
  - *Context Similarity*: conforme as recomendações são executadas, este agente passa a realizar uma análise de similaridade dos históricos de contextos dos projetos, identificando a sua semelhança;
  - *Context Storage*: armazena cada ocorrência no histórico.

- ***Database***: Responsável por armazenar as configurações da aplicação, como: (a) Dados relativos aos projetos; (b) Recomendações efetuadas pelo modelo; (b) *Feedback* dos *stakeholders* quanto às recomendações e (c) Históricos de contextos ocorridos ao longo do ciclo de vida dos projetos.

4.6 Protótipo e Aspetos da Implementação

Um protótipo de software foi desenvolvido para atender as definições do modelo, que estão dispostas no capítulo 4. Três aplicações integradas compõem o protótipo: (1) *Console Application*; (2) *RESTFull API Application* ou *WebService*; e (3) *Hybrid Application*, composta pelas interfaces *Web Application* e *Mobile App*. As duas primeiras aplicações são da camada *back-end*, executadas em um servidor. A terceira aplicação atua como software da camada de *front-end*, sendo utilizada por todos os usuários envolvidos, durante os processos da Engenharia de Requisitos. A Figura 18 mostra a visão geral do protótipo. Os códigos fonte do protótipo estão disponíveis para livre acesso da comunidade acadêmica (GITHUB, 2019a,b,c).

Figura 18 – Visão Geral do Protótipo



Fonte: Elaborado pelo autor

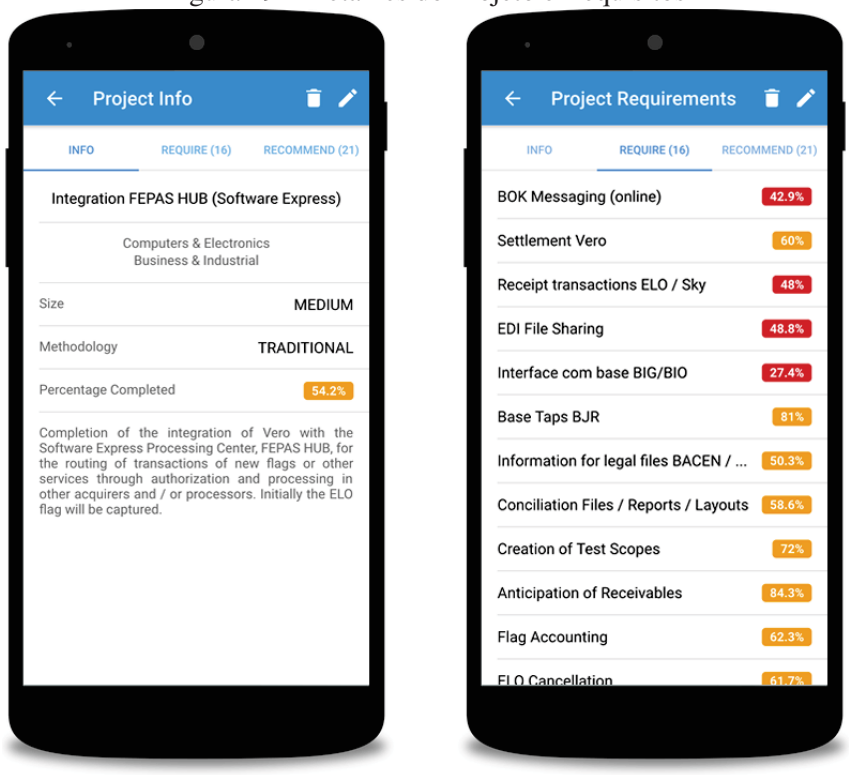
4.6.1 *Hybrid Application*

A aplicação intitulada *Hybrid Application* opera tanto em dispositivos móveis, quanto em computadores convencionais, através de um navegador *web*. Esta aplicação possui duas *interfaces* de comunicação com os usuários do sistema: (1) *Web Application*; e (2) *Mobile App*.

Esta aplicação possibilita a interação das equipes de projeto com o Nhatos, permitindo a gestão dos projetos, requisitos, atividades, recursos, registro das partes interessadas e avaliação das recomendações. O protótipo possibilita o acompanhamento dos projetos, a captura das informações de contexto e seus históricos, e pode ser utilizada durante todo o ciclo de vida dos projetos. Além disso, as recomendações realizadas pelo modelo Nhatos são apresentadas para o usuário através desta aplicação, permitindo assim, a coleta do *feedback* das partes interessadas.

A Figura 19a mostra a *interface* de apresentação do projeto. São mostradas também informações características do projeto, como: tamanho, metodologia, percentual de evolução e área de conhecimento. A listagem dos requisitos do projeto é exibida na Figura 19b, assim como seus respectivos percentuais de evolução.

Figura 19 – Detalhes do Projeto e Requisitos



(a) Projeto e Informações (b) Requisitos e Percentual das Atividades

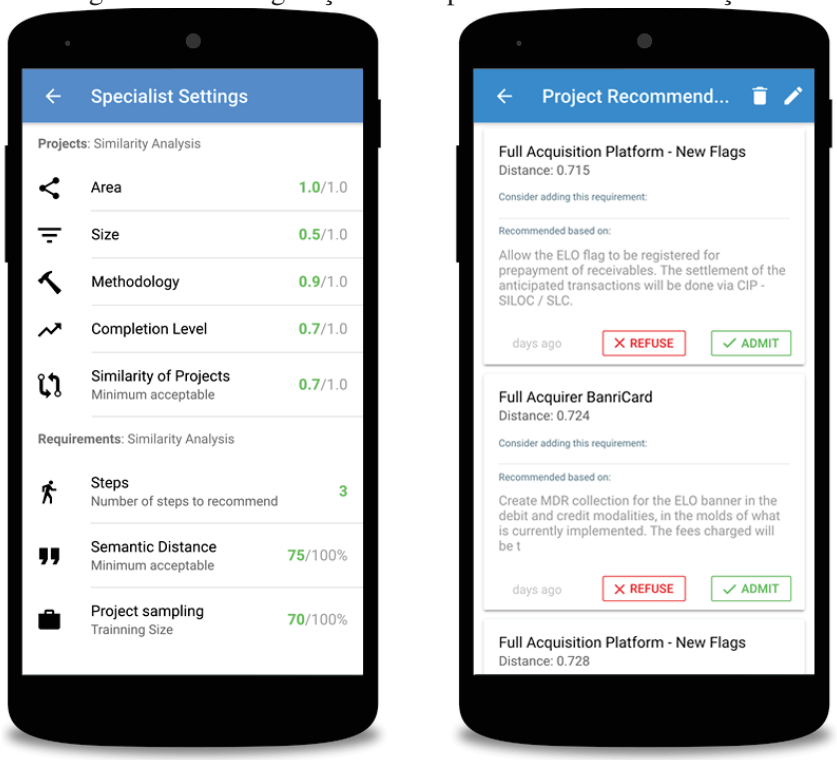
Fonte: Elaborado pelo autor

A Figura 20a exibe as configurações das variáveis de peso, que são definidas por um especialista. Estas variáveis definem a importância de cada aspecto do projeto e seus requisitos, durante o processo de recomendação. O usuário pode definir pesos referentes à: área do projeto,

tamanho, metodologia e nível de conclusão, assim como a proximidade semântica aceitável entre os requisitos que serão considerados para uma possível recomendação.

Por fim, na Figura 20b mostra as recomendações de saída do modelo. Nesta *interface*, o requisito recomendado é apresentado, assim como o requisito que originou esta recomendação, e a distância semântica entre os requisitos comparados. A área de interação ainda permite que o usuário forneça o seu *feedback*, selecionado entre as opções aceitar ou recusar a recomendação. Esta decisão do usuário é armazenada, para futuramente avaliar a aceitabilidade das recomendações pelos *stakeholders*.

Figura 20 – Configurações do Especialista e Recomendações



Fonte: Elaborado pelo autor

4.6.2 *Console Application*

Desenvolvida em linguagem *Python*, esta aplicação trata-se de um software encapsulado que atua no formato de um serviço. Este software utiliza o conceito de sistemas Multi-agentes, proposto por Padgham e Winikoff (PADGHAM; WINIKOFF, 2004), sendo responsável pela execução da camada *Agents* do modelo Nhatos. A aplicação independe da interação direta com os usuários do sistema, sendo acionada a partir de modificações detectadas no ambiente.

Conforme descrito previamente, a camada do modelo intitulada *Agents* possui 6 agentes. Cada agente é responsável por um tipo de processamento distinto. Sendo estes: (1) *Recommendation Engine*; (2) *Translate*; (3) *Projects Similarity*; (4) *Requirements Similarity*; (5) *Context*

Similarity; e (6) *Context Storage*.

O agente *Recommendation Engine* é executado sistematicamente a partir da detecção de mudanças implicadas pelo ambiente. Sempre que ocorre a inserção de um novo projeto ou requisito, este agente é acionado. Uma vez desempenhado, este agente é responsável por iniciar a execução de todos os demais agentes.

O agente *Translate* tem por objetivo traduzir todos os conteúdos inseridos (pelo usuário) para o idioma inglês. Possibilitando assim, a execução dos agentes subsequentes. Uma vez que, como premissa, é necessário que todos os conteúdos estejam cadastrados em idioma inglês. Pois, no caso de uso deste estudo utiliza-se PLN. E, o *corpus* de textos obtidos para a aplicação do estudo é escrito neste idioma. A Figura 21 mostra o específico trecho de código, onde o software carrega os projetos e seus requisitos, presentes na base de dados, traduzindo-os através do uso da *API Google Cloud Translate* (GOOGLE, 2019a). Os projetos são carregados a partir da base de dados (linha 1) e, consequentemente são traduzidos e atualizados (6). O mesmo ocorre com os requisitos (linhas 8 e 13).

Figura 21 – Tradução dos Conteúdos dos Projetos e Requisitos

```
1  projects = get_all_projects()
2  for i, p in enumerate(projects):
3      translate_client = translate.Client()
4      translation = translate_client.translate(p['description'],
5                                               target_language='en')
6      update_project(translation['translatedText'], p['id'])
7
8  requirements = get_all_requirements()
9  for i, r in enumerate(requirements):
10     translate_client = translate.Client()
11     translation = translate_client.translate(r['description'],
12                                              target_language='en')
13     update_requirement(translation['translatedText'], r['id'])
```

Fonte: Elaborado pelo autor

Uma vez traduzidos os conteúdos dos projetos e seus requisitos, o agente *Projects Similarity* é acionado. Este agente analisa a similaridade entre todos os projetos da base de dados. O agente utiliza as características dos projetos, agrupando-os separadamente. O software considera as informações presentes na ontologia representada na Figura 12, considerando todos os projetos da base de dados de acordo com: tamanho, área de conhecimento, metodologia de gerenciamento e nível de completude (cronograma). O Algoritmo 22 mostra a codificação executada por este agente. Após consultar todos os projetos (linha 1), o algoritmo rotula cada projeto (linha 4), classificando-os (linha 7) para que a próxima etapa do algoritmo seja iniciada.

O agente *Requirements Similarity* emprega o uso de PLN para encontrar requisitos que contenham objetivos equivalentes, assim como mesmo número de atores envolvidos. Este agente de software também considera a análise de similaridade entre os projetos, executada previamente. Deste modo, o algoritmo analisa a similaridade entre os requisitos de projetos considerados se-

Figura 22 – Análise de Similaridade dos Projetos

```
projects = get_all_projects()

for i, p in enumerate(projects):
    label = get_project_label(p.area, p.methodology, p.size,
                              p.requirements)

    label_project(label)
```

Fonte: Elaborado pelo autor

melhantes. Abaixo, está disposto o trecho do algoritmo na Figura 23, onde o agente analisa a similaridade entre os requisitos encontrados.

Figura 23 – Análise de Similaridade dos Requisitos entre Projetos Semelhantes

```
distance, steps, sample, counter = 0.25, 5, 0.7, 0
delete_recommendations(distance, sample, steps)
projects = get_all_projects()

for i, prj in enumerate(projects):
    requirements = get_requirements_by_project_id(prj['id'])
    prj_to_compare = get_projects_by_domain(prj['domain'])

    for i, pc in enumerate(prj_to_compare):
        if (prj['id'] == pc['id']): continue
        req_to_compare = get_requirements_by_project_id(pc['id'])
        loop = min(int(round(len(requirements) * sample)),
                   len(req_to_compare)) - 1

        for i in range(loop):
            compare = get_req_distance(requirements[i]['id'],
                                       req_to_compare[i]['id'])
            if (compare is None): continue
            if (compare['distance'] <= distance): counter += 1
            else: counter = 0

            if (counter == steps and i != len(req_to_compare)):
                insert_rec(prj['id'], req_to_compare[i + 1]['id'],
                           requirements[i]['added'], distance,
                           sample, steps)
                counter = 0
```

Fonte: Elaborado pelo autor

Nesta etapa, o algoritmo se apropria das novas configurações de distância, passos e amostragem, iniciando assim um novo processamento (linha 1). Em seguida, remove as recomendações anteriores (se houver), para inicializar um novo processo de recomendações, através do método *delete recommendations* (linha 2). Este método remove todas as recomendações que originaram-se de uma mesma configuração de distância, número de passos e amostragem. Pois, considera-se que ao longo do ciclo de vida, os requisitos podem ter sofrido modificações

referentes aos seus objetivos ou atores envolvidos (PMI, 2017a).

Depois, o algoritmo obtém todos os projetos da base de dados (linha 3). Para cada projeto, são obtidos os seus requisitos, através do método *get requirements by project id* (linha 6). Cada requisito obtido é confrontado com os requisitos de projetos semelhantes, adquiridos a partir do método *get projects by domain* (linha 7). Nesta etapa, verifica-se se o objetivo de ambos os requisitos atende aos parâmetros estabelecidos de distância, assim como se a quantidade de atores (linha 16). Por fim, uma vez que a quantidade de requisitos encontrados, de modo sequencial, for igual à quantidade de passos configurados, este requisito é considerado recomendável ao projeto (linha 23).

O agente *Context Similarity* analisa se as recomendações, realizadas dentro da amostragem selecionada pelo especialista, se tornaram presentes no decorrer do projeto. A Figura 24 mostra o trecho da codificação responsável por esta análise. Portanto, o software carrega todas as recomendações realizadas (linha 1). Assim, para cada recomendação, o algoritmo seleciona os requisitos subsequentes à sua data de cadastro, do projeto ao qual a recomendação fora realizada (linha 2). Se dentre os requisitos encontrados for encontrado um requisito com mesma distância semântica ao requisito comparado e, contenha o mesmo número de atores, esta recomendação é considerada assertiva (linha 6). Do contrário, a recomendação realizada é considerada não assertiva. Por fim, esta decisão é armazenada na base de dados (linha 10).

Figura 24 – Análise de Similaridade dos Históricos de Contextos

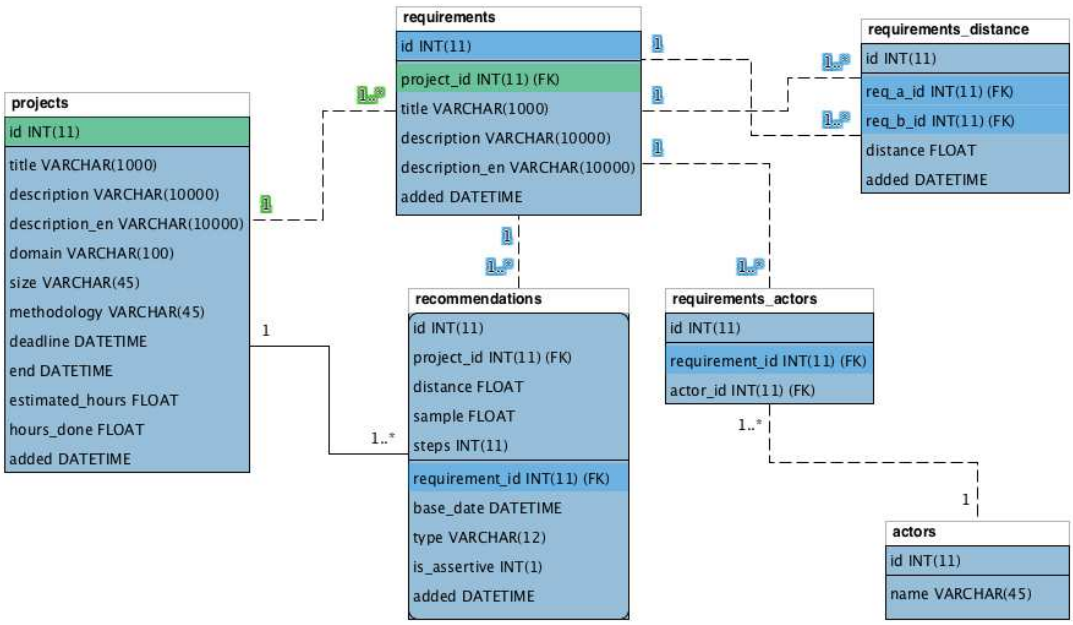
```
1  for i, rec in enumerate(get_all_recommendations()):
2      reqs = get_req_by_date(rec['project_id'], rec['base_date'])
3
4      is_assertive = False
5      for i, req in reqs:
6          cp = get_req_distance(rec['requirement_id'], reqs[i]['id'])
7          if (cp is None): continue
8          if (cp['distance'] <= rec['distance']): is_assertive = True
9
10     update_rec(rec['id'],
11                (lambda assertive: 1 if is_assertive == True
12                 else 0)(is_assertive))
```

Fonte: Elaborado pelo autor

Por fim, o agente *Context Storage* armazena as informações do modelo em uma base de dados com quatro entidades principais: (1) *projects*; (2) *requirements*; (3) *requirements distance*; e (4) *recommendations*. A Figura 25 mostra do diagrama de entidade relacional da base de dados do modelo.

A entidade *projects* é responsável por armazenar as informações referentes aos projetos. Nesta entidade, além de informações básicas dos projetos, são armazenadas as suas informações características, como: termo de abertura (*description*), área de conhecimento (*domain*), tamanho (*size*) e metodologia aplicada durante o seu desenvolvimento (*methodology*).

Figura 25 – Entidade Relacional do Modelo Nhatos



Fonte: Elaborado pelo autor

A entidade *requirements* é responsável por armazenar os requisitos do modelo. Um mesmo projeto pode conter diversos requisitos, conforme relacionamento presente no diagrama (1..n). Os atores de cada requisito são memorizados na entidade *actors* e, um ator pode estar vinculado a diversos requisitos e vice versa, assim como mostrado na entidade *requirements_actors* (n..n).

A entidade *requirements_distance* armazena as informações de distância entre os requisitos processados. A entidade armazena o requisito original (*req_a_id*) e o requisito comparado (*req_b_id*), assim como a respectiva distância (*distance*) semântica entre eles.

Por fim, a entidade *recommendations* é responsável por armazenar as recomendações inferidas pelo Nhatos. Cada recomendação é direcionada a um projeto (*project_id*) e o requisito que gerou tal recomendação (*requirement_id*). A avaliação da assertividade de cada recomendação é guardada na propriedade *is_assertive*.

4.6.3 *RESTFull API Application* ou *Web Service*

A *RESTFull API Application* é a aplicação responsável por prover um canal de comunicação entre a aplicação *Hybrid_Application* e a aplicação *Context Storage*. Fazendo uso do protocolo *RESTFull*, a aplicação permite o tráfego de dados no formato *JSON* entre as aplicações. Ela possibilita a troca de dados entre a aplicação híbrida, utilizada pelos usuários, e as informações já processadas pela *Console_Application*, que armazena suas informações na base de dados do modelo Nhatos.

4.7 Considerações sobre o Capítulo

Este capítulo apresentou a descrição do modelo Nhatos, assim como sua representação de domínio através de uma ontologia. Também foi detalhada a arquitetura do projeto do modelo e como é realizada a análise de similaridade entre os projetos e seus requisitos, mostrando os detalhes da implementação. Também foi mostrado o protótipo desenvolvido para apoiar o modelo durante sua avaliação e seus detalhes de funcionamento. O capítulo seguinte mostra os resultados obtidos através do uso deste protótipo em um ambiente real de projetos, onde equipes de software foram expostas às recomendações geradas pelo modelo.

5 AVALIAÇÃO

As questões de pesquisa apresentadas neste trabalho foram respondidas através da aplicação de um estudo de caso em uma empresa de desenvolvimento de software. Esta empresa desenvolve soluções de aplicativos para uma instituição bancária. A partir da definição das questões de pesquisa, este estudo teve como objetivo confirmar a hipótese da utilização dos históricos de contextos de projetos, para assim recomendar requisitos a projetos novos ou em fase de andamento.

Um histórico contendo 183 projetos foi utilizado como base para o estudo de caso. Estes projetos ocorreram no período entre 2017 e 2019. O banco de dados histórico possui projetos com diferentes características, e os projetos são desenvolvidos de forma distribuída ou local. O uso de diferentes características dos projetos permite uma análise dos diferentes contextos para recomendação de requisitos.

O estudo foi aplicado em dois cenários: (1) Duas equipes avaliaram o uso do protótipo durante a implementação de 5 projetos; (2) Considerou-se 17 projetos concluídos para avaliar as recomendações feitas pelo modelo Nhatos, comparando as recomendações com os requisitos nos projetos originais.

5.1 Cenário 1: Avaliação Durante a Execução dos Projetos

No primeiro cenário, o estudo de caso foi realizado contendo a participação de duas equipes, com um total de 12 profissionais. Requisitou-se a estes participantes a validação das recomendações efetuadas pelo modelo. A Tabela 7 mostra o perfil das equipes que participaram do estudo de caso.

5.1.1 Recomendação na Fase Inicial do Projeto

Inicialmente foram inseridas na base de dados as informações referentes aos projetos (escopo e informações descritivas, termo de abertura, recursos, cronograma e tarefas/atividades). Esta interação foi realizada através da interface de integração (*Integration Service*). Posteriormente, os times de projeto identificaram e cadastraram os requisitos fazendo uso do protótipo por meio de dispositivos móveis.

A análise de similaridade dos projetos foi efetuada através da utilização de PLN. Através do uso de PLN, classificou-se cada projeto de acordo com sua respectiva área de conhecimento. A identificação da área de conhecimento dos projetos envolveu a utilização do termo de abertura. O termo de abertura contém a descrição do projeto em alto nível. A classificação foi realizada através do uso da API *Google Natural Language* (GNL). A GNL fornece recursos para a análise de textos não estruturados, como classificação de conteúdos e identificação de entidades. A classificação de conteúdo analisa um documento e resulta em uma lista de categorias que se

Tabela 7 – Perfil das Equipes Participantes durante a Avaliação

Equipe	Estrutura	Papel	Experiência	Localização
Equipe A	Local	Scrum Master	15+	Rio Grande do Sul, Brasil
		Product Owner	10+	Rio Grande do Sul, Brasil
		Designer	5+	Rio Grande do Sul, Brasil
		Desenvolvedor	10+	Rio Grande do Sul, Brasil
		Desenvolvedor	5+	Rio Grande do Sul, Brasil
		Desenvolvedor	5+	Rio Grande do Sul, Brasil
		Analista de Testes	5+	Rio Grande do Sul, Brasil
Equipe B	Distribuída	Gerente de Projetos	25+	São Paulo, Brasil
		Desenvolvedor	10+	São Paulo, Brasil
		Desenvolvedor	5+	Rio Grande do Sul, Brasil
		Desenvolvedor	5+	Rio Grande do Sul, Brasil
		Analista de Testes	5+	Rio Grande do Sul, Brasil

Fonte: Elaborado pelo autor

aplicam ao texto encontrado. A classificação ainda pode conter diversos níveis, especificando maior profundidade de detalhes sobre a área de conhecimento em questão (GOOGLE, 2019b).

Atualmente a GNL processa apenas sentenças do idioma inglês. Deste modo, necessitou-se traduzir previamente o conteúdo descritivo dos projetos antes de efetuar o processo de classificação. A tradução foi realizada de forma automatizada pelo protótipo, através do uso da API *Google Cloud Translation* (GCT) (GOOGLE, 2019a). A GCT recebe como entrada uma sentença e, independente do idioma inserido, identifica a sua linguagem e resulta em sua tradução para o idioma selecionado pelo usuário. Durante o estudo, todas as sentenças foram traduzidas para o idioma inglês. Após a tradução, o Nhatos classificou os projetos fazendo uso da GNL.

A Tabela 8 mostra os resultados obtidos nesta etapa, mostrando a classificação efetuada a partir da base do histórico. Nesta tabela, são exibidas as categorias com maior número de projetos classificados, para fim de demonstração dos resultados.

Tabela 8 – Categorização dos Projetos

Categoria	Qtd. Projetos	Percentual [1]
Finance	23	15,03 %
Business and Industry	20	13,07 %
Computers and Eletronics	10	6,53 %
Credit and Lending	6	3,92 %
Accounting and Auditing	4	2,61 %

[1] Percentual em relação à quantidade total de projetos

Fonte: Elaborado pelo autor

Além de definir a área de conhecimento de cada projeto, para análise de similaridade, foram levadas em consideração as variáveis registradas no módulo *Configuration*. Os pesos foram atribuídos a estas variáveis, que refletem a singularidade de cada projeto. A configuração de peso permite que o algoritmo gere recomendações diferentes de acordo com as características registradas pelo especialista. A Tabela 9 apresenta os valores de requisitos recomendados para

cada projeto e os valores de requisitos que foram adicionados aos novos projetos por meio das recomendações.

Tabela 9 – Recomendações Realizadas por Projeto

Projeto	**Área**	**Recomendações**	**Aceitas**	**% Aceitas**
Operações Renegociadas - Reestruturadas	Finance	16	11	68.7
Alteração Renov Autom Cheque Especial PF	Finance	14	8	57.1
Movimentações de CDB no M-BANKING	Business & Industrial	14	7	50.0
Parametrização de Indexadores	Finance	14	9	64.2
Renovação automática de travas	Finance	13	8	61,6
Percentual de Aprovação				75.1

Fonte: Elaborado pelo autor

5.1.2 Recomendação Durante a Execução do Projeto

Para avaliar as recomendações realizadas durante a execução do projeto, 41 requisitos foram incluídos nos projetos ao longo do estudo de caso. Assim, sempre que um novo requisito é adicionado, o agente *Requirements Similarity* identifica o evento e realiza a análise de proximidade semântica. O agente compara a nova descrição do requisito com os requisitos armazenados no histórico do projeto.

A Tabela 10 mostra um exemplo de um requisito adicionado a um projeto e os requisitos recomendados com base nesta inserção. O Nhatos considera o objetivo de cada requisito e também o número de atores relacionados.

Tabela 10 – Análise Semântica para Recomendação dos Requisitos

Requisito Incluído	**Requisitos Recomendados**	**Distância**
The software must allow the manager to request airline tickets (2 actors).	*The software must allow the manager to order supplies (2 actors).*	0,21
	The software must allow the manager to request resource transfers between projects (2 actors).	0,32

Fonte: Elaborado pelo autor

Nesta fase, a análise é aplicada a todo o banco de dados, independentemente dos projetos terem sido comparados na primeira etapa da recomendação. Pois, neste momento, trata-se da semântica do texto incluído e, os objetivos dos requisitos podem sofrer modificações ao longo do projeto (ELIZABETH H. JEREMY D., 2017). Esta etapa também permite que os requisitos do projeto que não foram originalmente recomendados sejam analisados e considerados. A análise é realizada neste ponto sobre os requisitos, considerando o agrupamento dos projetos com base em suas características. A distância semântica, que representa o histórico de contexto do objetivo do requisito, possui um valor de ponto flutuante entre 0.0 e 1.0. A distância mais próxima de 0 indica que o requisito recomendado é semanticamente mais próximo do original.

Consequentemente, quanto mais próxima a distância semântica de 1, menos similar o requisito recomendado é considerado quando comparado ao original. Os atores de cada requisito também são considerados durante esta etapa da análise. Este histórico de contexto é definido pela quantidade dos atores ao qual um requisito está relacionado.

Durante o estudo de caso, para cada um dos 5 projetos, o modelo analisou a similaridade de acordo com a Equação 4.1, recomendando os requisitos entre projetos semelhantes. Logo em seguida, a equipe analisou os requisitos recomendados para os 5 projetos registrados. Uma média de 75,1% dos requisitos foi aprovada, conforme mostra a Tabela 9 (Taxa de aprovação). A apropriação média das recomendações apresentada para novos projetos mostra a aceitação do modelo de recomendação de requisitos avaliado pelas equipes, com o objetivo de fornecer mais informações aos gerentes desde o início do projeto.

5.2 Cenário 2: Avaliação Através da Análise dos Históricos de Contextos

Neste cenário, 183 projetos concluídos foram utilizados para avaliar as recomendações de requisitos feitas pelo modelo Nhatos. As recomendações foram comparadas com os requisitos originalmente registrados nos projetos. Assim, é possível inferir se as recomendações realizadas, considerando uma amostra de 70% do andamento dos projetos, de fato foram inseridos no projeto durante seu percentual restante.

Projetos com características diferentes foram utilizados para avaliar as recomendações feitas em várias situações. A maioria dos projetos (112) foi desenvolvida seguindo a metodologia ágil, usando o *framework* SCRUM (SUTHERLAND J., 2019). Os demais (71), foram geridos através de metodologia tradicional, baseada nas boas práticas propostas pelo PMBOK (PMI, 2017a). A classificação de tamanho dos projetos foi definida pela quantidade de tempo necessária para executá-lo. Deste modo, cada projeto foi classificado em uma de três categorias de tamanho: pequeno (até 500 h), médio (até 3000 h) e grande (acima de 3000 h).

Neste cenário, uma amostragem de 70% referente à execução de cada projeto foi utilizada como aprendizado para o modelo Nhatos. Com esse aprendizado, o Nhatos gerou recomendações de requisito para os mesmos projetos. Assim, este caso de uso teve como objetivo avaliar se os requisitos recomendados estão contidos nos 30% restantes dos projetos executados.

Para analisar a semelhança entre os projetos, o Nhatos considera os seus respectivos históricos de contextos. O modelo busca por sequências semelhantes de contextos durante a evolução no ciclo de vida dos projetos. Em seguida, o mesmo recomenda os requisitos que ocorreram após a sequência encontrada para o projeto que está sendo implementado, desde que estas sequências sejam encontradas em projetos similares. Deste modo, comparar uma sequência de eventos que gerou históricos de contextos aproximada aumenta a probabilidade de que os cronogramas de andamento dos projetos sejam equivalentes.

Diferentes parametrizações foram configuradas junto ao modelo para encontrar o melhor cenário de recomendações. Ao todo, foram configurados 9 cenários de testes. Cada teste gerou

uma nova rodada de processamento envolvendo toda a base de dados histórica. A Tabela 11 mostra os resultados das diferentes configurações aplicadas. Um cenário de testes é caracterizado por diferentes combinações das variáveis *Distância*, *Passos* e *Amostra*.

Tabela 11 – Avaliação das Recomendações Realizadas pelo Modelo Nhatos

#	Distância	Passos	Amostra	Recomendações Não Assertivas	Recomendações Não Assertivas (%)	Recomendações Assertivas	Recomendações Assertivas (%)	Total
1	0,25	3	0,7	248	51,56	233	48,44	481
2	0,25	4	0,7	19	70,37	8	29,63	27
3	0,25	5	0,7	1	50	1	50	2
4	0,3	3	0,7	555	29,65	1317	70,35	1872
5	0,3	4	0,7	194	30,03	452	69,97	646
6	0,3	5	0,7	56	28,43	141	71,57	197
7	0,35	3	0,7	904	16,96	4427	83,04	5331
8	0,35	4	0,7	443	17,51	2087	82,49	2530
9	0,35	5	0,7	237	17,52	1116	82,48	1353

Fonte: Elaborado pelo autor

Todos os cenários de testes foram configurados para receber uma amostra de valor 0,7. Sendo assim, considerou-se como amostra de treinamento 70% da evolução do cronograma (ciclo de vida) dos projetos para gerar as recomendações. Os testes 1, 2 e 3 foram configurados com uma distância semântica mínima entre os requisitos de 0,25 (75% semelhantes). Todas as 3 configurações de passos para estes testes resultaram em uma taxa de assertividade igual ou inferir a 50%. O cenário 2 e 3 gerou poucas recomendações, 27 e 2 respectivamente. O teste 1 gerou um total de 481 recomendações. Mas, como mencionado, sua assertividade não alcançou o percentual de 50%.

Por outro lado, os testes 7, 8 e 9 geraram uma quantidade significativa de recomendações. A taxa de assertividade se mostrou alta, acima de 80% nos três casos. No entanto, a distância se mostrou relativamente abrangente, considerando requisitos apenas 65% similares, dada a sua distância semântica. Portanto, uma quantidade elevada de recomendações de requisitos foi inferida durante estes testes (um total de 9214). Uma aplicação ubíqua deve ser minimamente intrusiva (Satyanarayanan, 2001), e a alta quantidade de recomendações destes cenários não atenderia este requisito, uma vez que o sistema emitiria mais recomendações do que as equipes poderiam avaliar.

Os cenários 4, 5 e 6 mostraram-se como mais promissores durante os testes realizados. Ambos obtiveram taxa percentual de assertividade próximas ou superiores a 70%. No entanto, uma quantidade de recomendações aceitável para se obter uma análise através do caso de uso foi apresentada apenas nos cenários 4 e 5.

O cenário 4 foi configurado com uma distância mínima de 0,3 (70%) e 3 passos, onde obteve-se uma taxa de acerto de 70,35% quanto às recomendações assertivas. Das 1872 recomendações inferidas, 1317 foram consideradas corretas, enquanto que 555 foram consideradas incorretas. O cenário 5 foi configurado com uma distância mínima de 0,3 (70%) e 4 passos. Este cenário obteve uma taxa de acerto de 69,97% de recomendações assertivas. 452 das 646 recomendações foram corretas, enquanto que 194 não obtiveram sucesso.

A observação de diferentes cenários permitiu que uma nova rodada de testes fosse executada a cada nova configuração realizada pelo especialista. Assim, obtiveram-se as configurações mais assertivas (cenários de testes 4 e 5) para a base de dados utilizada.

5.3 Resultados e Respostas às Questões de Pesquisa

O estudo de caso corroborou para a observação de lacunas no gerenciamento de requisitos do projeto. Ao usar o modelo Nhatos, por meio do protótipo desenvolvido, as equipes puderam monitorar e analisar os requisitos. E, as equipes de projeto receberam recomendações no início e durante a implementação de cada projeto.

A aplicação do estudo de caso teve caráter de avaliação das hipóteses criadas com base nas questões da pesquisa deste estudo. As questões de pesquisa foram elaboradas para validar o uso do modelo Nhatos em duas dimensões: (i) recomendação de requisito considerando o históricos de contextos dos projetos; (ii) observação dos processos de gerenciamento de requisitos, permitindo seu uso de forma colaborativa. Nesse sentido, os resultados e as evidências apresentadas neste estudo demonstram a aderência do modelo Nhatos ao gerenciamento proativo de requisitos em projetos de software, pois: (a) O modelo demonstrou taxa de acerto de 83,04% em suas recomendações e (b) A colaboração dos times de projeto, durante os processos de Engenharia de Requisitos, contribuiu para a coleta de históricos de contextos e *feedback* das recomendações realizadas pelo modelo Nhatos.

O principal objetivo do Nhatos é recomendar requisitos considerando as características dos projetos e analisar os históricos de contexto, além de monitorar todo o ciclo de vida dos requisitos ao longo do projeto. A primeira questão de pesquisa apresentada (QP 1) questionou a possibilidade da utilização dos históricos de contextos dos projetos durante a recomendação de requisitos. A segunda questão de pesquisa (QP 2) teve como objetivo responder se a colaboração das equipes, ao longo de todo o ciclo de vida dos projetos, contribui para a recomendação de requisitos.

Assim, para responder à QP 1 através da aplicação do estudo de caso descrito, conclui-se como verdadeira a hipótese desenvolvida inicialmente na Figura 1 de que os históricos de contexto do projeto contribuem para gerar recomendações de requisitos aos projetos. Com uma taxa média de aprovação de recomendações de 75,1%, a partir das configurações dos especialistas, apresentada no cenário 1, o modelo Nhatos mostrou-se alinhado à recomendação de requisitos. Ao considerar as características de cada projeto para a recomendação, as informações recomendadas pelo modelo para os projetos se tornam cada vez mais assertivas. Assim, os engenheiros de requisitos, ao iniciar um novo projeto, terão um conjunto maior de informações aderentes ao projeto em andamento, proporcionando um melhor planejamento.

O segundo cenário da avaliação teve como objetivo explorar os resultados com base na análise de similaridade dos históricos de contexto dos projetos. Assim, diferentes configurações foram geradas no Nhatos para testar a assertividade das recomendações. O modelo conseguiu

atingir 83,04% de assertividade quanto aos requisitos recomendados, a partir da aplicação de diferentes cenários, reforçando a hipótese do uso dos históricos de contextos do projeto como verdadeiros para a recomendação de requisitos.

Quanto à QP 2, durante o período de acompanhamento do projeto no estudo de caso, foram registrados 41 novos requisitos. Isto possibilitou a recomendação de novos requisitos aos projetos semelhantes. Após o registro dos requisitos, os mesmos foram analisados de forma colaborativa, onde todos os membros da equipe puderam analisar cada requisito e contribuir com o prévio conhecimento durante os processos de elicitação, especificação e validação. Este aspecto permitiu que mais informações fossem inseridas aos projetos e consideradas pelo modelo, pois um grupo maior de pessoas pôde analisar tanto os requisitos quanto as recomendações à eles apresentadas, resultando assim em um total de 43 recomendações aceitas (75% do total). Deste modo, concluiu-se que a colaboração das equipes durante o gerenciamento de requisitos permitiu uma ampla análise e visualização dos impactos que poderiam ocorrer nos projetos de software. O protótipo desenvolvido permitiu que as equipes colaborassem durante todos os processos de gerenciamento de requisitos. Os times de projeto puderam avaliar as recomendações feitas pelo modelo, tanto no início da fase de planejamento como no decorrer do projeto, uma vez que o modelo atualiza suas saídas a cada modificação observada pelos seus agentes.

A partir da confirmação das hipóteses, o estudo apresenta, como principal contribuição científica, o uso da análise de similaridade do projeto através dos históricos de contextos para gerar recomendações de requisitos. Assim, fornecendo recomendações seguindo a semelhança do projeto e dos contextos capturados ao longo do seu ciclo de vida.

5.4 Considerações sobre o Capítulo

Este capítulo apresentou a avaliação do modelo Nhatos, seu detalhamento e os resultados obtidos. O capítulo seguinte apresenta as considerações finais deste trabalho. As contribuições detalham a colaboração acadêmica e o modo como este estudo foi organizado.

6 CONSIDERAÇÕES FINAIS

Esta dissertação apresentou o modelo Nhatos, um modelo para a recomendação de requisitos para projetos de software. O Nhatos foi desenvolvido baseado nas características comuns e nas lacunas identificadas nos trabalhos relacionados (XIE et al., 2017; HERRERA et al., 2009b; BAKAR et al., 2016; PORTUGAL et al., 2017; SHI et al., 2013; WILLIAMS et al., 2017; DUMITRU et al., 2011; NAKATANI et al., 2014; GARCIA et al., 2016; Zhao; Zhang, 2018; Liu et al., 2018; Kim; Dey; Lee, 2019; ZHIYING et al., 2019). Os estudos analisados apontam que a recomendação ou reuso de requisitos é uma promissora alternativa para auxiliar os engenheiros de requisitos durante a gestão dos projetos. Os estudos relacionados citados não exploraram a similaridade dos projetos ou seus históricos de contexto ao longo do ciclo de vida, sendo estes fatores presentes unicamente no modelo Nhatos.

O modelo faz uso da análise dos históricos de contextos dos projeto. Através das recomendações realizadas, os gestores são direcionados às atividades com maior relevância para a adição de benefícios ao projeto. Os resultados mostram a aplicabilidade da recomendação de requisitos para novos projetos a partir da análise de similaridade dos seus históricos. Deste modo, o modelo fornece recomendações considerando as características de cada novo projeto, fazendo com que o gestor inicie o seu planejamento com um conjunto maior de informações para uma gestão mais assertiva.

Durante a execução do projeto, a análise sobre os requisitos incluídos permitiu que novas recomendações fossem realizadas, considerando uma análise semântica do seu conteúdo e a sua proximidade aos requisitos recomendados. Isto permitiu que novos cenários para os projetos passassem a ser considerados em sua execução. Além disso, o Nhatos considera os históricos de contextos dos projetos durante a recomendação de novos requisitos. Pois, considera a linha de tempo dos contextos dos projetos similares e captura eventos ao longo da sua evolução.

A participação dos integrantes dos times durante a gestão dos requisitos proporciona ao modelo a colaboração dos *stakeholders*, uma abordagem que não está presente em todos os trabalhos pesquisados, uma vez que normalmente a gestão do projeto é realizado pela equipe responsável pelas atividades. Este diferencial torna possível a coleta de uma maior quantidade de informações durante a execução do projeto e traz consigo conhecimento técnico e prático sobre as definições dos requisitos.

A utilização de uma aplicação híbrida multi-plataformas durante as interações das equipes de projeto contribuiu para tornar o gerenciamento de requisitos compatível com a realidade de projetos de diferentes formatos de equipes, sejam estas distribuídas ou tradicionais. Esta abordagem não foi explorada nos demais trabalhos, e garante uma dinâmica que favorece o uso do modelo de maneira amigável e natural aos usuários, uma vez que o Nhatos pode ser acessado pelo *smartphone* dos usuários.

O modelo Nhatos contribui para a redução das incertezas durante a gestão dos projetos. Pois, ao fornecer recomendações contextualizadas às equipes provê o uso de informações históricas

que normalmente não seriam levadas em consideração durante os processos da Engenharia de Requisitos.

6.1 Trabalhos Futuros

Com base nos resultados obtidos através do estudo de caso, foram encontradas questões que podem ser melhor exploradas em trabalhos futuros. A utilização contínua deste modelo permite formar uma base de informações que contribui para a recomendação de requisitos em projetos futuros. A partir de uma base de dados com históricos de contextos constantemente atualizada, é possível realizar recomendações mais assertivas de requisitos.

Outro ponto observado, que pode se tornar uma extensão deste estudo, seria o uso de *Machine Learning* para identificar as configurações com maiores níveis de acerto, considerando diferentes cenários de aplicação. Utilizando esta tecnologia, seria possível identificar a configuração mais adequada para cada cenário dos pesos a serem utilizados na Equação 4.1, configurações estas que foram definidas manualmente no presente estudo.

Os algoritmos aqui apresentados na aplicação do modelo Nhatos para recomendação através da análise dos históricos de contextos, podem ser estendidos para uso das recomendações em outras áreas da gestão de projetos, tais como: (a) Escopo, através da análise e recomendação de requisitos; (b) Tempo, prevendo possíveis atrasos em cronogramas ou alocação de recursos; (c) Custos, o modelo Nhatos disponibiliza uma interface para gestão de requisitos, o que permite a captura das informações que formam a evolução do projeto, mas, ao se analisar os históricos de contextos é possível realizar uma análise sobre o cronograma e poder prever possíveis desvios dos requisitos, como sua qualidade e possíveis fraquezas em suas definições.

6.2 Contribuições

Através de um estudo de mapeamento sistemático para a identificação dos trabalhos relacionados, dois diferenciais acadêmicos são apresentados através do modelo Nhatos: (1) Utilização dos históricos de contextos dos projetos como forma de classificação e análise de similaridade; e (2) Colaboração entre os envolvidos.

O primeiro diferencial aborda a vantagem do uso de históricos de contextos de projetos como forma de identificar e prover o reuso de requisitos semelhantes entre projetos. Este trabalho defende que a análise deste histórico permite a identificação de requisitos no início de um projeto, provendo maiores subsídios para os gestores em seu planejamento, seja em termos de custo, escopo ou de tempo de projeto. O modelo propõe a captura de históricos de contextos dos projetos ao longo da sua evolução de cronograma. Esta estratégia possibilita a recomendação de requisitos através da análise de similaridade dos históricos de contextos dos projetos, uma vez que as informações tanto dos projetos quanto dos requisitos mudam com o passar do tempo.

Por fim, o segundo diferencial explorado pelo modelo Nhatos aborda os benefícios da cola-

boração entre os envolvidos no projeto. Isto se dá durante o processo de gestão dos requisitos. Através do engajamento das partes interessadas, considera-se que o processo de gestão seja apoiado com novas informações a qualquer instante e com diferentes pontos de vista referente aos requisitos durante a evolução do seu ciclo de vida.

REFERÊNCIAS

AMEYED, D.; MIRAOUI, M.; TADJ, C. A survey of prediction approach in pervasive computing. **International Journal of Scientific and Engineering Research**, [S.l.], v. 6, 2015.

BAKAR, N. H. et al. Extracting features from online software reviews to aid requirements reuse. **Applied Soft Computing**, [S.l.], v. 49, p. 1297–1315, 2016.

BARBOSA, J. Ubiquitous computing: applications and research opportunities. , [S.l.], p. 1–8, 2015.

BARBOSA, J. et al. Trailcare: an indoor and outdoor context-aware system to assist wheelchair users. **International Journal of Human-Computer Studies**, [S.l.], v. 116, p. 1 – 14, 2018.

BATISTA, M. et al. Chronos: a model for ubiquitous project. **IADIS International Conference on Applied Computing**, [S.l.], v. 1, p. 1–5, 2011.

BLAKE, M. B. et al. Shared service recommendations from requirement specifications: a hybrid syntactic and semantic toolkit. **Information and Software Technology**, [S.l.], v. 57, n. 1, p. 392–404, 2015.

BURBEY, I.; L. MARTIN, T. A survey on predicting personal mobility. **International Journal of Pervasive Computing and Communications**, [S.l.], v. 8, 2012.

CAMBRIDGE. **Statutory body meaning in the cambridge english dictionary**. [Online; accessed 19-July-2017].

CASAMAYOR, A. et al. Identification of non-functional requirements in textual specifications: a semi-supervised learning approach. **Information and Software Technology**, [S.l.], v. 52, n. 4, p. 436–445, 2010.

CHEN, C. et al. An in-process customer utility prediction system for product conceptualisation. **Expert Systems with Applications**, [S.l.], v. 34, n. 4, p. 2555–2567, 2008.

DEY, A. K.; ABOWD, G. D.; SALBER, D. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. **Hum.-Comput. Interact.**, Hillsdale, NJ, USA, v. 16, n. 2, p. 97–166, Dec. 2001.

DIAS, L. et al. Gamification and serious games in depression care: a systematic mapping study. **Telematics and Informatics**, [S.l.], v. 35, n. 1, p. 213–224, 2018.

DRIVE, G. **Shared data from systematic mapping**. Disponível em: https://https://drive.google.com/drive/u/1/folders/1ZzUwv0frY8LehXyPYUd20mpt0NWj9xDa. Acesso em: 19 de Junho de 2019, `https://drive.google.com/drive/u/1/folders/1ZzUwv0frY8LehXyPYUd20mpt0NWj9xDa`.

DUMITRU, H. et al. On-demand feature recommendations derived from mining public product descriptions. **Proceeding of the 33rd international conference on Software engineering - ICSE '11**, [S.l.], p. 181., 2011.

DUPONT, D.; BARBOSA, J. L. V.; ALVES, B. M. CHSPAM: a multi-domain model for sequential pattern discovery and monitoring in contexts histories. **Pattern Analysis and Applications**, [S.l.], June 2019.

ELIZABETH H. JEREMY D., K. J. **Requirements engineering**. Newtown Square, PA: Springer, Cham, 2017. v. 5.

FELFERNIG, A. et al. Recommendation and decision technologies for requirements engineering. , [S.l.], p. 11–15, 2010.

Filippetto, A. et al. A project management model based on an activity theory ontology. , [S.l.], p. 1–11, 2016.

FILIPPETTO, A.; LIMA, R.; BARBOSA, J. Lean risk – gestão de riscos em times distribuídos. **Universo PM**, [S.l.], v. 2, p. 07–13, 2011.

FOURNIER, D. et al. Towards ad hoc contextual services for pervasive computing. , [S.l.], 01 2006.

GARCIA, J. et al. Reqanalytics: a recommender system for requirements maintenance. **International Journal of Software Engineering and its Applications**, [S.l.], v. 10, n. 1, p. 129–140, 2016.

GASPARIC, M.; JANES, A. What recommendation systems for software engineering recommend: a systematic literature review. **Journal of Systems and Software**, [S.l.], v. 113, p. 101 – 113, 2016.

GITHUB. **Aplicação híbrida para recomendações de requisitos**. [Online; acessado em 20/12/2019], https://github.com/robsonklima/nhatos_front_end.

GITHUB. **Aplicação do tipo console para recomendações de requisitos**. [Online; acessado em 20/12/2019], https://github.com/robsonklima/nathospyv2.

GITHUB. **Aplicação do tipo api ou webservice para recomendações de requisitos**. [Online; acessado em 20/12/2019], https://github.com/robsonklima/nhatos_api.

GOOGLE. **Cloud translation dynamically translate between languages**. [Online; acessado em 11/03/2019], https://cloud.google.com/translate.

GOOGLE. **Cloud natural language derive insights from unstructured text using google machine learning**. [Online; acessado em 12/03/2019], https://cloud.google.com/naturallanguage.

HAMZA, M. et al. Recommending features and feature relationships from requirements documents for software product lines. **Proceedings - 4th International Workshop on Realizing Artificial Intelligence Synergies in Software Engineering, RAISE 2015**, [S.l.], p. 25–31., 2015.

HASTIE, S.; WOJEWODA, S. **Standish group 2015 chaos report**. Accessed: 2017-01-30, `https://www.infoq.com/articles/standish-chaos-2015`.

HASTIE, S.; WOJEWODA, S. **Standish group 2015 chaos report**. Disponível em: https://https://www.infoq.com/articles/standish-chaos-2015. Acesso em: 30 de Janeiro de 2017, `https://www.infoq.com/articles/standish-chaos-2015`.

HERRERA, C. et al. Using data mining and recommender systems to facilitate large-scale, open, and inclusive requirements elicitation processes. **Proceedings of the 16th IEEE International Requirements Engineering Conference, RE'08**, [S.l.], p. 165–168., 2008.

HERRERA, C. et al. **A recommender system for requirements elicitation in large-scale software projects**. New York, NY, USA: ACM, 2009. 1419–1426 p. (SAC '09).

HERRERA, C. et al. Enhancing stakeholder profiles to improve recommendations in online requirements elicitation. **Proceedings of the IEEE International Conference on Requirements Engineering**, [S.l.], p. 37–46, 2009.

KASPERBAUER, M. et al. Chronos mobi: uma aplicação móvel multiplataforma para o gerenciamento de projetos. **Revista Brasileira de Computação Aplicada**, [S.l.], v. 5, 2013.

KESHAV, S. How to read a paper. **ACM SIGCOMM Computer Communication Review**, [S.l.], v. 37, n. 3, p. 83, 2007.

Kim, M.; Dey, S.; Lee, S. Ontology-driven security requirements recommendation for apt attack. , [S.l.], p. 150–156, 2019.

KUSNER, M. J. et al. From word embeddings to document distances. **Proceedings of the 32Nd International Conference on International Conference on Machine Learning**, [S.l.], v. 37, p. 957–966, 2015.

LIU, L. et al. A statistical analysis approach to predict user's changing requirements for software service evolution. **Journal of Systems and Software**, [S.l.], v. 124, p. 187–194, 2017.

LIU, L. et al. Requirements cybernetics: elicitation based on user behavioral data. **Journal of Systems and Software**, [S.l.], v. 124, p. 187–194, 2017.

Liu, X. et al. Mining android app descriptions for permission requirements recommendation. , [S.l.], p. 147–158, 2018.

LUGER, G. F. **Inteligência artificial**. Pearson Brasil: Addison Wesley, 2016. v. 3.

LUISA, M.; MARIANGELA, F.; PIERLUIGI, N. Market research for requirements analysis using linguistic tools. **Requirements Engineering**, [S.l.], v. 9, n. 1, p. 40–56, 2004.

LUZ, S. et al. Chronos: a tool for interactive scheduling and visualisation of task hierarchies. **Proceedings of the International Conference on Information Visualisation**, [S.l.], p. 241–246, 07 2009.

MENDELEY. **Mendeley homepage**. Disponível em: https://www.mendeley.com/homepage2/?switchedFrom=. Acesso em: 02 de Abril de 2018.

MIKOLOV, T. et al. Efficient estimation of word representations in vector space. , [S.l.], p. 1–12, 2013.

MOBILAB. **Pesquisa da percepção das equipes de gerenciamento de projetos**. Disponível em: http://bit.do/eSWL7. Acesso em: 01 de Fevereiro de 2019.

MULLER, F.; FORNO, G. Fatores críticos em projetos de desenvolvimento de software. **Rev. Adm. UFSM, Santa Maria**, [S.l.], 2017.

MULLER, F.; FORNO, G. Construção e validação de um instrumento de avaliação dos fatores críticos em projetos de desenvolvimento de software. **Rev. Adm. UFSM, Santa Maria**, [S.l.], p. 725–746, 2017.

NAKATANI, T. et al. Predicting requirements changes by focusing on the social relations. , Auckland, New Zealand, v. 154, p. 65–70, 2014.

NARENDRA, N. C. et al. Functional and architectural adaptation in pervasive computing environments. **Proceedings of the 3rd International Workshop on Middleware for Pervasive and Ad-hoc Computing**, New York, NY, USA, p. 1–7, 2005.

PADGHAM, L.; WINIKOFF, M. Developing intelligent agent systems: a practical guide. , [S.l.], 01 2004.

PARK, S. et al. Requirements attributes to predict requirements related defects. , [S.l.], p. 42, 2010.

PEJOVIC, V.; MUSOLESI, M. Anticipatory mobile computing: a survey of the state of the art and research challenges. **ACM Computing Surveys**, [S.l.], v. 47, 2013.

PETERSEN, K. et al. Systematic mapping studies in software engineering. **12Th International Conference on Evaluation and Assessment in Software Engineering**, [S.l.], v. 17, p. 10, 2008.

PETERSEN, K.; VAKKALANKA, S.; KUZNIARZ, L. Guidelines for conducting systematic mapping studies in software engineering: an update. **Information and Software Technology**, [S.l.], v. 64, p. 1–18, 2015.

PMI. **A guide to the project management body of knowledge (pmbok® guide)**. [S.l.]: Project Management Institute, 2017. v. 6ª.

PMI. **Requirements management**: a practice guide. [S.l.]: Project Management Institute, 2017.

PORTUGAL, R. et al. Gh4re: repository recommendation on github for requirements elicitation reuse. **CEUR Workshop Proceedings**, [S.l.], v. 1848, p. 113–120.., 2017.

REIS, J. et al. Digital transformation: a literature review and guidelines for future research. , [S.l.], p. 411–421, 03 2018.

Reshma, E. U.; Remya, P. C. A review of different approaches in natural language interfaces to databases. **2017 International Conference on Intelligent Sustainable Systems (ICISS)**, [S.l.], p. 801–804, 2017.

ROBILLARD, M. et al. **Recommendation systems in software engineering**. [S.l.: s.n.], 2010.

ROBILLARD, M. et al. **Recommendation systems in software engineering**. [S.l.: s.n.], 2014.

ROSA, J. H. da; BARBOSA, J. L.; RIBEIRO, G. D. Oracon: an adaptive model for context prediction. **Expert Systems with Applications**, [S.l.], v. 45, p. 56 – 70, 2016.

ROSA, J. H. et al. A multi-temporal context-aware system for competences management. **International Journal of Artificial Intelligence in Education**, [S.l.], v. 25, n. 4, p. 455–492, Dec 2015.

SAGRADO del et al. Stability prediction of the software requirements specification. **Software Quality Journal**, [S.l.], v. 26, p. 585–605, 06 2018.

SAP. **Tam - the sap way combining fmc and uml**. [Acessado em 06/04/2019].

Satyanarayanan, M. Pervasive computing: vision and challenges. **IEEE Personal Communications**, [S.l.], v. 8, n. 4, p. 10–17, 2001.

SHI, L. et al. Learning from evolution history to predict future requirement changes. **2013 21st IEEE International Requirements Engineering Conference, RE 2013 - Proceedings**, [S.l.], p. 135–144., 2013.

SILVER, G. A. The use of ontologies in discrete-event simulation. **Global Journal of Researches in Engineering**, [S.l.], v. 8, 2014.

SOMMERVILLE, I. **Software engineering**. [S.l.]: Pearson Education, 2016. 1056 p. v. 9ª.

SUTHERLAND J., C. J. **A scrum book - the spirit of the game**. [S.l.]: Pragmatic Bookshelf, 2019. v. 2.

THESAURUS. **Roget's 21st century thesaurus - prediction synonyms, prediction antonyms**. Disponível em: https://http://www.thesaurus.com/browse/prediction?s=ts. Acesso em: 19 de Junho de 2017.

TRENDS, G. **Natural language processing on google trends**. Disponível em: https://trends.google.com/trends/explore?date=today%205-yq=%2Fm%2F05flf. Acesso em: 24 de Dezembro de 2019.

TRENDS, G. **Ubiquitous computing on google trends**. Disponível em: https://trends.google.com/trends/explore?date=today%205-yq=%2Fm%2F07v58. Acesso em: 24 de Dezembro de 2019.

VANSYCKEL, S.; BECKER, C. A survey of proactive pervasive computing. **Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication**, New York, NY, USA, p. 421–430, 2014.

VENKATESH, V. et al. User acceptance of information technology: toward a unified view. **MIS Quarterly**, [S.l.], v. 27, n. 3, p. 425–478, 2003.

Villela, K. et al. Towards ubiquitous re: a perspective on requirements engineering in the era of digital transformation. **2018 IEEE 26th International Requirements Engineering Conference (RE)**, [S.l.], p. 205–216, 2018.

VILLELA, K. et al. Ubiquitous requirements engineering: a paradigm shift that affects everyone. **IEEE Software**, [S.l.], v. 36, n. 2, p. 8–12, 2019.

Wang, D.; Su, J.; Yu, H. Feature extraction and analysis of natural language processing for deep learning english language. **IEEE Access**, [S.l.], p. 1–1, 2020.

WANG, J. et al. Automated software security requirements recommendation based on ft-sr model. **Proceedings of the International Conference on Software Engineering and Knowledge Engineering, SEKE 2017**, [S.l.], p. 382–385., 2017.

WANG, W. et al. A regression model based approach for identifying security requirements in open source software development. **2017 IEEE 25th International Requirements Engineering Conference Workshops (REW)**, [S.l.], p. 443–446., 2017.

WEISER, M. The computer for the 21st century. **SIGMOBILE Mob. Comput. Commun. Rev.**, New York, NY, USA, v. 3, n. 3, p. 3–11, July 1999.

WHITEHEAD, J. Collaboration in software engineering: a roadmap. , [S.l.], p. 214–225, 2007.

WIEDEMANN TIAGO; BARBOSA, J. L. V. R. S. J. A model for learning object recommendation using similarity of sessions. **Brazilian Journal of Computers in Education**, [S.l.], v. 22, p. 85, 2014.

WIEDMANN, T. et al. Recsim: a model for learning objects recommendation using similarity of sessions. **j-jucs**, [S.l.], v. 22, n. 8, p. 1175–1200, 2016.

WILLIAMS, G. et al. Mining twitter feeds for software user requirements. **Proceedings - 2017 IEEE 25th International Requirements Engineering Conference, RE 2017**, [S.l.], p. 1–10., 2017.

WOOLDRIDGE, M. **An introduction to multiagent systems**. 2nd. ed. [S.l.]: Wiley Publishing, 2009.

XIE, H. et al. A statistical analysis approach to predict user's changing requirements for software service evolution. **Journal of Systems and Software**, [S.l.], v. 132, p. 147 – 164, 2017.

ZHANG, X. L. et al. Non-functional requirement analysis and recommendation for software services. **Proceedings - IEEE 20th International Conference on Web Services, ICWS 2013**, [S.l.], p. 555–562, 2013.

Zhao, W.; Zhang, W. Collaborative filtering service recommendation algorithm based on trusted user and recommendation evaluation. , [S.l.], p. 2248–2255, 2018.

ZHIYING, T. et al. Crowdsourcing service requirement oriented requirement pattern elicitation method. , [S.l.], p. 1433–3058, 2019.

ANEXO A - QUESTIONÁRIO APLICADO AOS GESTORES DE PROJETOS

Este questionário faz parte de uma pesquisa realizada pelo Laboratório de Computação Móvel (Mobilab) e pelo Programa de Pós-Graduação em Computação Aplicada (PPGCA), ambos vinculados à Unisinos. Os objetivos desta pesquisa são:

- Coletar informações sobre os modelos utilizados para a gestão dos projetos;
- Entender a percepção dos gestores sobre as ferramentas utilizadas atualmente;
- Identificar como as ferramentas poderiam evoluir para atuar de forma proativa na gestão dos projetos.

A pesquisa destina-se a profissionais que tenham participação em projetos como gestores ou membros de equipes. As informações serão utilizadas para fins de pesquisa acadêmica, não sendo distribuídas ou compartilhadas para outros fins.
Mobilab - http://www.unisinos.br/mobilab/
PPGCA - http://www.unisinos.br/mestrado-e-doutorado/computacao-aplicada

1. Qual sua experiência como Gerente de Projetos ou como membro em equipes de projeto?

- Menos de 2 anos
- De 2 a 5 anos
- Mais de 5 anos

2. Qual o tamanho da empresa onde trabalha atualmente?

- Menos de 20 funcionários
- De 20 a 100 funcionários
- Mais de 100 funcionários

3. Qual a área de atuação da empresa?

4. Qual o tamanho médio das equipes que costuma trabalhar?

- Menos de 10 pessoas
- De 10 a 25 pessoas
- Mais de 25 pessoas

5. De quantas equipes ou projetos você costuma trabalhar simultaneamente?

- 1
- 2
- ...
- 9
- 10

6. Qual abordagem sua equipe utiliza atualmente?

- Ágil
- Tradicional (PMBoK, RUP, PRINCE2, etc)
- Bimodal (modelo híbrido, com características de projetos ágeis e tradicionais)
- Não utiliza nenhuma metodologia específica
- Outra: _____

7. Quais áreas você considera mais críticas para o sucesso do projeto?

- Integração
- Escopo
- Tempo
- Custos
- Qualidade
- Recursos Humanos
- Comunicações
- Riscos
- Aquisições
- Partes Interessadas

8. Pensando nos últimos 10 projetos no qual você participou, quantos deles tiveram problemas que impactaram em custo, tempo ou qualidade?

- 1
- 2

- ...
- 9
- 10

9. Nos projetos em que ocorreram problemas, quais foram as áreas em que os problemas foram identificados?

- Integração
- Escopo
- Tempo
- Custos
- Qualidade
- Recursos Humanos
- Comunicações
- Riscos
- Aquisições
- Partes Interessadas

10. Qual das informações abaixo você considera essencial em uma ferramenta para lhe auxiliar na prevenção de problemas em projetos?

- Um checklist com problemas comuns da área do projeto
- Um histórico dos problemas em projetos já realizados
- Uma lista com os riscos e problemas identificados em projetos similares
- Outra: _____

11. Quais ferramentas de gestão de projetos você tem utilizado atualmente?

12. Entre as ferramentas que você conhece para gestão de projetos, você acredita que elas auxiliem de uma forma proativa (sugerindo ou prevendo situações) a tomada de decisões no projeto?

- Sim
- Não

- Em partes

13. Você acredita que informações de outros projetos já concluídos poderiam auxiliar na gestão do projeto?

- Sim
- Não
- Em partes

14. Com qual frequência você utiliza dados históricos de projetos já concluídos para auxiliar na gestão?

- Sempre
- Às vezes
- Raramente
- Não utilizo

15. Quais tipos de sugestão você gostaria de receber de uma ferramenta proativa de gestão de projetos?

- Riscos para o projeto
- Alocação de recursos
- Possibilidade de atraso
- Possibilidade de exceder custos
- Sugestão de requisitos e novas funcionalidades
- Outra: _____

16. Deixe aqui suas sugestões em relação a ferramentas de gestão de projetos.

17. Caso tenha interesse em auxiliar ou receber informações de etapas futuras da pesquisa, informe seu e-mail.

ANEXO B - ARTIGOS PUBLICADOS

- FILIPPETTO, Alexsandro Souza; LIMA, Robson; BARBOSA, Jorge. Átropos: Towards a Risk Prediction Model for Software Project Management - International Journal of Agile Systems and Management (2020) - [Aceito para Publicação Aguardando DOI].

- FILIPPETTO, Alexsandro S.; LIMA, Robson K. ; BARBOSA, Jorge. L. V. ; FRANCISCO, Rosemary. ; KLEIN, Amarolinda Z. A Ubiquitous Project Management Model based on Context. In: International Journal of Business Information Systems, 2020. Disponível em: http://dx.doi.org/10.1504/ijbis.2020.10023985

- FILIPPETTO, Alexsandro Souza; LIMA, Robson; BARBOSA, Jorge. Um Modelo de Gerenciamento de Riscos para Projetos de Software com Equipes Distribuídas. iSys Revista Brasileira de Sistemas de Informação, [S.l.], june 2019. ISSN 1984-2902. Disponível em: <http://www.seer.unirio.br/index.php/isys/article/view/6612>. Acesso em: 27 de junho 2019

- FILIPPETTO, Alexsandro S.; LIMA, Robson K.; BARBOSA, Jorge. L. V. Lean Risk – Gestão de Riscos em Times Distribuídos. In: Universo PM, v. 02, p. 07-13, 2017

Artigos Submetidos ou em Revisão

- LIMA, Robson K.; FILIPPETTO, Alexsandro S.; BARBOSA, Jorge, Recommendation in Requirements Engineering: A Systematic Mapping Study. Automated Software Engineering (2020) - [Submetido Aguardando Processamento AUSE-D-19-00127]

- LIMA, Robson K.; FILIPPETTO, Alexsandro S.; BARBOSA, Jorge; Towards Ubiquitous RE: A Recommender Model Based on Contexts Histories. Information Processing and Management (2020) - [Submetido Aguardando Processamento IPM_2020_215]

ANEXO C - REGISTROS DE SOFTWARES

- LIMA, Robson K.; FILIPPETTO, Alexsandro S.; BARBOSA, Jorge L. V., Sistema Nathos. 2019. Patente: Programa de Computador. Número do registro: BR512019000899-5, data de registro: 14/05/2019, título: "Sistema Nathos", Instituição de registro: INPI - Instituto Nacional da Propriedade Industrial.
- FILIPPETTO, Alexsandro S.; LIMA, Robson. K. ; BARBOSA, Jorge. L. V. Sistema Átropos. 2019. Patente: Programa de Computador. Número do registro: BR512019000888-0, data de registro: 14/05/2019, título: Sistema Átropos, Instituição de registro: INPI - Instituto Nacional da Propriedade Industrial.