A tool to support the analysis of IoT software requirements using clustering techniques

José Ruan Rodrigues Sampaio[1], Bruno Carvalho da Silva[1], Rodrigo Pereira dos Santos[2], Davi Viana[1]
[1]Universidade Federal do Maranhão - UFMA, São Luís, Maranhão, Brasil
[2]Universidade Federal do Estado do Rio de Janeiro - UNIRIO, Rio de Janeiro, Rio de Janeiro, Brasil
{jrr.sampaio,bruno.carvalho1}@discente.ufma.br,davi.viana@ufma.br,rps@uniriotec.br

ABSTRACT

The increasing complexity of Internet of Things (IoT) applications has introduced new challenges in managing and organizing functional requirements. Traditional requirements engineering approaches often struggle to effectively address the heterogeneity and semantic variability inherent in IoT contexts. To address these issues, this work presents a tool designed to classify and cluster functional requirements in IoT applications. The solution is grounded in insights from an exploratory literature review that identifies characteristics to IoT systems. The tool automatically categorizes requirements and groups those with similar semantic features, improving the organization and analysis of system demands. This paper presents a full-stack application that supports the requirements engineering process in IoT environments. The tool enables requirements engineers to document, analyze, and manage requirements more efficiently, assisting them in decision-making processes in dynamic and heterogeneous IoT contexts.

Ferramenta: github.com/RuanSampaio-code/ ReqCluster4IoT-Tool
Apresentação (YouTube): https://www.youtube.com/watch?v=J3bvUnyNTPU
Apresentação (Zenodo): https://zenodo.org/records/15485952
Licença: MIT

KEYWORDS

Engenharia de Software, Engenharia de Requisitos, Algoritmos, IoT

1 INTRODUÇÃO

A Engenharia de Software (ES) é responsável pelo desenvolvimento sistemático, controlado e eficiente de sistemas computacionais. Ela abrange um conjunto de processos, técnicas e ferramentas voltados para a construção de software de qualidade, atendendo às necessidades dos usuários e aos objetivos de negócio das organizações [23]. Um de seus processos é a Engenharia de Requisitos (ER), que desempenha um papel de identificação, análise, especificação, validação e gerenciamento dos requisitos de um sistema. A Engenharia de Requisitos está intimamente ligada à aquisição e aplicação de conhecimento para a criação, o aperfeiçoamento e a implementação de sistemas de informação [27].

Os requisitos constituem a base sobre a qual o desenvolvimento de software é fundamentado. Requisitos mal compreendidos, incompletos ou mal gerenciados são uma das principais causas de falhas em projetos de software, resultando em retrabalho, atrasos e insatisfação do cliente [6, 20]. Nesse contexto, a ER surge como um processo crítico para garantir que o software atenda, de forma clara e precisa, às expectativas das partes interessadas. Destaca-se que documentos de requisitos são geralmente escritos em linguagem natural, conforme Franch et al. [8], facilitando sua produção, entretanto, possui os problemas inerentes desta linguagem, tais quais ambiguidades e linguagem específica do domínio.

Com o avanço das tecnologias e a crescente complexidade dos sistemas, novos desafios têm emergido no domínio da Engenharia de Requisitos. A variedade de domínios de aplicação, como Internet das Coisas (do inglês, *Internet of Things* ou IoT), sistemas ciberfísicos e aplicações distribuídas, impõe demandas cada vez mais específicas e dinâmicas [9]. Em especial, aplicações IoT envolvem requisitos altamente heterogêneos, múltiplos dispositivos, diferentes níveis de abstração e variações semânticas expressivas, dificultando a classificação, a organização e a manutenção dos requisitos ao longo do ciclo de vida do software [14].

A complexidade do contexto em aplicações de IoT ocasiona uma grande quantidade de requisitos nos projetos. Neste contexto, é relevante empregar técnicas com inteligência artificial para lidar com essa problemática, buscando agrupar os requisitos de forma que se possa trabalhar de forma mais otimizada. O uso do método de aprendizado de máquina do agrupamento permite reunir requisitos semelhantes ou relacionados, facilitando sua organização, a identificação de padrões e a priorização de funcionalidades. De acordo com Linden [16], técnicas de agrupamento têm se mostrado eficazes em diversas áreas, demonstrando sua capacidade de apoiar atividades que envolvem grande volume de dados e a necessidade de estruturação da informação.

Desta forma, este artigo apresenta uma ferramenta que implementa um método de agrupamento visando fornecer apoio à análise de requisitos em aplicações IoT. A ferramenta propõe facilitar a análise, organização e manutenção dos requisitos, promovendo melhor entendimento entre as partes interessadas. A aplicação combina processamento de linguagem natural, aprendizado de máquina e uma interface web, permitindo aos usuários classificar e agrupar requisitos de maneira automática. Além disso, esta ferramenta também considera as especificidades do domínio de IoT, destacando uma característica do domínio a cada grupo de requisitos, para dar mais subsídios para os analistas de requisitos.

O presente artigo está estruturado da seguinte forma: a Seção 2 apresenta os conceitos e fundamentação teórica nas aplicações de IoT. A Seção 3, detalha uma visão geral da ferramenta, evidenciando a arquitetura de software da ferramenta e fluxo de funcionamento. A Seção 4, detalha exemplos de uso da ferramenta. Já a Seção 5, expõe os trabalhos relacionados às aplicações IoT e a ferramenta. Por fim, a Seção 6 apresenta as conclusões.

2 FUNDAMENTAÇÃO TEÓRICA

Nesta seção, apresentam-se os principais conceitos que norteiam este trabalho, tais como aplicações web, definições de aplicações IoT e símbolos de classificação e agrupamento. Essas definições são fundamentais para compreender a base e o fluxo que estão por trás da aplicação em desenvolvimento.

2.1 Aplicações Web

Uma aplicação Web é um sistema baseado na Web (envolvendo servidor, rede, protocolo HTTP e navegador) em que as interações do usuário, como navegação e inserção de dados, influenciam diretamente o estado do negócio. Sendo assim, uma aplicação Web é, essencialmente, um sistema de software que gerencia estados empresariais e possui sua interface entregue por meio de recursos da Web [2].

Esse conceito é relevante para este trabalho, uma vez que a ferramenta proposta é desenvolvida como uma aplicação web. Esta abordagem implica na disponibilidade multiplataforma e facilita a interação dos usuários com os recursos de classificação e gerenciamento de requisitos.

2.2 IoT

IoT é um conceito atribuído a qualquer "coisa" que tenha conexão com a internet que promove a integração com outros dispositivos ou serviços [19] . Esse paradigma evolui continuamente e possui forte característica de diversidade de aplicações, dispositivos e contextos nos quais pode ser implementado. Existem diversas áreas de interesse para a aplicação da IoT, dentre elas: casas inteligentes, cidades inteligentes, saúde, manufatura inteligente, além de outras aplicações que abrangem desde o impacto da IoT em diferentes setores [15].

Dentro desses conceitos e dessa diversidade, Khanna and Kaur [14] sugere dividir o domínio da IoT em três áreas principais: dispositivos de IoT, Tecnologias e Aplicações. Dispositivos IoT são os elementos que capturam dados e interagem com o ambiente. Já as tecnologias dão suporte à comunicação e ao desenvolvimento das aplicações. Por fim, as aplicações são as soluções criadas a partir dessa combinação.

Essa diversidade de conceitos de IoT implicou em uma falta de padronização das características relativas à IoT. Neste cenário, alguns trabalhos presentes na literatura têm realizado revisões de definições de forma a elaborar *frameworks* de características. A seguir apresenta-se um compilado de três trabalhos que geraram 12 características do comínio de IoT [7, 18, 24]. A ferramenta descrita neste trabalho indica uma destas características para cada grupo de requisitos, de forma a dar mais subsídios para os analistas de requisitos. As características são:

- **Interação:** envolve a comunicação eficiente entre dispositivos, utilizando redes que permitam aplicações, não só a internet;
- **Coisas:** refere-se aos objetos físicos e virtuais, como sensores, atuadores e dispositivos inteligentes, que substituem computadores tradicionais e ampliam a conectividade;
- **Serviços:** abrange aplicações e soluções que geram valor estratégico, apoiando a inovação e a tomada de decisão;
- **Tecnologias Padronizadas:** destaca a importância da padronização de protocolos, arquiteturas e interfaces para garantir a interoperabilidade;
- **Dados:** trata da coleta, processamento e análise de informações, fundamentais para decisões automáticas e otimização de recursos;
- **Ubiquidade:** assegura a comunicação contínua e em tempo real entre dispositivos, em qualquer lugar e momento;
- **Singularidade:** cada objeto IoT deve ser identificado de forma única e automática;
- **Ação:** refere-se às respostas automáticas baseadas no processamento de dados, podendo ser realizadas pelo dispositivo ou por outros agentes;
- **Heterogeneidade:** destaca a necessidade de integração entre diferentes dispositivos, redes e plataformas;
- **Ecossistema:** representa a interação entre dispositivos, protocolos, plataformas, comunidades e objetivos em torno da IoT;
- **Inteligência:** a aplicação de IA e aprendizado de máquina para análise de dados e suporte à tomada de decisão inteligente;
- **Ambiente:** diz respeito ao espaço físico onde os objetos atuam, detectando, colaborando e respondendo a eventos e interações.

2.3 Algoritmo de classificação e agrupamento

2.3.1 Classificação. A classificação dos requisitos é uma etapa importante no processo de desenvolvimento de software, ao organizar os diferentes requisitos de forma estruturada. Na etapa de análise de requisitos, estes requisitos podem ser classificados em requisitos funcionais e não funcionais [10]. Os requisitos funcionais dizem respeito às funcionalidades que um sistema deve fazer para satisfazer seus objetivos. Por outro lado, os requisitos não funcionais estabelecem restrições ao funcionamento do sistema, com o objetivo de assegurar sua conformidade com as expectativas e necessidades dos usuários.

A classificação manual de requisitos representa uma atividade complexa, demorada e propensa à falhas, especialmente em projetos de software com um volume elevado de requisitos. Nesse contexto, evidencia-se a demanda por métodos automáticos de classificação, impulsionando investigações que explorem o uso de aprendizado de máquina [29]. A classificação em algoritmos de aprendizado de máquina integra a aprendizagem supervisionada, ou seja, o modelo é treinado com um conjunto de dados para depois prever rótulos em informações não vistas anteriormente.

Na classificação de requisitos, os algoritmos de aprendizado de máquina mais empregados na classificação binária incluem algoritmos clássicos como Máquina de Vetor de Suporte, Árvore de Decisão e *K-Nearest Neighbors* [13]. Contudo, estas abordagens possuem limitações, principalmente quando se tratam de dados com relações complexas. Sendo assim, abordagens novas como Transformers podem capturar relações de longo alcance de forma mais eficiente e precisa [26]. Nesta ferramenta, foi utilizado um modelo de linguagem pré-treinado que passou pelo processo de ajuste fino para a tarefa de classificação binária de requisitos, utilizando a base de dados Promise+ [22].

2.3.2 Agrupamento. Algoritmos de agrupamento de dados buscam reunir objetos em grupos, ou *clusters*, de objetos semelhantes ou relacionados [5]. Formalizando a ideia de grupos, Jain and Dubes [11] definem que grupos são compreendidos como um conjunto de pontos localizados em um espaço multidimensional, no qual os pontos dentro do mesmo grupo estão mais próximos entre si do que em relação a outros pontos de outros grupos.

Na literatura, existem três tipos de algoritmos de agrupamento. O primeiro é baseado em centróide, que necessita da quantidade de grupos à priori. O segundo tipo é baseado em densidade, que não precisa da quantidade de grupos à priori, porém em situações de alta dimensionalidade, como vetores textuais, este tipo possui uma difícil visualização. O último grupo é o baseado em hierarquia, que não necessita da quantidade de grupos e possui uma visualização simplificada, através de dendrogramas. Desta forma, nesta ferramenta, foi escolhida a abordagem baseada em hierarquia.

No agrupamento hierárquico, o algoritmo constrói uma estrutura hierárquica de agrupamento a partir de divisões sucessivas. Sendo cada partição associada a um nível de hierarquia. Dentro desta classe de algoritmos, destacam-se as seguintes abordagens para a determinação dos grupos:

- **Ward**: visa minimizar a variância em todos os clusters, uma abordagem semelhante ao K-means. É ideal para um ambiente com muitos clusters e com esses clusters desiguais.
- **Algoritmos de Link**: nessa categoria estão três tipos de algoritmos: *Complete Linkage*, que visa minimizar a distância entre dois grupos; *Average Linkage*, que minimiza a média das distâncias entre dois grupos; e, por fim, *Single Linkage*, que minimiza as distâncias entre os pontos mais próximos de dois clusters.

3 VISÃO GERAL

A ferramenta apresentada nesse artigo é uma aplicação web cujo principal objetivo é auxiliar analistas de requisitos e desenvolvedores na classificação e agrupamento automático de requisitos funcionais de aplicações de IoT, por meio da combinação de técnicas de Processamento de Linguagem Natural e Aprendizado de Máquina.

A solução proposta visa facilitar a análise e organização de requisitos, promovendo maior clareza no entendimento entre as partes interessadas. A ferramenta conta com uma arquitetura integrada composta por uma interface web e um back-end responsável pelo processamento e lógica de negócio.

3.1 Arquitetura da Ferramenta

A aplicação está dividida em módulos para facilitar a gerência de desenvolvimento. Conforme destaca Pereira et al. [21], a modularização facilita o desenvolvimento de software ao reduzir sua complexidade e promover organização. Para cumprir essa finalidade, a ferramenta foi estruturada em três módulos, como ilustrado na Figura 1.

3.1.1 Módulo de Banco de dados. Este módulo é responsável pela persistência dos dados da aplicação. Nesta implementação, foram utilizados dois bancos de dados: SQLite [1] e MongoDB [2]. O SQLite é utilizado para armazenar os dados de cadastro do usuário e informações iniciais do projeto. Já o MongoDB é empregado para guardar dados dos requisitos de cada projeto.

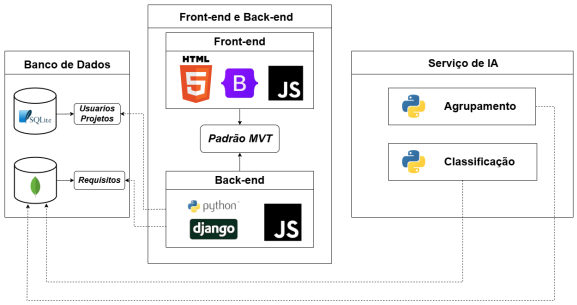

Figura 1: Arquitetura geral da ferramenta.

O SQLite, sendo um banco de dados relacional leve e de fácil integração, é ideal para armazenar dados estruturados e menos complexos, como os cadastros e configurações básicas. Por outro lado, o MongoDB, por ser um banco de dados NoSQL orientado a documentos, oferece maior flexibilidade na manipulação de dados com estruturas variadas, como os requisitos dos projetos, que podem mudar com frequência e exigir escalabilidade. Essa abordagem híbrida contribui para um melhor desempenho e organização da ferramenta.

Na relação Projeto e Requisitos, tem-se que cada projeto possui um conjunto de requisitos armazenado no MongoDB. A relação aqui é do tipo um para um (1:1). Como o MongoDB é utilizado para lidar com dados mais complexos e flexíveis, os requisitos são armazenados como documentos, permitindo o registro de informações estruturadas, como requisitos funcionais e não funcionais e grupos de requisitos.

3.1.2 Módulo Web (Front-end + Back-end). É o módulo responsável pela interface com que o usuário interage e as regras de negócio da aplicação. Durante esse desenvolvimento, a tecnologia principal escolhida foi framework Django[3], uma ferramenta baseada em Python, que segue o padrão de arquitetura *Model-View-Template* (MVT). Tal tecnologia consegue interagir tanto com a camada de front-end e back-end da aplicação.

3.1.3 Módulo de Serviço de IA. É o módulo responsável pela classificação e agrupamento dos requisitos. Este módulo é composto por três abordagens, uma para classificar os requisitos em funcionais e não funcionais, outra para agrupar requisitos funcionais e por fim, inferir característica IoT para o grupo.

O Algoritmo de Classificação de Requisitos, representado no Algoritmo 1, é utilizado para separar um conjunto de requisitos de software em dois grupos distintos: funcionais (F) e não funcionais (NF). Inicialmente, os conjuntos F e NF são criados vazios. Em seguida, o algoritmo carrega um modelo pré-treinado (M) e o utiliza para inicializar um classificador (C). Para cada requisito presente no conjunto de entrada R, o texto do requisito é processado pelo classificador, que atribui um rótulo indicando se o requisito é funcional ("F") ou não funcional ("NF"). Com base nesse rótulo, o requisito

[1] https://www.sqlite.org/
[2] https://www.mongodb.com/
[3] https://www.djangoproject.com/

é adicionado ao conjunto correspondente. Ao final do processo, o algoritmo retorna ambos os conjuntos resultantes, permitindo uma separação automatizada dos requisitos com base no modelo de classificação empregado.

Algorithm 1 Algoritmo de Classificação de Requisitos

Require: Conjunto de requisitos $R = \{r_1, r_2, ..., r_n\}$
Ensure: Dois conjuntos: F (funcionais) e NF (não-funcionais)

procedure ClassificarRequisitos(R)
  $F \leftarrow \emptyset$ ▷ Inicializa conjunto de funcionais
  $NF \leftarrow \emptyset$ ▷ Inicializa conjunto de não-funcionais
  Carregar modelo pré-treinado M
  Inicializar classificador C com modelo M
  for cada requisito $r_i \in R$ **do**
    $rotulo \leftarrow C.classificar(r_i.texto)$
    if $rotulo$ = "F" **then**
      $F \leftarrow F \cup \{r_i\}$
    else
      $NF \leftarrow NF \cup \{r_i\}$
    end if
  end for
  return (F, NF)
end procedure

Uma vez determinados os requisitos funcionais, eles são agrupados utilizando o algoritmo de agrupamento hierárquico, presente no Algoritmo 2. O processo inicia-se com a geração de embeddings para representar semanticamente os textos. Em seguida, o algoritmo testa diferentes combinações de métricas de dissimilaridade e métodos de ligação para construir dendrogramas, avaliando cada combinação com base na métrica de silhueta, a fim de identificar a mais adequada. Com a melhor configuração selecionada, realiza-se o agrupamento hierárquico dos documentos. Posteriormente, são extraídos tópicos de cada grupo por meio do modelo LDA (*Latent Dirichlet Allocation*). Esses tópicos são então associados a tópicos de referência (*seed topics*) utilizando uma medida de similaridade semântica. Ao final, o algoritmo retorna um dicionário que mapeia cada tópico identificado para a lista de documentos que lhe pertencem, fornecendo uma estrutura temática dos requisitos.

3.2 Funcionamento da ferramenta

Para a visualização do fluxo dos processos da aplicação, utiliza-se a notação de BPMN (*Business Process Modeling Notation*) que de acordo com White [28] é uma notação padronizada para representar visualmente fluxos de processos de negócios. Essa notação apresenta uma capacidade de abranger todo o processo integralmente, do início ao fim [1]. O processo é apresentado na Figura 2.

Observa-se que foram utilizadas três baias horizontais (*lanes*) para representar os principais participantes envolvidos no processo. Cada baia corresponde a uma função ou entidade distinta, permitindo uma visualização clara das responsabilidades e interações ao longo do fluxo. Na terceira baia destaca-se a utilização da biblioteca Mind Elixir [4] para criação e manipulaçaõ da modelagem visual do

[4]https://docs.mind-elixir.com/

Algorithm 2 Algoritmo de Agrupamento Hierárquico

procedure AgruparDocumentos(*requisitos*, *indices*)
  Entrada: Textos e seus identificadores
  Saída: Grupos com tópicos associados
  1. Gerar embeddings dos textos
  2. Para cada combinação métrica+método:
    a. Calcular matriz de dissimilaridade
    b. Construir dendrograma
    c. Avaliar candidatos com silhueta
  3. Selecionar melhor combinação
  4. Aplicar agrupamento hierárquico
  5. Extrair tópicos com LDA
  6. Associar a seed topics via similaridade
  7. Retornar $\{tpico : [documentos]\}$
end procedure

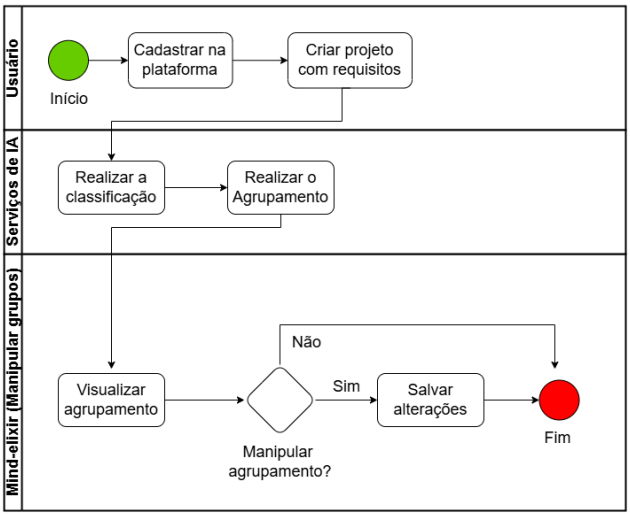

Figura 2: fluxo geral da aplicação

agrupamento de requisititos. A seguir, descreve-se cada uma das atividades.

- **Cadastrar na plataforma:** o processo da ferramenta tem início nesta atividade, onde o usuário cria seu usuário na plataforma preenchendo informações de *username*, e-mail e senha.
- **Criar projeto com requisitos:** nesta atividade, o usuário realiza a criação do projeto, informando nome, descrição e listando os requisitos do projeto.
- **Realizar a classificação:** o usuário classifica os requisitos em funcionais e não funcionais.
- **Realizar o agrupamento:** com os requisitos classificados, nesta atividade o usuário pode realizar o agrupamento de requisitos funcionais. Cada requisito será atribuído a um grupo e cada grupo será associado a uma característica de IoT presentes na Seção 2.2.
- **Visualizar agrupamento:** o usuário poderá visualizar agrupamento de requisitos do projeto por meio de um mapa

mental. Além disso, poderá manipular e alterar os requisitos entre os grupos e salvar as alterações realizadas. O mapa mental exibe os grupos definidos e os respectivos requisitos organizados em cada um, proporcionando uma visão clara e estruturada da distribuição dos requisitos.

4 EXEMPLO DE USO

Para melhor entendimento da ferramenta, segue um exemplo de uso. Na Figura 3, que representa a tela de criação de projetos, a qual o usuário cadastrará seus projetos e os requisitos. É permitido ao usuário inserir um nome ao projeto e uma descrição. Quanto aos requisitos, o usuário pode inserir um a um manualmente ou inserir todos os requisitos elicitados utilizando um arquivo .txt.

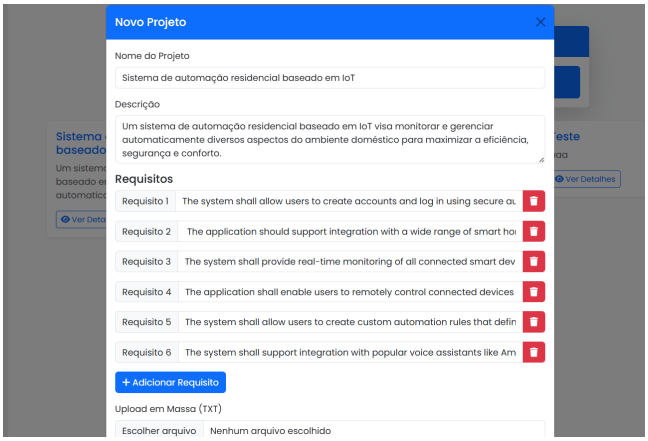

Figura 3: Tela de criação de projeto.

Com o projeto cadastrado, o usuário poderá visualizar em outra tela os detalhes do projeto. Nesta tela, será possível ver os requisitos e três botões nos quais será possível fazer a classificação, agrupamento e a visualização do agrupamento. Detalhes são apresentados na Figura 4.

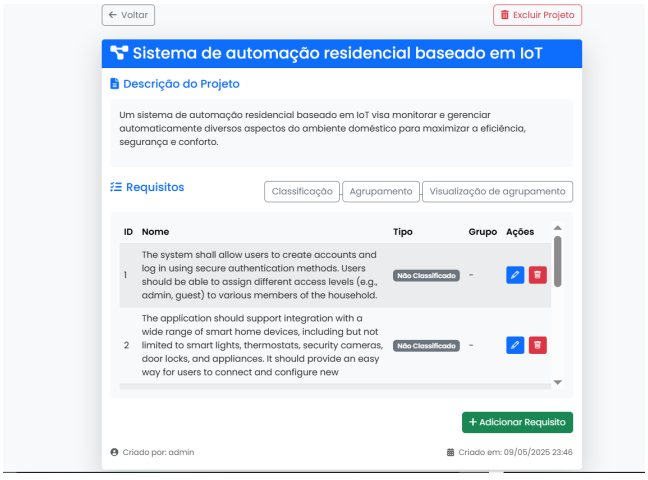

Figura 4: Tela de detalhes do projeto.

Após a classificação, a Figura 5 exibe os requisitos, com destaque para a coluna "Tipo", no qual cada requisito foi categorizado como funcional ou não funcional. É importante ressaltar que, se os requisitos não forem classificados, não será possível realizar a próxima etapa da aplicação: o agrupamento.

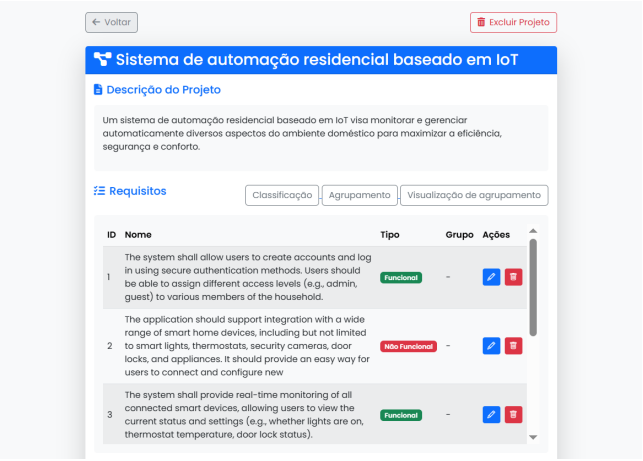

Figura 5: Tela com os requisitos do projeto classificados.

O passo seguinte é o agrupamento dos requisitos. Na Figura 6, os requisitos funcionais são organizados em grupos com características semelhantes, com base em sua similaridade semântica. Os grupos são identificados na coluna "Grupos", na qual cada requisito recebe um rótulo correspondente ao grupo ao qual pertence.

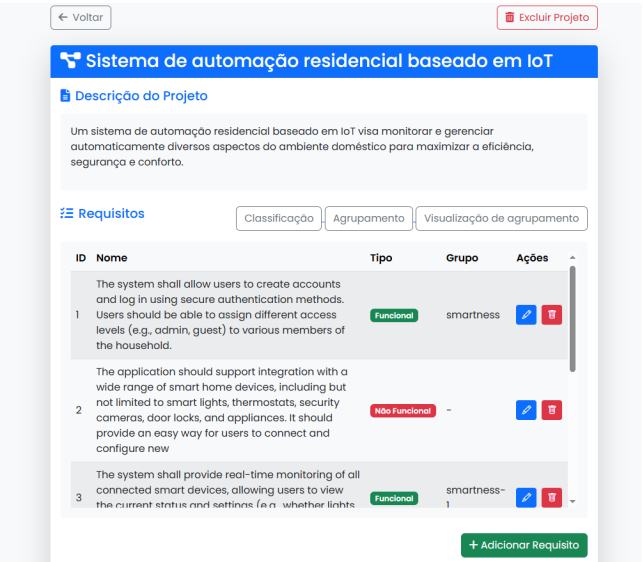

Figura 6: Tela dos requisitos do projeto agrupados.

E por fim, a Figura 7 apresenta a visualização do agrupamento e a possibilidade de manipular os requisitos entre os grupos formados. Essa interface permite ao usuário analisar graficamente os agrupamentos gerados, facilitando a identificação de requisitos

semelhantes. Além disso, é possível realocar requisitos entre os grupos, ajustando manualmente o resultado conforme o entendimento do analista.

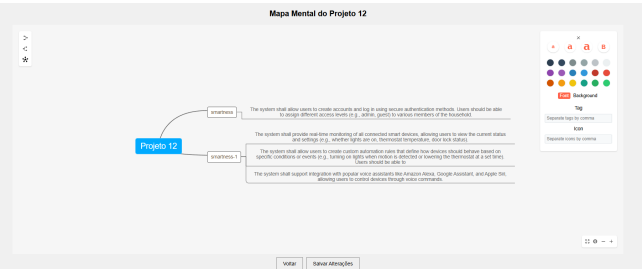

Figura 7: Tela de visualização do agrupamento.

5 TRABALHOS RELACIONADOS

Existem ferramentas e técnicas na literatura que fornecem apoio para Engenharia de Requisitos no contexto de IoT. Em Júnior et al. [12], é proposta a *RST-IoT: Uma ferramenta de apoio à especificação de requisitos de sistemas de software IoT*, uma ferramenta para automatizar o processo de especificação de requisitos de sistemas de software IoT. Em seus resultados, demonstraram que 87,5% dos participantes consideraram que a RST-IoT facilita o processo de especificação, enquanto 62,5% apontaram que a organização modular contribuiu para a clareza dos requisitos.

Outro trabalho a ser destacado é o de Souza et al. [25]. Neste trabalho, foi elaborado um *template* para elicitar requisitos de sistemas de software IoT, cujo objetivo é apoiar o engenheiro de software durante a fase de elicitação. Os autores destacam que realizaram experimentos durante o projeto para avaliar sua proposta e concluíram que a ferramenta satisfaz as necessidades de elaboração dos templates.

Seguindo uma linha semelhante, o trabalho de Da Silva et al. [3] demonstra a REIoT (*Requirements Engineering for software systems based on IoT*), uma tecnologia que se basea em duas tecnicas: o SCENARIoT e SCENARIoTCHECK. A primeira é uma técnica de especificação de cenários IoT e a segunda é uma técnica de verificação de cenários para sistemas de software baseados em IoT. Com essas duas técnicas em conjunto, a tecnologia tem como finalidade apoiar de forma estruturada a elaboração de documentos de requisitos, adaptando práticas convencionais à realidade de sistemas de software baseados em IoT.

Quando se considera a estratégia de agrupamento de requisitos, em Misra et al. [17], é proposta uma técnica de agrupamento baseada em LDA, voltada para requisitos sem erros linguísticos ou ambiguidade. O método é dividido em três etapas: (1) pré-processamento do corpus com técnicas de PLN (POS-tagging, *chunking*, *lemmatization* e *tokenization*), removendo stopwords; (2) geração do modelo LDA e cálculo da similaridade semântica via cosseno entre vetores; (3) identificação dos tópicos e formação de clusters, representados como um grafo não direcionado ponderado. A técnica foi avaliada com 41 requisitos e 10 grupos definidos conforme o documento original. Compararam-se os métodos *Theme-based Clustering*, *Hierarchical Average Linkage e Partitional Clustering*, com métricas de pureza, pureza inversa e F1.

Outro trabalho que envolve agrupamento é o de Eyal Salman et al. [4], no qual os autores propõem um método de agrupamento de requisitos funcionais dividido em quatro etapas: (1) seleção manual dos requisitos funcionais a partir do documento de especificação; (2) tokenização, remoção de *stopwords* e *stemming*; (3) cálculo da similaridade semântica com base na quantidade de tokens compartilhados entre os requisitos; e (4) aplicação de agrupamento hierárquico aglomerativo. A avaliação foi feita com quatro documentos de diferentes domínios, verificando-se a correção semântica dos clusters e sua correspondência com o número real de grupos. Os resultados evidenciaram alta precisão e recall, com número de clusters próximo ao ideal.

Destaca-se que ambos os trabalhos não classificam requisitos de forma automática para a seleção dos requisitos funcionais e nem dão suporte para a análise de requisitos de software para IoT. Embora essas iniciativas contribuam significativamente para etapas como elicitação, especificação e verificação de requisitos, observa-se que há poucas abordagens voltadas à análise, organização e agrupamento sistemático dos requisitos. Essas etapas são importantes para garantir a consistência e o sucesso dos sistemas IoT ao longo de seu ciclo de vida. Em especial, identifica-se uma carência de ferramentas que implementem métodos especializados para a classificação e o agrupamento de requisitos nesse contexto.

6 CONCLUSÃO

Sabe-se que existe uma grande complexidade das aplicações IoT, por apresentar uma alta heterogeneidade em seus dispositivos e serviços. Essa diversidade tecnológica impõe desafios significativos no processo de desenvolvimento, principalmente no que diz respeito aos tipos de requisitos, definição de grupos de requisitos, organização e gerenciamento dos requisitos do sistema.

Este artigo apresentou uma ferramenta que visa auxiliar analistas, engenheiros de requisitos e desenvolvedores no gerenciamento, agrupamento e classificação de requisitos no contexto de aplicações IoT. A ferramenta propõe uma abordagem sistemática estruturada em uma ferramenta web, implementada por meio de uma aplicação que utiliza técnicas voltadas à automatização dessas atividades.

Como trabalhos futuros, destacam-se incluir uma abordagem de compartilhamento de projeto entre usuários, permitindo assim trabalho colaborativo. Outra evolução seria incluir os requisitos não funcionais na análise. Requisitos não funcionais possuem aspecto transversal aos sistemas de software, ou seja, podem atingir mais de uma funcionalidade. Por isto, nesta ferramenta, optou-se pelo agrupamento dos requisitos funcionais.

AGRADECIMENTOS

Agradecemos a Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) - Código de Financiamento 001. Além de agradecermos ao PROCAD-Amazônia da CAPES (88887.200532/2018-00). Agradecemos a FAPEMA (BEPP-03906/23). Agradece-se ainda ao CNPq (Proc. 316510/2023-8) e à UNIRIO (PPQ 2023). Por fim, agradecemos ao INCT de Internet do Futuro para Cidades Inteligentes (CNPq proc. 465446/2014-0, FAPESP proc. 14/50937-1, and FAPESP proc. 15/24485-9).

REFERÊNCIAS

[1] Gart Capote. 2011. Guia para formação de analistas de processos. *Business Process Management. Rio de Janeiro: Bookess* (2011).

[2] Jim Conallen. 1999. Modeling web application architectures with UML. *Commun. ACM* 42, 10 (1999), 63–70.

[3] Danyllo Valente Da Silva, Bruno Pedraça De Souza, TS Gonçalves, and GH Travassos. 2020. Uma tecnologia para apoiar a engenharia de requisitos de sistemas de software iot. In *23rd Iberoamerican Conference on Software Engineering*.

[4] Hamzeh Eyal Salman, Mustafa Hammad, Abdelhak-Djamel Seriai, and Ahed Al-Sbou. 2018. Semantic Clustering of Functional Requirements Using Agglomerative Hierarchical Clustering. *Information* 9, 9 (2018). https://doi.org/10.3390/info9090222

[5] Katti Faceli, André Carlos Ponce de Leon Ferreira de Carvalho, and Marcílio Carlos Pereira de Souto. 2005. Algoritmos de agrupamento de dados. (2005).

[6] D Méndez Fernández, Stefan Wagner, Marcos Kalinowski, Michael Felderer, Priscilla Mafra, Antonio Vetrò, Tayana Conte, M-T Christiansson, Des Greer, Casper Lassenius, et al. 2017. Naming the pain in requirements engineering: Contemporary problems, causes, and effects in practice. *Empirical software engineering* 22 (2017), 2298–2338.

[7] Farshad Firouzi, Bahar Farahani, Markus Weinberger, Gabriel DePace, and Fereidoon Shams Aliee. 2020. *IoT Fundamentals: Definitions, Architectures, Challenges, and Promises*. Springer International Publishing, Cham, 3–50. https://doi.org/10.1007/978-3-030-30367-9_1

[8] Xavier Franch, Cristina Palomares, Carme Quer, Panagiota Chatzipetrou, and Tony Gorschek. 2023. The state-of-practice in requirements specification: an extended interview study at 12 companies. *Requirements Engineering* 28, 3 (01 Sep 2023), 377–409. https://doi.org/10.1007/s00766-023-00399-7

[9] Christopher Greer, Martin Burns, David Wollman, and Edward Griffor. 2019. Cyber-physical systems and internet of things. (2019).

[10] ISO/IEC/IEEE 29148. 2018. ISO/IEC/IEEE International Standard - Systems and software engineering – Life cycle processes – Requirements engineering. *ISO/IEC/IEEE 29148:2018(E)* (2018), 1–104. https://doi.org/10.1109/IEEESTD.2018.8559686

[11] Anil K Jain and Richard C Dubes. 1988. *Algorithms for clustering data*. Prentice-Hall, Inc.

[12] Albuquerque Júnior, Ednilson da Silva, et al. 2020. RST-IoT-uma ferramenta de apoio a especificação de requisitos de sistemas de software IoT. (2020).

[13] Kamaljit Kaur and Parminder Kaur. 2024. The application of AI techniques in requirements classification: a systematic mapping. *Artificial Intelligence Review* 57, 3 (2024), 23, 24, 25, 26, and 40.

[14] Abhishek Khanna and Sanmeet Kaur. 2020. Internet of things (IoT), applications and challenges: a comprehensive review. *Wireless Personal Communications* 114 (2020), 1687–1762.

[15] JR Emiliano Leite, Paulo S Martins, and E URSINI. 2017. A Internet das coisas (IoT): tecnologias e aplicações. In *Brazilian Technology Symposium*, Vol. 1. 1–6.

[16] Ricardo Linden. 2009. Técnicas de agrupamento. *Revista de Sistemas de Informação da FSMA* 4, 4 (2009), 18–36.

[17] Janardan Misra, Shubhashis Sengupta, and Sanjay Podder. 2016. Topic Cohesion Preserving Requirements Clustering. In *Proceedings of the 5th International Workshop on Realizing Artificial Intelligence Synergies in Software Engineering* (Austin, Texas) *(RAISE '16)*. Association for Computing Machinery, New York, NY, USA, 22–28. https://doi.org/10.1145/2896995.2896998

[18] Rebeca Motta, Káthia Oliveira, and Guilherme Travassos. 2019. On Challenges in Engineering IoT Software Systems. *Journal of Software Engineering Research and Development* 7 (09 2019), 5. https://doi.org/10.5753/jserd.2019.15

[19] Subhas Chandra Mukhopadhyay and Nagender K Suryadevara. 2014. *Internet of things: Challenges and opportunities*. Springer.

[20] Ayan Nigam, Neeraj Arya, Bhawna Nigam, and Deepika Jain. 2012. Tool for automatic discovery of ambiguity in requirements. *International Journal of Computer Science Issues (IJCSI)* 9, 5 (2012), 350.

[21] Ítalo Magno Pereira et al. 2011. Benefícios da modularização na obtenção de software de qualidade. (2011).

[22] Bruno Silva, Rodrigo Nascimento, Luis Rivero, Geraldo Braz, Rodrigo Santos, Luiz Martins, and Davi Viana. 2024. Promise+: expandindo a base de dados de requisitos de software Promise_exp. In *Anais do XXXVIII Simpósio Brasileiro de Engenharia de Software* (Curitiba/PR). SBC, Porto Alegre, RS, Brasil, 291–301. https://doi.org/10.5753/sbes.2024.3427

[23] Ian Sommerville. 2011. Engenharia de software, 9a. *São Palo, SP, Brasil* (2011), 5, 6.

[24] Krista Sorri, Navonil Mustafee, and Marko Seppanen. 2022. Revisiting IoT definitions: A framework towards comprehensive use. *Technological Forecasting and Social Change* 179 (2022), 121623. https://doi.org/10.1016/j.techfore.2022.121623

[25] Sabrina Rocha de Souza et al. 2020. TEl-IoT: template para elicitar requisitos de sistemas de software IoT. (2020).

[26] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention Is All You Need. arXiv:1706.03762 [cs.CL] https://arxiv.org/abs/1706.03762

[27] Carlos Eduardo Vazquez and Guilherme Siqueira Simões. 2016. Engenharia de Requisitos: software orientado ao negócio. (2016), 2.

[28] Stephen A White. 2004. Introduction to BPMN. *Ibm Cooperation* 2, 0 (2004), 0.

[29] Liping Zhao, Waad Alhoshan, Alessio Ferrari, Keletso J. Letsholo, Muideen A. Ajagbe, Erol-Valeriu Chioasca, and Riza T. Batista-Navarro. 2021. Natural Language Processing for Requirements Engineering: A Systematic Mapping Study. *ACM Comput. Surv.* 54, 3, Article 55 (apr 2021), 41 pages. https://doi.org/10.1145/3444689