Uma Tecnologia para Apoiar a Engenharia de Requisitos de Sistemas de Software IoT

Danyllo Valente da Silva, Bruno Pedraça de Souza, Taisa Guidini Gonçalves[0000-0002-7108-1763] e Guilherme Horta Travassos[0000-0002-4258-0424]

Universidade Federal do Rio de Janeiro – PESC/COPPE, Rio de Janeiro, Brasil
{dvsilva, bpsouza, taisa, ght}@cos.ufrj.br

Abstract. Sistemas de software contemporâneos (internet das coisas, indústria 4.0, cidades inteligentes, dentre outros) incorporam novas preocupações e características em sua construção e inerentes à rede, software, hardware, sensibilidade ao contexto, e interoperabilidade. Nesse sentido, a engenharia de requisitos (ER) desempenha um papel fundamental para garantir a correta construção desses sistemas na perspectiva das necessidades do negócio e dos usuários finais. Diversas tecnologias de software para apoiar a ER estão disponíveis na literatura, porém nem todas cobrem as atividades e especificidades de sistemas de software contemporâneos, notadamente daqueles baseados em *IoT*. Nesse sentido, o presente trabalho apresenta a tecnologia RE_{IoT} (*Requirements Engineering for software systems based on IoT*), que tem como objetivo prover apoio metodológico, técnico e ferramental para a ER de sistemas de software *IoT*. A RE_{IoT} apoia a construção do documento de requisitos utilizando uma técnica de descrição de cenários *IoT*, um *checklist* para a verificação dos cenários, um processo construtivo e *templates* ajustados para sistemas de software *IoT*. As técnicas de descrição e verificação de cenários, bem como os *templates*, apresentam indicação de viabilidade de sua utilização em projetos de software *IoT*.

Keywords: Engenharia de Requisitos, Especificação de requisitos, Cenários, Verificação, Internet das Coisas, Engenharia de Software.

1 Introdução

Sistemas de software contemporâneos, tais como aqueles inerentes ao paradigma da Internet das Coisas (*IoT*), Indústria 4.0, Cidades Inteligentes, dentre outros são complexos quando comparados a sistemas de software tradicionais. Essa complexidade provém da inclusão de novas preocupações e características relacionadas à rede, software, hardware, sensibilidade ao contexto, interface, interoperabilidade [1][2].

Particularmente, os sistemas de software baseados em *IoT* buscam promover o entrelaçamento de tecnologias e dispositivos que, por meio de uma rede, são capazes de capturar e trocar dados, tomar decisões e atuar, unindo os mundos real e virtual por meio de objetos e *tags*. Devido as suas características tecnológicas específicas, construir

sistemas *IoT* não é simples, exigindo tecnologias de software adaptadas e/ou inovadoras para criar e garantir a qualidade do produto construído [1].

A qualidade do desenvolvimento de sistemas de software contemporâneos depende de tecnologias de software que respondam as novas preocupações e características inerentes a esses sistemas. Como para qualquer outro produto construído com base em princípios de engenharia, uma atividade fundamental do desenvolvimento de sistemas de software *IoT* é a construção do documento de requisitos. Defeitos presentes no documento de requisitos podem ocasionar aumento do tempo, custo e esforço para o projeto; clientes e usuários finais insatisfeitos; baixa confiabilidade do sistema de software; alta quantidade de falhas; entre outros [3][4].

A engenharia de requisitos (ER) é responsável pelo ciclo de vida (concepção, elicitação, negociação, análise, especificação, verificação, validação e gestão) do documento de requisitos [4][5] e garante a construção adequada deste. A literatura técnica apresenta diversas tecnologias de software com o objetivo de apoiar a ER para sistemas de software, porém nem todas abrangem as diferentes etapas da ER e, principalmente, as especificidades de sistemas de software *IoT*.

Considerando a necessidade de tecnologias de software apropriadas para o desenvolvimento de sistemas de software *IoT*, e entendendo a importância que um documento de requisitos tem para a estabilidade, adequação, e qualidade de um projeto, este trabalho apresenta a versão inicial da RE_{IoT} (***R**equirements **E**ngineering for software systems based on **IoT***). O termo tecnologia refere-se ao apoio metodológico, técnico e ferramental oferecido pela RE_{IoT}, cujo objetivo é apoiar a construção de documentos de requisitos para sistemas de software *IoT*.

A tecnologia RE_{IoT} é composta de uma técnica de especificação de requisitos baseada na descrição de cenários *IoT* - $SCENARI_{OT}$ [6], de uma técnica de inspeção de cenários *IoT* - $SCENARI_{OT}CHECK$ [7], de um processo construtivo – versão inicial [8] e *templates* de artefatos (escopo do projeto, proposta de solução e descrição de casos de uso *IoT*) que apoiam as atividades do processo e compõem o documento de requisitos. As técnicas $SCENARI_{OT}$ e $SCENARI_{OT}CHECK$ foram avaliadas anteriormente através de estudos experimentais os quais indicaram sua viabilidade [6][9] e vêm sendo usadas nos projetos de sistemas de software *IoT* desenvolvidos pelo Grupo de Engenharia de Software Experimental (ESE) da COPPE/UFRJ.

Esse artigo tem como objetivo apresentar a tecnologia RE_{IoT} e uma prova de conceito envolvendo a comparação de seus *templates* com aqueles utilizados para apoiar a construção de documentos de requisitos de sistemas de software convencionais. Os resultados indicam que os *templates* da RE_{IoT} permitem capturar as informações necessárias para projetos *IoT* e que estão prontos para serem avaliados na construção de documentos de requisitos de sistemas de software *IoT*. Em complemento, é possível observar que a tecnologia RE_{IoT} provê cobertura para as principais fases da ER no que tange as especificidades de sistemas de software *IoT*.

Além desta introdução, este artigo apresenta outras cinco seções. A seção 2 descreve a base tecnológica da RE_{IoT}. A seção 3 apresenta a tecnologia RE_{IoT}. A seção 4 demonstra o estudo conduzido. A seção 5 apresenta alguns trabalhos relacionados encontrados na literatura. Por fim, a seção 6 apresenta os trabalhos futuros e conclui o artigo.

2 A Base Tecnológica da RE*IOT*

Esta seção apresenta a base tecnológica da *RE*$_{IoT}$ (**R**equirements **E**ngineering for software systems based on ***IoT***), construída para apoiar a ER de sistemas de software *IoT*. Esta tecnologia está inserida no contexto de uma abordagem de engenharia de sistemas, que contempla as grandes etapas do desenvolvimento de sistemas de software contemporâneos [1]. A base tecnológica é composta das técnicas *SCENARI*$_{OT}$ [6] e *SCENARI*$_{OT}$*CHECK* [7] que possuem indicação experimental de viabilidade e vêm sendo utilizadas em projetos de sistemas de software *IoT* em desenvolvimento pelo Grupo ESE no DELFOS - Observatório da Engenharia do Software Contemporâneo da COPPE/UFRJ.

2.1 SCENARI$_{OT}$

SCENARI$_{OT}$ é uma técnica de especificação de cenários *IoT* [6]. A técnica adapta e desenvolve tecnologias convencionais de cenários de software para apoiar a especificação de cenários em sistemas de software *IoT*, considerando características (adaptabilidade, conectividade, privacidade, inteligência, interoperabilidade, mobilidade, dentre outras) e comportamentos (identificação, sensoriamento e atuação) específicos do paradigma *IoT* [10].

A combinação das características com os três comportamentos permitiu a criação de nove arranjos de interação *IoT* (AIIs). Esses arranjos orientam os engenheiros de software a capturar informações essenciais sobre o sistema de software que será construído, como os tipos de dados a serem coletados, identificação das “coisas” no sistema *IoT*, entre outros.

Além disso, cada AII possui um catálogo contendo as informações de todas as características a serem capturadas na descrição do cenário, conforme apresentado na Fig 1. Os AIIs (isolados ou combinados) resultam em um artefato de descrição de cenários para sistemas de software *IoT*.

O artefato de especificação de cenários produzido pela *SCENARI*$_{OT}$ contém as seguintes informações: 1) Identificação dos principais elementos dos sistemas de software *IoT* que irão compor a solução, bem como a forma como estes elementos são organizados; 2) Descrição do domínio do problema, como saúde, cidades inteligentes, engenharia automotiva, entre outros; 3) Descrição do papel de cada ator dentro do cenário; e 4) Descrição das interações entre os atores e o sistema *IoT*.

2.2 SCENARI$_{OT}$CHECK

SCENARI$_{OT}$*CHECK* é uma técnica de verificação de cenários para sistemas de software baseados em *IoT* [7]. Esta técnica foi criada para apoiar a verificação dos artefatos produzidos pela técnica de especificação de cenários - *SCENARI*$_{OT}$ [6]. O objetivo da *SCENARI*$_{OT}$*CHECK* é aumentar a qualidade da especificação de cenários *IoT* realizada com a técnica de especificação *SCENARI*$_{OT}$. Ela consiste em um *checklist*, composto de duas partes, que auxilia os inspetores (com nenhum ou baixo conhecimento em sistemas de software *IoT*) a identificar defeitos na descrição dos cenários.

Catálogo – Exibição de dados	
Componente	Informação Relacionada
Produtores de dados	• Quem coleta os dados? (Sensores, leitores de *tags, wearables*)
	• Quais dados são coletados? (temperatura, umidade etc.)
	• Fonte dos dados (salas, xícara de café, geladeira, terra, etc.)
Exibidor de dados	• O que exibe dados? (Ex. Dispositivos executando aplicativos de usuário)
Consumidor de dados	• Quem acessa os dados? (Ex. Pessoa, sistema, etc.)
	• Semântica de dados (significado dos dados de acordo com quem os visualiza)

Fig 1. Exemplo do AII - coleta e exibição de dados e seu catálogo (baseado em [6])

A primeira parte do *checklist*, com base nos AIIs da técnica *SCENARI$_{OT}$*, visa verificar problemas gerais, capturando algumas características como domínio do problema, fluxo alternativo e de exceção, a interação e identificação entre atores, sistema e dispositivos. A segunda parte do *checklist* leva em consideração propriedades não funcionais de sistemas de software *IoT*. Essas propriedades são definidas de acordo com seis facetas: ambiente, coisas, comportamento, conectividade, interatividade e inteligência [1]. O *checklist* completo é apresentado em [9].

Após a especificação dos cenários *IoT* com a técnica *SCENARI$_{OT}$*, os inspetores aplicam a técnica *SCENARI$_{OT}$CHECK* para verificar a descrição dos cenários. Por fim, após a reunião de discriminação, o documento de especificação de cenários é corrigido a partir dos defeitos encontrados.

3 A Tecnologia *RE$_{IoT}$*

A tecnologia *RE$_{IoT}$* (***R**equirements **E**ngineering for software systems based on **IoT***) é composta pela base tecnológica descrita na seção 3, de um processo construtivo (seção 3.1), e *templates* (seção 3.2) para apoiar a construção do documento de requisitos (lista de requisitos, casos de uso *IoT*, cenários *IoT*, arranjos de interação *IoT* incluindo seus respectivos catálogos) em conformidade com os princípios da ER.

3.1 Processo de Construção do Documento de Requisitos da a RE$_{IoT}$

O processo de construção do documento de requisitos da *RE$_{IoT}$* é baseado nas principais fases da ER: concepção, elicitação, análise, especificação, verificação, validação e gerenciamento. Entretanto, a *RE$_{IoT}$* adapta e inclui novas atividades para atender as especificidades de sistemas de software *IoT*. Uma visão geral do processo construtivo é apresentada na Fig 2. Nos próximos parágrafos um breve resumo de cada fase é apresentado.

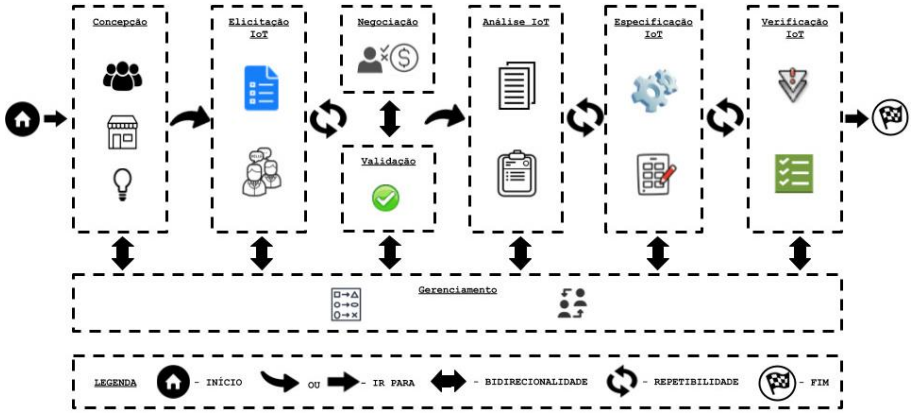

Fig 2. Visão geral da estrutura do processo de construção oferecido pela RE_{IoT}

Durante a fase de **concepção** deve-se buscar entender se o problema ou oportunidade pode ser resolvido utilizando tecnologias *IoT*. Um estudo de viabilidade também deve ser realizado para determinar se existem tecnologias disponíveis e equipe qualificada para a resolver o problema.

A fase de **elicitação *IoT*** busca refinar e transformar as necessidades de negócio e partes interessadas em requisitos. Durante essa fase é empregada a técnica $SCENARI_{OT}$ [6] para apoiar a identificação dos requisitos. Além disso, cenários *IoT* podem ser descritos para auxiliar no entendimento do comportamento do sistema. Por último, a lista de requisitos final deve organizar e classificar os requisitos em "requisitos *IoT"* e "requisitos não-*IoT"*.

As fases **validação** e **negociação** ocorrem em paralelo. Durante a fase de **validação** os requisitos são validados, atestando que um entendimento comum sobre o sistema foi alcançado. Paralelamente, a fase de **negociação** busca junto as partes interessadas a priorização dos requisitos e a resolução de conflitos levando em consideração dependências e precedências entre os requisitos. A análise de custo, esforço e riscos para a construção do sistema também deve ser realizada.

Durante a fase de **análise *IoT*** são gerados modelos do sistema. Os possíveis casos de uso *IoT* e os arranjos de interação *IoT* do sistema são listados. O diagrama de casos de uso *IoT* e o preenchimento do catálogo dos arranjos de interação identificados devem ser produzidos durante essa fase. A fase de **especificação *IoT*** engloba a descrição dos casos de uso *IoT* identificados. Essa fase é executada paralelamente a fase de **análise** e fluxos iterativos entre as fases podem ocorrer.

A fase de **verificação *IoT*** engloba verificar o documento de requisitos para assegurar sua qualidade. A técnica $SCENARI_{OT}CHECK$ [7] é aplicada nessa essa fase. Na fase de **gerenciamento** a tecnologia propõe o controle de versão dos artefatos e a rastreabilidade entre requisitos, cenários *IoT*, arranjos de interação *IoT* e casos de uso *IoT*. Além disso, a RE_{IoT} oferece o gerenciamento de mudanças para que mudanças nos requisitos possam ser refletidas nos artefatos.

3.2 Templates da RE_{IoT}

Os *templates* fornecidos pela RE_{IoT} apoiam as fases de **elicitação** (ELI) (artefato "Escopo do projeto"), **análise** (ANA) (artefato "Proposta de solução") e **especificação** (ESP) (artefato "Descrição de casos de uso *IoT*"). As fases de concepção (CON), **negociação** (NEG) e **validação** (VAL) são minimamente atendidas pelo *template* "Escopo do projeto".

Os *templates* "Proposta de solução" e "Descrição de casos de uso *IoT*" apoiam a fase de **gerenciamento** (GER) mantendo a rastreabilidade entre os requisitos e os modelos de análise. Além disso, as técnicas descritas na seção 3 apoiam as fases de **elicitação** (ELI), **especificação** (ESP) e **verificação** (VER).

A subseção que segue apresenta a descrição global dos *templates* (escopo do projeto, proposta de solução e descrição de casos de uso *IoT*) definidos pela tecnologia RE_{IoT}. A Fig. 3 apresenta um extrato de cada *template* proposto pela RE_{IoT}, os quais estão disponíveis em *http://bit.ly/359uTIN*.

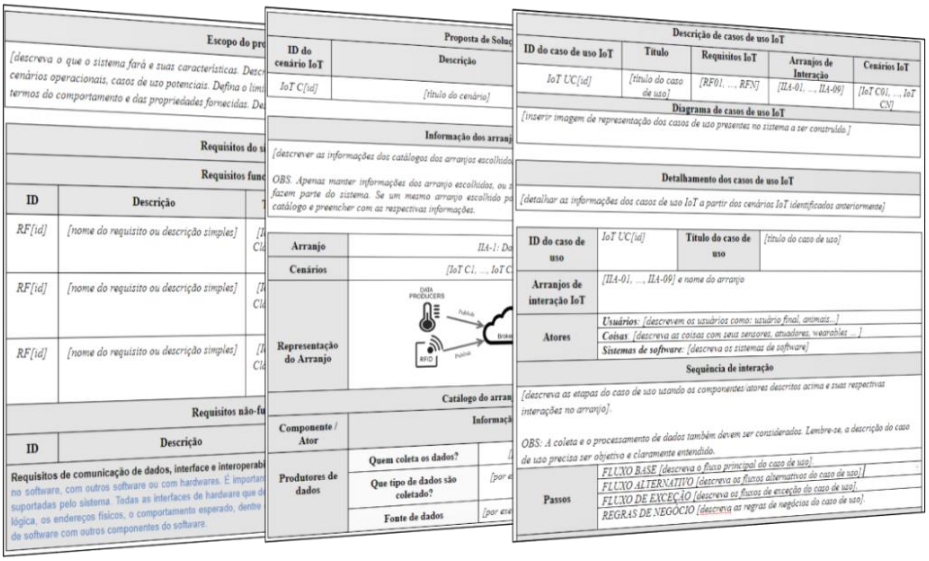

Fig. 3. Extrato dos *templates* propostos pela RE_{IoT}

3.2.1 Template "Escopo do projeto"

Este *template* apoia documentar as atividades iniciais do projeto, o problema a ser resolvido, os envolvidos no projeto, os perfis de usuários e as necessidades dos usuários que englobam as necessidades de negócio. Ele contempla a identificação e descrição de requisitos do sistema (funcionais, não funcionais, restrições, entre outros) e das regras de negócio. Além disso, a validação do documento de requisitos é viabilizada através de acordo explícito (assinatura ou cópia de e-mail). O *template* proposto é usado nas

fases iniciais do projeto, ou seja, as fases de concepção (CON), elicitação (ELI), negociação (NEG) e validação (VAL).

3.2.2 Template "Proposta de solução"

Este *template* apoia documentar a descrição da solução. Nele são identificados e descritos, a partir do uso da técnica *SCENARI$_{OT}$* [6], os cenários *IoT*, os componentes *IoT* e os arranjos de interação *IoT* (AIIs) do sistema. Além disso, fornece o detalhamento dos AIIs escolhidos para cada cenário *IoT* por meio do preenchimento dos catálogos correspondentes [6]. A rastreabilidade entre requisitos, cenários *IoT*, arranjos de interação *IoT* e seus respectivos catálogos é mantida. Este *template* deve ser utilizado nas fases de elicitação (ELI), análise (ANA), especificação (ESP) e gerenciamento (GER) visando identificar, descrever e refinar o comportamento do sistema mantendo a rastreabilidade dos requisitos.

3.2.3 Template "Descrição de casos de uso *IoT*"

Este *template* contempla a descrição dos casos de uso *IoT*. Os casos de uso são identificados e descritos fornecendo uma visão ampla do comportamento do sistema. O diagrama de casos de uso do projeto é inserido neste *template*. A rastreabilidade entre requisitos, cenários *IoT*, arranjos de interação *IoT* e casos de uso *IoT* é mantida. Este *template* deve ser usado nas fases de análise (ANA), especificação (ESP), verificação (VER) e gerenciamento (GER). A técnica *SCENARI$_{OT}$CHECK* [7] é aplicada durante a fase de verificação para identificar inconsistências na descrição de cenários *IoT* e seus componentes e na escolha dos AIIs.

4 Avaliando a Viabilidade dos *Templates*

A *RE$_{IoT}$* tem como objetivo apoiar engenheiros de software durante as atividades da ER. Duas das tecnologias utilizadas pela *RE$_{IoT}$* já foram avaliadas experimentalmente e têm sido utilizadas em projetos de sistemas de software *IoT*. Entretanto, a inclusão de novas facilidades para apoiar a ER com a *RE$_{IoT}$* demanda a observação inicial de viabilidade, antes da realização de estudos experimentais mais robustos. Dessa forma, esta seção apresenta uma prova de conceito da viabilidade de uso dos *templates* da *RE$_{IoT}$* em projetos de sistemas de software *IoT*.

4.1 Design

Na prova de conceito consideramos a estrutura de dois artefatos (lista de requisitos (LR) e descrição de casos de uso *IoT* (DUC1)) para sistemas convencionais que foram utilizados em projetos de sistemas software *IoT*; comparados com a estrutura dos *templates* (escopo do projeto (EP), proposta de solução (PS) e descrição de casos de uso *IoT* (DUC2)) da *RE$_{IoT}$* descritos na seção 4.

Os artefatos LR e DUC1 foram construídos a partir de *templates* convencionais em três projetos que representam problemas a serem solucionados utilizando o paradigma

IoT. São eles: i) **Projeto A** - sistema de software para apoiar a coleta de marcadores ambientais (Ex.: temperatura, umidade, particulados, nível de CO_2, e gases tóxicos); ii) **Projeto B** - sistema de software para apoiar o monitoramento de um ambiente computacional de alto desempenho (*datacentro*) envolvendo a coleta de informações de temperatura, umidade do ambiente, consumo de energia, e qualidade de fornecimento de energia; e iii) **Projeto C** - sistema de software para apoiar o monitoramento da temperatura, umidade, velocidade, e direção do vento em diferentes regiões de uma cidade.

Os artefatos LR e DUC1 foram produzidos por estudantes de graduação durante uma disciplina de Engenharia de Software na UFRJ. A disciplina contou com a participação de 21 alunos do quarto ano dos cursos de engenharia envolvendo computação. Estes alunos foram organizados em três times de desenvolvimento, com sete participantes cada um. Os times foram equilibrados e continham participantes com níveis equivalentes de conhecimento em *software* e *hardware*. Os alunos tiveram aulas expositivas sobre diferentes tópicos de engenharia de software e mentoria ao longo do projeto. Não houve intervenção dos mentores no conteúdo dos artefatos.

Os problemas tratados representavam demanda real e um *stakeholder* (totalmente externo ao curso e ao grupo de pesquisa) atuou junto com os desenvolvedores, incluindo a aceitação dos requisitos. Alguns dos tópicos ministrados na disciplina foram: engenharia de requisitos; cenários *IoT*; técnica de verificação para sistemas *IoT*, diagramas UML, entre outros. As técnicas *SCENARI*$_{OT}$ e S*CENARI*$_{OT}$*CHECK* foram apresentadas aos participantes, embora não tenham sido condicionadas ao uso. Os times tinham liberdade de organizar seus projetos. O documento de requisitos representava um dos marcos de projeto. Um produto mínimo viável representa um resultado concreto a ser entregue ao final do curso.

4.2 Execução

Após a entrega do documento de requisitos pelos times, os artefatos LR e DUC1 produzidos pelos três projetos foram analisados. As estruturas dos artefatos gerados extraída com base nas informações encontradas nos mesmos foram comparadas com a estrutura de informações dos *templates* EP, PS e DUC2. Para a comparação dos *templates* foi utilizado um *checklist* que será apresentado na próxima subseção.

4.3 Resultados e Discussão da Prova de Conceito

A **Tabela 1** apresenta o *checklist* usado para comparar a estrutura de cada um dos *templates* (sistemas convencionais e tecnologia *RE*$_{IoT}$) e o resultado da análise. Analisando a **Tabela 1**, é possível observar que:

- O objetivo do projeto/sistema e o domínio do problema não são abordados pelo *template* LR. Ter conhecimento sobre o domínio do problema é uma informação essencial para sistemas *IoT* [1][2];
- O *template* LR apresenta uma descrição parcial das partes interessadas. Ele não inclui uma descrição dos perfis dos diferentes usuários que é importante para o desenvolvimento do sistema e da interface do usuário;

Tabela 1. *Checklist* de mapeamento da estrutura de informações dos *templates*

Informação do projeto/sistema	***Templates* convencionais**		***Templates* da Tecnologia *RE$_{IoT}$***		
	LR	**DUC1**	**EP**	**PS**	**DUC2**
Nome do projeto/Responsável do projeto	T	T	T	T	T
Controle de versão	T	T	T	T	T
Acordo explícito	T		T		
Objetivo do projeto/sistema	N		T		
Domínio do problema	N		T		
Escopo do projeto	T		T		
Glossário de termos	T		T		
Descrição das partes interessadas	P		T		
Descrição das necessidades do negócio/das partes interessadas	N		T		
Requisitos funcionais	P		T		
Requisitos não-funcionais	T		T		
Negociação dos requisitos	T		T		
Regras de negócio	N	T	T		T
Análise do projeto	N		P		
Cenários *IoT*	N	P		T	T
Descrição dos componentes *IoT*		N		T	T
Arranjos *IoT*		P		T	T
Diagrama de casos de uso *IoT*		T			T
Descrição de casos de uso *IoT*		T			T
Rastreabilidade		P		T	T
Referências (outros documentos)	T		N		

***** P - Coleta Parcialmente; T - Coleta Totalmente; N - Não coleta a informação; Cinza - Não se aplica

- A descrição das necessidades do negócio/das partes interessadas não é abordada pelo *template* LR. A identificação das necessidades do negócio/das partes interessadas representa a etapa inicial do projeto. Nesta etapa buscamos entender a real necessidade do cliente, que futuramente será transformada em requisitos do sistema;
- Diferentemente do *template* LR que não identifica os requisitos *IoT*, a tecnologia *RE$_{IoT}$* identifica desde o início os requisitos que direcionarão a solução *IoT*;
- O *template* DUC1 trata as informações dos cenários *IoT* e dos arranjos *IoT* parcialmente, e não trata da descrição dos componentes *IoT*. Diferentemente, a tecnologia *RE$_{IoT}$* trata estas informações totalmente nos *templates* "proposta de solução" (PS) e "descrição de casos de uso *IoT*" (DUC2);
- A rastreabilidade é tratada parcialmente pelo *template* DUC1 e tratada totalmente pela tecnologia *RE$_{IoT}$* em dois *templates*;
- O *template* LR apresenta um campo para as referências (outros documentos), o que não é abordado pela tecnologia *RE$_{IoT}$*.

Os diferentes pontos de convergência e divergência entre os *templates* (LR e DUC1) de sistemas convencionais e os *templates* da RE_{IoT} (EP, PS e DUC2), oferecem indícios de que a tecnologia RE_{IoT} é mais robusta, uma vez que trata as informações relacionadas à *IoT* desde o início do projeto.

De acordo com os resultados, foi observado que a tecnologia RE_{IoT} apresenta um potencial significativo quanto à sua utilização em sistemas de software *IoT*. Isto se deve ao fato de que ela incorpora em seus *templates* informações especificas de sistemas de software *IoT* diferentemente dos *templates* convencionais. A falta dessas informações pode ocasionar problemas na construção do documento de requisitos. No entanto, para assegurar a validade da tecnologia é necessária uma avaliação experimental para verificar, dentre outras coisas, se o encadeamento do processo construtivo bem como os *templates* são úteis, completos, corretos e intuitivos.

4.4 Ameaças à Validade e Limitações

Uma limitação é a própria prova de conceito, mesmo sendo parte da tecnologia avaliada e validada por estudos experimentais. Entretanto, os resultados apresentam indícios de que os *templates* podem capturar informação pertinentes quando comparada a uma tecnologia convencional, no que diz respeito ao uso de artefatos de projetos.

Uma **ameaça externa** diz respeito aos participantes (estudantes de graduação) que foram convidados para participar da prova de conceito. Em relação à **validade de constructo** não houve controle da criação artefatos produzidos durante a prova de conceito. Entretanto, os projetos eram equivalentes quanto ao tamanho, complexidade, e tecnologias *IoT* a serem utilizados. Pode-se destacar que os grupos receberam treinamento e mentoria em ER. Por fim, a **ameaça de conclusão** está relacionada ao estudo em si e o tamanho reduzido e homogeneidade da amostra.

5 Trabalhos Relacionados

A construção de um documento de requisitos com qualidade requer engenharia. As fases e atividades da ER podem diferir de acordo com o domínio da aplicação, pessoas envolvidas, processos, cultura organizacional, entre outros. No entanto, podemos observar algumas fases e atividades recorrentes na engenharia de requisitos como: **concepção, elicitação, negociação, análise, especificação, verificação, validação** e **gerenciamento**.

Assim sendo, esta seção apresenta um conjunto de trabalhos relacionados encontrados na literatura técnica, que abordam tecnologias para as diferentes fases da ER, citadas anteriormente. A Tabela 2 apresenta a comparação desses trabalhos em ordem cronológica).

Com relação a fase de **concepção (CON)**, as técnicas *GSEM-IoT* [11], *Ignite* [12] e a técnica de Laplante *et al.* [13] realizam a análise das partes interessadas envolvidas no sistema. Além disso, as técnicas *Ignite* e *CORE* [14] fornecem mecanismos para realizar a análise de negócio. A análise de viabilidade é parcialmente tratada pela técnica *IoT Methodology* [12].

Tabela 2. Tecnologias para requisitos x fases da ER

Tecnologia / Fase*		CON	ELI	NEG	ANA	ESP	VER	VAL	GER
Aziz *et al.* [15]					X	X			X
Mahalank *et al.* [16]						X			
Prieto-Gonzalez *et al.* [17]				X					
Takeda *et al.* [18]					X	X			
Touzani *et al.* [19]					X				
IoT-RML [20]					X	X	X		
Yamakami [21]							X		
GSEM-IoT [11]		X	X						
Carvalho *et al.* [22]							X		
Curumsing *et al.* [23]			X		X		X		X
IoT System Development Methods	*Ignite* [12]	X	X	X	X			X	
	IoT Methodology [12]	X	X					X	
Laplante *et al.* [13]		X	X		X				
Lim *et al.* [24]			X	X				X	
IoTReq [25]			X		X	X			
CORE [14]		X	X		X	X			
SCENARI$_{OT}$ [6]			X			X			
SCENARI$_{OT}$CHECK [7]							X		
TrUStAPIS [26]			X			X			X
RET$_{IoT}$		X	X	X	X	X	X	X	X

* CON – Concepção; ELI – Elicitação; NEG – Negociação; ANA – Análise; ESP – Especificação; VER – Verificação; VAL – Validação; GER – Gerenciamento

No que diz respeito à fase de **elicitação (ELI)**, Lim *et al.* [24] realizou uma revisão sistemática da literatura com o objetivo de elencar as técnicas de elicitação para sistemas *IoT*. As técnicas *SCENARI$_{OT}$* [6], *Ignite* [12], *IoT Methodology* [12], Laplante *et al.* [13], *CORE* [14], Curumsing *et al.* [23], *IoTReq* [25] e *TrUStAPIS* [26] oferecem recursos para a coleta dos requisitos. *GSEM-IoT, IoTReq e IoT Methodology* propõem mecanismos para transformar as necessidades dos usuários em requisitos.

Para a fase de **negociação (NEG)**, Lim *et al.* [24] apresenta uma contribuição para a análise e resolução de conflitos. A técnica *Ignite* e a técnica de Prieto-Gonzalez *et al.* [17] abordam a análise de impacto. A análise de riscos é tratada apenas pela técnica *Ignite*.

Na fase de **análise (ANA)**, *Ignite*, a técnica de Laplante *et al.* [13], *CORE* [14], a técnica de Curumsing *et al.* [23], *IoTReq* [25], a técnica de Takeda *et al.* [18] e a técnica de Touzani *et al.* [19] usam diagramas da UML para elaborar os modelos de análise. A

técnica de Aziz *et al.* [15] e a técnica *IoT-RML* [20] tratam artefatos e modelos passíveis de reuso.

Para a fase de **especificação (ESP)**, *IoTReq* [25], *CORE* [14], *TrUStAPIS* [26], a técnica de Takeda *et al.* [18] e *IoT-RML* [20] usam modelos formais (próprios) para a especificação de requisitos. A técnica de Aziz *et al.* [15] e a técnica de Mahalank *et al.* [16] fornecem *templates* para a especificação dos requisitos. A técnica $SCENARI_{OT}$ [6] propõe a especificação de cenários *IoT* usando arranjos de interação.

Quanto à fase de **verificação (VER)**, $SCENAR_{IOT}CHECK$ [7] e a técnica de Carvalho *et al.* [22] propõem mecanismos para verificar requisitos. A técnica de Curumsing *et al.* [23], *IoT-RML* [20], a técnica de Carvalho *et al.* [22] e a técnica de Yamakami [21], oferecem mecanismos para verificar requisitos conflitantes.

Com relação a fase de **validação (VAL)**, *Ignite* [12], *IoT Methodology* [12] *e* Lim *et al.* [24], oferecem a técnica de prototipação para garantir que o produto atenda às necessidades dos usuários.

Por último, quanto a fase de **gerenciamento (GER)**, alguns trabalhos oferecem mecanismos para viabilizar a rastreabilidade como as técnicas de Curumsing *et al.* [23] e de Aziz *et al.* [15]. A técnica *TrUStAPIS* [26] provê a rastreabilidade e o gerenciamento de mudanças dos requisitos.

A tecnologia RE_{IoT} (apresentada nas seções 3 e 4) permeia as diferentes fases da ER oferecendo apoio metodológico e técnico por meio de um processo construtivo e seus *templates*.

6 Conclusão e Trabalhos Futuros

Neste trabalho apresentamos a tecnologia RE_{IoT} (***R**equirements **E**ngineering for software systems based on **IoT***), que tem como objetivo prover apoio metodológico através de um processo construtivo (incluindo *templates* dos artefatos), técnico (técnicas de especificação e verificação de cenários *IoT*) e ferramental para a ER de sistemas de software *IoT*.

Uma prova de conceito foi realizada para comparar os *templates* (artefatos) definidos pela tecnologia RE_{IoT} com *templates* (artefatos) de sistemas convencionais (não específico para sistemas *IoT*). A comparação dos *templates* oferece indícios de que os artefatos da tecnologia RE_{IoT} podem ser mais completos do ponto de vista das informações relacionadas à *IoT*.

Alguns dos trabalhos futuros reservados para a tecnologia RE_{IoT} são: i) projeto e execução de estudos experimentais para avaliar a tecnologia RE_{IoT} em projetos de sistemas de software *IoT* nos contextos acadêmico e industrial; ii) desenvolvimento de apoio ferramental que integre o processo construtivo e os *templates* da RE_{IoT}; e iii) a partir do documento de requisitos gerado pela RE_{IoT}, apoiar a criação de casos de teste para sistemas de software *IoT*.

Referências

1. Motta, R.C., Oliveira, K.M., Travassos, G.H.: On challenges in engineering IoT software systems. J. Softw. Eng. Res. Dev. 7, 5:1-5:20 (2019). https://doi.org/10.5753/jserd.2019.15.
2. Nguyen-Duc, A., Khalid, K., Shahid Bajwa, S., Lønnestad, T.: Minimum Viable Products for Internet of Things Applications: Common Pitfalls and Practices. Future Internet. 11, Paper 50 (2019). https://doi.org/10.3390/fi11020050.
3. Arif, S., Khan, Q., Gahyyur, S.A.K.: Requirements engineering processes, tools/technologies, & methodologies. Int. J. Rev. Comput. 2, 41–56 (2009).
4. Vegendla, A., Duc, A.N., Gao, S., Sindre, G.: A Systematic Mapping Study on Requirements Engineering in Software Ecosystems: J. Inf. Technol. Res. 11, 4:1-4:21 (2018). https://doi.org/10.4018/JITR.2018010104.
5. Pandey, D., Suman, U., Ramani, A.K.: An Effective Requirement Engineering Process Model for Software Development and Requirements Management. In: International Conference on Advances in Recent Technologies in Communication and Computing. pp. 287–291. IEEE, Kottayam, India (2010). https://doi.org/10.1109/ARTCom.2010.24.
6. Silva, V.M.: SCENARIoT Support for Scenario Specification of Internet of Things-Based Software Systems, (2019).
7. Souza, B.P., Motta, R.C., Travassos, G.H.: The first version of SCENARIotCHECK: A Checklist for IoT based Scenarios. In: XXXIII Brazilian Symposium on Software Engineering. pp. 219–223. ACM Press, Salvador, Brazil (2019). https://doi.org/10.1145/3350768.3350796.
8. Silva, D., Gonçalves, T.G., Rocha, A.R.C.: A Requirements Engineering Process for IoT Systems. In: XVIII Brazilian Symposium on Software Quality. pp. 204–209. ACM Press, Fortaleza, Brazil (2019). https://doi.org/10.1145/3364641.3364664.
9. Souza, B.P., Motta, R.C., Costa, D.O., Travassos, G.H.: An IoT-based Scenario Description Inspection Technique. In: XVIII Brazilian Symposium on Software Quality. pp. 20–29. ACM Press, Fortaleza, Brazil (2019). https://doi.org/10.1145/3364641.3364644.
10. Motta, R.C., Silva, V.M., Travassos, G.H.: Towards a more in-depth understanding of the IoT Paradigm and its challenges. J. Softw. Eng. Res. Dev. 7, 3:1-3:16 (2019). https://doi.org/10.5753/jserd.2019.14.
11. Zambonelli, F.: Key Abstractions for IoT-Oriented Software Engineering. IEEE Softw. 34, 38–45 (2017). https://doi.org/10.1109/MS.2017.3.
12. Giray, G., Tekinerdogan, B., Tüzün, E.: IoT System Development Methods. In: Hassan, Q., Khan, A.R., and Madani, S.A. (eds.) Internet of Things. pp. 141–159. CRC Press/Taylor & Francis, New York (2018).
13. Laplante, N.L., Laplante, P.A., Voas, J.M.: Stakeholder Identification and Use Case Representation for Internet-of-Things Applications in Healthcare. IEEE Syst. J. 12, 1589–1597 (2018). https://doi.org/10.1109/JSYST.2016.2558449.
14. Hamdi, M.S., Ghannem, A., Loucopoulos, P., Kavakli, E., Ammar, H.: Intelligent Parking Management by Means of Capability Oriented Requirements Engineering. In: Wotawa, F., Friedrich, G., Pill, I., Koitz-Hristov, R., and Ali, M. (eds.) Advances and Trends in Artificial Intelligence. From Theory to Practice. pp. 158–172. Springer International Publishing, Cham (2019). https://doi.org/10.1007/978-3-030-22999-3_15.

15. Aziz, M.W., Sheikh, A.A., Felemban, E.A.: Requirement Engineering Technique for Smart Spaces. In: International Conference on Internet of things and Cloud Computing. pp. 54:1–54:7. ACM Press, Cambridge United Kingdom (2016). https://doi.org/10.1145/2896387.2896439.
16. Mahalank, S.N., Malagund, K.B., Banakar, R.M.: Non Functional Requirement Analysis in IoT based smart traffic management system. In: International Conference on Computing Communication Control and Automation. pp. 1–6. IEEE, Pune, India (2016). https://doi.org/10.1109/ICCUBEA.2016.7860147.
17. Prieto-Gonzalez, L., Tamm, G., Stantchev, V.: Towards a Software Engineering Approach for Cloud and IoT Services in Healthcare. In: Gervasi, O., Murgante, B., Misra, S., Rocha, A.M.A.C., Torre, C.M., Taniar, D., Apduhan, B.O., Stankova, E., and Wang, S. (eds.) Computational Science and Its Applications. pp. 439–452. Springer International Publishing, Cham (2016). https://doi.org/10.1007/978-3-319-42089-9_31.
18. Takeda, A., Hatakeyama, Y.: Conversion Method for User Experience Design Information and Software Requirement Specification. In: Marcus, A. (ed.) Design, User Experience, and Usability: Design Thinking and Methods. pp. 356–364. Springer International Publishing, Cham (2016). https://doi.org/10.1007/978-3-319-40409-7_34.
19. Touzani, M., Ponsard, C.: Towards Modelling and Analysis of Spatial and Temporal Requirements. In: 24th International Requirements Engineering Conference. pp. 389–394. IEEE, Beijing, China (2016). https://doi.org/10.1109/RE.2016.60.
20. Costa, B., Pires, P.F., Delicato, F.C.: Specifying Functional Requirements and QoS Parameters for IoT Systems. In: 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress. pp. 407–414. IEEE, Orlando, FL (2017). https://doi.org/10.1109/DASC-PICom-DataCom-CyberSciTec.2017.83.
21. Yamakami, T.: Horizontal Requirement Engineering in Integration of Multiple IoT Use Cases of City Platform as a Service. In: IEEE International Conference on Computer and Information Technology (CIT). pp. 292–296. IEEE, Helsinki, Finland (2017). https://doi.org/10.1109/CIT.2017.54.
22. Carvalho, R.M., Andrade, R.M.C., Oliveira, K.M.: Towards a catalog of conflicts for HCI quality characteristics in UbiComp and IoT applications: Process and first results. In: 12th International Conference on Research Challenges in Information Science (RCIS). pp. 1–6. IEEE, Nantes (2018). https://doi.org/10.1109/RCIS.2018.8406651.
23. Curumsing, M.K., Fernando, N., Abdelrazek, M., Vasa, R., Mouzakis, K., Grundy, J.: Emotion-oriented requirements engineering: A case study in developing a smart home system for the elderly. J. Syst. Softw. 147, 215–229 (2019). https://doi.org/10.1016/j.jss.2018.06.077.
24. Lim, T.-Y., Chua, F.-F., Tajuddin, B.B.: Elicitation Techniques for Internet of Things Applications Requirements: A Systematic Review. In: VII International Conference on Network, Communication and Computing. pp. 182–188. ACM Press, Taipei City, Taiwan (2018). https://doi.org/10.1145/3301326.3301360.
25. Reggio, G.: A UML-based proposal for IoT system requirements specification. In: 10th International Workshop on Modelling in Software Engineering. pp. 9–16. ACM Press, Gothenburg, Sweden (2018). https://doi.org/10.1145/3193954.3193956.
26. Ferraris, D., Fernandez-Gago, C.: TrUStAPIS: a trust requirements elicitation method for IoT. Int. J. Inf. Secur. 19, 111–127 (2020). https://doi.org/10.1007/s10207-019-00438-x.