Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Técnicas para Realizar a Validação de Requisitos no Contexto de Internet das Coisas (IoT)

Felipe Brito Ayres

Monografia apresentada como requisito parcial
para conclusão do Curso de Engenharia da Computação

Orientadora
Prof.a Dr.a Edna Dias Canedo

Brasília
2021

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Técnicas para Realizar a Validação de Requisitos no Contexto de Internet das Coisas (IoT)

Felipe Brito Ayres

Monografia apresentada como requisito parcial
para conclusão do Curso de Engenharia da Computação

Prof.a Dr.a Edna Dias Canedo (Orientadora)
CIC/UnB

Prof.a Dr.a Roberta Barbosa Oliveira
CIC/UnB

Prof.a Dr.a Letícia Lopes Leite
CIC/UnB

Prof. Dr. João José Costa Gondim
Coordenador do Curso de Engenharia da Computação

Brasília, 21 de outubro de 2021

Dedicatória

Dedico este trabalho a minha família que sempre me apoiou, tanto na minha vida pessoal quanto acadêmica. Foram 5 anos de muito esforço, dedicação e a presença deles tornou esse tempo mais tranquilo para que pudesse me formar. Minha mãe por sempre estar presente, receptível e paciente e a meu pai que, por ser da área de engenharia, foi um norte e que pude me espelhar para seguir firme durante o curso.

Agradecimentos

Aos meus pais Elana e Florêncio, que estiveram comigo durante toda minha jornada e fizeram o possível para me incentivar e ajudar.

Á minha família, em especial, aos meus avós maternos, Deijayme e Maria José, e aos meus falecidos avós paternos, Florêncio e Amanda, pela garra que tiveram durante a vida e que possibilitaram o crescimento de uma família grande e próspera.

Agradeço ao Lucas, parceiro durante a pesquisa exploratória para este trabalho e que facilitou muito o desenvolvimento das atividades.

Á minha orientadora, Edna Dias Canedo, que sanou minhas dúvidas durante o processo do trabalho de graduação e esteve sempre disponível.

Por fim, agradeço aos meus amigos dos ensinos primário, médio e universitário, parceiros de minha jornada estudantil, que somaram experiências e vivências durante este período especial e de muitos aprendizados.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES), por meio do Acesso ao Portal de Periódicos.

Resumo

A internet das coisas vem ocupando um espaço cada vez maior em equipes de desenvolvimento de software e na sociedade. O nível de aplicação da IoT é abrangente. Tráfego de pessoas, casas inteligentes, ambientes otimizados e gestão de água/energia são alguns dos exemplos da sua aplicabilidade. Nesse universo de possibilidades, desenvolvedores e empresas de tecnologia devem estar preparados para adaptar seus projetos e absorver essa tecnologia em expansão. Como essa tecnologia é recente, falhas de projeto e retrabalho acontecem com frequência e dificultam o desenvolvimento de produtos de alta qualidade atualmente. O objetivo deste trabalho é identificar por meio de uma pesquisa exploratória, processos e técnicas de validação, voltadas ao contexto da internet das coisas. Além disso, investigamos a percepção dos desenvolvedores de software IoT sobre as suas atividades relacionadas a Engenharia de Requisitos em seus projetos. A percepção dos profissionais foi coletada através de entrevistas onde eles relataram as dificuldades e desafios que enfrentam durante suas atividades diárias. Foram encontrados 22 processos e 9 técnicas de validação para o contexto de IoT na literatura. A partir das entrevistas, foi possível perceber que stakeholders de projetos IoT não utilizam um processo formal de engenharia de requisitos. Normalmente, são utilizadas técnicas distintas como reuniões e diagramas, sempre com base na demanda e na necessidade do projeto. Apesar dos profissionais e stakeholders acharem importante a Engenharia de Requisitos, a adesão à processos e técnicas voltadas a IoT não é unânime devido a curva de aprendizado para adotar novos métodos e a falta de maleabilidade nos processos durante o desenvolvimento de software.

Palavras-chave: Requisitos, IoT, internet das coisas, técnicas de validação, validação, entrevistas, pesquisa exploratória.

Abstract

Internet of things occupies more and more space in development teams and in society in general. The applicability that IoT covers is huge. Smart houses, water/energy consuption, traffic management and smart buildings are some examples of what has been made in this context. In this vast universe of possibilities, developers and tech companies need to be prepared and adapt their projects to cover it.

With that in mind, failures/reworks in projects happens more easily and makes it more difficult to produce high standards products. The objective of this paper is to identify, based on a exploratory research, processes and validation techniques in IoT context. Furthermore, this work investigates the professionals‘ perception in their activities with requirenment engineering in IoT projects. Their reports were collected through interviews so they could explain the difficulties and problems that arise in their daily work.

In total, 22 processes and 9 validation techniques has been found in literature. From the interviews, it had been realized that stakeholders don´t use formal processes in their IoT projects. Usually, single techniques are used, like reunions and diagramans, to handle the requirements engineering.The stakeholders implement these methods based on the demand and size of the project. Although stakeholders thinks that RE is a important part inside a project, the use of processes and techniques for IoT development isn´t unanimous due to the learning curve to adopt such methods and the lack of flexibility in these processes during the development phase.

Keywords: requirements, IoT , internet of things, techniques of validation, interviews, exploratory research

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

5W1H O que, Como, Quem, Quando e Por quê.

ADL Linguagem de Descrição de Arquitetura.

AIIs Arranjos de Interação IoT.

AIOTI Aliança para a Inovação em Internet das Coisas.

BPMN Notação para Modelagem de Processos de Negócio.

CADP Construção e Análises de Processos Distribuídos.

CORE Engenharia de Requisitos Orientada a Capacidade.

CPPS Sistemas Cibernéticos de Produção Física.

CSS Sistema de Software Contemporâneo.

DE Expositores de Dados.

DSPL Linha de Produto de Software Dinâmico.

DSR Design Science Research.

EoI Entidade de Interesse.

GIS Sistema de Informação Geográfica.

GORE Engenharia de Requisitos Orientada por Objetivos.

GSEM-IoT Metodologia Geral de Engenharia de Software para a Internet das Coisas.

HID Dispositivo de Interface Humana.

IIA Arranjos de Interação em IoT.

IoT Internet das Coisas.

IoT-AD Desenvolvimento de Aplicativos IoT.

IoT-RML Linguagem de Modelagem de Requisitos em IoT.

KAOS Manter Todos os Objetivos Satisfeitos.

LNT Nova Tecnologia LOTOS.

M2M Máquina para Máquina.

MBD Design Baseado em Modelo.

RE Engenharia de Requisitos.

REM4DSPL Método para Engenharia de Requisitos para Linhas de Produtos de Software Dinâmicos.

RR Revisão Rápida.

SAL Linguagem de Arquitetura Srijan.

SDL Linguagem de Implantação Srijan.

SDM Métodos de Desenvolvimento do Sistema.

SRS Especificação de Requisitos de Software.

SVL Linguagem de Vocabulário Srijan.

UbiComp Computação Ubíqua.

UFRJ Universidade do Rio de Janeiro.

UXD Design de Experiência do Usuário.

WoT Rede das Coisas.

Capítulo 1

Introdução

De acordo com Filho [10], a Engenharia de Requisitos (RE) possui como responsabilidade principal a identificação e análise de requisitos, com o objetivo de minimizar erros e economizar custos ao simplificar e corrigir eventuais falhas nos requisitos elicitados, antes de ser repassado para as fases seguintes do processo de desenvolvimento de software, incluindo a validação de requisitos. Para Schwab [11], um dos desafios da RE no contexto da Internet das Coisas (IoT) é a necessidade de realizar uma constante validação de dados, motivando a análise das técnicas existentes na RE para obter informações mais precisas.

A Internet das Coisas proporciona que objetos do dia-a-dia se comuniquem com a Internet, desde que seus recursos eletrônicos permitam esta tarefa [12]. A comunicação com a Internet é realizada através de um dispositivo eletrônico com módulo integrado de conexão à rede. Desde sua criação, a Internet das Coisas propõe o compartilhamento de informações em objetos, fornecendo dados e permitindo que os dispositivos controlem a si mesmos e a outros [13].

A Internet das Coisas ganhou um destaque significativo nos últimos anos para pessoas e empresas, cujos interesses se convergem em ter maior praticidade e economia em suas atividades cotidianas [12], uma vez que equipamentos interligados podem proporcionar redução no consumo elétrico e maior conforto para os usuários. No entanto, a RE enfrenta um desafio com a IoT [11] uma vez que a definição e escolha das técnicas para validar os requisitos dos dispositivos da IoT se mostra importante para que seu funcionamento possa ser garantido, pois os dispositivos são meros coletores de dados, porém os mesmos tendem a ter mais inteligência e necessidades [14].

Os sistemas no contexto da IoT são complexos e heterogêneos e as pesquisas na área de Requisitos de Software para IoT são limitados e muitas vezes específicos a um determinado tipo de projeto [15], [16].

1.1 Problema de pesquisa

A engenharia de requisitos no contexto de internet das coisas é marcada pela falta de uma abordagem sistemática para o desenvolvimento de aplicações IoT [2], bem como não possui técnicas específicas para validar os requisitos funcionais e não funcionais dessas aplicações [2].

1.2 Justificativa

Com o crescimento da internet das coisas, diversas aplicações foram surgindo [2], fazendo com que a engenharia de requisitos passasse a ter um papel fundamental nas fases iniciais do projeto, garantindo uma melhor qualidade no desenvolvimento dos sistemas. Propor processos e técnicas específicas para IoT permite um melhor produto e um menor retrabalho por parte das equipes de desenvolvimento de sistemas. Além disso, mapear as dificuldades e as percepções dos desenvolvedores a respeito da área permite a criação de processos que se adaptem melhor aos utilizadores.

1.3 Objetivos

Essa seção destina-se a descrever o objetivo central do trabalho e os objetivos secundários a serem alcançados.

1.3.1 Objetivo Geral

Este trabalho tem como objetivo fazer a conexão entre a literatura da engenharia de requisitos no contexto de IoT com o desenvolvimento em situações reais de produtos IoT por empresas e desenvolvedores.

1.3.2 Objetivos Específicos

Para atingir o Objetivo Geral, os seguintes objetivos específicos foram definidos:

1. Realizar uma pesquisa exploratória para identificar na literatura os processos de RE existentes no contexto IoT;
2. Realizar uma pesquisa exploratória para identificar na literatura técnicas de validação de requisitos no contexto IoT;
3. Avaliar como as empresas de IoT utilizam os processos da engenharia de requisitos e as técnicas de validação de requisitos no desenvolvimento de seus projetos;

4. Levantar as dificuldades encontradas por desenvolvedores e equipes para desenvolverem projetos IoT;

5. Analisar as percepções dos stakeholders em IoT a respeito da engenharia de requisitos;

1.4 Resultados Esperados

Espera-se com esse trabalho identificar as propostas de processos e técnicas de validação da RE na literatura e que possam ser utilizadas pela IoT, obter das equipes de desenvolvimento de software a utilidade desses processos e técnicas para lidar com a validação dos requisitos.

1.5 Metodologia de Pesquisa

A metodologia utilizada neste trabalho consiste das seguintes etapas:

1. **Pesquisa Exploratória:** Revisão da literatura para identificar as técnicas de validação de requisitos e os processos da engenharia de requisitos no contexto da IoT.

2. **Entrevista semi-estruturada com profissionais de IoT:** Etapa responsável por obter a percepção dos profissionais da área de desenvolvimento de software IoT sobre a aplicabilidade/dificuldade da implantação dos processos da RE além de uma explanação sobre a realidade da área e suas dificuldades.

3. **Análise e descrição dos resultados:** A partir das entrevistas e da revisão de literatura descreveremos os resultados obtidos.

Figura 1.1: Metodologia

1.6 Estrutura do Trabalho

O Capítulo 2 apresenta o referencial teórico com a finalidade de aprofundar os conhecimentos gerais sobre Engenharia de Requisitos e Internet das Coisas e a pesquisa exploratória para identificar os processos e técnicas de validação de requisitos no contexto da IoT.

O Capítulo 3 é destinado a parte metodológica das entrevistas realizadas. Nele descrevemos a estrutura utilizada para as conversas, além dos participantes envolvidos e objetivos a serem alcançados pelas entrevistas.

O Capítulo 4 descreve os resultados obtidos a partir da pesquisa exploratória e a partir das entrevistas e questões de pesquisa propostas no capítulo 3.

O Capítulo 5 conclui este trabalho e descreve os próximos passos ou melhorias que poderão ser incorporadas a este trabalho.

Capítulo 2

Embasamento Teórico

Este Capítulo apresenta os conceitos necessários para o entendimento deste trabalho. Na Seção 2.1 é apresentando os conceitos gerais relativos a Engenharia de Requisitos. A Seção 2.2 apresenta os processos existentes para validação de requisitos em dispositivos IoT. A Seção 2.3 realiza uma síntese dos processos existentes na validação de requisitos IoT. Já na Seção 2.4 são apresentadas as técnicas propostas para a validação de requisitos no contexto da IoT. Com base nisso, a Seção 2.5 destaca uma visão geral das técnicas de validação de requisitos em IoT.

2.1 Engenharia de Requisitos

Engenharia de Requisitos (RE), uma subárea da Engenharia de Software, possui relevância para a elaboração de um software por possuir a responsabilidade de levantar todas as solicitações requeridas pelo stakeholder, elaborar uma documentação e repassar todos estes requisitos para todas as partes interessadas de maneira clara, concisa e sem ambiguidades [10].

Outra principal responsabilidade desta área é a redução dos custos ao longo do processo. Este processo, chamado de Validação de Software, possui como objetivo identificar informações conflitantes (seja por ambiguidade ou conflitos com a legislação) e solucioná-las antes de ser iniciada a construção do software. De acordo com Kalinowski e Spínola [17], os custos para reparar um eventual erro cometido por falha na análise de requisitos pode variar de acordo com o nível de evolução do desenvolvimento, sendo bastante caro caso alguma inconsistência seja encontrada na fase final do produto de software.

Todas estas questões ocorrem através de um ponto de partida mutualmente relacionado, os stakeholders. Bourque e Fairley [18] afirmam que, em todo projeto relacionado a RE, é necessário definir os atores, escutar suas necessidades, para que posteriormente

possa ser feito a validação destes requisitos, seguida pela compilação e verificação com os usuários. Os atores são:

- Usuário: pessoa que irá utilizar o produto final, sem nenhuma relação com o desenvolvimento de um software;
- Clientes: parte principal do projeto de software, são as pessoas que encomendam o software para proporcionar uma boa experiência de seus serviços para os clientes da companhia;
- Analistas de mercado: ator que tem o papel de sondar novas tecnologias e tendências no meio externo, podendo adicionar algumas ideias para o projeto;
- Reguladores: tipo de ator que deve possuir domínio na área de atuação para projetos de software que envolvam portarias e outras normas, como bancos, mineração e logística;
- Engenheiros de Software: primordial para todo o projeto de elaboração do software, possui a responsabilidade de ouvir os clientes, validar seus requisitos (isto é, verificar inconsistências), compilar e verificar o sistema junto com os clientes enquanto aguarda um feedback. Todas estas funções devem ser executadas de modo a proporcionar uma redução nos custos do projeto, mesmo que possa ser difícil para softwares cujo funcionamento dependa de autorizações alheias às partes interessadas.

Avila e Spínola [19] definem a Engenharia de Requisitos (RE) como um conjunto de atividades que coordenam a produção e gerência de um projeto de software. As atividades são:

- Produção de Requisitos: construção do projeto de software, em que os requisitos, uma vez categorizados, são registrados através de uma documentação detalhada, para que possam ser verificados (com a finalidade de eliminar possíveis inconsistências, como ambiguidades e outros tipos de problemas) e validados com os stakeholders.
- Gerência de Requisitos: é a parte sensível da RE em decorrência de mudanças alheias ao projeto - mudanças na legislação ou mercado - e por eventuais falhas que passaram despercebidas pela fase de verificação de requisitos. Neste contexto, possui a responsabilidade de coordenar mudanças, gerenciar configurações (permite realizar as mudanças necessárias sem ferir a integridade do software), realizar a rastreabilidade (isto é, criar e manter um escopo para as funções) e gerenciar a qualidade dos requisitos, esta última, análoga à Produção de Requisitos.

2.1.1 Classificação de Requisitos

Vazques e Simões [20] definem requisitos como as informações dadas pelos stakeholders de uma organização, informando suas particularidades, descrevendo permissões, restrições e todas as condições necessárias para que um sistema satisfaça suas necessidades. Segundo Bourque e Fairley [18], existem duas categorias de requisitos vigentes na RE:

- Requisitos funcionais: descreve as funções principais que um software deve executar a cada comando de usuário, como o ato de informar credenciais;
- Requisitos não funcionais: ferramentas que envolvem o escopo dos requisitos funcionais, com a finalidade de proporcionar maior controle sobre um sistema, concedendo ou restringindo permissões para usuários.

Segundo Sommerville [21], os requisitos não funcionais, por realizar restrições, são categorizados em três modos, de acordo com as políticas internas da companhia ou por fatores externos:

- Requisitos de produto: concedem permissões ou negam acesso para alguns tipos de usuário;
- Requisitos organizacionais: são regras impostas pelo cliente (de acordo com as políticas internas da empresa) para acesso a certas funcionalidades do software;
- Requisitos externos: sistemas cujo funcionamento dependem de permissões concedidas por fatores alheios aos stakeholders, como certificações de segurança para bancos.

2.1.2 Processos em Engenharia de Software

Engenharia de requisitos é uma parte fundamental do desenvolvimento de um software como um todo. O procedimento de coletar, analisar e documentar os requisitos é fundamental para o produto final ter a qualidade desejada [1]. Processos em RE podem ser desenvolvidos de algumas maneiras distintas que, apesar de não haver o modelo de ideal de fazer o processo, todos esses procedimentos podem resultar em um ótimo modelo de engenharia de requisitos. São quatro modelos de processos de RE [22]:

1. Modelo de processo de RE linear

   Um dos processos mais simples na literatura de engenharia de requisitos e é destinado para projetos pequenos e com pouca escalabilidade. O desenvolvimento é composto por cinco etapas, conforme apresentado na Figura 2.1.

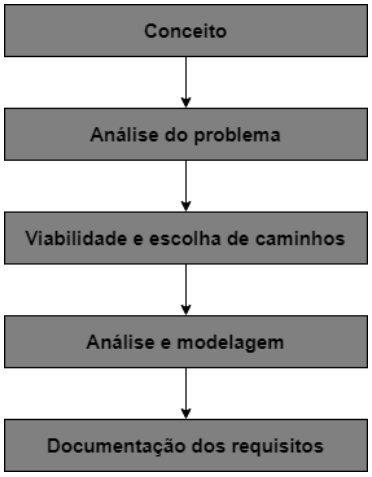


Figura 2.1: Modelo linear [1]

2. Modelo de processo de RE linear iterativo
   Este modelo é parecido em questão de etapas com o processo linear, porém, conforme apresentado na Figura 2.2, determina uma maior precisão nas especificações e na validação de requisitos, pelo fato de que essa etapa no processo é efetuada até os stakeholders ficarem satisfeitos [1].

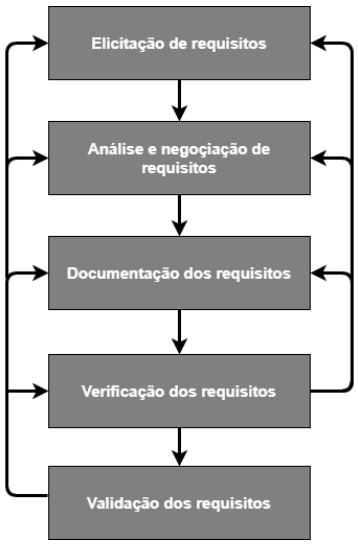
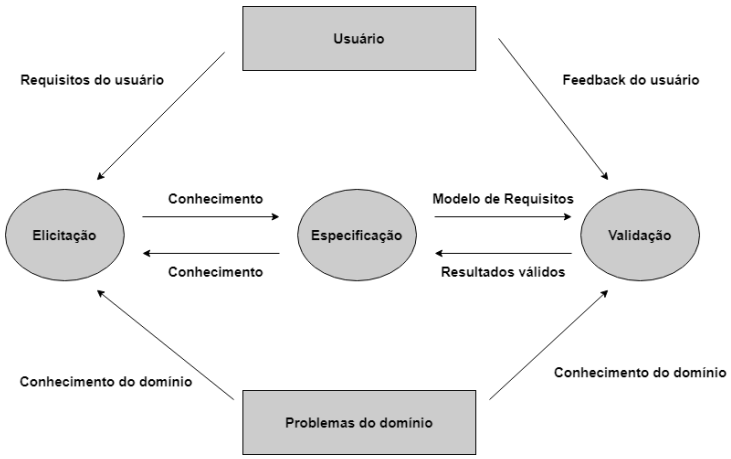


Figura 2.2: Modelo linear Iterativo [1]

3. Modelo de processo de RE iterativo
   Este modelo é composto de três fases principais: elicitação, especificação e validação dos requisitos. De acordo com o apresentado na Figura 2.3, segue um caminho não linear e retira os requisitos se baseando nos stakeholders e problemas do domínio. [1].

Figura 2.3: Modelo Iterativo não linear [1]



4. Modelo de processo de RE espiral
   Modela o processo de RE em uma espiral, em que cada volta na espiral significa passar por todos os processos envolvidos para os requisitos. A espiral é divida em quatro quadrantes em que cada um possui uma tarefa na RE, como pode ser observado na Figura 2.4 [1].

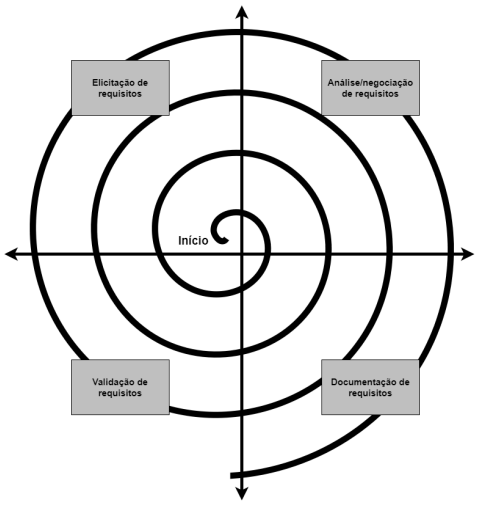


Figura 2.4: Modelo em Espiral [1]

2.1.3 Etapas do Processo em RE

Os modelos de processo em RE tem como principal diferença entre si a ordem e o meio de execução das fases do processo. Contudo, apresentam etapas semelhantes durante todo seu processo. No processo de desenvolvimento do sistema, a engenharia de requisitos deve fazer a elicitação de requisitos com os interessados do projeto, documenta-los de forma

correta, validar e verificar se esses requisitos estão em conforme com o projeto como um todo, além de geri-los por todo o ciclo de vida do sistema [23].

- **Elicitação**
  A elicitação de requisitos é executada pelo conhecimento do contexto do sistema a partir das diversas fontes de requisitos. Os requisitos podem vir a partir de três fontes diferentes. Elas são divididas em stakeholders, documentos e algumas vezes sistemas antigos em operação. Durante essa etapa, uma boa organização e comunicação com os stakeholders são essenciais para que o processo seja feito de uma maneira completa e bem feita [23].

- **Documentação**
  Por muitas vezes, o projeto apresenta uma quantidade muito vasta de requisitos, essa etapa se mantêm importante para que o pessoal envolvido e não envolvido com o projeto consigam compreender todos os requisitos. Além disso, a localização e modificação de requisitos conforme o andamento do projeto é muito facilitada [23].

- **Validação e verificação**
  Os requisitos elicitados e documentados precisam sofrer uma validação para que se garanta que as ideias dos stakeholders estejam sendo respeitadas e se os critérios de qualidade previamente definidos estão sendo compridos. Com isso, o objetivo da validação e verificação é descobrir erros nos requisitos documentados. Incompletude, contradições e ambiguidade são problemas que podem ser encontrados e solucionados nessa etapa do processo [23]. Apesar de serem similares, a etapa de verificação pode ser diferenciada com a de validação por meio dessa análise [24]:

  - Validação: "Estou fazendo o produto correto?"
  - Verificação: "Estou fazendo corretamente o produto?"

- **Gerência**
  Gerenciar os requisitos também é uma das principais etapas e garante a disponibilidade dos requisitos por todo o ciclo de vida do sistema. Essa manutenção ocorre tanto em requisitos individuais como em uma documentação de requisitos completa. Versionamento, rastreabilidade e designar atributos aos requisitos são algumas das técnicas utilizadas para gerir-los [23].

2.1.4 Técnicas para Realizar a Validação e Verificação de Requisitos

O processo de validação e verificação de requisitos é composto de atividades menores como revisar, analisar e testar para que se tenha certeza que o sistema desenhado é compatível com seus requerimentos [24].

Para esses passos serem bem estruturados foram criadas e aprimoradas diversas técnicas na literatura que permitem um melhor uso de tempo e recursos pelas pessoas envolvidas no sistema. Técnicas de validação de manuais ou documentação de requisitos,também conhecidas como revisões, são muito utilizadas nesse processo [23]. Elas podem ser classificadas principalmente por seis tipos de revisões:

1. **Revisão de um especialista**
   Esta técnica se baseia em utilizar os conhecimentos de outra pessoa, um especialista, para fazer um julgamento a respeito da qualidade dos requisitos. O especialista revê todos os requisitos e busca por problemas como inconsistência ou ambiguidade. Quando é encontrado algum problema, o revisor comenta na documentação os erros e possíveis soluções para consertá-los.

2. **Inspeção dos requisitos**
   Esse método normalmente é constituído de múltiplas fases e pessoas, portanto, é um processo detalhado e minucioso para verificar presença de erros nos requisitos. A equipe de uma inspeção é formada de seis funções distintas, sendo elas [23]:

   - **Organizador:** Planeja e controla o processo de inspeção;
   - **Moderador:** Lidera as sessões, mantendo sempre o seguimento da inspeção conforme combinado nas etapas iniciais;
   - **Autor:** Responsável para explicar os requisitos para os inspetores nas fases iniciais do projeto. O autor é responsável também por corrigir os erros levantados pelas últimas etapas do processo;
   - **Leitor:** Membro da equipe que guiará os inspetores diante dos diversos requisitos a serem inspecionados. A função dele é tirar a responsabilidade do inspetor de interpretar o que o autor quis dizer, e eles poderem se concentrar no requisito em si;
   - **Inspetores:** Responsáveis por detectar as falhas e reporta-las para os outros membros do projeto;
   - **Secretário:** Responsável por pegar o resultado da sessão e compila-los, além de preparar a ata para a sessão;

A inspeção dentro do contexto de validação de requisitos pode ser dividida nas etapas de planejamento, visão geral, detecção de falhas e coleta de falhas [23]:

(a) **Planejamento:** É o começo de tudo, nesta etapa é traçado o objetivo da inspeção, o desenvolver do processo e as funções de cada participante;

(b) **Visão geral:** Nesta fase o autor detalha todos os requisitos a sua equipe para que todos estejam cientes do sistema em questão e possam esclarecer eventuais dúvidas;

(c) **Detecção de falhas:** Os requisitos são efetivamente examinados pelos inspetores nesta fase e documentadas para a próxima etapa. A busca de falhas pode ser feita tanto em equipe quanto individualmente. A busca em grupo possui a vantagem de comunicação e consequente sinergia entre os inspetores enquanto a busca individual faz com que todos fiquem focados no trabalho;

(d) **Coleta de falhas:** A última fase dessa técnica é para a coleta e consolidação de todas as falhas apresentadas pela etapa de detecção. Nesta etapa é feita a checagem em todos os erros em busca de erros duplicados ou erros que na verdade não são erros, mas foram detectados como tal por má interpretação dos requisitos.

3. **Walkthrough**
   O Walkthrough é uma técnica parecida com a inspeção, porém é uma versão mais simples e menos rígida [23]. A divisão da equipe é mais simplificada, tendo várias funções podendo ser desempenhadas pela mesma pessoa. Nessa técnica, são usadas as funções de pelo menos um revisor, autor, secretário e moderador. Esse método é feito para identificar falhas nos requisitos e compartilhar o entendimento do projeto entre as pessoas envolvidas no projeto.

4. **Verificação em diferentes perspectivas**
   Essa técnica faz a validação dos requisitos através uma interpretação do sistema levando em conta diversas perspectivas distintas [23]. Com diversos pontos de vista únicos, o foco e resultados obtidos na validação de requisitos tem uma melhora significativa. As perspectivas normalmente variam de sistema para sistema, mas é possível ao menos elicitar alguns tipos que ocorrem geralmente em todo projeto e que podem ser adaptados:
   Perspectivas dos stakeholders

- **Perspectiva do cliente:** O cliente ou usuário checa os requisitos a partir de sua perspectiva e verifica se condiz com a realidade e qualidade esperadas;
- **Perspectiva do desenvolvedor:** O desenvolvedor verifica se os requisitos contêm todas as informações e propriedades necessárias para que o projeto consiga ser feito;
- **Perspectiva de teste:** O testador verifica se com os requisitos presentes é possível fazer casos de teste que satisfazem todo o projeto.

Perspectivas de qualidade do sistema

- **Perspectiva de documentação:** Nessa perspectiva é assegurado que a documentação envolvendo os requisitos atenda todas os critérios anteriormente discutidos;
- **Perspectiva de conteúdo:** Perspectiva que trata do conteúdo dos requisitos em si, e foca na qualidade desse conteúdo;
- **Perspectiva de acordo:** Nesta perspectiva, o foco é assegurar que todos os envolvidos no projeto estejam de acordo com os requisitos levantados e se não há nenhum conflito a ser solucionado.

A técnica de validação usando diferentes perspectivas pode ser usada independentemente ou em conjunto com alguma outra técnica de revisão vista anteriormente. É importante, nesta técnica, detalhar o que será abordado em cada perspectiva para que todos estejam entendendo o que irá ser focado em cada uma visão, para que não haja retrabalho posteriormente.

5. **Validação por prototipagem**

   Protótipo é um modelo preliminar de um sistema completo. Com isso, o que essa técnica faz é usar esse modelo a favor da validação de requisitos. A partir de um protótipo do projeto, requisitos podem ser testados e validados de forma prática pelos stakeholders e é verificado se atendem ao objetivo originalmente traçado. Os protótipos podem ser divididos dependendo da destinação deles em protótipos descartáveis ou protótipos evolutivos [23]. Os descartáveis servem somente para o teste atual e não irá ser colocado em produção novamente, já o evolutivo é um protótipo que fica em constante evolução conforme o projeto se desenvolve e, por isso, requer um maior trabalho para ser feito. Para que um protótipo seja feito, é preciso fazer a seleção dos requisitos que ele ira contemplar. Esse conjunto normalmente não cobre todos os requisitos documentos já que traria custos e complexidades muito elevados.

Por isso, é preciso fazer um critério de seleção entre os requisitos e determinar quais são os mais adequados a serem utilizados no protótipo.

Junto com o protótipo em si, são necessários mais três documentos para que seja feita a validação dos requisitos [23], sendo:

- Um manual para os integrantes que vão utilizar o protótipo que conste todas as informações de uso e aplicação;
- Um documento que conste vários cenários para a validação dos requisitos e uma lista de verificações com todas os requisitos que estão querendo ser validados. Cenários que não constam nos documentos podem e devem ser testados depois de todo o processo para que se verifique o sistema em diversas situações e circunstâncias que não foram planejadas de início.

Após a finalização da validação, os resultados são compilados e discutidos. Alteração, remoção ou adição de requisitos são consequências durante essa etapa. Se as alterações forem muito grandes ou impactantes, é aconselhável a remodelação do protótipo para que abrange essas mudanças e seja possível de ser feito uma nova validação [23].

6. **Validação por checklist**
   A técnica de checklist é utilizada em projetos complexos e quando o sistema possui diversas questões que precisam ser levados em conta. As checklists podem ser utilizadas em conjunto com diversas outras técnicas e ela é composta de perguntas ou afirmações para identificar os erros contidos nos requisitos. A fonte para a montagem da checklist pode vir tanto dos stakeholders quanto de critérios de qualidade de requisitos que a literatura dispõe. A lista pode sempre estar sendo atualizada conforme o projeto avança e novas funcionalidades e requisitos surgem, por isso, remoção de perguntas ambíguas, edição e adição de informações são sempre bem vindas conforme o projeto vai se desenvolvendo. O sucesso da técnica vai depender da extensão e complexidade da checklist. Uma lista muito longa e cheia de detalhes pode fazer o processo ficar mais lento e inviável para o auditor. A mesma coisa pode acontecer se a lista for muito subjetiva, tornando o processo de validação lento e suscetível a falhas.

2.1.5 Aplicações da IoT

De desktops a smartphones, todos os dispositivos eletrônicos com capacidade de enviar ou receber dados através da Internet ou outros meios de conexão, proporciona a indústria e ao consumidor uma melhor automação em processos, redução de custos e maior controle

sobre dispositivos, todos estes atrelados a IoT. Todos os recursos com uma variedade de sensores embutidos permitem diversas aplicações em potencial [25]. Neste quesito, Lacerda [26] destaca cinco utilidades para a IoT.

A Internet das Coisas evoluiu de maneira significativa neste aspecto graças a capacidade dos dispositivos interligados em realizar um monitoramento efetivo de atividades físicas e condições de saúde do usuário [25]. Dispositivos vestíveis como smartwatch (relógio com funções e monitoramento integrado a um smartphone) são recomendados para o monitoramento de atividades físicas, bem como pode ser utilizado para monitorar doenças crônicas, como por exemplo o mal de Parkinson [27].

Para Souza [8], este monitoramento dos pacientes através da IoT torna mais rápido o processo de diagnóstico, de identificação e da análise das condições de saúde do paciente. Além destes, dispositivos IoT ligados a área de saúde também podem coletar dados e rastrear os locais onde a pessoa frequentou, embora este assunto afete questões de privacidade previstas por legislações de cada país [25].

Santos et al. [12] definiu esta categoria da IoT como *Smart Home* (casas inteligentes) devido ao fato dos dispositivos interligados poderem controlar, por si mesmos ou por ação humana, todos os equipamentos elétricos e eletrônicos dentro da residência do usuário. Em casas e apartamentos, a IoT se mostra bastante eficiente em vários aspectos. Para Milenkovic [25], a Internet das Coisas dentro de um ambiente residencial pode proporcionar: 1. Conforto: controle de luz, de temperatura ou ligar/desligar dispositivos; 2. Segurança: abrir/fechar portões a uma certa distância ou avisar sobre janelas e portas abertas; 3. Redução de custos: desligamento automático de luzes e equipamentos que não estão sendo utilizados por um tempo determinado.

Milenkovic [25] destacou que os recursos propostos na IoT no conceito das smartcities (cidades inteligentes) necessita de uma ampla rede de conexão capaz de transmitir dados para controle, gerar estatísticas e propor maior segurança a sua população, como antecipação às previsões do clima em um determinado dia. Apesar deste crescimento ser promissor, estas tecnologias ligadas a IoT ainda dividem espaço com dispositivos analógicos, que embora não seja recomendável, conseguem atender as necessidades de cada usuário de um produto ou serviço [26].

2.2 Processos de Engenharia de Requisitos no Contexto de IoT

A evolução significativa da Internet das Coisas e sua complexidade se tornou um desafio para a área de Engenharia de Requisitos, principalmente para a indústria 4.0 cujo enfoque é o uso de biotecnologia, energias renováveis e nanotecnologia [11]. Para Lim et

al. [28], atualmente, existem estudos e aplicações voltadas ao uso da IoT como recurso para validação de requisitos, possuindo como características individuais as vantagens e desvantagens.

Os processos encontrados por meio da pesquisa exploratória foram evidenciados nas subseções abaixo e intituladas com o nome do artigo publicado que contem o processo ou o nome que os autores deram ao processo.

2.2.1 A Requirements Engineering Process for IoT Systems

Devido a falta de procedimentos sistemáticos a respeito de RE em IoT, Silva et al. [2] descreveu um processo voltado para o desenvolvimento de sistemas em Internet das coisas utilizando Notação para Modelagem de Processos de Negócio (BPMN). O processo é composto por três sub-processos, em que cada um é feito de acordo com a ISO/IEC/IEEE 12207[1], ou seja, a norma que define processos em engenharia de software. Cada sub-processo e o processo em si possui entradas e saídas:

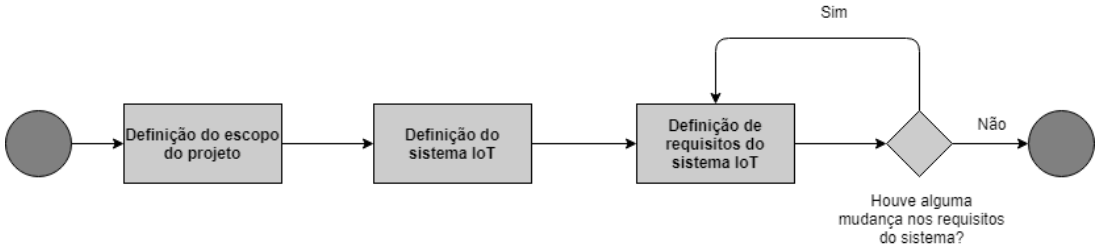


Figura 2.5: Modelo de processo em RE voltado a IoT [2].

1. **Definição do escopo do projeto**:

   Como primeiro sub-processo, descrito na Figura 2.6, a definição do escopo desempenha o papel importante de determinar o esquemático do projeto. O sub-processo determina primeiramente qual é o problema e o que o sistema irá abranger. Em seguida, determina quais são os stakeholders em potencial e, por consequência, quais são suas vontades e desejos a respeito do projeto a ser feito. Com todas essas informações agrupadas, é feita uma análise a respeito da viabilidade do sistema, se é possível fazer o sistema e se ele está de acordo com o plano. No passo seguinte, as necessidades dos stakeholders são transformadas em requisitos para o projeto. Esse processo inclui todas a identificação,controle de qualidade e análise dos requisitos. Por fim, o projeto precisa passar pela aprovação explícita de todos os envolvidos.

[1] `https://www.iso.org/standard/63712.html`

Este sub-processo tem como objetivo final que as necessidades dos stakeholders sejam transformadas em requisitos e que todas as pessoas envolvidas estejam cientes deles.

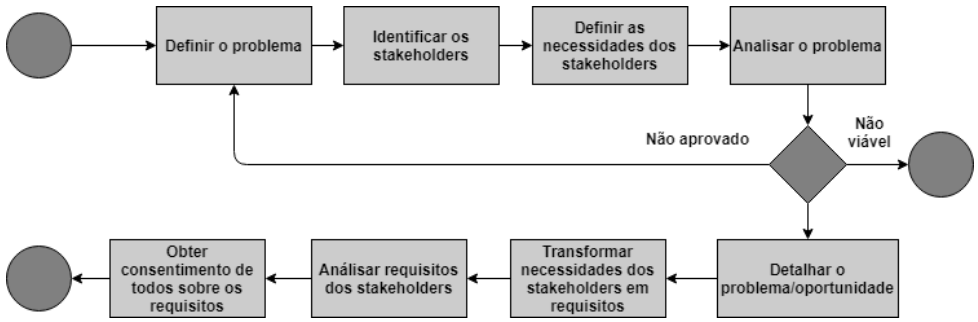


Figura 2.6: Sub-processo 1: Definição de escopo [2].

2. **Definição do sistema IoT**:

   Na segunda parte do processo, o aspecto IoT do projeto é levado muito em consideração em seu desenvolvimento. Seguindo o fluxograma proposto na Figura 2.7, o propósito dessa etapa é definir os cenários IoT, identificar os componentes presentes no sistema e identificar diversas soluções para que se possa escolher uma preferida. Esse sub-processo determina começa propondo uma solução em IoT para o sistema, se baseando nos requisitos dos stakeholders e do detalhamento do projeto,todos eles feitos no sub-processo anterior. Logo , se constrói cenários IoT em alto nível, normalmente em forma de narrativa ou esquemas, para o fácil entendimento de todas as pessoas envolvidas no projeto.

   Com o cenário definido, são feitas a identificação dos componentes do sistema IoT e suas ações. Dispositivos, sensores e outros são identificados e descritos de acordo com o cenário proposto e quais são suas ações dentro do sistema. Dessa etapa podem surgir diversos arranjos diferentes de componentes e ações diferentes, por isso cabe a equipe se reunir e decidir se as soluções encontradas atendem ao projeto e qual delas escolher para serem as mais viáveis.

   No final desse sub-processo os cenários IoT devem estar definidos e os componentes e ações do sistema IoT identificados. É preciso apresentar diversas composições para o sistema IoT além daqueles que são os mais propensos a se desenvolverem para realmente serem implementados.

3. **Definição de requisitos para o sistema IoT**:

   Última etapa do processo, e tem como objetivo a especificação dos componentes IoT e verificar toda a especificação do projeto. No final desta etapa, os componentes precisam estar especificados e verificados, casos de uso precisam estar criados

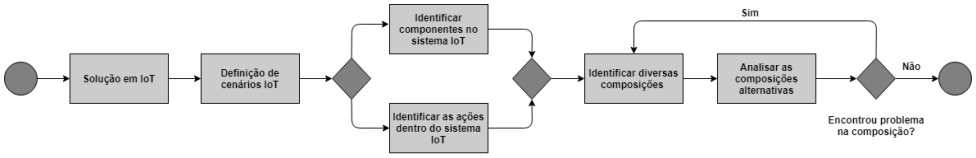


Figura 2.7: Sub-processo 2: Definição do sistema IoT [2].

e verificados. Além do vinculo dos requisitos dos stakeholders com o sistema IoT. O sub-processo inicia uma especificação dos componentes a fim de criar uma documentação detalhada sobre os dispositivos que compõem o sistema IoT. Logo em seguida a validação dessa especificação para que não haja nenhuma inconformidade. Com isso, é feito a especificação e a verificação dos componentes de software para que seja possível o desenvolvimento do projeto de acordo com o planejado.

Tendo por concluído a especificação e a verificação, são feitos e verificados os casos de uso se baseando em cada cenário criado em etapas anteriores. Estes casos abrangem o uso dos componentes, as exceções que podem ocorrer no software. Os casos de uso também devem abranger as regras de negócio que o sistema irá conter. Por fim, é feito um acompanhamento para saber se todos os desejos dos stakeholders foram atendidos nos requisitos e nas documentações. Além de verificar se os requisitos do sistema IoT atendem aos requisitos dos stakeholders. No final do processo, todos os componentes de software e de dispositivos devem estar especificados e verificados,casos de uso devem ter sido feitos e o acompanhamento de requisitos IoT e dos stakeholders deverão ter sido desempenhados.

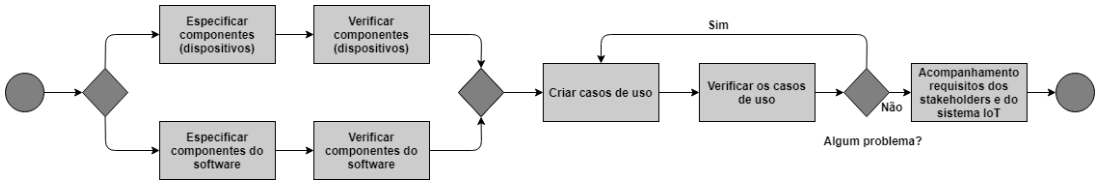


Figura 2.8: Sub-processo 3: Definição dos requisitos para o sistema IoT [2].

2.2.2 A UML-based Proposal for IoT System Requirements Specification

Tendo como motivo a falta de processos da engenharia de requisitos voltada para IoT, Reggio [15] apresentou o IoTReq, um processo para elicitação e especificação de requisitos em sistemas IoT. O processo tem como diferencial sua construção em UML para toda a construção do modelo. O IotReq primeiramente faz a modelagem em que o sistema IoT será inserido, tendo o seu domínio extenso e possuindo várias entidades e interações

entre elas. Ele é feito seguindo um paradigma orientado a serviços, e usando diagramas UML. Depois dessa fase inicial, o processo faz a descrição de diversos tipos de objetivos que o sistema tera como projeto. Esses objetivos são propriedades ou requisitos que os stakeholders anunciaram que o sistema deveria abranger, podendo ser divididos em:

1. **Objetivos estratégicos:** Primeiro requisito elicitado pelos stakeholders, normalmente são requisitos de alto nível e normalmente são as razões básicas que o projeto está sendo feito. Por isso, muitas vezes são genéricos/subjetivos e deles são criados outros requisitos;

2. **Objetivos operacionais:** Requisitos feitos a partir dos objetivos estratégicos, esses objetivos são propriedades e funções que o sistema IoT apresentará. Eles representam os requisitos funcionais do projeto, são essenciais para uma boa elicitação dos requisitos;

3. **Objetivos tecnológicos:** São os requisitos não funcionais do projeto, esses objetivos entram no detalhe da implementação e citam quais e como certas tecnologias serão usadas. São derivados dos objetivos operacionais e eles serão os requisitos de mais baixo nível levantados.

Ao final, o IotReq é feito a partir de três passos, a construção de um modelo para o domínio IoT, um esquemático para a visualização dos objetivos/relações do sistema e, por último, a especificação desses objetivos a partir de um diagrama UML. Esse processo foi colocado em prática com um caso de estudo realístico e conseguiu especificar os requisitos de forma precisa e com até certa facilidade de leitura [15].

2.2.3 TrUStAPIS: A Trust Requirements Elicitation Method for IoT

TruUStAPIS [29] é um método para elicitação de requisitos para um cenário IoT. Este método é feito a partir da criação de domínios e dividi-los em sete tipos diferentes (usabilidade, identidade, segurança, disponibilidade, privacidade, proteção e confiabilidade). Considerando a necessidade de haver confiança entre as entidades que atuam com IoT, o método proposto pelos autores é composto pelos seguintes elementos para a elicitação do requisito:

1. **Ator:** Humano ou entidade IoT que realiza o objetivo descrito;

2. **Ação:** Atividade que o Ator desempenha no sistema;

3. **Objetivo:** Objetivo que o requisito quer alcançar no sistema;

4. **Contexto:** Faz parte do domínio do requisito, envolve as circunstancias do projeto e os objetivos do requisito.

Para a documentação, os autores propuseram o uso de um JSON como template, tanto por ser suportável em várias linguagens como por ser uma linguagem fácil de ser entendida tanto por computadores como por humanos. Esse tipo de organização também é defendida pelos autores por ter capacidade de alta rastreabilidade entre seus requisitos, sendo possível ter um controle sobre as versões dos requisitos. Além disto, é adotado os diagramas UML para auxiliar nas fases de elicitação e especificação de requisitos em IoT.

O TrUStAPIS é dividido em quatro etapas [29]. Na primeira etapa foi expandido o sistema IoT para um esquemático envolvendo as entidades. Essa etapa torna mais simples a elicitação dos requisitos por facilitar a interpretação do sistema e das necessidades do stakeholders. A segunda parte já é a composição dos parâmetros nos JSONs baseando-se no modelo construído na fase anterior. A terceira parte é destinada a documentação escrita dos requisitos levantados pelo JSONs em tabelas, essa etapa tem como saída os próprios requisitos elicitados. A quarta e última etapa é manter a rastreabilidade dos requisitos, construindo-se novas tabelas para a associação de requisitos.

2.2.4 REM4DSPL: A Requirements Engineering Method for Dynamic Software Product Lines

O Método para Engenharia de Requisitos para Linhas de Produtos de Software Dinâmicos (REM4DSPL), criado por Sousa et al. [30], tem a finalidade de organizar um produto de software com o intuito de ser adaptativos às necessidades existentes em uma companhia, definido pelo Linha de Produto de Software Dinâmico (DSPL). O REM4DSPL é um método para a engenharia de requisitos voltada a produtos Linha de Produto de Software Dinâmico (DSPL). DSPL são linhas de produtos que possuem sensibilidade ao contexto que estão inseridos e precisam estar sempre se adaptando em tempo de execução para que suas finalidades possam ser feitas.

Sistemas IoT entram nessa categoria pois sempre são influenciadas pelo contexto presente no sistema, além da alta capacidade de adaptação necessária para poder executar corretamente as constantes mudanças. Tendo isso em mente o Método para Engenharia de Requisitos para Linhas de Produtos de Software Dinâmicos (REM4DSPL) pode ser um método utilizado em IoT, onde seu processo utiliza BPMN [30]. Esse método é composto de três fases no seu processo.

1. **Elicitação dos requisitos:** Colhe as características dos requisitos e identifica o contexto e as características que o sistema e o contexto em que ele está inserido

deve conter. O método propõe utilizar cenários e técnicas narrativas envolvendo o usuário e o sistema para detecta-los;

2. **Especificação dos requisitos:** Com base no que fora realizado na etapa anterior e utilizando uma documentação dos cenários levantados, é realizada uma identificação das regras de negócio e especificar as características de um Linha de Produto de Software Dinâmico (DSPL). Sendo a etapa mais complexa deste processo, é dividido em objetivos, sendo eles:

    - Especificar os requisitos; Identificar o modelo de negócio do sistema;
    - Especificar as funcionalidades do produto;
    - Associar as funcionalidades com os requisitos;
    - Identificar e especificar o contexto e suas características;
    - Identificar as variações do contexto e modela-las.

3. **Gerenciamento da mutabilidade do sistema:** Como etapa final, esta fase tem como objetivo de modelar todas as características da Linha de Produto de Software Dinâmico (DSPL) que serão implementadas e fazer a gerencia da mutabilidade e relação entre características e requisitos. Além disso, esta etapa faz uma verificação nos modelos criados em busca de inconsistências presentes.

2.2.5 Modeling IoT Aplications with SysML4IoT

Costa et al. [31] propuseram uma metodologia para o desenvolvimento inicial das aplicações IoT, desde a especificação dos requisitos a montagem de esquemáticos. Esse método é baseado em diagramas UML e foca na resolução de três problemas cruciais na fase de design de um sistema IoT, como a diversidade de componentes de hardware e software disponíveis para IoT, bem como a escassez de técnicas para a elicitação de requisitos e seu posterior planejamento por cenários. Como resolução ao primeiro problema, o SysML4IoT, usa um modelo que abstrai o sistema para especificar claramente os diversos componentes e softwares que um sistema IoT apresenta, facilitando assim o entendimento como um todo. Neste modelo o sistema é composto por dispositivos e serviços interagindo com outros sistemas ou usuários.

Para esclarecer melhor as necessidades dos stakeholders, Costa et al. [31] usaram tabelas para a separação dos stakeholders por habilidades e responsabilidades. Com isso, foi criado um outro modelo contendo vários pontos de vista diferentes a respeito da aplicação: visão funcional, de informação, operacional e outros, são citados como possíveis entradas para essa modelagem. Por último, o método IoT DevProcess foi proposto para

resolver o caso do terceiro problema, a falta de métodos de design das aplicações. Esse método utiliza da SO/IEC/IEEE 15288:2015 e da OOSEM (Object-Oriented SE Method) para a criação de um método para aplicações IoT em desenvolvimento.

O IoT DevProcess inicia por meio da análise dos requisitos do sistema. Os requisitos dos dispositivos são especificados, incluindo protocolos e formatos dos dados. Após essa etapa, é feita a modelagem das entidades, são especificados os dispositivos, serviços e recursos que o sistema terá quando estiver pronto. Nessa etapa também é criado um modelo para o formato dos dados que os serviços dentro do sistema irá conter. O próximo passo do método é decompor o sistema em diversos componentes que interagem entre si baseados nos requisitos levantados, e com esses componentes criar modelos para a visualização tanto em alto nível como com mais detalhes da composição.

2.2.6 Specifying Functional Requirements and QoS Parameters for IoT Systems

Costa et al. [32] descreveram sobre a não trivialidade de fazer uma especificação de requisitos em projetos IoT e propuseram um método para ajudar na especificação de requisitos em sistemas IoT. O processo é chamado de Linguagem de Modelagem de Requisitos em IoT (IoT-RML) e tem como objetivo especificar e modelar requisitos de diversos stakeholders de maneira precisa e resolver os conflitos que diversos requisitos possam possuir. O IoT-RML especifica tanto requisitos funcionais como não funcionais (Quality of Service) e utiliza a linguagem SysML para representar os modelos utilizados no processo, com base em diagramas UML.

O processo tem como base um modelo do domínio dos requisitos gerado quando requisitos são elicitados pelos stakeholders e modelados conforme estrutura contida [32]. Nele são descritos os conflitos entre requisitos, a sua categoria e suas características. Requisitos não funcionais são tratados como parâmetros QoS e considerados diferentemente conforme a camada da arquitetura do sistema. Vale destacar que o artigo descreve os conflitos de requisitos como de quatro tipos distintos, são eles:

1. **Oposto:** Conflito causado quando requisitos fazem ações opostas um do outro;
2. **Numérico:** Conflito em relação a unidade de medida entre um requisito e outro;
3. **Duração:** Aplicações requisitando o mesmo recurso mas com quantidade de tempo de uso diferente;
4. **Completude:** Uma aplicação não conseguir efetuar algum requisito por causa do uso do recurso por outra aplicação

Com os requisitos sendo abstraídos pelo modelo do domínio e baseado na SysML, é criado o IoT-RML SysML Profile, que efetua todo o gerenciamento de múltiplos stakeholders e conflitos entre requisitos. A partir dessas duas ferramentas, o processo entra na etapa de resolver os conflitos e o desenvolvimento propriamente do sistema IoT proposto.

2.2.7 Design Science and ThinkLets as an Holistic Approach to Design IoT/IoS Systems

O processo criado por Bernsteiner e Schlögl [33] utilizou dois parâmetros para a elaboração de um design de sistema IoT: o *Design Science* e a ferramenta *ThinkLets*. O primeiro parâmetro, o *Design Science*, é oriunda do termo Design Science Research (DSR), tendo consigo um papel fundamental para a resolução de problemas, criando artefatos que possam atender as necessidades e aos problemas de uma companhia [34].

Apiola e Sutinen [35] destacaram a imponência do DSR em relação a outros projetos de desenvolvimento de software, pela capacidade do DSR em utilizar um amplo e absoluto repertório de metodologias de pesquisa ao longo das fases do processo. Ainda, para os autores, estes processos são divididos em fases de explicação do problema, definição dos requisitos (funcionalidades e permissões), design, desenvolvimento e validação.

Segundo Bernsteiner e Schlögl [33], é importante destacar alguns fatores que influenciam o processo:

1. **Construtores:** Utilizados para definir um contexto geral sobre requisitos, problemas de domínio e realização de artefatos;

2. **Modelos:** Cria uma relação entre os construtores a fim de compartilhar informações em comum e, consequentemente, realizar configurações específicas para solução de problemas;

3. **Métodos:** Ferramentas que podem ser utilizadas para verificar se os construtores e modelos estão de acordo com o projeto de software, como exemplo, o uso de diagramas UML.

Hevner [3] destacou as três fases que envolvem a DSR, descritas na Tabela 2.1. O autor pondera a existência de pontes que interligam estas fases. Por exemplo, existe um **Ciclo de Relevância** - que prevalece entre as fases de Meio ambiente e DSR - e possui a responsabilidade de realizar testes de campo; o **Ciclo de Design**, que executa em loop apenas dentro do contexto da DSR; por fim, o **Rigor do Ciclo**, atuando entre a Base de conhecimento e o DSR na troca de informações.

ThinkLets, por outro lado, é definida como um padrão de construção genérica de blocos capaz de atender as especificações do projeto. O seu desenvolvimento se baseia nos

Fases	Propriedades
Ambiente do contexto	Aplicação de Domínio: • Pessoas • Sistemas de uma organização • Sistemas técnicos • Problemas e Oportunidades
DSR	Construção de um artefato de design e processos
	Avaliação
Base de Conhecimento	Fundamentos: • Teorias científicas • Métodos científicos • Experiência • Produtos e processos do design

Tabela 2.1: Três fases do processo envolvendo a DSR [3].

processos colaborativos, designando padrões genéricos de colaboração, que deixa claro as atividades para cada membro do projeto [33]. Consistindo de cinco fases, Kolfschoten e de Vreede [36] descreveram a visão geral sobre o ThinkLets na Figura 2.9. Cada item pode enviar e receber dados através das interações, as quais são:

- Diagnóstico da tarefa: através de uma conversa com os stakeholders, é discutido questões relevantes em relação a requisitos. Para a equipe de RE, os requisitos apresentados podem ser ajustados, negociados ou questionados;
- Decomposição da atividade: nesta fase, as atividades são divididas entre o grupo de desenvolvimento através do ThinkLets, onde todos devem possuir uma visão ampla do produto de software;
- Escolha da tarefa ThinkLets: após a conclusão da decomposição, o ThinkLets atua como responsável por atribuir as atividades separadas na fase de decomposição, fase esta que pode ser delicada;
- Construção de agenda: uma vez concluídas, as fases anteriores servem como parâmetro para esclarecer questionamentos, definições, objetivos, tempo que será gasto e estimativas;
- Validação do design: na última fase do processo, todos os requisitos propostos nas etapas anteriores são validados. Também é verificado questões acerca dos custos, tempo gasto e outras informações complementares;
- Satisfeita as cinco etapas deste processo colaborativo, uma documentação é elaborada.

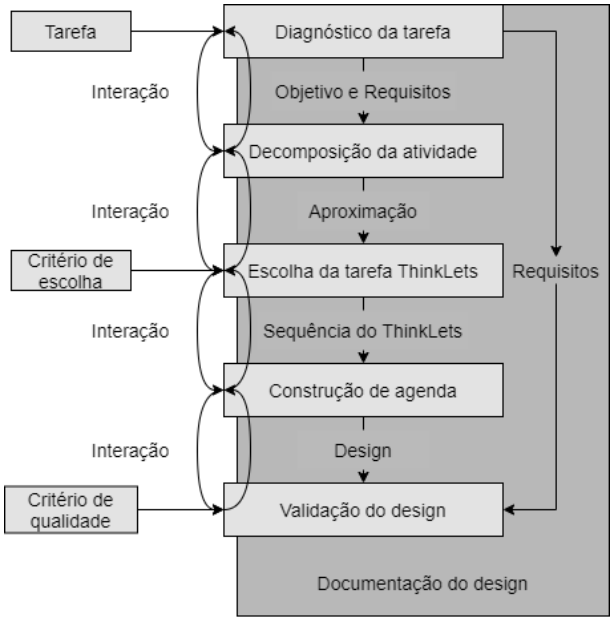


Figura 2.9: Visão geral sobre o funcionamento da ferramenta ThinkLets [3].

Levando em consideração os dois conceitos sobre DSR e ThinkLets vistos nesta seção, estas questões servem como base para a elaboração de uma ferramenta de design para IoT. Utilizando a técnica de brainstorming - chamada por Bernsteiner e Schlögl [33] como *DirectedBrainstorming* (Brainstorming Dirigido) - com o auxílio da ferramenta ThinkLets, utiliza cinco parâmetros seguindo os padrões colaborativos, os quais são:

1. Distribuição de papéis: consiste nos participantes do projeto realizarem um caso de uso em folha de papel, entregar a um moderador e, caso necessário, pode ser elaborado mais cenários;

2. Análise dos papéis: nesta fase, através de uma discussão e da ferramenta *FastFocus* (foco rápido) presente no ThinkLets, os participantes podem expressar uma visão geral dos casos de uso feitos na fase anterior, definindo portanto um caso de uso que não prejudique as fases seguintes;

3. Organização dos conceitos: por meio da ferramenta *ThinkLet PopCornSort*, tem o intuito de estabelecer relações e dependências nos casos de uso, feito por fases anteriores;

4. Descrição e desenvolvimento: ainda permanecendo nos termos de definição da fase anterior, nesta etapa os casos de uso passam por uma descrição acerca dos termos existentes na IoT - como sensoriamento, identificação e atuação - bem como meio de comunicação, interface de usuário e outros;

5. Avaliação: nesta última fase, os casos de uso trabalhados ao longo do processo de brainstorming passam por uma validação, que tem o objetivo de analisar as funções, situações que podem ser relevantes e eventuais falhas que podem ocorrer. Esta etapa conta com a ajuda da ferramenta *BucketWalk*.

Uma vez concluído as cinco etapas e atingido os objetivos em comum de cada fase (isto é, a conclusão de um caso de uso), é utilizado ainda no processo de avaliação uma ferramenta denominada *StrawPoll*, que permite a cada participante votar nos casos de uso que forem mais relevantes ao processo. Finalizando esta questão, todo o processo é dado como encerrado [33].

2.2.8 An Improved RE Framework for IoT-Oriented Smart Applications using Integrated Approach

Kaleem et al. [37] apresentaram um estudo para a criação de uma ferramenta de abordagem integrada utilizando o conceito de *smartness* (inteligência) - voltada para a Internet das Coisas - com base nos processos e técnicas de validação existentes na Engenharia de Requisitos. Os autores destacam a utilização de *frameworks* para definir os processos nesta proposta [37], adotando o uso de diagramas UML. Seu primeiro processo destaca o ponto de partida para o estudo dos processos, seguindo uma metodologia similar à proposta do processo linear elaborada por Rehman et al. [1], com foco em satisfazer as cinco etapas:.

1. **Identificação dos Stakeholders:** nesta etapa, as partes interessadas entram em uma conversa para debater o propósito do sistema, suas funcionalidades, permissões, bem como a definição de responsabilidades acerca das áreas da Engenharia de Software. Esta fase do processo é baseada no método Goal-Question-Metric, conceito criado pela Universidade de Maryland para unir objetivos, questões operacionais e métricas [38];

2. **Elicitação de Requisitos:** fase responsável por colher as necessidades dos stakeholders utilizando fatores socio-técnicos para mesclar fatores humanos, organizacionais e aspectos técnicos do sistema [39];

3. **Análise de Requisitos:** ocorre a checagem de todas as necessidades descritas pelos stakeholders com auxílio do Design Baseado em Modelo (MBD), ferramenta primordial para a análise, também incluindo aspectos de verificação e validação dos requisitos [40];

4. **Especificação dos Requisitos:** nesta fase, um workshop interativo se mostra presente dentro do framework da Engenharia de Requisitos para documentar, através da Especificação de Requisitos de Software (SRS), todos os processos e requisitos voltados a *smartness* envolvendo a IoT [37];

5. **Métodos:** ferramentas que podem ser utilizadas para verificar se os construtores e modelos estão de acordo com o projeto de software, como exemplo, o uso de diagramas UML. na última fase deste processo principal, é utilizado técnicas para design de casos de teste - podendo ser um *template* ou um protótipo - para validação e verificação do produto de software, envolvendo os stakeholders e a equipe de desenvolvimento. Caso receba um *feedback* positivo, este processo é declarado como encerrado.

2.2.9 Detecting IoT Applications Opportunities and Requirements Elicitation: A Design Thinking Based Approach

Dantas et al. [41] descreveram um processo de RE para IoT por meio da abordagem Design Thinking. Este processo tenta entender os objetivos dos usuários e stakeholders enquanto detecta os desafios e propõe soluções que não estavam sendo pensadas a primeira vista. O processo de Design Thinking descrito pelos autores [41] é iterativo, não linear e consta com cinco etapas principais, divididas em:

- **Compreensão:** Fase inicial e crucial para o sucesso no processo, nesta fase é feita a compreensão do ponto de vista dos stakeholders. São citadas técnicas como bodystorming e entrevistas;

- **Definição:** Define qual será as características que o design do sistema terá como foco. A *Interaction Design Foundation* é uma abordagem centralizada no ser humano, ampla para a criatividade e focada para que seja possível de ser concretizada;

- **Idealização:** Fase para se pensar em outras ideias além da qual já foi arquitetada. Essa etapa visa ampliar a visão do time para soluções inovadoras;

- **Prototipação:** Criação de protótipos de alta e baixa fidelidade para que se possa validar as ideias levantadas e adquirir mais feedbacks dos stakeholders;

- **Testagem:** Agrupamento dos feedbacks dos stakeholders e dos protótipos criados ao longo do desenvolvimento do sistema. A equipe pode analisa-los e deles surgirem uma nova ideia.

Dantas et al. [41] utilizaram três técnicas:

- **How Might We Questions (HMW):** Técnica de *brainstorming* que se baseia em perguntas de "Como pode ser feito?". Os times do projeto se reúnem para analisar os fatos e requisitos já conhecidos junto com as diversas perspectivas que cada time deve ter a respeito do projeto. Rescrever todos essas informações em perguntas HMW e verificar se com elas dá para se fazer outras soluções a partir de uma sessão de *brainstorming*;

- **Jornada do usuário:** Técnica utilizada para documentar o comportamento do usuário utilizando o produto. Cada passo que o usuário fizer no sistema é documentado, assim como o pensamento e sentimento do usuário a respeito do uso da plataforma. Essa técnica constrói uma perspectiva do usuário e pode ser essencial para a descoberta de algum problema ou inconsistência;

- **Diagrama de afinidade:** Agrupamento por setores de dados e resultados obtidos anteriormente a partir de um critério antes definido. Essa técnica facilita a organização e a fluidez das ideias conforme o projeto é desenvolvido.

Dantas et al. [41] descreveram algumas ferramentas feitas para IoT que podem ser utilizadas para o entendimento dos requisitos do sistema IoT e aplicação das técnicas mencionadas, são elas:

- **IoT Design Deck:** Utilizado para fazer a integração de times para a produção de um sistema IoT. Utiliza um deck de cartas para ajudar os times a utilizarem uma linguagem em comum e focar na experiencia do usuário. Por meio de várias técnicas e ferramentas, esse método garante a comunicação e entendimento do projeto IoT que normalmente contém uma grande multidisciplinaridade;

- **Tiles IoT ToolKit:** Projeto em open source também baseado em cartas para ajudar na elaboração de ideias, storyboards e explorar o cenário IoT por completo;

- **IoT Service Kit:** Ferramenta para IoT que baseia-se em cartões, onde existe a interação com o design e tokens para esquematizar interações de entidades no projeto.

2.2.10 An Evidence-Based Framework for Supporting the Engineering of IoT Software Systems Mid-stage Research

Motta [42] propôs um framework composto por três etapas, a fim de evitar soluções limitadas (considerando a coleta e processamento de dados), deixando lacunas em aberto no quesito de validação dos requisitos em Internet das Coisas, bem como visa reforçar a

qualidade de desenvolvimento baseando-se nos estudos de viabilidade do projeto. Com base na metodologia O que, Como, Quem, Quando e Por quê (5W1H), Motta apresentou um framework composto por três fases, cada uma destas possuindo sub-rotinas.

1. **Concepção:** nesta etapa, é feita uma análise detalhada sobre os requisitos solicitados para um dispositivo IoT.

   (a) Caracterização: primeira etapa do processo, é consultado na literatura as técnicas existentes, com suas definições, aplicações e características, a fim de encontrar possíveis inconsistências;

   (b) Preocupações: nesta etapa, é vista as questões legais e sociais que podem afetar um projeto em Internet das Coisas de forma negativa, como invasão a privacidade ou coleta de dados sem a devida precaução na segurança dos mesmos;

   (c) Investigação de facetas: visa a revisão de questões que podem envolver a IoT, como áreas de conhecimento e tópicos, etapa esta que pode ter um contexto amplo;

   (d) Proposta de um framework: com base nos dados colhidos nas etapas anteriores, é construído um Conjunto de Conhecimentos em IoT (BoK), baseando-se em questões como coisas, inteligência, interatividade, meio ambiente, conectividade e comportamento, aliando um panorama dos stakeholders com desafios e questionamentos.

2. **Desenvolvimento:** nesta etapa, é colocado em prática os conceitos vistos anteriormente. Existem três etapas:

   (a) Construção da base de conhecimento: fundamentado na criação da BoK na fase de constituição do framework, uma segunda revisão literária se torna necessária para analisar aspectos técnicos;

   (b) Definição do projeto: nesta fase, é vista as caracterizações dos artefatos vistos ao longo do projeto e do BoK;

   (c) Definição de orientações de engenharia: nesta fase, os desenvolvedores podem fazer algumas pontuações sobre o projeto, tais como definições, recursos e outras questões.

3. **Avaliação:** na última fase do projeto, a autora destaca dois tópicos cujo estudo é necessário para a aprovação final do projeto.

(a) Viabilidade: é feito um estudo para verificar se a estrutura realizada corresponde aos anseios propostos na etapa de concepção do processo e se a construção do mesmo é realizável. Em consonância, pode ser agregado também a opinião da equipe de desenvolvimento. Todas estas variáveis podem colaborar para uma melhoria do framework.

(b) Observações: sendo a última etapa, é visto a possibilidade do projeto em si possuir uma interpretação prática coerente com sua revisão teórica, questão esta que depende da aplicação do recurso IoT (por exemplo, automação residencial ou mobilidade urbana).

2.2.11 Non Functional Requirement Analysis in IoT based smart traffic management system

Mahalank et al. [43] levantaram a importância do estudo e entendimento dos requisitos não funcionais para o desenvolvimento de um sistema de software em IoT. Uma detecção no início do processo de desenvolvimento desses requisitos levam o desenvolvimento do sistema a ter um melhor gerenciamento de suas limitações já no começo do seu desenvolvimento, evitando assim um retrabalho no final do projeto. Os autores afirmaram que o conceito de requisitos não funcionais não é muito bem definido entre os desenvolvedores de software. Por isso, descreveram um método para analisar e especificar os requisitos não funcionais para sistemas IoT. O método conta com três características em que os requisitos não funcionais deverão ser analisados: performance, capacidade de armazenamento e limitações para manutenção.

Mahalank et al. [43] utilizaram um sistema de tráfego inteligente como exemplo para seu método, e para isso, cada elemento do sistema IoT a ser desenvolvido é analisado e descrito a partir dessas características principais. Com essa análise, o nível de funcionamento do sistema IoT é identificado e descrito. O desenvolvimento dessas três características principais ainda pode se desdobrar em outras características menores conforme a necessidade do projeto.

Por fim, o método proposto pelos autores [43] resumem a análise dos requisitos não funcionais em três templates e uma checklist em que contém as informações necessárias sobre o requisito especificado.

2.2.12 Horizontal Requirement Engineering in Integration of Multiple IoT Use Cases of City Platform as a Service

Yamakami [44] descreveu a falta de metodologias para o design de sistemas IoT e propôs um framework para ajudar a unir diferentes casos de uso dentro de sistemas IoT. Este framework destaca os desafios que compõem a união de casos de uso e cria formas de levá-los para os times envolvidos com o projeto, utilizando diagramas UML para fins de ilustração. Além disso, faz uma análise sobre a associação da engenharia de requisitos vertical com a horizontal, e o impacto de cada uma dentro da estrutura do IoT.

O framework é sucintamente composto de 3 etapas:

1. **Identificação da interação entre casos de uso:** Responsável por identificar a relação que os casos de uso possuem antes de se fazer a união. Essa interação pode ser de:

    - **Inclusão:** Um caso de uso é incluso em outro caso de uso;
    - **Sobreposição:** Há somente uma parte de um caso de uso que é compartilhada com outro;
    - **Diferenciais ocultos:** Existem aspectos ocultos no desenvolvimento,implantação ou no operacional entre esses casos de uso;
    - **Requisitos conflitantes:** Requisitos de diferentes casos de uso possuem conflitos entre si;

2. **Identificação de coordenação:** Essa fase, é responsável por fazer a coordenação entre as diferentes fases de arquitetura do sistema a ser construído. Muitas vezes, o projeto possui diversas equipes trabalhando de forma independente, por isso, é importante garantir que que os times tenham em mente o que cada parte tem como arquitetura planejada.Se o projeto estiver bem definido, essa fase irá somente confirmar os requisitos da arquitetura projetada.

3. **Identificação de impacto:** Essa etapa lida com os impactos que a união de casos de uso podem efetuar no sistema IoT. Os autores ainda quantificam a gravidade do impacto, e se não houve nenhum impacto nessas uniões que levaram a revisões e criação de novos requisitos e, com isso, impactando diversas partes [44].

Checklists e tabelas também são utilizadas para auxiliar as etapas do framework. Elas tem objetivo de tornar mais claro o processo para o utilizador. O framework proposto por Yamakami et al. [44] apesar de estar em estágios iniciais, tem como função guiar os times de desenvolvedores que trabalham no mesmo projeto mas em áreas diferentes, a identificar

e verificar com mais facilidades os aspectos do projeto, minimizando retrabalhos e conflitos de requisitos.

2.2.13 Towards Modelling and Analysis of Spatial and Temporal Requirements

Touzani e Ponsand [45] descreveram um modelo para integração e análise de requisitos de espaço-tempo no desenvolvimento de sistemas com contexto de IoT a partir da Engenharia de Requisitos e do Sistema de Informação Geográfica (GIS). Características temporais são estudadas e analisadas com muito mais ênfase do que características espaciais, estas só vistas em sistemas mais específicos. Apesar disso, Touzani e Ponsand [45] destacaram a importância que a união das características de tempo e espaço podem ter em revelar problemas antes não vistos e tornar mais claro o sistema a ser construído.

A modelagem e integração desses requisitos é feita a partir de um método baseado em Engenharia de Requisitos Orientada por Objetivos (GORE), o conceito de Manter Todos os Objetivos Satisfeitos (KAOS) e a representação dos modelos por um diagrama UML. Com isso, são feitas as integrações dos modelos de domínio e de objetivos com características especiais e temporais. A integração de requisitos espaço temporais gera um refinamento nos requisitos gerados baseando no fato das condições serem mais precisamente representadas. Outro ponto positivo é na checagem de requisitos, pois conforme as restrições são colocadas, mais pontos de validação são criados.

A proposta realizada pelos autores [45] ainda está em fase inicial de desenvolvimento mas já foi testada em alguns cenários e foi considerada útil para a fase de descoberta e estruturação dos requisitos do projeto.

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

Silva et al. [46] criaram o termo REIoT (Engenharia de Requisitos para sistemas de software baseados em IoT) com o intuito de ser um processo capaz de elaborar uma documentação seguindo os estudos sobre os cenários propostos pela SCENARIOT (abordado por Silva [6] na Seção 2.4.1), da técnica de inspeção por checklist apontada na SCENARIOTCHECK (realizado por Souza et al. [8] na Seção 2.4.2) e templates para apontar detalhes do projeto, de modo que seja uma ferramenta capaz de auxiliar o planejamento da equipe de desenvolvimento. Utilizando as cinco etapas principais da RE, a REIoT busca reforçar estes itens para atender os recursos IoT separadamente, além disto, todas as etapas são atreladas à fase de gerenciamento, responsável por realizar o monitoramento de todo o processo.

1. **Concepção:** baseando-se em uma visão geral sobre o problema, é estudado como este imbróglio pode ser solucionado ao ser adicionado os recursos propostos na IoT e se existem recursos - humanos e tecnológicos - para a dissolução;

2. **Elicitação de requisitos em IoT:** uma vez levantado os requisitos junto aos stakeholders, os cenários propostos na SCENARIoT [6] auxiliam na identificação e na aplicação destas requisições levantadas, cabendo à equipe de desenvolvimento a separação destes requisitos sobre a utilização em IoT;

3. **Validação e Negociação (executadas em paralelo):**

   (a) Validação: todos os requisitos levantados são validados a fim de encontrar possíveis inconsistências (como erros ou ambiguidades);

   (b) Negociação: nesta fase, questões como custos (desenvolvimento, processamento e outros) são levadas em consideração quando são definidas as prioridades do sistema;

4. **Análise, Especificação e Verificação em IoT (fases que podem se repetir):**

   (a) Análise em IoT: todo o entendimento levantado ao longo deste processo - casos de uso e arranjos (ambos em IoT) e o preenchimento das informações - são o ponto de partida para a criação de um modelo de sistema. Tem consigo a capacidade de definir um diagrama de casos de uso voltado para IoT;

   (b) Especificação em IoT: com o diagrama de casos de uso gerado na fase de Análise, nesta etapa ocorre a descrição destes diagramas;

   (c) Verificação em IoT: utilizando a técnica SCENARIoTCHECK [8], esta fase realiza a checagem do documento de requisitos a fim de reforçar a sua confiabilidade. Neste caso, é validado os requisitos no contexto da IoT, diferente da fase de validação, em que apenas os requisitos normais são levantados;

Complementando as etapas acima, os autores adotaram o uso de três *templates* para este processo sobre elaboração de documentos contendo informações sobre os requisitos gerais e os voltados para a IoT. São eles:

1. **Escopo do projeto:** utilizado pelas fases iniciais do processo - elicitação, negociação e validação - este *template* tem o intuito de registrar os usuários, suas necessidades (permissões e restrições), categorização dos requisitos e outras questões levantadas;

2. **Proposta de solução:** englobando as etapas de elicitação, análise, especificação e gerenciamento, este *template* utiliza a SCENARIoT [6] para ilustrar os cenários em IoT e detalhamento sobre os Arranjos de Interação IoT (AIIs), importantes para o acompanhamento das etapas da REIoT [46];

3. **Descrição de casos de uso em IoT:** aplicado nas etapas de análise, especificação, verificação e gerenciamento, este *template* utiliza a técnica SCENARIoTCHECK [8] a partir da fase de verificação para averiguar possíveis ambiguidades nas definições estabelecidas pelos diagramas de casos de uso, bem como em outras etapas da REIoT [46];

2.2.15 IoT Composer: Composition and Deployment of IoT Applications

Krishna et al. [4] criaram uma interface Web - o IoT Composer - capaz de atender as etapas de design, composição e desenvolvimento, através de uma entrada de requisições. Voltado para usuários finais que desejam utilizar algum dispositivo IoT direcionado a *smart homes* (casas inteligentes) sem a necessidade de conhecer profundamente a área de programação, este processo mescla decisões humanas e eletrônicas, através da linguagem de especificação Nova Tecnologia LOTOS (LNT), elaborada pela Construção e Análises de Processos Distribuídos (CADP) em 2005 [47], que possui o intuito de atuar contra deadlocks e inconsistências, sendo uma importante ferramenta para a validação de requisitos no contexto da IoT. Este processo consiste de quatro etapas, conforme diagrama UML apresentado na Figura 2.10.

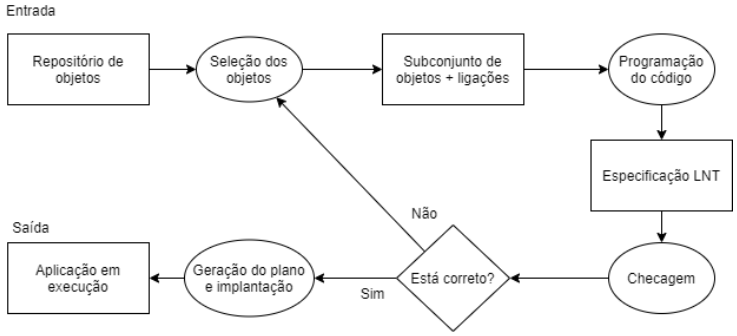


Figura 2.10: Funcionamento da ferramenta IoT Composer [4].

1. **Seleção dos objetos:** realizada por humanos, objetos são lançados no sistema web. Após isto, o usuário possui autonomia para definir os parâmetros do objeto, como ligações entre itens distintos ou relações. Nesta etapa, a fins de visão geral, é utilizado diagramas UML para fins de ilustração;

2. **Programação do código:** feito de forma eletrônica através da linguagem LNT, é gerado um código com a finalidade principal de verificar, sobretudo, a compatibilidade e outros problemas que podem acarretar uma falha na execução do dispositivo IoT;

3. **Checagem:** uma vez concluído, o LNT envia um parecer para esta fase. No caso do sistema ter encontrado alguma inconsistência, este processo retorna um erro para o usuário, caso contrário, avança para a fase posterior;

4. **Geração do plano e implantação:** tendo um parecer positivo por parte da fase de checagem, esta etapa finaliza o processo, resultando na execução da aplicação solicitada pelo usuário.

2.2.16 Conversion Method for User Experience Design Information and Software Requirement Specification

A complexidade e a utilização de múltiplos dispositivos são características presentes em sistemas IoT e tornam o uso do Design de Experiência do Usuário (UXD) cada vez mais necessário. Visando a falta de um Design voltado ao usuário, Takeda e Hatakeyama [48] apresentam um processo de especificação de requisitos em que UX e UXD estão inseridos. O processo é definido utilizando a notação em diagramas UML para facilidade de entendimento e faz a integração do UXD dividindo o processo de especificação de requisitos em três etapas.

A primeira etapa define o modelo de negócio e os recursos necessários. Logo em seguida entra a UXD com a função de entender o comportamento do usuário e arquitetar como a experiência do usuário deverá ser, para que finalmente chegue a última etapa de construção do sistema do projeto. Os autores mencionam algumas ferramentas com que a etapa de UXD pode ser feita, com o uso de entrevistas, criação de cenários , casos de uso e diagramas de atividade.

2.2.17 Key Abstractions for IoT-Oriented Software Engineering

Zambonelli [7] apresentou algumas questões chave que envolvem o conceito de Rede das Coisas (WoT) e sua aplicação prática, tendo como exemplo uma análise baseada em cenários em um hotel, visando a realização da sua operação onde os dispositivos IoT embarcados podem colaborar para tal. O autor define como questões chave todas as variáveis que são importantes em todo o processo envolvendo a criação de um dispositivo IoT. Com base nesta questão e no estudo de caso realizado em um hotel, a Tabela 2.2 apresenta os posicionamentos de Zambonelli sobre este processo [7].

Etapa	Questão chave	Descrição
Análise	Stakeholders e Usuários	Nesta fase da análise, os stakeholders e usuários são identificados, definindo para todos as suas funcionalidades, permissões, restrições, funcionalidades de um sistema e a criação de hierarquia, separados em gerência global (controlando todos os recursos IoT), gerência local (realiza o controle dos dispositivos IoT apenas em seu setor) e usuários (pessoas sem vínculo empregatício e que possui acesso limitado aos recursos IoT).
	Requisitos	Seguindo adiante no processo, é realizada uma análise completa sobre os requisitos levantados pelos stakeholders, com base nas políticas (globais, onde um diretor pode agir em todas as camadas, ou locais, que afeta apenas um setor), objetivos (metas estabelecidas para ocasiões planejadas ou não) e funcionalidades (que visa uma melhor experiência de todos os envolvidos) necessárias para uma operação com IoT. Esta última parte é ilustrada por diagramas UML.
Design	Grupos e coalisões	Todas as particularidades podem ser separadas em grupos, podendo agregar todos os atores. Para tal, o autor enfatiza a WoT, através de plataformas como RESTful e HTTP, para que todas as partes envolvidas possam acessar os recursos IoT definidos.
	Avatar	Nesta fase, é elaborado um design para cada avatar, com base nas etapas vistas anteriormente.

Desenvolvimento	Avatar	Etapa que depende das particularidades do sistema, nesta fase é implementada todos os requisitos revisados na etapa de análise, bem como outras etapas deste processo, sendo sugerido o uso de ferramentas WoT como o RESTful.
	Coisas Inteligentes	Como última etapa do processo, provê melhorias e eventuais correções de erros que podem ocorrer por alterações de firmware ou outras adversidades.

Tabela 2.2: Questões chave levantadas por Zambonelli para a elaboração de um sistema utilizando IoT [7].

2.2.18 Towards a catalog of conflicts for HCI quality characteristics in UbiComp and IoT applications: Process and first results

Carvalho et al. [5] apresentaram os eventuais conflitos que podem ocorrer entre a Computação Ubíqua (UbiComp) - termo criado em 1991 por Max Weiser, se tornando o que é hoje a Internet das Coisas - e requisitos não funcionais. Os autores afirmaram que requisitos não-funcionais conflitantes envolve critérios de satisfação de uma característica e que impacta negativamente na execução de outra característica. Já em uma harmonia, ocorre o oposto. Para solucionar este problema, os autores criaram um processo capaz de catalogar eventuais conflitos e proporcionar uma melhor experiência de sistema para os usuários finais, de acordo com sua hierarquia.

A Figura 2.11 apresenta o processo proposto pelos autores em forma de fluxograma BPMN, o qual contém cinco passos e há um armazenamento do conhecimento a partir do segundo passo até o último. Além disso, cada passo é composto de uma entrada - desde padrões a fontes literárias - e uma técnica ou processo de pesquisa [5].

1. **Seleção de uma característica de qualidade para ser analisada:** nesta etapa, é escolhida como entrada uma característica da UbiComp e a IoT - isto é, invisibilidade, consciência de contexto, mobilidade, atenção, calma e sincronismo - e que podem entrar em conflito com alguma característica dos requisitos não-funcionais. Aliado a esta seleção, também é realizada uma técnica para a análise deste conflito - que pode afetar a usabilidade - podendo ser um *survey*, entrevista ou outra técnica de validação que pode ser escolhida para este passo.

2. **Refinamento da característica de qualidade em subcaracterísticas:** a exemplo de uma árvore, uma vez selecionada, neste passo é utilizado a expansão da característica - que é a raiz do processo - em subcaracterísticas, ou ramificações. Para este passo ser realizado, deve possuir como entrada algum padrão de software voltado para o tema ou alguma pesquisa científica, aliado a uma metodologia proposta ao tema pesquisado.

3. **Identificação de métodos para a subcaracterística:** para cada subcaracterística analisada no passo anterior, é feita uma descrição destes subitens escolhidos. Para este passo, os autores recomendam o uso de técnicas para a validação - como uma entrevista ou histórias de usuários - e, assim como ocorre no segundo passo deste processo, também sugere como entrada a análise de padrões de software ou alguma fonte literária, aliado a uma teoria fundamentada.

4. **Análise de correlações:** nesta penúltima etapa, ocorre um detalhamento sobre duas características, sendo uma delas já selecionadas na primeira etapa e uma outra característica de qualidade - de escolha livre - a fim de que possam ser comparadas e que possam constatar correlações entre as mesmas. Assim como nos passos 2 e 3, deve possuir como entrada um padrão industrial ou alguma pesquisa científica para o embasamento teórico. Como um método científico para a validação, as autoras sugerem o uso de experimentação.

5. **Armazenamento de conhecimento para um catálogo:** a partir do segundo passo até o quarto, todo o conhecimento obtido nestas etapas é adicionado a um catálogo a fim de contribuir para o desenvolvimento do sistema e, consequentemente, agregando experiência para a criação de um sistema voltado para IoT.

2.2.19 Emotion-oriented requirements engineering: A case study in developing a smart home system for the elderly

Curumsing et al. [49] desenvolveram o SofiHub, um aplicativo que utiliza os recursos IoT para proporcionar conforto, praticidade e segurança para idosos, graças ao conceito de *smart homes* (casas inteligentes). Os autores também destacam a importância deste sistema para amenizar os efeitos da solidão - considerando que, em grande maioria, idosos moram sozinhos - e os cuidados com a saúde do morador, a fim de tranquilizar familiares e proporcionar um maior acolhimento do usuário, sobretudo pela crise sanitária ocasionada pelo coronavírus. Sendo orientado a emoções, o processo de desenvolvimento do SofiHub - em relação às interações - consiste em quatro passos que, através de uma técnica ou estudo

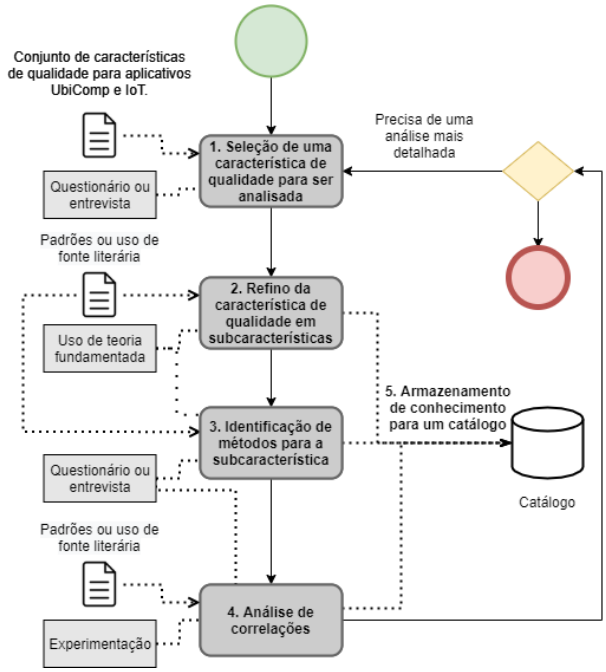


Figura 2.11: Passos do processo proposto por Carvalho et al. no contexto da UbiComp [5].

voltado para a área, chega a uma conclusão que serve de base para os passos seguintes [49]. Os passos são:

1. **Captura de requisitos:** de suma importância para a leitura das emoções humanas, o SofiHub utiliza:

    - **Modelos de funções**: capaz de moldar o perfil do usuário e de pessoas que frequentam uma casa, cuja finalidade principal é proporcionar maior conforto ao idoso nas suas rotinas cotidianas;
    - **Modelos de metas**: metas que são traçadas pelo sistema do SofiHub para prover uma melhor experiência ao usuário, fornecendo uma visão geral sobre os objetivos - funcionais, de qualidade ou emocionais - e as funcionalidades para cada uma destas questões);
    - **Cenários motivacionais**: estipula metas para usuários finais, exibindo aos usuários algumas sugestões de atividades físicas ou estipula objetivos para serem alcançados;
    - **Modelo de interação**: estabelece os atores e suas relações dentro do sistema. Neste caso, os autores estipulam as atividades destes atores através de diagramas UML, que podem utilizar o SofiHub para adicionar lembretes ou algum evento;

- **Modelos de cenário**: um cenário fornece todas as informações sobre as funcionalidades do sistema, os objetivos emocionais, como é iniciada e quem pode realizá-la e, por fim, define atividades que podem acontecer em paralelo;
- **Modelo de comportamento**: com base nos modelos de cenário, reforça os que os atores podem fazer dentro do sistema. São definidas atividades e condições preexistentes, que servem como base para um estudo sobre o comportamento dos usuários - em relação às metas - com o sistema.

2. **Modelagem de requisitos:** recebendo um modelo de emoções elaborado na primeira etapa, nesta etapa, utiliza-se a **análise de conteúdo** (técnica que utiliza palavras, simbologias, temas, imagens e outros, de modo a que possa identificar, cruzar os dados e rotular padrões) e o **diagrama de afinidade** (técnica responsável por organizar estes padrões com a finalidade de obter conhecimentos, essencial para a construindo um diagrama de baixo para cima, onde suas categorias podem ser levantadas por processo indutivo) para que todos os requisitos orientados para emoções possam ser levantados.

3. **Design e desenvolvimento:** ao receber as metas da segunda etapa, nesta etapa os autores não destacaram uma técnica voltada aos objetivos emocionais desta etapa, sendo portanto executada exclusivamente por seres humanos com base nos modelos de emoção e de metas;

4. **Avaliação:** sendo criado um sistema voltado para casa inteligente, esta última etapa visa colher um *feedback* dos usuários através de um questionário onde o usuário responde com uma escala de 0 a 5. O SofiHub utiliza diversos questionários para prover uma melhor experiência.

Com base neste processo voltado para as emoções, o SofiHub adota suas funcionalidades para uma *smart home*, cujo funcionamento se dá através de sensores aliados a inteligência artificial, que se adaptam à realidade do morador. O sistema também alerta o usuário sobre a saúde (horários de medicação, hidratação, sugestão de atividades físicas e tempo de sono), questões de segurança (como uma porta aberta ou alerta de movimentação no terreno) e utiliza atuadores para ações diversas (como ligar uma cafeteira ou proporcionar uma temperatura ambiente em situações de calor ou frio) [49].

Aliado a esta questão, o SofiHub trata sobre modelos voltados a emoções, a fim de proporcionar um conforto ao usuário para cada ameaça que possa ocorrer em sua residência, cuja solução se dá pelo conjunto de objetivos (funcionais, de qualidade e emocionais). Um dos exemplos citados pelos autores é a insegurança, que pode ser solucionada graças aos passos de detecção da anomalia (objetivo funcional), responsividade (objetivo de

qualidade) e proporcionar ao morador uma sensação de segurança (objetivo emocional) [49].

Por fim, a Figura 2.12 exibe todas as interações entre os objetivos funcionais, de qualidade ou emocionais. Esta versão é uma atualização do primeiro modelo proposto pelos autores, cujas modificações foram realizadas na emissão de notificações para o idoso, agregando mais objetivos emocionais como a segurança, o afeto - mesmo estando sozinho - e demonstrar preocupação com a saúde do idoso, bem como a adição de uma funcionalidade em seu aplicativo para mantê-lo ativo na vida do idoso, conversando de forma rotineira sobre os acontecimentos cotidianos. Com exceção destes itens, Curumsing et al. destacam os objetivos emocionais que foram atrelados ao projeto do SofiHub:

- **Tranquilizar:** tem o intuito de acalmar seus cuidadores e seus parentes sobre o bem-estar do idoso em suas atividades cotidianas;
- Ser independente: entende-se que os idosos, mesmo estando em uma idade que requer atenção, deseja realizar suas atividades sem a necessidade de serem ajudados por terceiros ou ser monitorado pelos mesmos;
- **Estar no controle:** ainda que possam estar sofrendo por limitações de locomoção, os idosos desejam ter uma vida autônoma, sem haver regulação por parte de familiares ou cuidadores;
- **Segurança:** tem o intuito de comunicar os atores - cuidadores e parentes - sobre problemas que podem ocorrer na rotina de uma casa, como porta aberta, queda do idoso ou movimentação no terreno sem o consentimento do morador;
- **Integrar:** prover ao idoso a integração de suas aplicações com o dia a dia, incluindo notícias da região, previsão do tempo ou outras informações que sejam relevantes ao usuário;
- **Sentir-se confortável:** proporcionar aos idosos uma confiança na execução de suas aplicações, preservando sua privacidade e mantendo-o integrado com a tecnologia SofiHub;
- **Estar capacitado:** análogo ao conforto, este objetivo emocional possui o intuito de proporcionar confiança no controle de suas atividades;
- **Privacidade protegida:** mesmo utilizando algoritmos de inteligência artificial para exibir notícias sobre seus gostos pessoais, o sistema tem o dever de deixar o idoso confiante sobre a proteção dos dados colhidos por seus sensores.

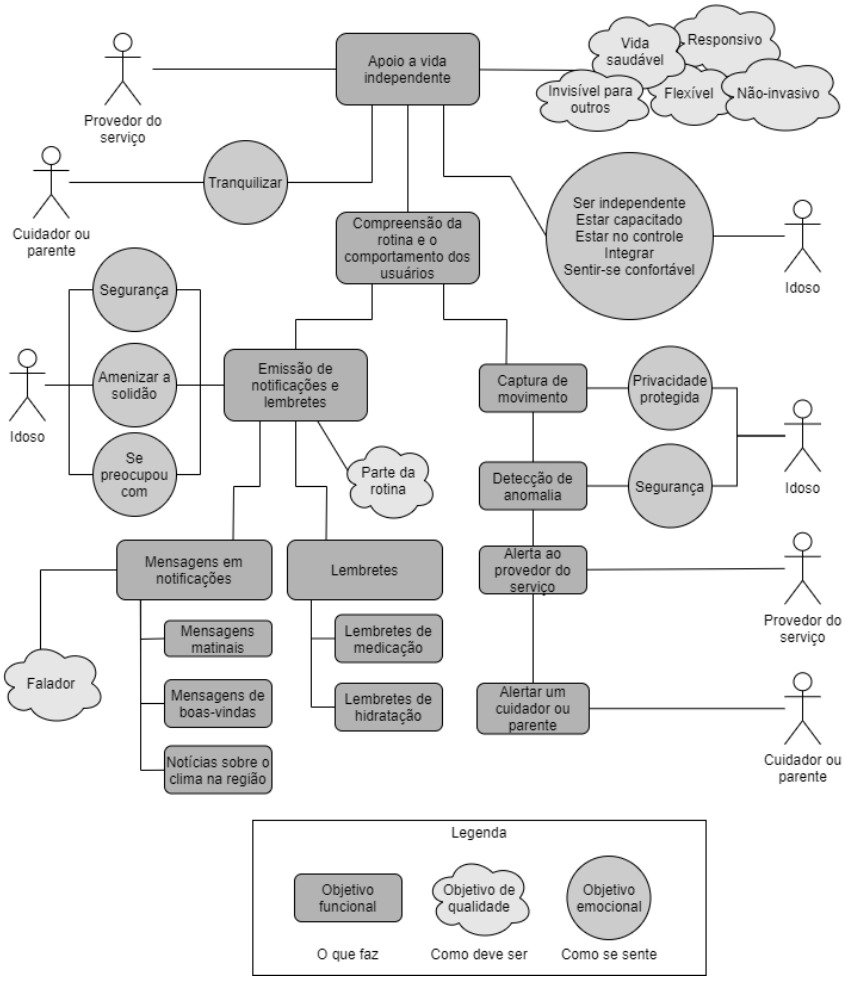


Figura 2.12: Modelo de Objetivos do SofiHub

2.2.20 IoT System Development Methods

Giray et al. [50] realizaram uma revisão dos artigos que adotam os Métodos de Desenvolvimento do Sistema (SDM) para servir como ponto de referência ao desenvolvimento de sistemas IoT e aplicou uma avaliação destes artigos abordados em relação a catorze parâmetros, desde passos do processo até o design e capacidade de manutenção. Seguindo a mesma linha, os autores destacam também o Bloco de Construção do Sistema IoT, proposto pela Aliança para a Inovação em Internet das Coisas (AIOTI) em 2016 e que descreve padrões para o funcionamento de dispositivos IoT, entre o uso de tags, sensores e atuadores, ligados a uma ação física do ser humano ou um comando dado por software.

Segundo Licorish et al. [51], o SDM envolve a criação de sistemas que seguem rigorosamente as regras e orientações propostas por stakeholders ou agentes externos, de modo que o processo é divisível em fases distintas. Além disso, propõe diretrizes a serem seguidas para a verificação de artefatos e também realiza a eliminação de quaisquer erros ocorridos na fase de desenvolvimento.

Enterprise Iot: Strategies and Best Practices for Connected Products and Services

Em seu livro, Slama et al. [52] propuseram o Ignite como um processo de alto nível baseado nas experiências dos stakeholders dentro da IoT, como gerentes de projeto, gerentes de produto, arquitetos de soluções e desenvolvedores. Este fluxo de processo busca abordar todas as questões que envolvam a IoT em um projeto, desde a concepção até o resultado final, graças ao uso de diagramas UML e diagramas de objetivos. O fluxo, dividido em dois grupos de metodologias, possuem um papel fundamental no funcionamento de um software, que são elas:

- **Execução de Estratégia em IoT**: tem o objetivo de estabelecer uma estratégia e um portfólio de projetos que agregam conteúdo para um desenvolvimento em IoT. Nesta metodologia, é vista questões que envolvem regras de negócios e oportunidades de conhecimento, servindo como base para o desenvolvimento de um protótipo e decisões sobre a operação deste recurso IoT;
- **Entrega de Solução em IoT**: ferramentas de apoio que pode ser utilizada reforçar os conceitos dos processos de design da solução e o gerenciamento de projeto, ambos envolvendo a IoT, aliado a alguns artifícios para que este suporte possa ser realizado, como checklists, desenhos e outros recursos.

Dentro da Entrega de Solução em IoT, existe um ciclo composto pelas fases de planejamento, construção e execução do sistema. Ao ser iniciado, o planejamento exige a entrega do design de solução e uma representação gráfica das iterações (podendo ser um organograma, diagrama ou outros) [50]. Seguindo adiante, estas tarefas se tornam gerenciáveis no projeto por sete fluxos de trabalho:

1. **Gerenciamento de projetos**: fluxo que agrega todas as atividades existentes dentro do processo, como o início, planejamento, execução, monitoramento, controle e encerramento;
2. **Tarefas transversais**: trabalha com fluxos de trabalho, que podem depender de outros fluxos para que a sua execução seja realizada;
3. **Infraestrutura e ferramentas utilizadas para a solução**: nesta fase, utiliza a adaptação do sistema para a execução destes fluxos de trabalho, onde são gerenciados a infraestrutura de hardware e software;
4. **Serviços utilizados em servidores**: envolvendo ferramentas de back-end, este fluxo se refere ao uso de serviços IoT que podem interagir com o usuário ou aos próprios recursos IoT, em todos os casos, são hospedados em nuvem;

5. **Serviços de comunicação**: parte do fluxo que envolve questões de infraestrutura de rede, desde a instalação até o gerenciamento;

6. **Componentes do ativo**: envolve a aquisição ou o desenvolvimento de hardware e softwares, cujo interesse é fazer a integração destes dois quesitos em um sistema IoT, através de placas de rede, torres de transmissão ou qualquer outro método para a comunicação dos componentes de hardware;

7. **Preparação do ativo**: envolve questões da criação - como a compra ou fabricação - de algum processo cuja automação se dá por IoT. Por exemplo, a energia de uma casa apenas funcionaria com um cartão RFID inserido em um *slot*.

Enabling High-level Application Development for the Internet of Things

Patel e Cassou [16] criaram o Desenvolvimento de Aplicativos IoT (IoT-AD), um processo sobre desenvolvimento de software voltado para aplicações IoT, reforçando outros projetos existentes na literatura sobre metodologias em IoT mas que abrangem áreas específicas, não existindo um detalhamento técnico na fase de implantação e sem oferecer suporte para manutenções póstumas. Para solucionar estes problemas, além da IoT-AD, os autores adotaram a IotSuite, feita por Soukaras et al. [53] para que estas modificações possam ser realizadas de modo que os stakeholders entendam o desenvolvimento em uma linguagem de alto nível.

Com o intuito de distinguir o desenvolvimento de aplicativos IoT em níveis e fornecer uma estrutura de desenvolvimento (escrita de código, tarefas a serem realizadas e técnicas), a IoT-AD dispõe de dois processos, um conceitual e outro voltado ao desenvolvimento, nos quais ambos auxiliam no fluxo de processo para o desenvolvimento de um software voltado a IoT [16]. Em seu modelo conceitual, os autores especificam quatro passos fundamentais para a área de desenvolvimento do software e dos componentes que serão utilizados pela IoT [16]. São elas:

1. **Domínio:** tem como intuito estudar o ambiente que um recurso IoT será aplicado e quais recursos podem ser automatizados pela IoT, utilizando sensores e atuadores, ambos controlado pela Entidade de Interesse (EoI), que coleta dados captados sobre a temperatura, presença ou outros parâmetros, informando o sistema e o usuário;

2. **Funcionalidades:** separado por elementos computacionais (que une os dados do sistema e criam políticas de restrições ao acesso), consiste no envio de um pedido (feito por seres humanos ou eletronicamente), que por sua vez é processado e, após, é atendido por atuadores. Nesta etapa, existe uma forte troca de mensagens (dados) e comandos entre componentes de software, bem como o uso de diagramas de classe;

3. **Implantação:** é visto as questões relativas às informações - coletadas ou executadas - por sensores e atuadores e onde estão localizados fisicamente;

4. **Plataforma:** softwares capazes de realizar a troca de mensagens entre dispositivos de hardware e softwares. É necessário configurar e gerar definições para drivers de sensores (para coletar dados), atuadores (para controle em recursos automatizados), bem como prover serviços de armazenamento para os dados coletados (banco de dados), cujo objetivo é simplificar a rotina do usuário final, auxiliando-o nas suas tarefas e se fazendo presente por meio de notificações de sistema.

Com base no modelo conceitual, os autores definiram um processo de desenvolvimento estruturado e que seguem os diagramas UML, atribuindo funções para cada stakeholder conforme suas habilidades e suas preocupações [16]. São elas:

1. **Domínio:** está relacionada com possíveis inconsistências que possam ocorrer dentro da escrita. Através da Linguagem de Vocabulário Srijan (SVL), os autores utilizam linguagem de alto nível para gerar uma estrutura de desenvolvimento para auxiliar desenvolvedores, designers de software e gerenciadores de rede;

2. **Funcionalidades:** utilizando a Linguagem de Arquitetura Srijan (SAL), uma derivação da Linguagem de Descrição de Arquitetura (ADL), responsável por gerar uma especificação de serviços computacionais e cruzar suas interações com outros componentes, os autores resolvem as problemáticas envolvendo a arquitetura do aplicativo (onde o desenvolvedor pode realizar o agrupamento de dispositivos em uma sala, de modo a agilizar a troca de dados), especificar a arquitetura de compilação (onde o desenvolvedor recebe um framework já configurado, podendo realizar poucos ajustes) e a implementação da lógica do aplicativo (onde o desenvolvedor cria um ambiente lógico no aplicativo com base no framework recebido);

3. **Implantação:** nesta etapa, faz valer a Linguagem de Implantação Srijan (SDL) para amenizar as preocupações acerca da especificação da implantação de destino e mapeamento. A especificação de implantação leva em consideração as especificações de cada dispositivo e os recursos hospedados, já o mapeamento considera regras de localização para cada serviço computacional, permitindo ao mapeador decidir onde a implantação irá ocorrer para cada dispositivo;

4. **Plataforma:** envolve as preocupações acerca dos drivers de sensores e atuadores, onde as classes concretas possuem métodos de interação com outros componentes do software. Nesta fase, o desenvolvedor do dispositivo tem a responsabilidade de desenvolver drivers que se comuniquem com o software;

5. **Linking:** através de uma infraestrutura de rede, um código - gerado por vários estágios - se tornam pacotes, que podem ser acessados por uma rede interna. Nesta etapa, estruturas de arquitetura, parte lógica, mapeamento, drivers e vocabulário são de suma importância, pois uma vez mesclados, dão suporte para a fase de implantação do aplicativo e, consequentemente, realizando um sistema de software específico, com hospedagens e proporcionando uma melhor experiência na automação;

6. **Questões voltadas a evolução:** levando em consideração a evolução tecnológica dos dispositivos IoT e em sua capacidade de integração, p framework da IoT-AD divide o desenvolvimento de aplicativos IoT em setores, a fim de proporcionar um desenvolvimento iterativo.

Towards a General Software Engineering Methodology for the Internet of Things

Zambonelli [54] apresentou a Metodologia Geral de Engenharia de Software para a Internet das Coisas (GSEM-IoT) como um processo que visa detalhar os processos de desenvolvimento de software voltados para a IoT. Apesar de não reforçar questões voltadas para a validação e não destacar diretamente as ferramentas voltadas para a manutenção, o GSEM-IoT dispõe de melhorias propostas anteriormente pelo autor a fim de englobar diversas atividades econômicas, na disponibilização de diagramas UML para um estudo aprofundado e ferramentas necessárias para que um produto chegue a sua fase final (entre produtos e alterações de software) [54]. No processo apresentado pelo autor e descrito na Tabela 2.2, baseado em um fluxo de desenvolvimento sistemático, a GSEM-IoT provê melhorias nas fases de:

1. **Análise:**

   - Reforço na identificação dos atores envolvidos no sistema, cabendo aos gestores globais as questões de gerenciamento dos usuários já vistas;
   - Reforço em sua estrutura, considerando a gradativa evolução da IoT e a necessidade de algumas companhias em englobar mais funcionalidades para garantir a sua operação. Neste caso, é recomendável fazer um estudo das características e particularidades, para que um desenvolvedor possua embasamento para a construção - ou manutenção - de um software;
   - Realização de uma elicitação de requisitos, cujos dados passam por um estudo para a identificação de objetivos, funções e políticas da companhia para

que possam ser apresentados os papéis de cada ator, podendo ser utilizado o diagrama de atores.

2. **Design:**

   - Criação de um desenho para que stakeholders possam visualizar um *design* global capaz de mostrar as funcionalidades, políticas e objetivos propostos para cada ator;
   - Possibilidade em haver novas melhorias em sua infraestrutura, podendo ser dispositivos ou serviços.

2.2.21 Stakeholder Identification and Use Case Representation for Internet-of-Things Applications in Healthcare

IoT no contexto de assistência médica vem crescendo em um ritmo acelerado nos últimos anos. Bio-sensores, dispositivos vestíveis e pesquisas relacionadas com big data transformam o ambiente médico e como os seus serviços são efetuados [55]. Laplante et al. [56] descreveram como identificar stakeholders e modelar as atividades efetuadas em uma sala de emergência de um hospital para que um sistema IoT possa ser integrado ao funcionamento da sala. Ele divide a aplicação de IoT para esse sistema em três categorias: rastreamento de pessoas, rastreamento de equipamentos e rastreamento de ambos simultaneamente.

Primeiramente, para esse método, é realizado diagramas UML para identificar todos os stakeholders, definir qual é o objetivo e os limites do sistema final. *Rich pictures* é uma técnica utilizada nessa etapa e se baseia em desenhos simples que ilustram o sistema com regras não tão rígidas. Essa técnica elicita todas os elementos presentes no sistema de uma maneira simples para que haja um amplo entendimento. Com as *rich pictures* desenvolvidas, os stakeholders são consultados para a produção e especificação dos casos de uso. *Rich pictures* e casos de uso também podem ser utilizadas para outras etapas da RE como para a verificação e validação de requisitos.

Laplante et al. [56] utilizaram essas técnicas e descreve um sistema IoT para ser implementado em uma sala de emergência. Essa sala possui rastreabilidade de pessoas, materiais e resultaria em um gerenciamento muito mais eficiente dentro de um hospital. Apesar do relato dos autores ser na área de assistência médica, o processo e a lógica aplicadas são úteis para diversas outras áreas que a IoT abrange. As técnicas utilizadas e as análises feitas podem ser utilizadas para outros projetos.

2.2.22 Intelligent Parking Management by Means of Capability Oriented Requirements Engineering

Hamdi et al. [57] citaram as características que uma cidade integrada e inteligente possuem e as vantagens que elas proporcionam para seus cidadãos. Os autores descreveram a implementação de um sistema inteligente de estacionamento capaz de ser implantado em grandes cidades e gerenciar até grandes eventos. Através do Modelo de Objetivos, os autores ilustram o sistema a partir de capacidades, diagramas de objetivos, dependência de atores, além dos relacionamentos entre esses modelos. O sistema foi feito utilizando a Engenharia de Requisitos Orientada a Capacidade (CORE), metodologia responsável pela elicitação dos requisitos.

A metodologia CORE utiliza um conjunto de entidades que são modeladas e relacionadas entre si para a descrição do sistema e seus requisitos. Capacidades e ativos do sistema são modelados,relacionados e categorizados para que requisitos/objetivos possas ser associados a eles. A metodologia CORE foi utilizada em três etapas e executadas de maneira iterativa, são elas:

1. **Elicitação de informações:** Essa fase tem como objetivo coletar informações relacionadas ao que o sistema está proposto a fazer. Por meio de diversas maneiras e técnicas, descrições textuais são captadas dos stakeholders a respeito das regras do negócio e da qualidade do sistema para que se tenha uma boa descrição do domínio da aplicação;

2. **Modelagem dos requisitos de negócios:** Fase responsável pela modelagem dos requisitos de negócios baseando-se nas informações coletadas na etapa anterior. O modelo é gerado com base no que o empreendimento é capaz, objetivos e atores presentes no sistema. O resultado desta etapa são modelos que os stakeholders possam utilizar para certificar que os requisitos foram precisos e bem explicados;

3. **Modelagem dos requisitos de sistema:** Como última etapa, ocorre a transformação dos requisitos de negócios em requisitos do sistema. O comportamento do sistema é descrito nessa etapa através das necessidades do usuário descritos nas etapas anteriores. Técnicas como diagramas de atividade e diagramas UML podem ser utilizadas para tal propósito.

2.3 Síntese dos Processos de Engenharia de Requisitos no contexto da IoT

A Tabela 2.3 apresenta uma síntese de todos os processos de engenharia de requisitos identificados na literatura para o contexto da IoT:

Seção	Nome do processo	Breve resumo
2.2.1	A Requirements Engineering Process for IoT Systems [2]	Os autores descreveram como um processo de RE no contexto de IoT deve operar e que etapas deve conter. Os autores utilizaram BPMN.
2.2.2	A UML-based Proposal for IoT System Requirements Specification [15]	Os autores utilizaram os diagramas da UML para realizar a elicitação e especificação de requisitos no contexto da IoT. Os autores não realizaram a validação em um contexto prático.
2.2.3	TrUStAPIS: A Trust Requirements Elicitation Method for IoT [29]	Os autores propuseram um método de elicitação e especificação de requisitos que utiliza a confiabilidade e outras propriedades para descrever os requisitos do sistema. Os autores utilizaram templates e diagrama UML.
2.2.4	REM4DSPL: A Requirements Engineering Method for Dynamic Software Product Lines [30]	Os autores descreveram um método para RE destinada a Linha de Produtos de Software Dinâmico (DSPL) utilizando BPMN.
2.2.5	Modeling IoT Aplications with SysML4IoT [31]	Os autores utilizaram diagramas da UML e a norma ISO/IEC/IEEE15288:2015 para criação de um método no contexto da IoT para a modelagem e design do sistema.

Seção	Nome do processo	Breve resumo
2.2.6	Specifying Functional Requirements and QoS Parameters for IoT Systems [32]	Os autores utilizaram os diagramas da UML para encontrar possíveis inconsistências nos requisitos funcionais e não funcionais.
2.2.7	Design Science and ThinkLets as as Holistic Approach to Design IoT/IoS Systems [33]	Os autores utilizaram o DSR e a ferramenta ThinkLets para a criação de uma documentação detalhada do sistema IoT, com o auxílio dos diagramas da UML e Brainstorming.
2.2.8	An Improved RE Framework for IoT-Oriented Smart Applications using Integrated Approach [37]	Os autores utilizaram os diagramas da UML e o uso de um framework para definir os processos da *smartness* e IoT.
2.2.9	Detecting IoT Applications Opportunities and Requirements Elicitation: A Design Thinking Based Approach [41]	Os autores descreveram um método para elicitação e concepção de requisitos em sistemas IoT, utilizando diagramas de afinidade. Propuseram o uso de ferramentas já existentes como IoT Design Deck, Tiles IoT TookKit (para elaboração de ideias) e IoT Service Kit para realizar a checagem com os stakeholders.
2.2.10	An Evidence-Based Framework for Supporting the Engineering of IoT Software Systems Mid-stage Research [42]	A autora adotou a metodologia 5W1H e propôs um framework de três passos com sub-rotinas, para eliminar a ocorrência de problemas futuros envolvendo a validação de requisitos em IoT.
2.2.11	Non Functional Requirement Analysis in IoT based smart traffic management system [43]	Os autores discutiram a importância dos requisitos não funcionais para o sistema IoT e utilizaram templates com a finalidade de propor o uso de um framework para a especificação de requisitos.

Seção	Nome do processo	Breve resumo
2.2.12	Horizontal Requirement Engineering in Integration of Multiple IoT Use Cases of City Platform as a Service [44]	Os autores utilizaram os diagramas da UML e propuseram um framework de três etapas para para validar os requisitos do sistema.
2.2.13	Towards Modelling and Analysis of Spatial and Temporal Requirements [45]	Os autores propuseram a inclusão, por meio de artefatos, de requisitos de espaço-tempo. Os autores utilizaram os diagramas da UML para representar os modelos de objetos do KAOS.
2.2.14	Uma Tecnologia para Apoiar a Engenharia de Requisitos de Sistemas de Software IoT [46]	Os autores elaboraram uma documentação utilizando templates e as técnicas SCENARIoT [6] e SCENARIoTCHECK [8].
2.2.15	IoT Composer: Composition and Deployment of IoT Applications	Os autores utilizaram uma interface Web para os usuários finais elaborarem sistemas inteligentes em IoT mesmo sem experiência em programação. Os autores utilizaram os diagramas da UML.
2.2.16	Conversion Method for User Experience Design Information and Software Requirement Specification [48]	Os autores propuseram um processo utilizando os diagramas da UML, Design de Experiência do Usuário (UXD) e requisitos de UX para realizar a especificação de requisitos de sistemas no contexto de IoT.
2.2.17	Key Abstractions for IoT-Oriented Software Engineering [7]	Os autores reforçaram questões chave para a elaboração de um sistema em IoT utilizando os diagramas da UML.

Seção	Nome do processo	Breve resumo
2.2.18	Towards a catalog of conflicts for HCI quality characteristics in UbiComp and IoT applications: Process and first results [5]	Os autores propuseram um processo para identificar requisitos não-funcionais. Ao concluir cada etapa é realizada uma análise documental utilizando BPMN.
2.2.19	Emotion-oriented requirements engineering: A case study in developing a smart home system for the elderly [49]	Os autores propuseram um Modelo de Objetivos para ilustrar as funcionalidades de um sistema. Os autores utilizaram questionários para realizar a validação.
2.2.20	IoT System Development Methods [50]	Os autores realizaram um estudo sobre publicações já existentes na literatura que utilizam SDM. Os autores utilizam BPMN para representar o funcionamento dos sistemas propostos.
2.2.21	Stakeholder Identification and Use Case Representation for Internet-of-Things Applications in Healthcare [56]	Os autores utilizaram os diagramas da UML e a técnica (*rich picture*), para identificar os stakeholders, os requisitos e as restrições do sistema.
2.2.22	Intelligent Parking Management by Means of Capability Oriented Requirements Engineering [57]	Os autores descreveram, por meio da Engenharia de Requisitos Orientada a Capacidade (CORE), uma metodologia para elicitação e análise de requisitos utilizando os diagramas de afinidade e diagramas da UML.

Tabela 2.3: Visão geral dos Processos de Engenharia de Requisitos no contexto da IoT

2.4 Técnicas de Validação de Requisitos no Contexto de IoT

As técnicas de validação encontradas por meio da pesquisa exploratória foram evidenciados nas subseções abaixo e intituladas com o nome do artigo publicado que contem a

técnica ou o nome que os autores deram a ela.

2.4.1 SCENARIoT: Support for scenario specification of internet of things-based software systems

A dissertação de mestrado defendida por Silva [6] pela Universidade do Rio de Janeiro (UFRJ), teve como finalidade criar uma ferramenta capaz de auxiliar a validação de requisitos no contexto de IoT. A ferramenta utiliza uma metodologia definida em modelos de objetivos, seus processos envolvem os três preceitos da IoT - identificação, sensoriamento e atuação - que tem por finalidade gerar dados, auxiliando no processo de validação. Como passo principal de todo o processo envolvendo a SCENARIoT, Silva [6] destacou o termo Arranjos de Interação em IoT (IIA), ferramenta necessária para descrever as interações entre componentes, seja com tecnologia IoT embarcada ou não.

Para avançarmos no estudo do processo desta estrutura, os passos principais que envolvem a IIA destacados na Figura 2.13 devem ser compreendidos para a escolha de uma das utilidades da IoT - atuação, identificação ou sensoriamento - e, através desta predileção, associá-la com as 29 propriedades propostas na IoT pela autora [6][2], importante para a definição da categoria do dispositivo.

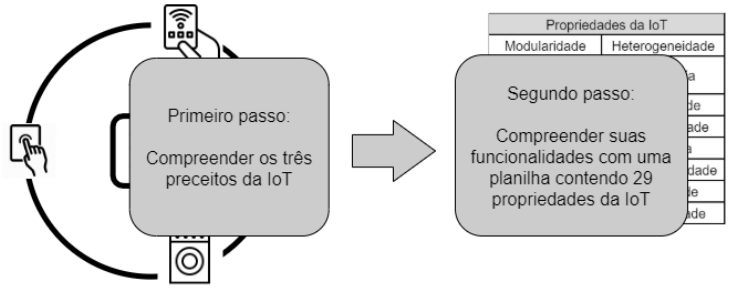


Figura 2.13: Processo de estruturação de uma IIA segundo Silva [6].

Em sua documentação, Silva destaca a finalidade de dois componentes fundamentais para o funcionamento de um dispositivo IoT [6]. São elas:

- **Produtores de Dados:** São meios existentes em dispositivos IoT - como a identificação e sensoriamento - que podem ser utilizados para a criação de dados, como registro de frequência cardíaca ou cálculo do tempo de um equipamento em funcionamento;
- **Atuadores:** São dispositivos IoT que obedecem aos comandos realizados por ação humana ou por outros dispositivos IoT interligados, através de uma interface de

[2]Acesso: `https://goo.gl/cZVVDc`

software. Como exemplo, podemos ligar lâmpadas ou definir uma temperatura para aparelhos de ar condicionado.

Cenários

Silva destacou nove cenários existentes para atuação da IoT como meio de validação dos requisitos propostos na RE [6].

1. **Produtor de dados**
   Neste cenário, elementos de identificação ou sensoriamento publicam os dados coletados para um intermediário. Por sua vez, o intermediário publica os dados para os Expositores de Dados (DE), sendo vistos pelo usuário por um Dispositivo de Interface Humana (HID). Por fim, estes dados mostrados ao usuário são retornados ao intermediário.

   (a) Desencadeado por um indivíduo
   Um indivíduo, através do HID, publica uma requisição para o intermediário, enviando uma ordem para um atuador. No final do ciclo, o atuador retorna o que realizou para a ferramenta intermediária.

   (b) Desencadeado por um software
   Nesta situação a ocorrência é similar ao cenário anterior, com a diferença de que softwares podem enviar requisições para o intermediário, criando uma ordem para o atuador. O atuador, por sua vez, retorna o que realizou durante a ordem.

   (c) Desencadeado por um indivíduo com base nos dados em IoT
   Produtores de dados (como identificador ou sensor) enviam requisições para um intermediário principal e envia as informações por um HID para o usuário. Com estes dados na tela, o usuário pode permitir ou negar uma requisição. Caso aceite, uma requisição é enviada para um segundo intermediário que ordena a execução de um atuador, o retorno deste idem os cenários anteriores. Oposto a isso, a requisição feita pelos produtores de dados retorna para o intermediário.

   (d) Desencadeado por um software com base nos dados em IoT
   Situação análoga ao cenário anterior, com a diferença que não existe um parecer do usuário para ordenar a execução de um atuador, decisão esta que é realizada pelo sistema de software.

   (e) Desencadeado por um software com base em dados não oriundos da IoT
   Neste cenário, os produtores de dados é representado por sistemas de software, que tem a responsabilidade de enviar dados para o segundo sistema de software,

cabendo a este aceitar a requisição ou recusar. Caso aceite, passa por um intermediário, que envia a ordem de atuação e espera um retorno dos dados emitidos pelo atuador.

(f) Não acionado por IoT, desencadeada por um software baseado em dados IoT
Produtores de dados, como identificadores e sensores, enviam informações até um intermediário, enviando estes dados para um sistema de software capaz de decidir se aceita ou não tal requisição. Caso aceite, a requisição é enviada para outro sistema de software que irá executar o pedido. Caso não seja aceita, a requisição retorna ao intermediário.

(g) Desencadeada por um indivíduo com base em dados não oriundos da IoT
Nesta perspectiva, os produtores de dados não-IoT - representados pelos sistemas de software - conversa constantemente com o usuário através da HID e DE, cabendo ao usuário aceitar ou não esta requisição. Caso aceite, a requisição é enviada para um intermediário, que por sua vez ordena a atuação e aguarda seu retorno para concluir a operação.

(h) Não acionado por IoT, desencadeado por um indivíduo com base em dados IoT
Situação oposta ao cenário anterior, neste caso os produtores de dados enviam suas requisições para um intermediário, enviando os dados por DE para visualização do usuário por uma HID. Uma vez aceito pelo indivíduo, há a execução de softwares não ligados a IoT, caso contrário a requisição é retornada ao intermediário.

2.4.2 SCENARI$_{oT}$CHECK: Uma Técnica de Leitura Baseada em Checklist para Verificação de Cenários IoT

A proposta apresentada por Souza et al. [8], consiste em melhorar a qualidade do processo envolvendo a SCENARI$_{oT}$ - destacado na Seção 2.4.1 - criada por Silva [6]. O autor enfatiza algumas questões-chave que podem tornar este método inviável, como ausência de uma linguagem de baixo nível, ausência de plataformas para elaboração dos *templates* por parte dos engenheiros de software e falta de aplicação prática deste processo a nível comercial.

Esta técnica envolve a utilização de checklists, para que os engenheiros de software possam verificar eventuais inconsistências antes de partir para a fase de desenvolvimento [58]. Este questionário, composto por 32 perguntas - sendo 23 gerais e 9 específicas - segue os arranjos propostos pelos Arranjos de Interação em IoT (IAA) [6]: 1) Questões gerais: é perguntado questões como domínio, passos, permissões ou restrições para cada ator,

aspectos de sistemas e outros; 2) Questões específicas: nesta fase é perguntada aspectos dos preceitos básicos da IoT, como conectividade, coisas, comportamento, interatividade, ambiente e inteligência.

A Tabela 2.4 destaca o processo criado por Souza [8] para a criação de uma checklist, realizando a inspeção sobre cada IIA proposto por Silva [6] e, posteriormente, realizando as alterações em um novo cenário.

Item de Avaliação - Questão 1	
Descrição do Item	Há riqueza de detalhes?
Resultado esperado	É esperado resultados satisfatórios
Se "não", descreva o tipo de defeito	Pode ocorrer inconsistências ou ambiguidades
Objetivo da Questão	Propor uma solução para a solicitação do usuário
Exemplo de Aplicação	
Descrição de um cenário utilizando a IAA, proposto por Silva [6]	Aplicação de um sistema de identificação por RFID nas catracas, com o intuito de registrar o ponto dos colaboradores.

Tabela 2.4: Checklist utilizada por Souza [8] para esclarecer possíveis inconsistências na SCENARIoT.

Com base na Tabela 2.4 e nos conceitos apresentados sobre a SCENARIoT [6], Souza [8] enfatiza, conforme apresentado na Figura 2.14, um método para correção de eventuais problemas que podem ocorrer durante o processo de elaboração e validação dos requisitos, resguardando os engenheiros de software de elevações nos custos para correção destas falhas.

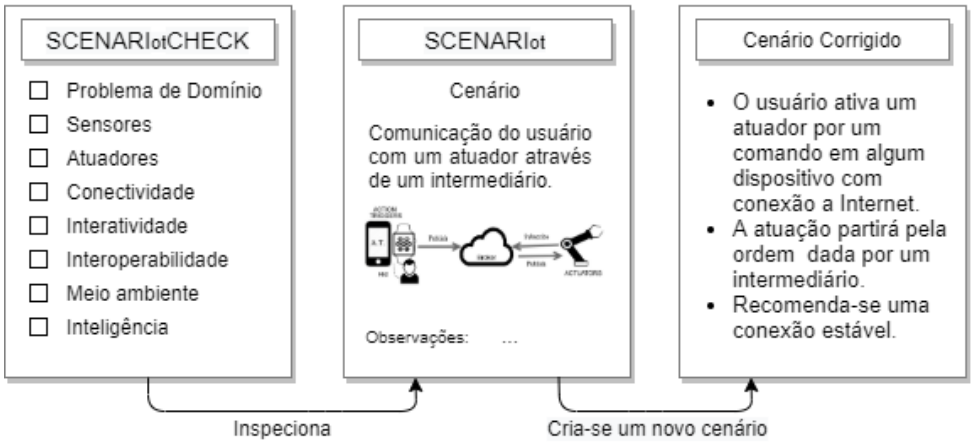


Figura 2.14: Processo da SCENARIoTCHECK com base na melhoria do processo SCENARIoT proposto por Silva [6].

2.4.3 Towards the Description and Representation of Smartness in IoT Scenarios Specification

Técnica criada por Souza et al. [9], tem como destaque o aprofundamento em relação ao termo Sistema de Software Contemporâneo (CSS), que cresceu de maneira significativa para a industrialização de países ao mesmo tempo que gerou um desafio para os engenheiros de software. Nesta técnica, os autores especificam os requisitos cujas representações são dadas por cenários, com base em *templates* e técnicas de inspeção vigentes na RE.

Trabalhando com os conceitos voltados à inteligência, como *smart device* (dispositivo inteligente) e autonomia, estas notabilidades são relevantes para todos os equipamentos com tecnologia embarcada e conexão com a Internet [9]. Para isto, Souza trabalha com o conceito de Revisão Rápida (RR), proposto por Cartaxo et al. [59], que consiste em reduzir os custos e tempo com métodos pesados/complexos, sendo portanto uma metodologia interessante para as empresas de desenvolvimento.

Este procedimento ocorreu com um grupo de pessoas, que realizou um estudo baseado no *snowballing procedures* (bola de neve), termo na literatura dado por um estudo de citações, procedimento este que parte de trás para frente. Souza et al. [9], realizaram um estudo com uma turma de dezesseis alunos, em que foram realizadas perguntas - gerais e específicas - sobre as questões do CSS, consequentemente, envolvendo a IoT e o conceito de *smartness* (inteligência), realizado após a leitura e revisão de artigos.

Pensando nas características principais envolvendo a IoT - sensores, tomadores de decisão, atuadores e inteligência artificial - Souza et al. [9] destacaram algumas técnicas existentes na RE para elicitar os requisitos, utilizando *templates*. A primeira, através de uma leitura dos artigos, é estudado os cenários de acordo as características sobre o conceito de *smartness*, conforme apresentado na Tabela 2.5. Por último, a Tabela 2.6 destaca a análise aprofundada de todos os cenários estudados na fase anterior.

Informação	**Descrição**
Ambiente	
Obtenção de dados	Definição de qual dado será capturado para apreciação, bem como seu tipo. Como exemplo, biometria, velocidade e batimento cardíaco.
Análise de dados	Uma vez colhidos, estes dados passam por uma observação para que possam ser revelados de acordo com as necessidades do usuário.

Transmissão de dados	Vencida a fase de análise, este item parte para diretrizes para que o usuário aprecie os resultados, seja para monitoramento em tempo real ou para ações complementares.
Inteligência Artificial	
Algoritmo de Inteligência Artificial	Qual conhecimento tecnológico será utilizado? (Linguagem de Máquina, Big Data, dentre outros).
Internet das Coisas	
Atuadores	O que tal dispositivo IoT deve fazer ao comando de um usuário ou sistema.
Sensores	Quais sensores estarão embarcados? (como exemplo, medidores de temperatura, presença, frequência cardíaca)
Tomadores de decisão	O que um sistema deve fazer ao ato de uma decisão tomada, realizada pelo usuário ou software.

Tabela 2.5: Elaboração de um cenário baseado em *smartness* (inteligência), segundo Souza [9].

Informação	**Descrito no cenário da 2.5?**	**Que tipo de informação foi obtida?**
Ambiente		
A obtenção de dados foi concluída?	Sim	Biometria, reconhecimento facial, frequência cardíaca.
Os dados obtidos foram analisados?	Sim	Dados da pessoa, podendo ser diversos.
Como estes dados transmitidos foram obtidos?	Sim	Através de dispositivos com transmissão em tempo real.
Inteligência Artificial		
Os algoritmos de inteligência artificial foram descritos?	Sim	Reconhecimento facial
Internet das Coisas		
Os atuadores foram identificados?	Sim	Aberturas de portas e ativação/desativação de equipamentos.

Os sensores foram identificados?	Sim	Smartwatch, leitor biométrico, presença.
Foi identificado o tomador de decisão?	Sim	Partiu de ambos (usuário e software)

Tabela 2.6: Análise do cenário produzido pela técnica [9].

2.4.4 Requirement Engineering Technique for Smart Spaces

Aziz et al. [60] propuseram esta técnica voltada a espaços inteligentes, cujo intuito é fazer com que os equipamentos troquem informações com outros dispositivos conectados através do uso de componentes embarcados como sensores e atuadores, ambos com conexão a Internet e que podem ser aplicados em diversas situações cotidianas, desde a programação de funcionamento de um ar-condicionado em um horário determinado até a localização de vagas disponíveis em um estacionamento. No entanto, a programação destes dispositivos IoT demanda uma considerável complexidade em decorrência de algumas aplicações exigirem precisão no seu uso.

Com isso, esta técnica tem como objetivo simplificar os processos vigentes na RE - isto é, elicitação, validação e verificação - e aplicá-los no contexto da IoT, concedendo uma margem para futuras alterações que possam ser necessárias ao projeto. Além disto, também é utilizado os Casos de Uso Dirigidos para representar os requisitos propostos pelos stakeholders através dos cenários que simulam o uso dos atores em um sistema, sendo portanto uma ferramenta importante para deixar esclarecido - a nível de usuário - como o software proposto irá funcionar.

Em relação ao processo, Aziz et al. [60] estabelecem passos simples. Primeiro, os stakeholders destacam seus requisitos de acordo com suas funcionalidades, podendo ser diversas (como exemplo, automação residencial, segurança e gerenciamento energético). Por último, as solicitações são programadas com o auxílio dos Casos de Uso. Todas estas partes do processo deverão ser anotados em uma tabela contendo um identificador, o título do caso de uso e um breve resumo, para que possam servir como referência para a elaboração de um caso de uso. Os autores destacam cinco passos existentes para cada caso de uso. São eles:

1. Identificar as ações levantadas pelos stakeholders, criando um caso de uso que pode envolver um ou mais atores;

2. Descrever as funcionalidades do caso de uso;

3. Identificar os atores nestes casos de uso que terão consigo a permissão de executar tais ações, podendo ser pessoas ou sistemas. Caso um ator não esteja habilitado a executar uma funcionalidade determinada, significa que não está habilitada;

4. Modelar os modelos de casos de uso voltados a dispositivos *smart space* com base na identificação dos atores, proposto na etapa anterior;

5. Gerar uma breve descrição do caso de uso, contendo o que o ator irá fazer no sistema.

2.4.5 Requirements for Testing and Validating the Industrial Internet of Things

Antão et al. [61] criaram uma técnica voltada para a área industrial, visando melhorar a qualidade de seus processos utilizando a IoT como recurso para validação de requisitos, caminho que segue em consonância com a evolução da Indústria 4.0 [11]. Os autores destacam a implementação dos Sistemas Cibernéticos de Produção Física (CPPS) como uma ferramenta de suporte para a operação industrial, separado em cinco camadas - negócio (controles de negócio), aplicação (representação gráfica), tomada de decisão (feita graças a análise de dados), redes (agregando tecnologias como WiFi, Bluetooth e 4G) e percepção (objetos físicos que coletam dados por sensores e entram em atuação quando requerido) - e que precisam trabalhar em paralelo para um correto funcionamento de todo o sistema.

Os autores consideram que a validação dos requisitos na área industrial é a força motriz para a operação nas fábricas. Assim, os autores realizaram casos de testes nos requisitos. Em oito destes, foram considerados como principais e que devem ser levados em consideração pelo CPPS, incluindo termos como o Máquina para Máquina (M2M) e plataformas Cloud. Além destes testes, foram vistos a camada da IoT utilizada, bem como seus custos (implementação, gastos de processamento e outros contratempos que influenciam negativamente na atividade) [61].

Ainda no cenário entre robô e pessoa, os cenários de casos de uso se mostram viáveis para o estudo em questões de segurança (na garantia de prevenção à invasão), no tempo (para medir a precisão do tempo de reação e, consequentemente, medir o cansaço), prevenção a acidentes (testes de estresse em máquinas para verificar se, de alguma forma, pode ferir um ser humano), critérios de recuperação (realizar o reinício do sistema e validar a possibilidade de retomar ao trabalho em um tempo preferencialmente pequeno) e a interoperabilidade (testando a capacidade da plataforma em trabalhar de forma integrada com API de terceiros, bem como acrescentar novos sensores de forma automática) [61].

2.4.6 IoT Roadmap: Support for Internet of Things Software Systems Engineering

Motta et al. [62] desenvolveu o IoT Roadmap, uma ferramenta que auxilia a definição de todos os sistemas em IoT, atribuições para a equipe de desenvolvimento e a utilização de uma checklist com três atribuições (feito, para fazer e sem aplicação), Todos estes parâmetros são ligados a um ciclo com quatro atividades, com a finalidade de agregar maior conhecimento para um projeto:

1. **Ler o roteiro**, que possui como objetivo propor uma visão geral sobre as facetas;
2. **Considerar questões** antes de seguir com o processo, como exemplo, fatores regulatórios e restrições de mercados;
3. **Executar** a estratégia escolhida para o projeto;
4. **Combinar** o roteiro planejado com outros processos em uso, caso haja necessidade.

O Iot Roadmap realiza a combinação de fases, facetas e itens, havendo uma ligação direta entre o tema atual e o seu sucessor [62]. Dividido em subrotinas, os detalhes de cada uma das três etapas são:

1. **Fases:** Composta por três fases genéricas - definição do conceito, definição do sistema e realização do sistema, incluindo hardware e software - estas subrotinas contribuem para o ciclo de vida de um produto de software IoT, cobrindo suas definições, necessidades, restrições, que contribuem para satisfação das necessidades, uma vez identificadas. As três fases propostas podem evoluir ao longo do projeto;
2. **Facetas:** Com base nas áreas de conhecimento que englobam a IoT, os autores propõe oito facetas - problema de domínio, coisas, comportamento, interatividade, conectividade, inteligência, dados e ambiente - e propõe uma checklist para cada faceta com três tipos de resposta de forma que seja possível chegar a uma conclusão na etapa seguinte;
3. **Itens:** Dentro de cada faceta, existem questões cujas respostas - feito, para fazer ou não se aplica - servem como parâmetro para a contabilização de itens realizados, para fazer ou sem aplicação, cujo cálculo é realizado de forma incremental.

Em seu artigo, Motta et al. [62] destaca a aplicação desta checklist proposta com a soma dos itens, uma vez preenchidos.

2.5 Síntese das técnicas de validação em Engenharia de Requisitos no contexto da IoT

A Tabela 2.7 apresenta as técnicas de validação de requisitos no contexto da Internet das Coisas identificadas na literatura.

Seção	Nome do processo	Técnicas usadas
2.2.11	Non Functional Requirement Analysis in IoT based smart traffic management system [43]	Realiza uma checklist para avaliação do framework.
2.2.12	Horizontal Requirement Engineering in Integration of Multiple IoT Use Cases of City Platform as a Service [44]	Utiliza *checklists* para a validação dos relacionamentos.
2.2.19	Emotion-oriented requirements engineering: A case study in developing a smart home system for the elderly [49]	Utiliza questionários para avaliar a experiência do usuário.
2.4.1	SCENARIoT: Support for scenario specification of internet of things-based software systems [6]	Utiliza fluxos iterativos para os IAAs e, em casos específicos destes, utiliza *personas*.
2.4.2	SCENARIoTCHECK: Uma Técnica de Leitura Baseada em Checklist para Verificação de Cenários IoT [8]	Propõe a utilização de checklist.
2.4.3	Towards the Description and Representation of Smartness in IoT Scenarios Specification [9]	Utiliza cenários, com base em *templates* para elicitar e validar os requisitos,
2.4.4	Requirement Engineering Technique for Smart Spaces [60]	Utiliza casos de uso.
2.4.5	Requirements for Testing and Validating the Industrial Internet of Things [61]	Utiliza casos de teste voltados para aplicações industriais.
2.4.6	IoT Roadmap: Support for Internet of Things Software Systems Engineering [62]	Utiliza uma checklist.

Tabela 2.7: Visão geral das técnicas de validação de requisitos existentes em Internet das Coisas

2.6 Síntese do Capítulo

Neste capítulo foi abordado os conceitos relacionados à Engenharia de Requisitos e Internet das Coisas. Foram apresentados os processos e técnicas de validação de requisitos no contexto da IoT. No Capítulo 3, serão propostas entrevistas a serem efetuadas com desenvolvedores e stakeholders em projetos IoT a respeito dos processos e técnicas descritos neste capítulo.

Além disso, as entrevistas procuram saber as dificuldades no desenvolvimento IoT e as opiniões de desenvolvedores a respeito da adoção de processos definidos na literatura em seus projetos.

Capítulo 3

Entrevistas

Neste trabalho foi realizado uma revisão de literatura para identificar os processos da RE no contexto da IoT e as técnicas utilizadas para realizar a validação desses requisitos. Após a identificação dos processos e técnicas, foi conduzido entrevistas semi-estruturadas com os profissionais da área de IoT para analisar como eles realizam as suas atividades diárias para elicitar e validar os requisitos, bem como se eles conhecem os processos da RE e as técnicas existentes para realizar a validação de requisitos de software, além das percepções gerais sobre esta área.

As entrevistas tiveram como objetivo estabelecer conexão entre a literatura e a realidade da área, verificando os pontos fortes e de fragilidade no desenvolvimento de projetos IoT, e que podem ser melhorados com as informações obtidas. Assim, as entrevistas foram estruturadas para responderem as seguintes questões:

- **QP.1** - Como é feito processo da engenharia de requisitos em empresas com projetos no contexto IoT?
- **QP.2** - Qual a importância dada para todo o processo de engenharia de requisitos pelos stakeholders de um projeto IoT?
- **QP.3** - Qual o conhecimento que desenvolvedores e stakeholders possuem de técnicas e processos existentes na literatura sobre RE em IoT?
- **QP.4** - Como é feita a etapa de validação de requisitos em uma empresa com projetos IoT?
- **QP.5** - Quais características do desenvolvimento IoT diferem em comparação com outros contextos de software?

Para responder a questão de pesquisa principal deste trabalho, **QP.1. Qual a percepção das equipes de desenvolvimento de software em relação a aplicabilidade**

dos processos e técnicas para realizar a validação de requisitos no contexto de IoT?, foram conduzidas 20 entrevistas com 20 profissionais da área de IoT. O roteiro da entrevista foi feito com o objetivo de responder com detalhes as questões, conhecer as opiniões dos desenvolvedores e ter uma boa ideia de como o desenvolvedor e os stakeholders de projetos IoT interagem com este contexto. A entrevista foi dividida em 3 partes:

1. **Informações demográficas do entrevistado**
    - Idade e formação;
    - Tempo de experiência em desenvolvimento de software IoT;
    - Papel que desempenha atualmente ou já desempenhou no processo de desenvolvimento de software em IoT: Analista de requisitos, desenvolvedor, testador, gerente de projetos e etc;
2. **Informações relacionadas ao processo de desenvolvimento de software em IoT**
    - Breve descrição relacionada aos projetos IoT que o entrevistado participou;
    - Conhecimento do entrevistado em relação às técnicas para elicitação e validação de requisitos. Se sim, qual técnica? Quais os desafios enfrentaram com o uso dessa técnica?
    - Procedimento para elicitação e validação dos requisitos dentro dos projetos que participou;
    - A importância dada a etapa do processo de elicitação de requisitos, especificação dos requisitos e validação dos requisitos em relação as demais etapas do projeto;
    - Conhecimento do entrevistado em relação aos processos da engenharia de requisitos no contexto IoT;
3. **Percepção do entrevistado**
    - Comparação dos desafios que o profissional enfrenta atualmente no desenvolvimento de software para IOT em comparação com o desenvolvimento de software em outros contextos, principalmente em relação a elicitação de requisitos, documentação, validação;
    - A relevância, para o entrevistado, da adoção de um processo de engenharia de requisitos nos projetos de desenvolvimento de software;
    - Importância de uma ferramenta para apoiar o profissional durante a elicitação, documentação e validação de requisitos;

3.1 Seleção dos participantes

As primeiras entrevistas foram realizadas com pessoas de círculo social próximo do entrevistador e que tiveram experiência com desenvolvimento de projetos comerciais de IoT. A medida que essas entrevistas foram sendo feitas, novas sugestões de entrevistados eram sugeridas pelos participantes e, por meio disso, foi possível estender a entrevista para outras áreas do país, passando a entrevistar pessoas de outras cidades e estados. Essa rede formada foi essencial para ser possível entrevistar pessoas de realidades diferentes e que possuem décadas de experiência na área de desenvolvimento. Outro meio utilizado para seleção de participantes foi a rede social LinkedIn. Através dela foi possível filtrar os perfis desejados e comunicá-los sobre a possibilidade de uma futura entrevista.

3.2 Processo das entrevistas

O processo durou 1 mês e meio, com 32 convites enviados e 20 entrevistas realizadas. As entrevistas foram efetuadas por meio de um software online de reuniões e tiveram duração média de 14 minutos, sendo que a mais longa durou 22 minutos. Os áudios das entrevistas foram gravadas com consentimento do entrevistado e armazenadas para futuras consultas durante a realização do trabalho.

3.3 Perfil dos entrevistados

Os 20 entrevistados são brasileiros, foram enumerados de P-1 a P-20, de acordo com a ordem cronológica das entrevistas e para facilitar a menção, quando necessário. O perfil buscado para os entrevistados sempre foi de desenvolvedores e stakeholders de projetos IoT voltados para o comércio. O entrevistado precisa possuir um nível de experiência básico nessa área.

#	Idade	Formação	Experiência IoT	Papel
1	39	Engenharia mecatrônica	8 anos	Gerente de projetos
2	24	Engenharia elétrica(graduando)	4 anos	Criador do produto
3	24	Ciência de dados(graduando)	2 anos	Gerente de projetos
4	37	Engenharia mecatrônica	10 anos	Analista de requisitos
5	25	Engenharia mecatrônica	2 anos	Gerente de projetos
6	25	Engenharia mecatrônica	3 anos	Desenvolvedor
7	23	Engenharia de computação	2 anos	Desenvolvedor
8	25	Engenharia mecatrônica	2 anos	Desenvolvedor
9	42	Ciência da computação	9 anos	Gerente de projetos
10	32	Engenharia mecatrônica	10 anos	Desenvolvedor
11	35	Engenharia elétrica	5 anos	Desenvolvedor
12	27	Engenharia de computação(graduando)	2 anos	Gerente de projetos
13	23	Engenharia de redes(graduando)	2 anos	Desenvolvedor
14	49	Sistemas de informação	9 anos	Gerente de projetos
15	47	Engenharia eletricista	25 anos	Gerente de projetos
16	32	Engenharia de computação	6 anos	Gerente de projetos
17	31	Engenharia de computação	10 anos	Desenvolvedor
18	47	Engenharia eletrônica	20 anos	Gerente de projetos
19	34	Engenharia elétrica	6 anos	Gerente de projetos
20	23	Engenharia de computação	2 anos	Desenvolvedor

Tabela 3.1: Informações demográficas dos participantes da entrevista

Capítulo 4

Resultados

Nesta seção são apresentados os resultados obtidos através da pesquisa exploratória descrita no Capitulo 2 e as entrevistas semi-estruturadas no Capítulo 3.

4.1 Pesquisa exploratória

Para esta etapa do trabalho, foram utilizados sites como DBLP, Google Scholar para a busca de artigos que abordassem sobre os processos e técnicas de validação para IoT. Diversas strings de busca foram utilizadas para abranger a maior quantidade de artigos possíveis, tornando assim o resultado mais completo.

4.1.1 Processos no contexto IoT

A pesquisa exploratória efetuada no período deste trabalho identificou no total 22 processos de engenharia de requisitos no contexto IoT, conforme apresentados e descritos pelo capítulo 2. Esses processos utilizam técnicas e metodologias únicas entre si, mas, em sua maioria, fazem uso de técnicas e métodos de representação já conhecidos e por isso foi possível classifica-los em sete categorias distintas, são elas:

1. **Diagramas:** Utilizado em mais de 50% dos processos encontrados [15], [31], [32], [33], [37], [41], [44], [4], [57], [48], [7], [56]. Diagramas UML foram os mais utilizados entre eles, possivelmente devido a sua ampla utilização e, portanto, uma maior familiaridade;

2. **BPMN**: é uma forma de construir fluxos de fácil leitura e comunicação entre equipes para montagem de processos. Utilizado em 4 dos processos encontrados [2], [30], [5], [50];

3. **Diagramas e templates**: Diagramas utilizados em conjunto com templates criados para o processo específico corresponderam por 2 dos processos encontrados [29], [46];

4. **Modelo de objetivos**: Modelo centrado nos objetivos do produto e no que ele deve alcançar. Foi utilizado em 3 dos processos [45], [49], [57];

5. **Templates**: 1 processo descrito utilizou templates como meio para efetuar o processo [43];

6. **5W1H**: Ferramenta utilizada para organizar ideias e utilizada em 1 dos processos pesquisados [42];

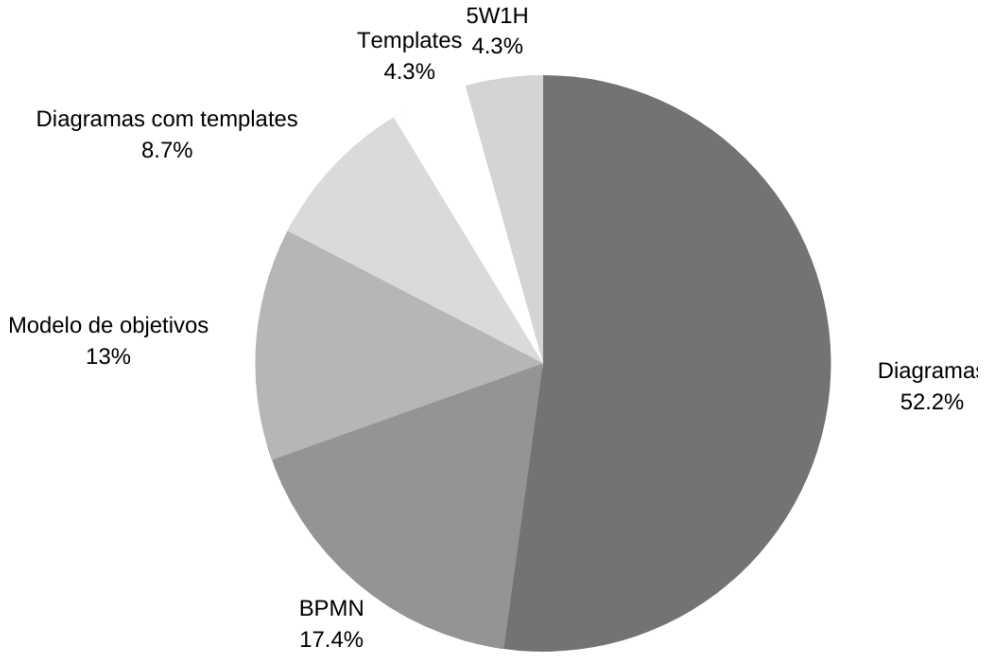


Figura 4.1: Resultado dos métodos utilizados em processos

Conforme os processos foram sendo descritos, foi possível perceber um enfoque maior nas etapas de elicitação, análise e especificação de requisitos dentro de RE. Etapas como a validação não tiveram uma abordagem direta tão clara e por isso o número de técnicas e métodos encontrados para fazer a validação de requisitos foi consideravelmente menor.

4.1.2 Técnicas de validação

A pesquisa exploratória efetuada durante a elaboração deste trabalho encontrou 9 técnicas de validação de requisitos. Foram elas:

	Fases da engenharia de requisitos							
Processos	**Concepção**	**Elicitação**	**Negociação**	**Análise**	**Especificação**	**Verificação**	**Validação**	**Gerenciamento**
2.2.1 [2]	X	X		X	X	X		X
2.2.2 [15]		X		X	X			
2.2.3 [29]		X			X			X
2.2.4 [30]		X	X	X				X
2.2.5 [31]		X			X			
2.2.6 [32]				X	X	X		
2.2.7 [33]	X	X	X	X		X		
2.2.8 [37]	X	X		X	X		X	
2.2.9 [41]	X	X	X	X	X	X	X	
2.2.10 [42]	X	X	X	X	X			
2.2.11 [43]					X			
2.2.12 [44]						X		
2.2.13 [45]				X	X	X		
2.2.14 [46]	X	X	X	X	X	X	X	X
2.2.15 [4]	X	X	X	X	X	X	X	X
2.2.16 [48]			X	X	X			
2.2.17 [7]	X	X						
2.2.18 [5]						X		
2.2.19 [49]		X		X		X		X
2.2.20 [50]	X	X	X	X	X		X	
2.2.21 [55]	X	X		X				
2.2.22 [57]		X		X				
Total	**10**	**16**	**8**	**16**	**14**	**10**	**5**	**6**

Tabela 4.1: Fases abordadas em cada processo

1. **Checklist:** foi a técnica mais adotada nos estudos identificados, e dois deles utilizaram-na como principal técnica [62], [43];

2. **Checklist e Cenário:** Checklist associado a criação de cenários foi utilizado em 1 dos estudos[8];

3. **Checklist e Casos de uso:** A utilização de casos de uso em conjunto com checklists foi adotado em 1 dos estudos para a fase de validação[44];

4. **Questionário:** Questionários foram identificados em 1 dos estudos como principal ferramenta[49];

5. **Cenário:** O uso de cenários foi feito por 1 dos estudos identificados[9];

6. **Cenários e Personas:** O uso de cenários(fluxos iterativos) e Personas foi apresentado por 1 dos estudos identificados [6];

7. **Casos de teste:** A técnica de casos de teste foi utilizado por 1 dos estudos identificados como principal maneira de validar requisitos [61];

8. **Casos de uso:** Casos de uso foi utilizado por 1 dos estudos identificados [60];

4.2 Entrevistas

Esta seção apresenta os resultados obtidos através das 20 entrevistas efetuadas com pessoas participantes do cenário de desenvolvimento no contexto IoT.

A. QP1- Como é feito o processo da engenharia de requisitos em empresas com projetos no contexto IoT?

Em todas as empresas e projetos descritos pelas entrevistas, o processo de engenharia de requisitos voltada a projetos IoT é efetuado de acordo com a demanda do projeto e sem seguir nenhum processo especificamente feito para IoT.

Os entrevistados citaram reuniões com clientes, entrevistas, análise de demandas do mercado e recepção das dores do cliente como formas de reunir requisitos e começar o desenvolvimento do projeto.

Apesar de nenhum entrevistado seguir processos de RE voltados a IoT, o P.14 adota abordagens descritas no Swebok [18] no desenvolvimento de seus projetos, P.9 estudou sobre técnicas para melhorar o desempenho na engenharia de requisitos, optando por utilizar diagramas, casos de uso e entrevistas para desenvolver os requisitos dos projetos realizados e uma melhor comunicação com o cliente.

Em projetos menores ou equipes não tão desenvolvidas notou-se uma falta de procedimento formal para a engenharia de requisitos. O P.3 faz parte de uma equipe pequena e com recursos limitados e, por isso, teme que um grande foco nesta etapa atrasaria consideravelmente um projeto que não tem muitos recursos humanos e financeiros.

A maior preocupação dos entrevistados foi com a etapa de surgimento do projeto, ou seja, as fases de elicitação, concepção e negociação. Entender como resolver a dor do cliente, ter reuniões e entrevistas com os stakeholders foram descritas pelos entrevistados como principais métodos a serem utilizados em processos de engenharia de requisitos. Essas técnicas foram informadas como utilizadas à medida que o projeto exija, mas nenhum dos entrevistados disse que o processo de engenharia de requisitos seguia um processo formal e documentado.

B. QP2- Qual a importância dada para todo o processo de engenharia de requisitos pelos stakeholders de um projeto IoT?

A engenharia de requisitos obteve níveis de importância diferentes considerando os entrevistados e o nível de complexidade do projeto desenvolvido. Todos os participantes deixaram claro que essa etapa é importante para o projeto, mas nem todos acreditam que uma preocupação maior com essa fase seria necessária nos projetos.

O P.15 relatou experiências em projetos passados que, vendo hoje em dia, teria menos complicações se houvesse uma atenção maior nesse etapa durante o desenvolvimento. Relata ainda que existiram projetos que não seguiram adiante por divergências com clientes e que poderiam ter sido evitadas caso um processo mais bem definido na área de requisitos tivesse sido efetuado.

No entanto, o P.3, por participar de uma equipe de 2 pessoas e com poucos recursos, acredita que ter um processo RE no projeto é importante, mas que não deve consumir muito tempo da equipe para não impactar no desenvolvimento do projeto.

Equipe maiores e mais experientes apesar de não adotarem processos bem definidos e voltados a IoT, admitem a importância desta etapa e destaca que ela é importante para atender expectativas do cliente e deixar a equipe toda ciente do que está sendo executado.

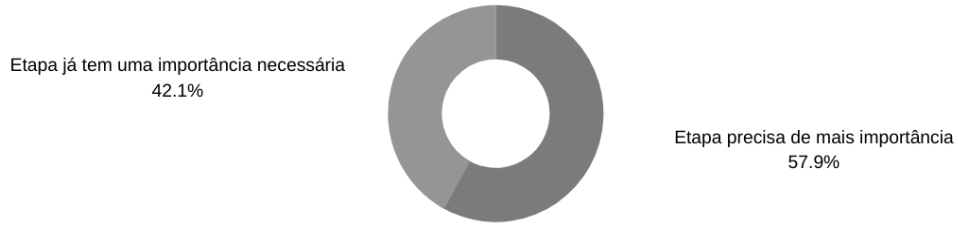


Figura 4.2: Relato dos entrevistados a respeito da importância da etapa de engenharia de requisitos

A maioria dos entrevistados descreveu que os projetos em que participaram deveria ter uma atenção maior a essa etapa. P.20 citou problemas sobre funções desenvolvidas desnecessariamente e um excesso de reuniões para discutir funcionalidades já prontas. Ele acredita que se a engenharia de requisitos fosse adotada com mais seriedade, muito tempo poderia ser economizado, independente do tamanho do projeto.

C. QP3- Qual o conhecimento que desenvolvedores e stakeholders possuem de técnicas e processos existentes na literatura sobre RE em IoT?

Ao serem perguntados sobre processos de RE destinados ao contexto IoT, a resposta foi simples e unânime de que não conheciam. Nenhum entrevistado conhecia os processos listados neste trabalho. Como dito na QP1, os que se dedicam um tempo maior a esta etapa do projeto, a executam por meio de técnicas comuns como diagramas, casos de uso, reuniões e brainstorms. Sem que estejam ligados a um processo definido pela literatura.

O P.9 relatou que já adotou alguns modelos de processos em projetos passados mas que não eram voltados especificamente a IoT. Desistiu de continuar utilizando pois a

flexibilidade que projetos IoT possuem dificultaria a implementação de um processo definido e mais engessado. Por isso, prefere flexibilizar essa etapa durante o projeto de acordo com a complexidade e a demanda que for exigida.

O P.1 tentou adotar um processo fora do contexto IoT para a equipe de desenvolvimento em que participa, mas não houve uma adesão satisfatória ou interesse por parte da empresa e da equipe. Com isso, o processo acabou sendo descartado e o que se efetua lá para a engenharia de requisitos é feito de acordo com a demanda do projeto.

D. QP4- Como é feito a etapa de validação de requisitos em uma empresa com projetos IoT?

Como visto pelas questões de pesquisa anteriores, a forma que as empresas e equipes abordam as fases de engenharia de requisitos é informal e baseada na demanda do projeto. A fase de validação não é diferente.

P12 destacou a produção de protótipos para a validação dos requisitos no ambiente em que participou. P1 descreveu a utilização de protótipos com o nome de mockups. Esses protótipos são o produto IoT ainda não finalizado, mas com capacidade de funcionamento da maioria ou de alguns requisitos e que poderiam ser instalado para testes. Mencionou, por exemplo, um projeto que gerenciava fluxo de pessoas em ambiente fechado e teve mockups instalados para verificar se seu o funcionamento atendia aos interesses dos stakeholders.

Além disso, os entrevistados ainda mencionaram reuniões com o cliente, entrevistas com stakeholders e opiniões de pessoas de fora do projeto como métodos para validação de requisitos de um projeto IoT. Nenhum dos entrevistados utilizam esses métodos em um processo definido, mas sempre conforme a necessidade e viabilidade dentro do projeto.

E. QP5- Quais características do desenvolvimento IoT diferem em comparação com outros contextos de software?

O desenvolvimento IoT possui características distintas de outros desenvolvimentos na área de tecnologia. Os entrevistados descreveram, com base em experiências anteriores, diversas comparações com o desenvolvimento em outros contextos. Os que mais se destacaram foram:

(a) **Interdisciplinaridade:** IoT envolve diversas disciplinas diferentes, não somente a área de programação. Como a IoT pode existir em qualquer ambiente e com diversas funções, faz se necessário um conhecimento de áreas diferentes. Dois entrevistados mencionaram sua importância;

(b) **Múltiplas camadas:** Seis entrevistados citaram a presença de múltiplas camadas de desenvolvimento ao produzir um produto IoT. Desde a escolha do hardware, montagem do circuito e programação, até a comunicação adotada pelo projeto com a internet e a arquitetura em nuvem do produto. Normalmente um desenvolvedor não tem conhecimento sozinho para efetuar todas essas camadas;

(c) **Falta de base acadêmica:** Falta de matérias que abordem IoT em faculdades e em cursos técnicos. Por ser uma área muito nova, programadores não tem contato com nada que envolva IoT na faculdade e saem despreparados para o mercado. Dois entrevistados mencionaram essa dificuldade;

(d) **Limitação de hardware:** Produtos IoT tendem a ser pequenos e o mais eficiente possível. Com isso, o desenvolvedor precisa de atenção constante quanto ao uso de memória e processamento do seu programa. Seis entrevistados salientaram esse tópico;

(e) **Especificidade de equipamentos:** Equipamentos de IoT são diversos e trabalham de maneira diferente e possui limitações próprias. Dois entrevistados citaram esse problema;

(f) **Falta de suporte(documentação):** Ao desenvolver projetos IoT, usualmente programadores utilizam bibliotecas recentes ou que são geridas por equipes pequenas e que não possuem estrutura para atender a demanda de dúvidas e bugs que ocorrem. Quatro entrevistados destacaram essa dificuldade;

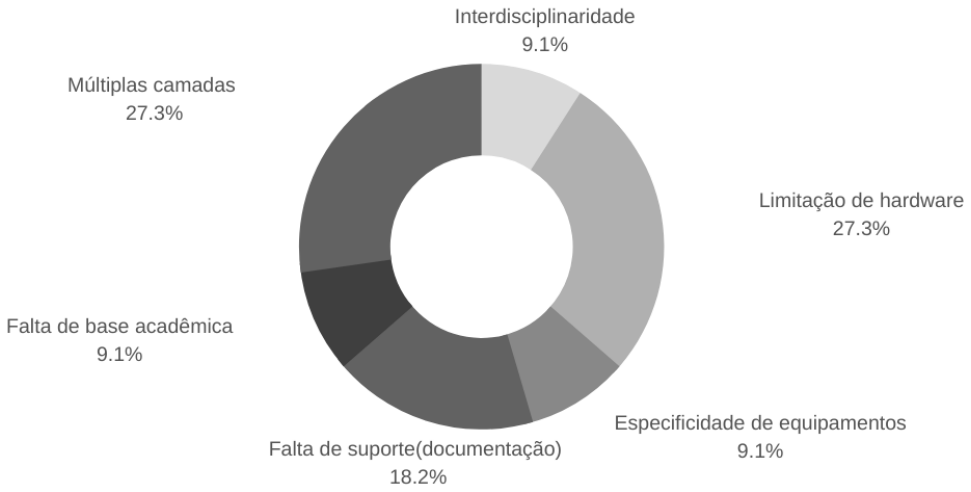


Figura 4.3: Relato dos entrevistados sobre as dificuldades em desenvolvimento de projetos IoT

4.2.1 Ameaças a Validação

Apesar do estudo contar com relatos de pessoas com os mais variados níveis de experiência, as entrevistas possuem limitações que podem ameaçar sua validação. Primeiramente, a quantidade de entrevistados que participaram da pesquisa, pois apesar das 20 pessoas serem um número considerável, alguns entrevistados possuíam a mesma experiência em IoT já que em algum momento de suas vidas, haviam trabalhado em conjunto na mesma empresa, duplicando assim alguns relatos.

Outra ameaça para a pesquisa advém do fato dos entrevistados serem em sua totalidade brasileiros. A realidade brasileira no contexto IoT pode não ser capaz de generalizar a realidade de outros países.

Capítulo 5

Conclusão e Trabalhos Futuros

O desenvolvimento de produtos no contexto IoT acompanhou o movimento de crescimento deste cenário na sociedade, mas observou-se uma escassez de material de apoio sobre engenharia de requisitos e o escasso material existente é pouco utilizado pelas equipes de desenvolvimento. Por meio deste trabalho, foi possível reunir processos e técnicas e tornar clara a importância de existir e adotar métodos durante o desenvolvimento de produtos. Além disso, este trabalho consultou pessoas da área e deixou claro as dores e motivos pela não adoção de processos definidos por parte de equipes e stakeholders.

A pesquisa exploratória apresentou os mais diversos processos e técnicas de validação. Existem processos de pequena complexidade de adoção e de maiores complexidades. Processos mais intuitivos e com curvas de aprendizado menor são ótimas saídas para empresas com pouco recurso ou tempo de aprendizado.

Com as entrevistas, foi possível perceber a realidade que desenvolvedores e empresas IoT enfrentam no desenvolvimento de seus produtos. Entrevistados com mais experiência, em geral, davam mais importância a etapa de engenharia de requisitos do que desenvolvedores e stakeholders mais jovens. IoT é uma área muito ampla e participativa. Empresas de 2 funcionários quanto de mais de 15 pessoas foram entrevistas e com elas foi possível perceber o por quê que este contexto apresenta e apresentou tanto potencial de crescimento.

O desenvolvimento de software de IoT possui diferenças que o distingue do desenvolvimento de software comum. Essas diferenças dificultam ainda mais a adoção de processos padrões utilizados no desenvolvimento de software tradicional. Além disso, a falta de adaptação ou interesse das equipes de IoT em adotar processos da engenharia de requisitos impossibilita que estes processos criados para IoT tenham uma utilização considerável na área.

Como trabalho futuro, a introdução de algum processo ou técnica descrito nesse trabalho em empresas IoT poderia ser implementado. Analisar o acolhimento da equipe e

reunir feedbacks sobre a adoção desse processo e seus efeitos tanto positivos quanto negativos. As entrevistas também podem ser expandidas para empresas de outros países para ter uma base de dados ainda mais concreta e confiável. Países mais tecnológicos e com maior renda podem gerar resultados interessantes e não visualizados nesse trabalho.

Referências

[1] Rehman, Tousif ur, Muhammad Naeem Ahmed Khan e Naveed Riaz: *Analysis of requirement engineering processes, tools/techniques and methodologies*. International Journal of Information Technology and Computer Science (IJITCS), 5(3):40, 2013. x, 7, 8, 9, 26

[2] Silva, Danyllo, Taisa Guidini Gonçalves e Ana Regina C da Rocha: *A requirements engineering process for iot systems*. Em *Proceedings of the XVIII Brazilian symposium on software quality*, páginas 204–209, 2019. x, 2, 16, 17, 18, 49, 68, 70

[3] Hevner, Alan R: *A three cycle view of design science research*. Scandinavian journal of information systems, 19(2):4, 2007. x, xi, 23, 24, 25

[4] Krishna, Ajay, Michel Le Pallec, Radu Mateescu, Ludovic Noirie e Gwen Salaün: *Iot composer: composition and deployment of iot applications*. Em Atlee, Joanne M., Tevfik Bultan e Jon Whittle (editores): *Proceedings of the 41st International Conference on Software Engineering: Companion Proceedings, ICSE 2019, Montreal, QC, Canada, May 25-31, 2019*, páginas 19–22. IEEE / ACM, 2019. `https://doi.org/10.1109/ICSE-Companion.2019.00028`. x, 34, 68, 70

[5] Carvalho, Rainara Maia, Rossana M. C. Andrade e Káthia Marçal de Oliveira: *Towards a catalog of conflicts for HCI quality characteristics in ubicomp and iot applications: Process and first results*. Em *12th International Conference on Research Challenges in Information Science, RCIS 2018, Nantes, France, May 29-31, 2018*, páginas 1–6. IEEE, 2018. `https://doi.org/10.1109/RCIS.2018.8406651`. x, 37, 39, 52, 68, 70

[6] Silva, Valeria Martins da: *Support for scenario specification of internet of things-based software systems*. UFRJ/COPPE, Rio de Janeiro, Brazil, 2019. x, 32, 33, 34, 51, 53, 54, 55, 56, 62, 70

[7] Zambonelli, Franco: *Key abstractions for iot-oriented software engineering*. IEEE Softw., 34(1):38–45, 2017. `https://doi.org/10.1109/MS.2017.3`. xi, 35, 37, 51, 68, 70

[8] Souza, Bruno Pedraça de: *SCENARIOTCHECK: UMA TÉCNICA DE LEITURA BASEADA EM CHECKLIST PARA VERIFICAÇÃO DE CENÁRIOS IOT*. Tese de Doutoramento, Universidade Federal do Rio de Janeiro, 2020. xi, 15, 32, 33, 34, 51, 55, 56, 62, 70

[9] Souza, Bruno Pedraça de, Rebeca Campos Motta e Guilherme Horta Travassos: *Towards the description and representation of smartness in iot scenarios specification.* Em Carmo Machado, Ivan do, Rodrigo Souza, Rita Suzana Pitangueira Maciel e Cláudio Sant'Anna (editores): *Proceedings of the XXXIII Brazilian Symposium on Software Engineering, SBES 2019, Salvador, Brazil, September 23-27, 2019*, páginas 511–516. ACM, 2019. https://doi.org/10.1145/3350768.3351797. xi, 57, 58, 59, 62, 70

[10] Pádua Paula Filho, Wilson de: *Engenharia de software*, volume 2. LTC, 2003. 1, 5

[11] Schwab, Klaus: *A quarta revolução industrial.* Edipro, 2019. 1, 15, 60

[12] Santos, Bruno P., Lucas A. M. Silva, Clayson S. F. S. Celes, João B. Borges Neto, Bruna S. Peres, Marcos Augusto M. Vieira, Luiz Filipe M. Vieira, Olga N. Goussevskaia e Antonio A. F. Loureiro: *Internet das coisas: da teoria à prática.* Departamento de Ciência da Computação, Universidade Federal de Minas Gerais (UFMG), Belo Horizonte, MG, Brasil, 2016. 1, 15

[13] Pal, Kamalendu e Ansar-Ul-Haque Yasar: *Convergence of internet of things and blockchain technology in managing supply chain.* J. Ubiquitous Syst. Pervasive Networks, 14(2):11–19, 2021. https://doi.org/10.5383/juspn.14.02.002. 1

[14] Motta, Rebeca Campos, Káthia Marçal de Oliveira e Guilherme Horta Travassos: *On challenges in engineering iot software systems.* J. Softw. Eng. Res. Dev., 7:5, 2019. https://doi.org/10.5753/jserd.2019.15. 1

[15] Reggio, Gianna: *A uml-based proposal for iot system requirements specification.* Em *Proceedings of the 10th international workshop on modelling in software engineering*, páginas 9–16, 2018. 1, 18, 19, 49, 68, 70

[16] Patel, Pankesh e Damien Cassou: *Enabling high-level application development for the internet of things.* J. Syst. Softw., 103:62–84, 2015. https://doi.org/10.1016/j.jss.2015.01.027. 1, 44, 45

[17] Kalinowski, Marcos e Rodrigo Oliveira Spínola: *Introdução à inspeção de software.* Revista Engenharia de Software: Qualidade de software, 1, 2008. 5

[18] Bourque, Pierre, Richard E Fairley *et al.*: *Guide to the software engineering body of knowledge (SWEBOK (R)): Version 3.0.* IEEE Computer Society Press, 2014. 5, 7, 71

[19] Ávila, Ana Luiza e Rodrigo Oliveira Spínola: *Introdução à engenharia de requisitos.* Engenharia de software Magazine, páginas 46–52, 2007. 6

[20] Vazquez, Carlos Eduardo e Guilherme Siqueira Simões: *Engenharia de Requisitos: software orientado ao negócio.* Brasport, 2016. 7

[21] Sommerville, Ian: *Engenharia de software.* PEARSON BRASIL, 2011, ISBN 9788579361081. https://books.google.com.br/books?id=H4u5ygAACAAJ. 7

[22] Shams-Ul-Arif, Q Khan e SAK Gahyyur: *Requirements engineering processes, tools/technologies, & methodologies*. International Journal of Reviews in Computing, 2(6):41–56, 2009. 7

[23] Pohl, Klaus: *Requirements engineering fundamentals: a study guide for the certified professional for requirements engineering exam-foundation level-IREB compliant*. Rocky Nook, Inc., 2016. 10, 11, 12, 13, 14

[24] Laplante, Phillip A: *Requirements engineering for software and systems*. CRC Press, 2017. 10, 11

[25] Milenkovic, Milan: *Internet of Things: Concepts and System Design*. Springer, 2020, ISBN 978-3-030-41345-3. https://doi.org/10.1007/978-3-030-41346-0. 15

[26] Lacerda, Flávia: *Arquitetura da informação pervasiva: projetos de ecossistemas de informação na internet das coisas*. Universidade de Brasília, 2015. 15

[27] Sigcha, Luis, Ignacio Pavón, Nélson Costa, Susana Costa, Miguel Gago, Pedro M. Arezes, Juan Manuel López e Guillermo de Arcas: *Automatic resting tremor assessment in parkinson's disease using smartwatches and multitask convolutional neural networks*. Sensors, 21(1):291, 2021. https://doi.org/10.3390/s21010291. 15

[28] Lim, Tek-Yong, Fang-Fang Chua e Bushra Binti Tajuddin: *Elicitation techniques for internet of things applications requirements: A systematic review*. Em *Proceedings of the VII International Conference on Network, Communication and Computing, ICNCC 2018, Taipei City, Taiwan, December 14-16, 2018*, páginas 182–188. ACM, 2018. https://doi.org/10.1145/3301326.3301360. 16

[29] Ferraris, Davide e Carmen Fernandez-Gago: *Trustapis: a trust requirements elicitation method for iot*. International Journal of Information Security, 19(1):111–127, 2020. 19, 20, 49, 69, 70

[30] Sousa, Amanda Oliveira de, Anderson G. Uchôa, Eduardo Fernandes, Carla I. M. Bezerra, José Maria Monteiro e Rossana M. C. Andrade: *REM4DSPL: A requirements engineering method for dynamic software product lines*. Em Albuquerque, Adriano Bessa e Ana Luiza Bessa de Paula Barros (editores): *Proceedings of the XVIII Brazilian Symposium on Software Quality, SBQS 2019, Fortaleza, Brazil, October 28 - November 1, 2019*, páginas 129–138. ACM, 2019. https://doi.org/10.1145/3364641.3364656. 20, 49, 68, 70

[31] Costa, Bruno, Paulo F Pires e Flávia C Delicato: *Modeling iot applications with sysml4iot*. Em *2016 42th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, páginas 157–164. IEEE, 2016. 21, 49, 68, 70

[32] Costa, Bruno, Paulo F Pires e Flávia C Delicato: *Specifying functional requirements and qos parameters for iot systems*. Em *2017 IEEE 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 (DASC/PiCom/DataCom/CyberSciTech)*, páginas 407–414. IEEE, 2017. 22, 50, 68, 70

[33] Bernsteiner, Reinhard e Stephan Schlögl: *Design science and thinklets as a holistic approach to design iot/ios systems.* Em Uden, Lorna, Wei Lu e I-Hsien Ting (editores): *Knowledge Management in Organizations - 12th International Conference, KMO 2017, Beijing, China, August 21-24, 2017, Proceedings*, volume 731 de *Communications in Computer and Information Science*, páginas 520–533. Springer, 2017. https://doi.org/10.1007/978-3-319-62698-7_43. 23, 24, 25, 26, 50, 68, 70

[34] Deng, Qi e Shaobo Ji: *A review of design science research in information systems: Concept, process, outcome, and evaluation.* Pac. Asia J. Assoc. Inf. Syst., 10(1):2, 2018. https://aisel.aisnet.org/pajais/vol10/iss1/2. 23

[35] Apiola, Mikko e Erkki Sutinen: *Design science research for learning software engineering and computational thinking: Four cases.* Comput. Appl. Eng. Educ., 29(1):83–101, 2021. https://doi.org/10.1002/cae.22291. 23

[36] Kolfschoten, Gwendolyn L. e Gert-Jan de Vreede: *A design approach for collaboration processes: A multimethod design science study in collaboration engineering.* J. Manag. Inf. Syst., 26(1):225–256, 2009. http://www.jmis-web.org/articles/552. 24

[37] Kaleem, S., S. Ahmed, F. Ullah, M. Babar, N. Sheeraz e F. Hadi: *An improved re framewrok for iot-oriented smart applications using inetgrated approach.* Em *2019 International Conference on Advances in the Emerging Computing Technologies (AECT)*, páginas 1–6, 2020. 26, 27, 50, 68, 70

[38] Caldiera, Victor R Basili1 Gianluigi e H Dieter Rombach: *The goal question metric approach.* Encyclopedia of software engineering, páginas 528–532, 1994. 26

[39] Baxter, G. e I. Sommerville: *Socio-technical systems: From design methods to systems engineering.* Interacting with Computers, 23(1):4–17, 2011. 26

[40] Jensen, Jeff C., Danica H. Chang e Edward A. Lee: *A model-based design methodology for cyber-physical systems.* Em *Proceedings of the 7th International Wireless Communications and Mobile Computing Conference, IWCMC 2011, Istanbul, Turkey, 4-8 July, 2011*, páginas 1666–1671. IEEE, 2011. https://doi.org/10.1109/IWCMC.2011.5982785. 26

[41] Dantas, Douglas Lima, Lucia Vilela Leite Filgueiras, Anarosa Alves Franco Brandão, Maria Cristina Machado Domingues e Maria Rosilene Ferreira: *Detecting iot applications opportunities and requirements elicitation: A design thinking based approach.* Em *International Conference on Human-Computer Interaction*, páginas 85–100. Springer, 2020. 27, 28, 50, 68, 70

[42] Motta, Rebeca C.: *An evidence-based framework for supporting the engineering of iot software systems.* SIGSOFT Softw. Eng. Notes, 44(3):22–23, novembro 2019, ISSN 0163-5948. https://doi-org.ez54.periodicos.capes.gov.br/10.1145/3356773.3356795. 28, 50, 69, 70

[43] Mahalank, Shubham N, Keertikumar B Malagund e RM Banakar: *Non functional requirement analysis in iot based smart traffic management system.* Em *2016 International Conference on Computing Communication Control and automation (ICCUBEA)*, páginas 1–6. IEEE, 2016. 30, 50, 62, 69, 70

[44] Yamakami, Toshihiko: *Horizontal requirement engineering in integration of multiple iot use cases of city platform as a service.* Em *2017 IEEE International Conference on Computer and Information Technology (CIT)*, páginas 292–296. IEEE, 2017. 31, 51, 62, 68, 70

[45] Touzani, Mounir e Christophe Ponsard: *Towards modelling and analysis of spatial and temporal requirements.* Em *2016 IEEE 24th International Requirements Engineering Conference (RE)*, páginas 389–394. IEEE, 2016. 32, 51, 69, 70

[46] Silva, Danyllo, Bruno Pedraça de Souza, Taisa Guidini Gonçalves e Guilherme Horta Travassos: *Uma tecnologia para apoiar a engenharia de requisitos de sistemas de software iot.* Em Ayala, Claudia P., Leonardo Murta, Daniela Soares Cruzes, Eduardo Figueiredo, Carla Silva, Jose Luis de la Vara, Breno de França, Martín Solari, Guilherme Horta Travassos e Ivan Machado (editores): *Proceedings of the XXIII Iberoamerican Conference on Software Engineering, CIbSE 2020, Curitiba, Paraná, Brazil, November 9-13, 2020*, páginas 342–355. Curran Associates, 2020. 32, 34, 51, 69, 70

[47] Garavel, Hubert, Frédéric Lang e Wendelin Serwe: *From lotos to lnt.* Em *ModelEd, TestEd, TrustEd*, páginas 3–26. Springer, 2017. 34

[48] Takeda, Ayumi e Yosuke Hatakeyama: *Conversion method for user experience design information and software requirement specification.* Em *International Conference of Design, User Experience, and Usability*, páginas 356–364. Springer, 2016. 35, 51, 68, 70

[49] Curumsing, Maheswaree Kissoon, Niroshinie Fernando, Mohamed Abdelrazek, Rajesh Vasa, Kon Mouzakis e John Grundy: *Emotion-oriented requirements engineering: A case study in developing a smart home system for the elderly.* Journal of Systems and Software, 147:215–229, 2019, ISSN 0164-1212. `https://www.sciencedirect.com/science/article/pii/S0164121218301341`. 38, 39, 40, 41, 52, 62, 69, 70

[50] Tekinerdoğan, Bedir, Eray Tüzün e G Giray: *Iot system development methods.* Em *Internet of things challenges, advances, and applications*, páginas 141–159. Chapman & Hall/CRC Press, 2018. 42, 43, 52, 68, 70

[51] Licorish, Sherlock A., Johannes Holvitie, Sami Hyrynsalmi, Ville Leppänen, Rodrigo O. Spínola, Thiago Souto Mendes, Stephen G. MacDonell e Jim Buchan: *Adoption and suitability of software development methods and practices.* CoRR, abs/2103.10653, 2021. `https://arxiv.org/abs/2103.10653`. 42

[52] Slama, Dirk, Frank Puhlmann, Jim Morrish e Rishi M Bhatnagar: *Enterprise IoT: Strategies and Best practices for connected products and services.* " O'Reilly Media, Inc.", 2015. 43

[53] Soukaras, Dimitris, Pankesh Patel, Hui Song e Sanjay Chaudhary: *Iotsuite: a toolsuite for prototyping internet of things applications.* Em *The 4th International Workshop on Computing and Networking for Internet of Things (ComNet-IoT), colocated with 16th International Conference on Distributed Computing and Networking (ICDCN)*, página 6, 2015. 44

[54] Zambonelli, Franco: *Towards a general software engineering methodology for the internet of things*. arXiv preprint arXiv:1601.05569, 2016. 46

[55] Farahani, Bahar, Farshad Firouzi e Krishnendu Chakrabarty: *Healthcare iot*. Em *Intelligent Internet of Things*, páginas 515–545. Springer, 2020. 47, 70

[56] Laplante, Nancy L, Phillip A Laplante e Jeffrey M Voas: *Stakeholder identification and use case representation for internet-of-things applications in healthcare*. IEEE systems journal, 12(2):1589–1597, 2016. 47, 52, 68

[57] Hamdi, Mohamed Salah, Adnane Ghannem, Pericles Loucopoulos, Evangelia Kavakli e Hany H. Ammar: *Intelligent parking management by means of capability oriented requirements engineering*. Em Wotawa, Franz, Gerhard Friedrich, Ingo Pill, Roxane Koitz-Hristov e Moonis Ali (editores): *Advances and Trends in Artificial Intelligence. From Theory to Practice - 32nd International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, IEA/AIE 2019, Graz, Austria, July 9-11, 2019, Proceedings*, volume 11606 de *Lecture Notes in Computer Science*, páginas 158–172. Springer, 2019. https://doi.org/10.1007/978-3-030-22999-3_15. 48, 52, 68, 69, 70

[58] Souza, Bruno Pedraça de, Rebeca Campos Motta, Daniella de O. Costa e Guilherme H. Travassos: *An iot-based scenario description inspection technique*. Em Albuquerque, Adriano Bessa e Ana Luiza Bessa de Paula Barros (editores): *Proceedings of the XVIII Brazilian Symposium on Software Quality, SBQS 2019, Fortaleza, Brazil, October 28 - November 1, 2019*, páginas 20–29. ACM, 2019. https://doi.org/10.1145/3364641.3364644. 55

[59] Cartaxo, Bruno, Gustavo Pinto e Sérgio Soares: *The role of rapid reviews in supporting decision-making in software engineering practice*. Em Rainer, Austen, Stephen G. MacDonell e Jacky W. Keung (editores): *Proceedings of the 22nd International Conference on Evaluation and Assessment in Software Engineering, EASE2018, Christchurch, New Zealand, June 28-29, 2018*, páginas 24–34. ACM, 2018. https://doi.org/10.1145/3210459.3210462. 57

[60] Aziz, Muhammad Waqar, Adil Amjad Sheikh e Emad A. Felemban: *Requirement engineering technique for smart spaces*. Em *Proceedings of the International Conference on Internet of Things and Cloud Computing, Cambridge, UK, March 22-23, 2016*, páginas 54:1–54:7. ACM, 2016. https://doi.org/10.1145/2896387.2896439. 59, 62, 70

[61] Antão, Liliana, Rui Pinto, João Reis e Gil Gonçalves: *Requirements for testing and validating the industrial internet of things*. Em *2018 IEEE International Conference on Software Testing, Verification and Validation Workshops, ICST Workshops, Västerås, Sweden, April 9-13, 2018*, páginas 110–115. IEEE Computer Society, 2018. http://doi.ieeecomputersociety.org/10.1109/ICSTW.2018.00036. 60, 62, 70

[62] Motta, Rebeca Campos, Káthia Marçal de Oliveira e Guilherme Travassos: *Iot roadmap: Support for internet of things software systems engineering*. CoRR, abs/2103.04969, 2021. https://arxiv.org/abs/2103.04969. 61, 62, 70