UNIVERSIDADE FEDERAL DE JUIZ DE FORA
INSTITUTO DE CIÊNCIAS EXATAS
PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Matheus Henrique da Silva Muniz

IoT-InterArch: uma arquitetura para apoiar a interoperabilidade em ecossistemas de IoT

Juiz de Fora
2022

Matheus Henrique da Silva Muniz

IoT-InterArch: uma arquitetura para apoiar a interoperabilidade em ecossistemas de IoT

Dissertação apresentada ao Programa de Pós-graduação em Ciência da Computação, do Instituto de Ciências Exatas da Universidade Federal de Juiz de Fora como requisito parcial a obtenção do título de Mestre em Ciência da Computação

Orientador: Prof. D.Sc José Maria Nazar David

Juiz de Fora

2022

Ficha catalográfica elaborada através do Modelo Latex do CDC da UFJF
com os dados fornecidos pelo(a) autor(a)

Muniz, Matheus Henrique da Silva Muniz.
IoT-InterArch : uma arquitetura para apoiar a interoperabilidade em ecossistemas de IoT / Matheus Henrique da Silva Muniz. – 2022.
85 f. : il.

Orientador: José Maria Nazar David
Dissertação (Mestrado) – Universidade Federal de Juiz de Fora, INSTITUTO DE CIÊNCIAS EXATAS. PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO, 2022.

1. Internet-das-Coisas. 2. IoT. 3. Ecossistemas-de-software. 4. Interoperabilidade. I. David, José Maria Nazar, orient. II. Título.

Matheus Henrique da Silva Muniz

IoT-InterArch: uma arquitetura para apoiar a interoperabilidade em ecossistemas de IoT

Dissertação apresentada ao Programa de Pós-graduação em Ciência da Computação, do Instituto de Ciências Exatas da Universidade Federal de Juiz de Fora como requisito parcial a obtenção do título de Mestre em Ciência da Computação

Aprovada em (dia) de (mês) de (ano)

BANCA EXAMINADORA

Prof. D.Sc José Maria Nazar David - Orientador
Universidade Federal de Juiz de Fora

Profa. D.Sc - Regina Maria Maciel Braga
Universidade Federal de Juiz de Fora

Profa. D.Sc - Frâncila Weidt Neiva
Centro Universitário Estácio de Juiz de Fora

“Dedico esse trabalho a Deus, minha esposa, meus pais e minha irmã”

AGRADECIMENTOS

Gostaria de expressar extrema gratidão

A Deus, a razão de todas as coisas e sem Ele nada seria possível;

A minha esposa por todo o apoio, amor e paciência no decorrer nesta fase e em todos as outras;

Ao meus pais e minha irmã pelo apoio e suporte;

Ao meu orientador por todo apoio, direcionamento e encorajamento para que eu alcançasse o melhor resultado;

A todos os meus colegas de pós-graduação por todas as conversas produtivas e pelo convívio;

E finalmente a todos que contribuíram diretamente e indiretamente a este trabalho;

“O sucesso geralmente vem para aqueles que estão ocupados demais para procurá-lo.” – Henry David Thoreau

RESUMO

A Internet das Coisas (IoT) pode ser considerada como a internet do futuro e a crescente utilização e produção de dispositivos inteligentes, provenientes de diversos fabricantes geram barreiras na comunicação para a IoT. Na literatura, vários trabalhos foram realizados para mitigar o problema de interoperabilidade, em um contexto geral e na IoT. No entanto, o foco da maioria destes trabalhos se encontra no nível sintático e semântico. Um mapeamento sistemático foi realizado no contexto desta pesquisa e foi observado que poucos são os trabalhos que focam no nível pragmático em IoT. O objetivo deste trabalho é propor uma arquitetura, denominada IoT-InterArch, para apoiar a interoperabilidade em ecossistemas de IoT levando em consideração as limitações da IoT (processamento, memória e largura de banda). Para a avaliação da arquitetura proposta, um estudo de caso foi realizado utilizando como contexto uma estufa inteligente. Foram criados cinco cenários de avaliação que tem por objetivo avaliar os atributos de qualidade de software (escalabilidade, extensibilidade e flexibilidade) a partir da adição de novos fluxos. A execução da avaliação demonstrou evidências que comprovam que a arquitetura proposta apoia a interoperabilidade em ecossistemas de IoT levando em consideração as limitações da IoT.

Palavras-chave: Internet-das-Coisas. IoT. SECO. Ecossistemas-de-software. Interoperabilidade. Pragmática.

ABSTRACT

The Internet of Things (IoT) can be considered as the internet of the future and the growing use and production of smart devices from different manufacturers generate barriers in communication for the IoT. In the literature, several works have been carried out to mitigate the interoperability problem, in a general context and in the IoT. However, the focus of most of these works is on the syntactic and semantic level. A systematic mapping was carried out in the context of this research and it was observed that there are few works that focus on the pragmatic level in IoT. The objective of this work is to propose an architecture, called IoT-InterArch, to support interoperability in IoT ecosystems taking into account IoT limitations (processing, memory and bandwidth). For the evaluation of the proposed architecture, the case study was carried out using an intelligent greenhouse as a context. Five evaluation scenarios were created to evaluate software quality attributes (scalability, extensibility and flexibility) from the addition of new flows. The execution of the assessment showed evidence that proves that the proposed architecture supports interoperability in IoT ecosystems taking into account the limitations of the IoT.

Keywords: Internet-of-Things. IoT. SECO. Software-Ecosystem. Interoperability. Pragmatic.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

WEF	*World Economic Forum*
IoT	*Internet of Things*
SOM	*Service-Oriented Middleware*
SOA	*Service-Oriented architecture*
LCIM	*The Level of Conceptual Interoperability Model*
XML	*eXtensible Markup Language*
RDF	*Resource Description Framework*
SECO	*Software Ecosystem*
SSNs	*Software Supply Networks*
MS	Mapeamento Sistemático
PICOC	*Population, Intervention, Comparison, Outcomes and Context*
EDA	*Event-driven architecture*
SOEDA	*Service-Oriented Event-Driven Architecture*
REST	*Representational State Transfer*
ESB	*Enterprise Service Bus*
FIPA-AAS	*Abstract Architecture Specifications*
FIPA-ACL	*Foundation for Intelligent Physical Agents*
RuleML	*Rule Markup Language*
W3C	*RIF Rule Interchange Format*
USDL	*Unified Service Description Language*
BPEL	*Business Process Execution Language*
HTTP	*Hypertext Transfer Protocol*
CoAP	*Constrained Application Protocol*
AMQP	*Advanced Message Queuing Protocol*
MAS	*Multi-agent Systems*
IIoT	*Industrial Internet of Things*
TbPSI	*Topic-based Publish Subscribe Infrastructure*
WSN	*Wireless Sensor Network*

SUMÁRIO

1 INTRODUÇÃO

Este capítulo tem como objetivo apresentar a motivação que inspirou a realização deste estudo, os problemas principais da área, os objetivos do trabalho e também a organização deste.

1.1 Motivação

A internet das coisas (IoT - Internet of things) pode ser considerada como um paradigma que visa interconectar objetos (dispositivos e aplicações) inteligentes em rede, de uma forma que estes possam interagir para alcançar objetivos em comum [Lee and Lee 2015][Veiga et al. 2018]. A interação destes objetos inteligentes permitirá a construção de aplicações mais complexas visando a coleta de dados produzidos e disponibilizados por meio da Web, a fim de dar um suporte às atividade diárias, seja estas industriais ou pessoais [Oliveira et al. 2016][Xia et al. 2012]. Para alguns autores, a IoT já pode ser considerada como uma das áreas mais importantes da tecnologia do futuro, visto que será uma parte vital da indústria, principalmente com o advento de novas tecnologias para apoiá-las [Saxena et al. 2017][Skouby and Lynggaard 2014].

Um estudo realizado em 2010, relatou uma nova era de onipresença, onde a interação/comunicação não será mais só entre pessoas, mas sim entre pessoas e objetos inteligentes [Tan and Wang 2010]. Tais objetos ultrapassam o número de seres humanos conectados na internet. Esta informação pode ser corroborada com o estudo realizado por [Rivera and Meulen 2014], que estima uma quantidade de 26 bilhões de dispositivos em um futuro próximo. Tal crescimento impactará setores da indústria, comércio e da saúde que fornecem alguma utilidade para qualquer ser vivo. Como por exemplo, podemos citar um ecossistema de um hospital, o qual pode abranger tanto sistemas que estão ligados de forma direta a um software de uma ambulância, ou de forma indireta, como um sistema do corpo de bombeiros, de trânsito ou sistemas de desastres naturais. Sistemas estes que juntos podem gerar um valor maior.

Em alguns setores da indústria e do comércio, a IoT já está sendo utilizada como forma de alavancagem no meio competitivo. O **Fórum econômico Mundial** (WEF - *World Economic Forum*), como forma de corroborar tal fato, realizou uma pesquisa com um conjunto de empresas [Forum 2019]. As informações expostas pela pesquisa da WEF apresentaram, de uma forma geral, uma melhoria significativa no gerenciamento das empresas estudadas. Como exemplo, podemos citar o caso da empresa "Danfoss, Commercial Compressors". As informações expostas sobre esta empresa, demonstraram que a utilização de sensores em conjunto com ferramentas de gerenciamento e monitoramento, resultaram em um aumento na produtividade do trabalho, em cerca de 30%. Além disso, em período de dois anos após a implementação das tecnologias, houve uma diminuição de

57% nas reclamações de clientes.

1.2 Problema

Contudo, a elaboração de mecanismos, que possibilitem uma comunicação entre dispositivos IoT e sistemas de gerenciamento, análise, suporte e *bussiness inteligent* de forma completa, são necessários para que o real valor da IoT possa ser alcançado. Porém, a heterogeneidade de fabricantes e padrões de comunicação, acabam possibilitando a geração de silos verticais de dados. Para que isto não aconteça, as barreiras da interoperabilidade entre IoT devem ser rompidas, para que seja plausível a elaboração de funcionalidades mais complexas, oriundas da interação entre dispositivos/aplicações IoTs. Nos **desafios de Sistemas de Informação** [Maciel et al. 2017], foi demonstrado a importância em se alcançar a interoperabilidade plena no contexto de IoT. No contexto desse estudo, a interoperabilidade pode ser definida como a habilidade em trocar informações entre pelo menos dois objetos inteligentes, a fim de gerar novas funcionalidades [Rezaei et al. 2014].

Visando abstrair a interoperabilidade plena em camadas, um modelo com cinco níveis de interoperabilidade foi descrito por [Tolk and Muguira 2003], estes são: técnico, sintático, semântico, pragmático/dinâmico e organizacional/conceitual. Dentre estes, os quatro primeiros podem ser ditos como os já alcançáveis. Porém a maioria dos estudos sobre a interoperabilidade, de forma geral e em IoT, focam nos níveis sintáticos e semântico [Maciel et al. 2017]. No entanto, como relatado por [Asuncion and Sinderen 2010], a descrição somente do formato (sintático) e do significado (semântico), não são suficientes quando o assunto é a colaboração entre aplicações.

Com base nessas informações, atingir o nível pragmático para IoT é de suma importância, pois além de expor as intenções dos dispositivos na rede, impacta de forma indireta na segurança da rede e também possibilita uma interação entre dispositivos/aplicações somente em determinadas situações e contextos. Como exemplo, podemos citar a utilização de *smart objects* em um ambiente hospitalar, onde estes monitoram a saúde dos pacientes deste ambiente ou em outro que esteja vinculado a este. A partir dos dados que são produzidos, ações podem ser tomadas. Como por exemplo acionar algum responsável do ambiente ou ambulância, caso o paciente esteja tendo algum tipo de crise e não possua alguém por perto. Tal ação é possível, através de regras e intenções sobre dados provenientes de um ou mais dispositivos/aplicações. Além disso, informações de contexto devem ser consideradas como essenciais para a tomada de decisão.

No estudo secundário realizado por [Asuncion and Sinderen 2010], foram relatadas várias definições para interoperabilidade pragmática. Para esse estudo, a interoperabilidade pragmática será definida como a intenção sobre um dado produzido levando em consideração o seu significado [Bravo and Alvarado 2008]. Isto implica que a interação no nível pragmático deve acontecer se a intenção esperada, tanto pelo produtor, quanto pelo

consumidor for a mesma.

1.3 Objetivo

Este trabalho tem por objetivo apoiar a interoperabilidade entre dispositivos/aplicações IoTs através da elaboração de uma arquitetura que leve em consideração as questões de escalabilidade, extensibilidade, flexibilidade e também as limitações de dispositivos IoTs, como memória, processamento e largura de banda. A arquitetura citada, denominada como IoT-InterArch, foi elaborada também para apoiar a interoperabilidade sintática, semântica e pragmática, de forma separada ou em conjunto, em ecossistemas externos ao que foi desenvolvido.

Com o objetivo de encontrar pesquisadores e trabalhos que tratem a interoperabilidade pragmática entre dispositivos IoT, um mapeamento sistemático da literatura foi realizado [Muniz et al. 2019]. No entanto, dentro dos estudos primários identificados, a maioria desses somente propuseram **conceitos** e **modelos** para tratar a interoperabilidade pragmática em IoT.

A arquitetura construída teve como foco:

a) Apoiar a interoperabilidade sintática através da conversão de unidades básicas e também da utilização de um message broker capaz de apoiar a escalabilidade.

b) Apoiar a interoperabilidade semântica através da utilização de informações semânticas de base de dados externas e também através da utilização de cálculos de similaridade entre sentenças.

c) Apoiar a interoperabilidade pragmática através da construção de um módulo que permita a execução de regras em cima de dados que são produzidos pelos dispositivos/aplicações.

d) Apoiar a interoperabilidade pragmática através da utilização da inteligência artificial (*Machine learning*) para possibilitar a inferência de novas informações e assim possibilitar a mudança na intenção dos dados, mediante a mudança de contexto.

e) Apoiar a construção de aplicações IoT, através de uma plataforma que permita a descrição de dispositivos/aplicações.

f) Construção de ontologias para descrever as informações geradas pelo serviço e assim inferir novas informações.

g) Apoiar os três níveis de interoperabilidade atacados neste trabalho em um ambiente externo, através da construção de endpoints que utilizem as informações armazenadas/inferidas pelas ontologias e pelo módulo de IA.

1.4 Organização

O restante do trabalho está organizado como se segue. O capítulo 2 é apresentado os conceitos necessários para esse trabalho, assim como o resultado do mapeamento sistemático e os trabalhos relacionados. O capítulo 3 apresenta a solução proposta para apoiar a interoperabilidade sintática, semântica e pragmática em IoT. O capítulo 4 apresenta a avaliação da solução, sendo apresentando o seu planejamento, execução e os resultados. O capítulo 5 apresenta as considerações finais do trabalho, tendo como objetivo expor as contribuições dos estudos, as limitações identificadas e trabalhos futuros.

2 REFERENCIAL TEÓRICO

Neste capítulo são apresentados os conceitos fundamentais relacionados à proposta do trabalho. Informações sobre a Internet das Coisas, Interoperabilidade, Ontologias e Ecossistemas de software são apresentadas de forma a embasar a proposta. Além disso, o mapeamento sistemático [Muniz et al. 2019] é apresentado, assim como os principais trabalhos relacionados identificados através deste estudo secundário.

2.1 Internet das coisas

A Internet das coisas (IoT) é um paradigma de tecnologia que visa interconectar dispositivos e máquinas em um ambiente, a fim que estes possam interagir entre si [Lee and Lee 2015]. A IoT possibilita uma série de benefícios ao abstrair a complexidade das coisas, ao mesmo tempo que possibilita a construção de aplicações mais complexas através da coleta, processamento e da análise dos dados. Como exemplo, podemos citar a construção de sistemas de suporte à decisão [Krytska, Skarga-Bandurova and Velykzhanin 2017], gerenciadores de fazendas [Dhivya, Parameswaran et al. 2020] e sistemas *Health-Care* distribuídos [Koop et al. 2008] .

Em relação às áreas de atuação, muitos podem ser destacados. Dentre esses, podemos citar alguns exemplos de uso como, da indústria (*Smart industry*) dentro da chamada indústria 4.0, na área da saúde (*Smart Health*) auxiliando médicos a monitorar os seus pacientes, em construções (*Smart building*) monitorando os diversos aparelhos presentes neste [Wortmann and Flüchter 2015]. Vale ressaltar que todos os domínios podem ser utilizados em conjunto, como é o que acontece no domínio de Smart City. A internet das coisas pode ser considerada também como um ecossistema contendo vários dispositivos que colaboram entre si.

Mas para que isto aconteça, muitos autores argumentam sobre a necessidade em abstrair os softwares de dispositivos [Brezolin et al. 2018], visto que estes possuem limitações físicas. Dispositivos que possuem este tipo de limitação, devem ser capazes de somente produzir dados de forma bruta (sem tratamento), como por exemplo, ações, temperatura, umidade, pressão e transmitir estas informações para quem queria usufruir destas [Rahmani et al. 2015]. Os dados produzidos por estes dispositivos IoT, tendem a ser simples devido a limitação de banda presente nestes dispositivos e as aplicações que as consomem tendem a ser mais complexas.

Visando resolver tais limitações, uma gama de plataformas Middleware orientadas a serviços (*Service-oriented Middleware - SOM*) foram propostas para a IoT, levando em consideração os conceitos do paradigma da arquitetura orientada a serviços (*service-oriented architecture -SOA*), como o conceito de abstração de serviços [Issarny et al. 2016]. A escolha na utilização de SOM é baseada no fato da abstração, visto que serviços originados

desta arquitetura, trazem boa parte do processamento para servidores. A partir desta, outras foram surgindo visando resolver os problemas da IoT relacionados a limitações físicas [Kjær 2007][Issarny et al. 2016].

No entanto, devido a heterogeneidade de fabricantes, uma variedade de padrões de comunicação começaram a surgir, possibilitando a geração de silos verticais de dados. Os middlewares de forma isolada, só eram capazes de resolver tais problemas de interoperabilidade em um nível muito superficial.

Para que então a IoT possa alcançar o seu real valor, as questões de interoperabilidade devem ser tratadas em níveis mais altos. Uma gama de trabalhos surgiu com o objetivo de tratar estas questões em diferentes níveis de interoperabilidade em IoT. A interoperabilidade nos diferentes níveis e os trabalhos que a abordam dentro do contexto da IoT, serão apresentados nas próximas seções.

2.2 Interoperabilidade

Para esse trabalho, a interoperabilidade entre dispositivos, pode ser definida com a habilidade em estabelecer uma comunicação entre dois ou mais dispositivos de uma forma completa e independente a fim de gerar funcionalidades mais complexas através da colaboração entre os dispositivos[Rezaei et al. 2014]. Para que a comunicação possa ser independente, tais dispositivos devem possuir um baixo acoplamento com a rede e devem ser de fácil substituição. A interoperabilidade pode ocorrer em vários níveis. Com a intenção de alcançar uma maior compreensão sobre o assunto, Tolk e Muguira (2003) desenvolveram um modelo baseado em camadas para alcançar este objetivo, o LCIM [Tolk and Muguira 2003]. Os quatro níveis que já são alcançáveis, serão descritos nas subseções a seguir.

2.2.1 Nível 0 - Técnico - *System Specific Data*

O nível 0 está relacionado aos dispositivos mais proprietários, onde os dados são vistos como recursos deste e não possui uma cultura organizacional que os dados podem ser compartilhados. Muitas das vezes, dispositivos que estão neste nível, possuem o formato do seus dados incorporado no próprio código como é o caso de sistemas *legacy*, em arquivos locais como planilhas ou em banco de dados. Nos dois últimos casos não existe uma documentação do significado dos dados.

2.2.2 Nível 1 – Sintático - *Documented Data*

O nível 1 tem o propósito de resolver o problema citado no nível anterior, que nada mais é do que a documentação do formato dos dados. A partir disso, dispositivos que possuem esta documentação, além terem um melhor gerenciamento dos seus dados, podem

ser utilizados como recursos externos por aplicações, através de camadas de mapeamento para interconectar os dados. Este nível também é conhecido como interoperabilidade sintática.

2.2.3 Nível 2 - Semântico - *Aligned Static Data*

O nível 2 foca no alinhamento dos dados e dos modelos de objetos, o qual muitos trabalhos da área acreditam ser algo necessário para alcançar a interoperabilidade. Sendo isto possível, através do uso de ontologias e modelos de referência comuns para padronizar os elementos dos dados e os seus significados [Falbo et al. 2004]. Mesmo sendo algo necessário não é suficiente, visto que dos quatros conflitos existentes definidos por [Spaccapietra, Parent and Dupont 1992] para agrupar diferentes fontes de dados, um dos conflitos não é totalmente resolvido.

Dos conflitos citado acima, o primeiro é o de descrição, pois para os mesmos conceitos podem existir homônimos e sinônimos. O segundo é o de heterogeneidade, pois diferentes substancialidades são descritas por diferentes metodologias. O terceiro é o estrutural, visto que diferentes estruturas podem ser empregadas para descrever o mesmo conceito. E o último é o semântico, o qual este nível não consegue resolver completamente, pois considera que todo mundo tem a mesma coisa em mente quando conversa sobre algo. Este nível é também conhecido como interoperabilidade semântica.

2.2.4 Nível 3 - Pragmático/Dinâmico – *Aligned Dynamic Data*

O nível 3 está relacionado com o comportamento de sistemas, mais especificamente, o comportamento dinâmico. Este comportamento está vinculado a mudança de estado/intenção sobre os dispositivos em tempo de execução. No nível semântico somente o comportamento estático (sistema, subsistema e componentes desses) e o funcional (interface e fluxo de dados) são documentados. Então para alcançar o terceiro nível, é necessário documentar o uso dos dados nos dispositivos e também o resultado da mudança de estado/intenção.

De acordo com [Maciel et al. 2017], a maioria dos trabalhos, tanto de forma geral, quanto no domínio de IoT, trazem soluções para os níveis sintático e semântico.

2.3 Ontologias

Para que a interoperabilidade fosse apoiada por meio do compartilhamento de informações entre vários sistemas, tornou-se necessário a criação de ferramentas para estabelecer um vocabulário comum entre diversos sistemas. Uma das mais difundidas é a Web Semântica.

A **web semântica** pode ser considerada como uma extensão da internet atual, pois por meio dela computadores e máquinas podem comunicar entre si através de documentos semânticos [Pickler 2007]. Tais documentos são estruturados através de marcações como tags, para o fácil entendimento das máquinas que irão ler esses arquivos [Berners-Lee, Hendler and Lassila 2001] [Gruber 1995]. Como por exemplo, podemos citar o exemplo de uma *Smart City* que possui diversos dispositivos, cada um possuindo um vocabulário próprio, o que impossibilita a interoperabilidade plena sem um documento que traga um relacionamento entre os vocabulários.

Vários documentos surgiram com a intenção de descrever informações sintáticas e semânticas, como documentos XML e RDF. Contudo o poder se expressão destes documentos é muito fraco. Com isso surgiram as ontologias como a terceira parte da internet, com objetivo de dar poder às máquinas de fazer comparações entre dados e verificar se esses são semelhantes. A ontologia pode ser definida, dentro da computação, como um documento ou um arquivo que expressa formalmente relacionamentos entre termos em formas de taxonomias e regras de inferências (na maioria dos casos) [Berners-Lee, Hendler and Lassila 2001].

Como exemplo de taxonomia, seria uma taxonomia que contenha em sua hierarquia as classes “País”, “Estado”, “Cidade” e por meio de uma simples inferência seria identificado que uma cidade pertence a um determinado estado. Tais inferências podem ainda descobrir conhecimento implícito graças as *properties chains*. Por meio dessas, informações que não foram descritas podem gerar novos conhecimentos em cima das que foram. Utilizando ainda o exemplo das da taxonomia de país, estado e cidade, pode-se fazer a inferência que se uma cidade pertence a um estado e este pertence a um país, então essa cidade pertence a esse país [Rees 2003] [Berners-Lee, Hendler and Lassila 2001].

Basicamente a ontologia pode ser considerada com uma especificação explícita de conceitos [Gruber 1995], onde novas informações podem ser inferidas. A ontologia utiliza-se de classes (relacionamento unário) para descrever elementos de um domínio, enquanto que utiliza de propriedades (relacionamento binário) para a relação entre elementos [Doan, Halevy and Ives 2012].

As ontologias também podem ser utilizadas para representar a parte pragmática de forma parcial, pois considerando o comportamento dinâmico de dispositivos/aplicações, ferramentas adicionais são necessárias. Principalmente em relação à mudança na intenção de dados.

2.4 Ecossistema de Software

A construção de software nos últimos anos está passando por uma grande evolução, visto que a demanda por softwares customizados tende só a aumentar. Tal fato acarreta também no aumento da complexidade dos softwares, no que tange a construção e manuten-

ção de código. A exigência da parte de clientes, na elaboração de funcionalidade em curto espaço de tempo, tornaram as fábricas de softwares ineficientes. Com o objetivo de serem mais competitivas, as empresas começaram a olhar para software de terceiros provenientes de outras empresas ou de desenvolvedores [Berk, Jansen and Luinenburg 2010]. Esta mudança, possibilitou a construção de softwares modulares, através da colaboração entre várias empresas inseridas em uma mesma rede. O paradigma gerado por esta mudança foi nomeado com ecossistemas de software (Software Ecosystem - SECO).

SECO foi definido por [Bosch 2009] como um conjunto de serviços e produtos que permitem a automatização de atividades e transações entre um ecossistema de negócios (ou um ambiente social) e organizações que fornecem esses conjuntos de soluções. No caso de empresas, estas podem ser consideradas como um ecossistema se a sua plataforma for disponibilizada além dos limites organizacionais. Uma outra definição para ecossistema é definido por [Hanssen 2012] como uma comunidade de organizações em rede que possuem um interesse em comum, interligados por um software central.

Existem duas razões porque uma organização poderia se interessar em se tornar um ecossistema [Bosch 2009]. A primeira está relacionada à quantidade de funcionalidades que devem ser desenvolvidas para atender seus clientes, onde na maioria das vezes não pode ser obtido através de um investimento em pesquisa em um período razoável de tempo. Algo necessário, visto que atender uma vasta quantidade de clientes é uma garantia para o sucesso. A segunda razão está diretamente ligada à demanda de aplicações customizadas, onde é necessário um investimento em pesquisa e desenvolvimento para se alcançar a produção de software de alta qualidade. Com a estratégia de estender um produto a partir de componentes desenvolvidos por terceiros, a personalização de software se torna algo mais viável.

Além dessas razões, um ecossistema possibilita a ideia de uma plataforma ser mais aderente a outras, atrair mais usuários, atrair a colaboração com outras plataformas e o aumento do valor das suas funcionalidades.

Com base nos diferentes estudos encontrados sobre ecossistemas de software, são observadas diferentes perspectivas a serem exploradas dentro do contexto, como por exemplo, plataformas mobile, arquiteturas e redes sociais. Um modelo de SECO foi desenvolvido no trabalho [Boucharas, Jansen and Brinkkemper 2009], o qual conta com três níveis de perspectiva para entender os desafios da área. O primeiro nível é o escopo organizacional onde as organizações e os seus atores dentro do contexto de ecossistemas de software são os objetos de estudo. O segundo é o escopo do SECO onde, os *software supply networks* (SSNs) e os seus relacionamentos são os objetos de estudo. Os relacionamentos do SSNs incluem os stakeholders e também as características de saúde e estabilidade do ecossistema de software. O terceiro é relacionado ao escopo onde existe mais de um ecossistema de software, onde eles e seus relacionamentos são os alvos de estudo.

A IoT pode ser tratada como um SECO, visto que abrange também conceitos de um ecossistema e de ecossistemas de negócio trazidos originalmente por [Moore 1993] e corroborado por[Leminen et al. 2012]. O SECO no contexto da IoT, pode ser considerada como um novo paradigma, o Ecossistema de IoT [Mazhelis, Luoma and Warma 2012] [Delicato et al. 2013].

2.5 Mapeamento sistemático da Literatura

Visando identificar estudos que abordem as questões de interoperabilidade pragmática em IoT, nós conduzimos um estudo secundário [Muniz et al. 2019]. O estudo foi conduzido seguindo as diretrizes de [Kitchenham 2004].

Segundo [Kitchenham 2004], métodos baseados em evidências devem ser adotados no âmbito da engenharia de software, por meio de estudos secundários. Um dos tipos de estudo secundários que podem ser usados para encontrar evidências é o Mapeamento Sistemático (MS). O objetivo deste tipo de estudo é identificar trabalhos em um contexto mais amplo e responder as questões que podem ser relevantes para uma melhor compreensão do campo pesquisado. As informações coletadas podem estar relacionados aos estudos primários identificados, como por exemplo, local de publicação, autor, ano e/ou informações sobre quais métodos foram utilizados no trabalho.

2.5.1 Planejamento

Com esse objetivo, esse MS foi realizado para evidenciar estudos primários que exponham soluções para tratar a interoperabilidade pragmática em IoT.Como primeiro passo, questões de pesquisa foram elaboradas para o MS, a fim de se alcançar o objetivo proposto. Questões essas, discutidas a seguir:

Q1 Qual é a distribuição dos estudos no decorrer dos anos?

Q2 Quais são os autores na área?

Q3 Em quais domínios a interoperabilidade pragmática em IoT é empregada? (p. ex.: medicina, meio ambiente, indústria, etc.)

Q4 Quais são os tipos de suporte computacional para a interoperabilidade pragmática em IoT? (framework, modelo, metodologia, etc.)

Q5 Quais os métodos de avaliação das soluções propostas têm sido utilizados?

Q6 Quais são os métodos, padrões ou tecnologias utilizados (ou propostos) para dar o suporte à interoperabilidade pragmática em IoT?

Tabela 1 – PICOC

Population (P)	Soluções que implementam a interoperabilidade
Intervention (I)	Soluções de interoperabilidade pragmática
Comparison (C)	Nenhuma comparação
Outcomes (O)	Métodos, modelos, *frameworks*, algoritmos, arquitetura, entre outros
Context (C)	Soluções no domínio de IoT

Em vista de responder às questões de pesquisa propostas, o modelo PICOC, definido por [Petticrew and Roberts 2008], como pode ser visto na Tabela 1.

Para elaborar uma string de busca, os principais termos relacionados ao estudo secundário foram definidos. Visando abranger o campo de busca, sinônimos aos termos foram também adicionados. Os termos e os sinônimos foram definidos baseados no modelo PICOC, como pode ser visto na Tabela 2. Para escolha dos termos e sinônimos, outros estudos secundários relacionados foram utilizados como base [Venceslau et al. 2019] [Neiva et al. 2016][Noura, Atiquzzaman and Gaedke 2019].

Tabela 2 – *string* de busca de acordo com o PICOC

	Palavras-chave	**Sinônimos**
Population	Interoperability	Interoperate,Interoperable, Interoperation,Integrate, Integration, Integrating
Intervention	Pragmatic	Pragmatics, Pragmatism
Comparison	-	-
Outcomes	Approach,Architecture,Framework, Infrastructure,Method,Model, Solution,Technique,Tool,Platform, Process,Software	-
Context	Internet of Things	IoT, Web of Things,Internet of Everything

Após a definição dos termos e dos sinônimos, a *string* de busca foi elaborada utilizando o operador booleano *OR* para agrupar os sinônimos de um termo e o operador *AND* para unir os grupos do PICOC. Como Resultado, a String de busca foi construída:

((interoperability OR interoperate OR interoperable OR interoperation OR integrate OR integration OR integrating) AND (pragmatic OR pragmatics OR pragmatism) AND (solution OR method OR technique OR model OR tool OR framework OR architecture OR infrastructure OR approach OR platform OR software OR process) AND (iot OR "internet of things" OR "internet of Everything" OR "Web of things"))

2.5.2 Condução

A condução do MS foi realizada utilizando-se seis estágios de filtragem. Visando auxiliar a condução, a ferramenta Parsif.al [1] foi utilizada para realizar os processos de filtragem a partir da importação dos estudos primários. Todas as decisões relacionadas às configurações do mapeamento, como por exemplo as bases de dados, artigos de controle e critérios de inclusão e exclusão, estão no artigo completo do MS gerado nesse trabalho [Muniz et al. 2019]. Os seis estágios de filtragem são respectivamente:

1. Buscas automáticas nas bases digitais escolhidas (Sete base no total), através da execução da *string* de busca elaborada;
2. Remoção de artigos duplicados;
3. Leitura dos metadados dos artigos (título, resumo e palavras-chave), com o objetivo de eliminar artigos que não estão dentro do contexto da pesquisa;
4. Leitura dos artigos por completo, a fim de identificar estudos que respondessem às questões de pesquisa propostas;
5. Aplicação do método de *snowballing backward* e *forward* nos estudos primários restantes, a fim de identificar estudos não identificados no MS. Para isto, os estágios do 2 ao 4 foram aplicados nos estudos identificados.
6. Aplicação do estágio 5 novamente nos artigos que foram retornados no *snowballing*;

Como resultado dos processos de filtragens, 1079 artigos foram identificados na busca automática, 1018 artigos sobraram depois da remoção de duplicados, 80 artigos restaram após a leitura dos metadados e 8 artigos sobraram após leitura do texto. Dois *Snowballing* foram realizados e no final, 12 artigos foram selecionados para o estudo. Logo após, os dados necessários e suficientes foram extraídos dos estudos primários selecionados e das bases onde estes estão hospedados, com o objetivo de responder às questões de pesquisas.

2.5.3 Resultados

Logo após a extração dos dados, análises foram feitas para responder às Questões de Pesquisa. As questões de pesquisa de Q1 a Q6 foram respondidas de forma reduzida nessa dissertação. As respostas em sua totalidade podem ser observadas no artigo publicado [Muniz et al. 2019].

No que se refere à Primeira questão de pesquisa (**Qual é a distribuição dos estudos no decorrer dos anos?**), poucos são os estudos identificados. Os artigos

[1] https://parsif.al/

identificados foram publicados no período entre 2012 e 2018, possuindo cinco citações entre estes. As citações são referentes a estudos que são continuações, para corroborar uma afirmativa, ou para citar a existência de um modelo de interoperabilidade baseado no modelo de [Tolk and Muguira 2003].

No que se refere a segunda questão de pesquisa (**Quais são os autores na área?**), 40 autores foram identificados[2], sendo 11 destes principais e 29 coautores. Visando identificar os possíveis autores mais influentes, dados da base digital foram extraídos para definir a relação entre os autores (autores que publicaram em conjunto, não restrito ao contexto da MS) e o nível de influência (relação entre artigos publicados com as citações dos mesmos). Foi observado que mais de 50 % dos autores possuem um certo nível de significância para área.

No que se refere à terceira questão de pesquisa (**Em quais domínios a interoperabilidade pragmática em IoT são empregados?**), foram identificados seis domínios: Estes são respectivamente *healthcare* (25 %), enterprise (25 %), *Industry* 4.0 (17 %), *smart computing* (8 %), e-Commerce (8 %) e um domínio mais geral (17 %).

No que se refere a quarta e a quinta questão de pesquisa (**Quais são os tipos de suporte computacional para a interoperabilidade pragmática em IoT? e quais são os métodos de avaliação dos trabalhos?**), foram identificados cinco tipos de suporte computacional. Estes são: Metodologia (5 estudos - 2 avaliados); Modelo (3 estudos - 1 avaliado); *Framework* (2 estudos - 1 avaliado); Metamodelo (1 estudo); Arquitetura (1 estudo). Foram identificados três tipos de avaliações: Estudo de caso (2 artigos); Teste de desempenho (1 artigo); Comparação entre estudos relacionados (1 artigo).

No que se refere a sexta questão de pesquisa (**Quais são os métodos, padrões ou tecnologias (MPT) utilizados, ou propostos, para dar o suporte à interoperabilidade pragmática em IoT?**), foram identificados 19 métodos, padrões ou tecnologias (MPT), sendo estes categorizados em quatro grupos, como pode ser visto na Tabela 3.

Os MPT capturados nos estudos primários, apoiavam a alcançar a interoperabilidade pragmática. No entanto, esses apoiavam em conjunto ou de forma isolada a alcançar a interoperabilidade semântica em conjunto com outras ferramentas. Ferramentas estas, que na maioria das vezes, salvo RuleML e FIPA-ACL(em conjunto com FIPA-AAS), não foram expostas pelos trabalhos. No que tange aos MPT de um forma geral, nota-se uma evolução/adaptação para que o domínio da IoT possa ser abrangido. Como exemplo podemos citar as arquiteturas SOA e EDA em conjunto para formar a SOEDA. Outro exemplo é a utilização de protocolos de comunicação como CoAP e AMQP, ao invés do tradicional HTTP.

[2] https://drive.google.com/open?id=1Z82SyYCAvb3AuQ-FAuHkqVUllENKVx2kqi-n2krQTpk

Tabela 3 – RQ 6 - Métodos, padrões ou tecnologias (MPT) utilizados nos trabalhos

	MPT
Arquiteturas	SOA (*Service-oriented architecture*)
	EDA (*Event-driven architecture*)
	SOEDA (*Service-Oriented Event-Driven Architecture*)
	REST (*Representational State Transfer*)
	ESB (*Enterprise Service Bus*)
	FIPA-AAS (*Abstract Architecture Specifications*)
Tecnologias para descrição ou execução	FIPA-ACL (*Foundation for Intelligent Physical Agents - Agents Communication Language*)
	RuleML (*Rule Markup Language*)
	W3C RIF (*Rule Interchange Format*)
	Linked USDL (*Unified Service Description Language*)
	BPEL (*Business Process Execution Language*)
Protocolos de comunicação	HTTP (*Hypertext Transfer Protocol*)
	CoAP (*Constrained Application Protocol*)
	AMQP (*Advanced Message Queuing Protocol*)
Outros	RabbitMQ
	Jena
	Apache Synapse
	Apache ODE
	Esper Engine

2.6 Trabalhos Relacionados

Esta seção tem por objetivo expor os 6 estudos primários identificados no mapeamento sistemático, que mais estão aderentes ao nosso trabalho.

[Wassermann and Fay 2017]

No trabalho de [Wassermann and Fay 2017], é proposto um modelo para dar suporte à interoperabilidade utilizando *Multi-agent systems (MAS)*. Os autores optaram pela utilização do MAS, pois além de ser uma tecnologia sofisticada e flexível, possui um grande potencial em oferecer soluções de problemas de uma forma descentralizada, o que favorece áreas como *Industrie* 4.0 e *Industrial Internet of Things* (IIoT).

Como solução, a tecnologia para agentes FIPA-ACL foi escolhida, pois além de ser uma linguagem popular para agentes e ser muito similar a outras linguagens, também possibilita o suporte à interoperabilidade até o nível semântico e parcialmente do pragmático. Para alcançar o nível pragmático e o conceitual, os autores propuseram a utilização de regras de interoperabilidade para MAS, visto que tais regras podem ser utilizadas para verificação de MAS em tempo de produção e execução.

[Zörrer et al. 2018]

No trabalho de [Zörrer et al. 2018], é proposto uma nova metodologia para SRI(Service Resolution Infrastructure) no domínio da *Industrial Internet of Things* (IIoT), focando no

aspectos pragmáticos da interoperabilidade. Para que isto fosse possível, um *Topic-based Publish Subscribe Infrastructure* (TbPSI), foi utilizado para disponibilizar dispositivos IoTs limitados. TbPSI foi escolhido como tecnologia para troca de mensagens visto que permite a comunicação entre sistemas, a partir da produção de mensagens para um determinado tópico, sendo necessário somente se inscrever neste para receber o conteúdo produzido.

Para alcançar o nível semântico, tópicos, tipos de mensagens e protocolos foram descritos. No entanto, devido a possível variedade de tipos de mensagens, os autores adotaram um padrão, que se baseia em aspectos pragmáticos, para diminuir os tipos de mensagens. Para que isto ocorra, uma documentação detalhada é necessária.

[Weigand and Paschke 2012]

No trabalho de [Weigand and Paschke 2012], é proposto a utilização de regras para se alcançar a Web Pragmática. De acordo com os autores, regras são importantes para alcançar o nível pragmático. Pois por meio deles, dados podem ser transformados em informação de uma maneira automática e contextual (nível sintático). Tais informações podem ser interpretadas como conhecimento (semântico) e a partir disso, conclusões e decisões podem ser feitas por meio do conhecimento já existente. Além disso, são importantes para que mudanças de comportamento possam acontecer.

Para expressar as regras, as tecnologias RuleML e W3C RIF foram utilizadas para representar decisões e comportamentos lógicos de agentes semânticos como, Dublin Core, vCard, Bibtex, FOAF e SIOC. Também, foram utilizados para trocar mensagens entre eles.

[Thoma et al. 2013]

No trabalho de [Thoma et al. 2013], é proposto uma visão sobre *sensing enterprise* (Sistemas empresariais IT de *context-aware e real-world aware*) a partir da utilização de **Linked Services** para acessar sensores de dispositivos que foram descritos por esta tecnologia. De acordo com os autores, a coleta de dados em tempo real em contextos reais, são um dos fatores cruciais para tomada de decisão de negócio.

A ideia na utilização de *Linked Services* é fornecer um serviço de descrição que tem como base a utilização de serviços de descrição da Web Semântica, como por exemplo, ontologias. A solução proposta da suporte à interoperabilidade semântica e pragmática através da utilização de ontologias leves. No entanto, como argumentado pelos autores, a solução não resolve todos problemas que surgem nestes níveis de interoperabilidade.

[Purnomosidi et al. 2014]

No trabalho de [Purnomosidi et al. 2014], é proposta uma arquitetura para Web pragmática para que a automação da IoT possa ser melhorada. Os autores argumentaram que a Web Pragmática é um boa escolha para o domínio da IoT, principalmente para automação residencial, visto que por meio dela, aplicações podem providenciar serviços e

também o contexto em que os dados estão inseridos.

Como solução, os autores propuseram um *e-commerce* para que a interação entre dispositivos inteligentes e a Web Pragmática possa acontecer, através da utilização de webservices RESTfull. Assim como no trabalho [Wassermann and Fay 2017], a tecnologia FIPA-ACL foi utilizada para a comunicação, sendo necessário também, uma melhoria para que o nível pragmático pudesse ser alcançado.

[Deng, Liu and Li 2016]

No trabalho de [Deng, Liu and Li 2016], é proposto um sistema de sumarização de ontologias para enriquecer a colaboração semântica em IoT com objetivo de dar suporte à interoperabilidade pragmática. O motivo que levou a esta proposta está relacionado ao grande volume de informação que está contido dentro de ontologias, o que dificulta a inferência sobre as informações. Os autores reportam que a sumarização no nível pragmático, como também no semântico, adiciona barreiras nas inferências de informações implícitas.

2.7 Atualização do mapeamento

Com o intuito de identificar novos estudos após a realização do MP em 2019, um ***SnowBalling - Forward*** foi realizado, em 2022, sobre os 12 artigos identificados. Após a realização da leitura dos metadados e leitura do texto, quatro estudos primários foram identificados [Ribeiro et al. 2021] [Jones 2020] [Hooi, Hassan and Sharifff 2018] [Robles, Narendra and Kiviranta 2020]. Os estudos realizados abordam a interoperabilidade pragmática em seus contextos e observa-se a presença de ontologias, Web pragmáticas e outros métodos para alcançar o objetivo. No entanto, os atributos de qualidade abordados nesta dissertação são diferentes dos apresentados nestes estudos primários.

2.8 Considerações Finais

Nesse capítulo, os conceitos necessários para entender a dissertação, assim como o mapeamento sistemático realizado e os trabalhos relacionados derivados deste.

No que tange ao mapeamento e aos trabalhos relacionados, observa-se que poucos são os trabalhos que propõem soluções para dar suporte à interoperabilidade pragmática em IoT e dentro destes, um minoria propõe uma solução mais concreta, como um framework. Dentro das tecnologias, métodos e padrões, observa-se que parte destes levam em consideração as limitações enfrentadas pela IoT, como limitação de *hardware* e escalabilidade.

No entanto, observa-se que nos estudos primários, estes foram utilizados para sanar somente parte dos problemas enfrentados pela IoT. Como por exemplo, podemos citar a utilização da tecnologia para agentes FIPA-ACL, sendo utilizada na Web pragmática.

Esta tecnologia permite a escalabilidade, pois permite ser distribuída, mas o protocolo de comunicação utilizado, o HTTP, não foi construído pensando nas limitações da IoT.

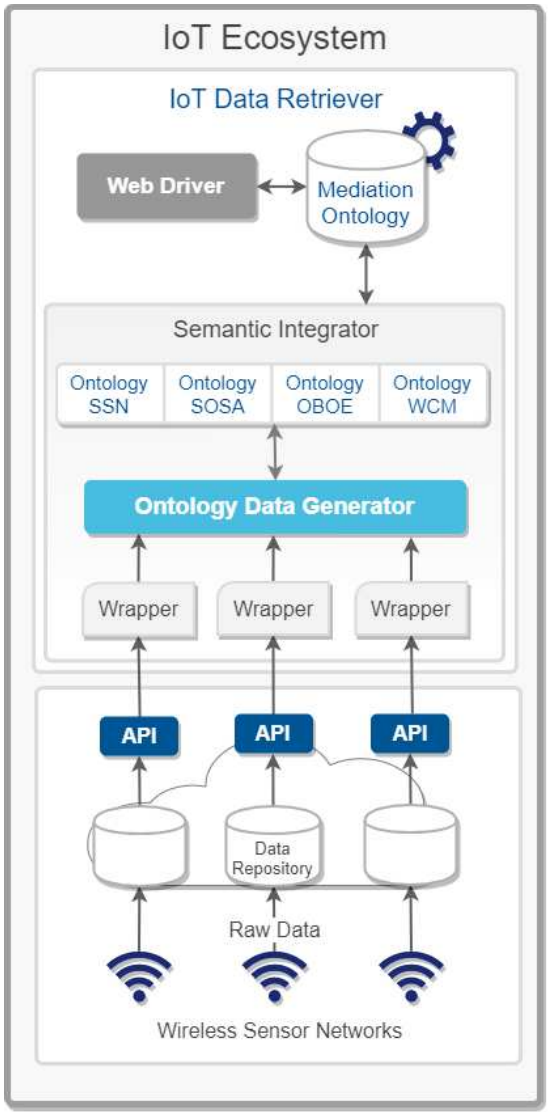
Visando propor uma solução para dar suporte à interoperabilidade pragmática em IoT, levando em consideração as limitações da IoT (memória, processamento, largura de banda), uma arquitetura foi proposta. Esta será descrita no próximo capítulo.

3 INTEROPERABILIDADE EM ECOSSISTEMAS DE IOT

Este capítulo tem por objetivo apresentar uma arquitetura para dar suporte à interoperabilidade sintática, semântica e pragmática em ecossistemas de IoT. A arquitetura desenvolvida foi integrada como parte de um ecossistema de IoT. Dentro deste ecossistema de IoT, uma plataforma para integração semântica dos dados de IoT já está inserida, o IoT Data Retriever [Ferreira 2019]. A solução desenvolvida nesta dissertação, tem por objetivo apoiar a interoperabilidade dentro deste ecossistema. Neste capítulo, primeiramente, será exposto um resumo sobre o que já existe no ecossistema e logo após, todos os detalhes sobre o projeto, implementação e teste funcionais da arquitetura proposta.

3.1 IoT Data Retriever

Figura 1 – Arquitetura do ecossistema de IoT [Ferreira 2019]



O IoT *Data retriever* é uma solução que busca integrar, de forma semântica, dados

de IoT [Ferreira 2019]. A arquitetura desse trabalho pode ser observada na Figura 1. Como relatado por [Ferreira 2019], a integração semântica é necessária para pesquisadores, visto que, dados de IoT possuem uma natureza descentralizada e heterogênea, o que dificulta a coleta e a análise dos dados. Como parte da solução, uma ontologia foi desenvolvida a partir de outras quatro. A integração semântica é realizada entre dados de diferentes *clouds* de armazenamentos de WSN *Wireless Sensor Network*). Os dados são fornecidos por estas *clouds*, através de APIs. A solução desenvolvida, tem como foco o domínio da pesquisa científica e a colaboração entre pesquisadores.

3.2 Especificação

Nesta seção o enfoque da solução da dissertação é apresentado, levando em consideração os problemas identificados no mapeamento sistemático [Muniz et al. 2019]. Além disso, os requisitos funcionais e não funcionais são elicitados.

Como relatado no Capítulo 2, a maioria dos trabalhos, de uma forma geral e também na IoT, tratam os níveis sintático e semântico. No mapeamento sistemático da literatura, foram identificados 12 trabalhos que propõem soluções para dar suporte à interoperabilidade pragmática em IoT. Dentre estes, a maioria somente propõe algo mais abstrato para tratar o nível pragmático.

Dos métodos, padrões e tecnologias capturados nos trabalhos o FIPA-ACL, uma linguagem para descrição de agentes, se destacou como tecnologia que pode alcançar o nível pragmático de forma superficial e de forma mais completa com auxílios de regras. No entanto, a utilização de agentes para a IoT, não leva em consideração todas as limitações da IoT, uma vez que, utiliza-se do protocolo HTTP (protocolo síncrono). A utilização de regras, foi observado em mais de um dos estudos selecionados no mapeamento sistemático, visto que tais auxiliam a alcançar o nível pragmático em conjunto o contexto em que os dispositivos estão inseridos.

Como enfoque da solução, essa dissertação visa propor uma solução para dar suporte à interoperabilidade sintática, semântica e pragmática em IoT levando em consideração as limitações da IoT. Além disso, dar suporte à produção destes aplicativos em um ambiente Web de uma forma que a descrição dos dispositivos venha a ser algo simples e objetivo. Para apoiar o nível pragmático, regras foram escolhidas em conjunto com uso de *Machine Learning* para identificar contextos em que as regras não foram declaradas.

A solução proposta visa apoiar a escalabilidade, flexibilidade e extensibilidade através da abstração da carga do processamento dos dispositivos IoT. Para que seja possível, a solução foi construída utilizando-se três módulos, sendo que dois destes podem possuir inúmeras instâncias em regiões geográficas distintas.

3.2.1 Requisitos funcionais

RF1 A solução deve permitir que usuários possam realizar uma descrição sintática de forma simplificada por meio de uma plataforma Web.

RF2 A solução deve permitir que através de uma única descrição entre duas unidades básicas, outras possam ser geradas (Através da ligação entre as próprias unidades).

RF3 A solução deve permitir que usuários possam realizar uma descrição semântica de forma simplificada por meio de uma plataforma Web.

RF4 A solução deve permitir uma descrição semântica com menos viés, através do uso de ferramentas que possibilitem traçar uma relação com outras descrições.

RF5 A solução deve permitir a descrição de dispositivos/aplicações IoTs de forma simplificada por meio de uma plataforma Web.

RF6 A solução deve permitir que dispositivos/aplicações IoT possam interoperar nos níveis sintáticos, semântico e pragmático.

RF7 A solução deve fornecer mecanismos para que ecossistemas externos possam capturar as informações sintáticas, semântica e pragmática, com o objetivo de apoiar a interoperabilidade em ambientes externos.

RF8 A solução deve permitir a mudança no contexto dos IoTs e consequentemente nas regras vinculadas a este contexto.

3.2.2 Requisitos não funcionais

RNF1 Escalabilidade - A solução deve permitir o crescimento no número de dispositivos conectados na rede de uma forma que o desempenho não venha ser afetado.

RNF2 Flexibilidade - A solução proposta deve ser capaz de se adaptar às mudanças do sistema e das regras que foram elaboradas para os contextos. A solução deve ser livre de estruturas rígidas que possam impactar na evolução do sistema.

RNF3 Extensibilidade - A solução proposta deve possuir a capacidade de ser extensível, de uma forma que módulos possam ser indexados ou modificados. Isto implica que a solução deve ter um baixo nível de acoplamento tanto entre os próprios módulos, quanto com os dispositivos IoTs e entre estes também.

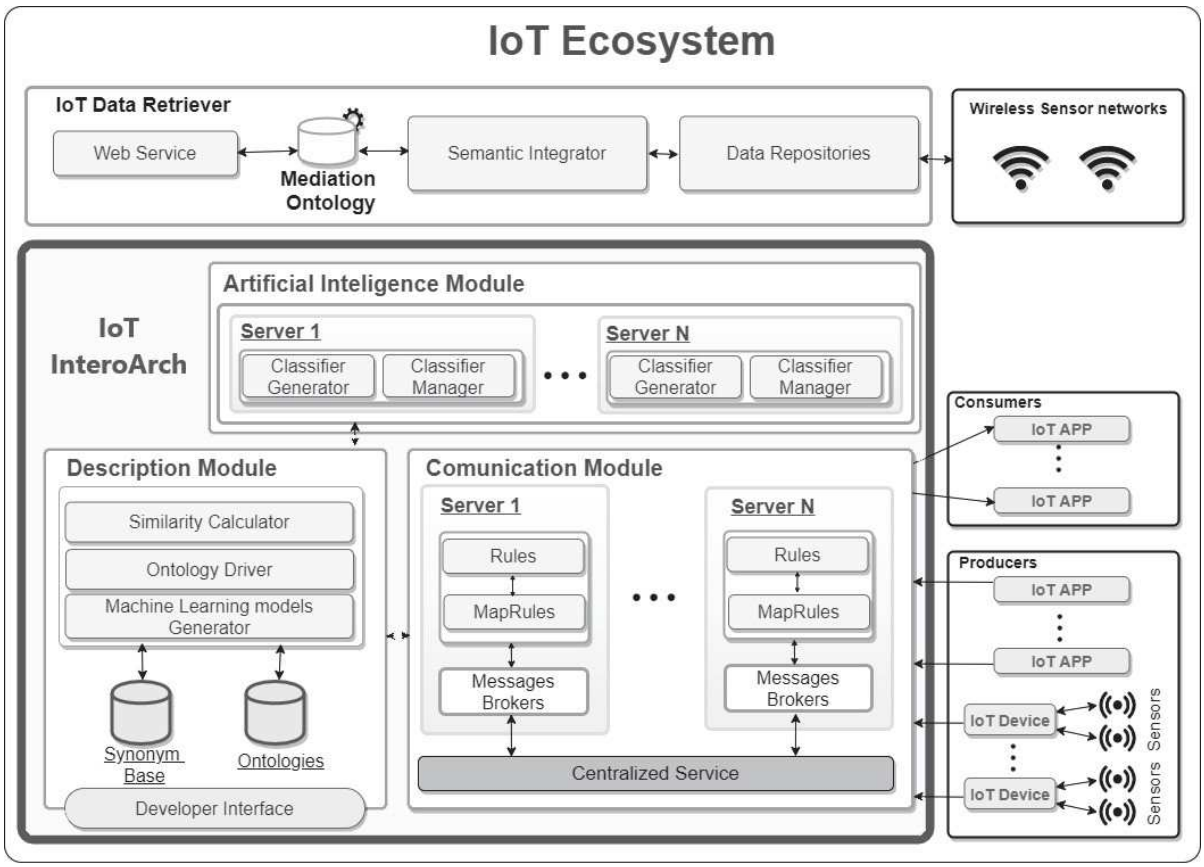
RNF4 Portabilidade - A solução proposta deve ser capaz de abstrair boa parte da lógica dos dispositivos IoT, de uma forma que a complexidade de desenvolvimento dos softwares embarcados seja diminuída, para que a interação possa acontecer dentro da rede criada de uma forma mais simples, replicável para outras linguagens e com

um baixo custo de processamento e armazenamento, visto que dispositivos IoT que produzem dados, tendem a ter limitações de hardware.

3.3 Projeto e Implementação

A solução proposta nessa dissertação, a IoT-InterArch, foi incorporada ao ecossistema de IoT já existente. Visando uma representação mais clara sobre a real contribuição do trabalho, o módulo do IoT data-retriever, foi representado de forma simplificada visto que não faz parte da solução proposta. A nova arquitetura de ecossistema de IoT gerada pode ser visualizada na Figura 2. O módulo referente a contribuição dessa dissertação foi destacado através do uso de uma borda mais grossa.

Figura 2 – Arquitetura do ecossistema de IoT (estendida de [Ferreira 2019])



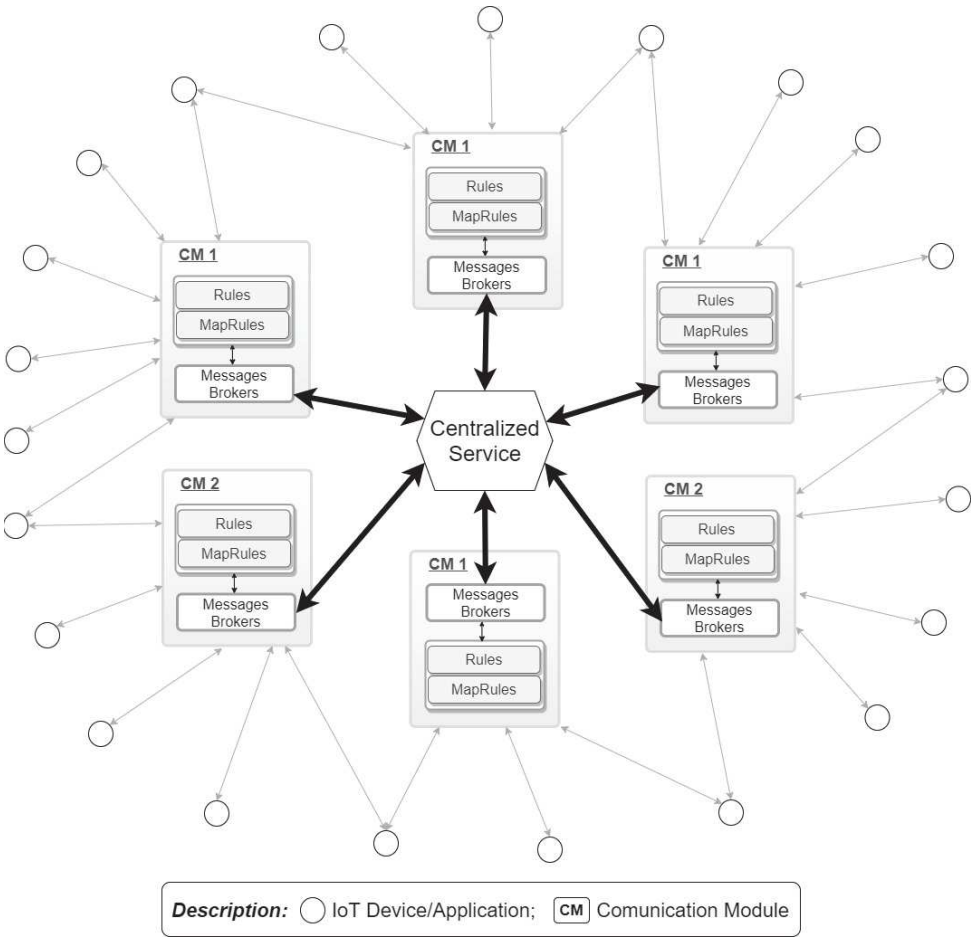
Para apoiar a interoperabilidade em IoT, levando em consideração os requisitos funcionais e não funcionais elicitados, a solução foi elaborada tendo como base três módulos principais: (1) *Comunication Module*; (2) *Description Module*; (3) *Artificial Inteligence Module*. Estes serão descritos a seguir.

3.3.1 *Comunication Module - CM*

O módulo de comunicação foi projetado para que a interoperabilidade em tempo de execução pudesse acontecer. Através deste módulo, dispositivos IoT podem trocar

informações no nível sintático, semântico e pragmático levando em consideração os requisitos não funcionais. Para que isto que seja possível, este módulo foi projeto para que possua diversas instâncias de uma forma que estas sejam independentes entre si, através do uso de *Messages Brokers* do tipo *publisher/subscriber* em conjunto com um serviço de centralização. Na Figura 3 é possível visualizar o processo básico de comunicação entre os dispositivos.

Figura 3 – Módulo de Comunicação



Messages Brokers, do tipo *publisher/subscriber*, foram escolhidos para a base da comunicação da IoT por permitirem abstração no processo de comunicação entre as partes dentro de um contexto geral e também na IoT [Guner, Kurtel and Celikkan 2017].

Por meio do módulo desenvolvido, dispositivos podem se conectar em mais de uma instância do mesmo, garantindo assim uma maior disponibilidade do serviço. A comunicação entre os dispositivos/aplicações IoT até o nível semântico é feita da mesma forma que acontece em um *Message Broker* comum, com a diferença que existe um processo essencial para que a interoperabilidade possa acontecer de fato, através da requisição de informações do **módulo de descrição**. Para alcançar o nível pragmático, um mapeamento de regras é realizado no mesmo módulo. Os processos de preparo para produzir ou consumir dados dentro de um contexto e o processo para mapear as regras será exposto em seções

futuras.

Para a implementação, o *message broker* **Apache Kafka** [1] foi escolhido em conjunto com o serviço centralizador **Apache Zookeeper** [2]. A combinação destas duas tecnologias permite que várias instâncias distribuídas Apache Kafka possam existir, através do Apache Zookeeper. Muitos são os *message brokers* existentes, mas no que se refere à escalabilidade horizontal, ao armazenamento temporário mais prolongado (ou até mesmo fixo através de utilização de banco de dado) das informações produzidas e a possibilidade em se criar canais seguros/privados, o Apache Kafka é uma das soluções mais utilizadas no mercado [Thein 2014]. No que se refere à IoT, várias são as soluções que a utilizam [D'silva et al. 2017] [Ogawa et al. 2019] [Syafrudin et al. 2018].

O Apache Kafka utiliza-se do protocolo HTTP para a comunicação. No entanto, devido a sua natureza de *open source*, é possível identificar diversos projetos que realizam a integração com outros *message brokers*. Dentro destes, vale citar o **Eclipse Mosquitto**, que utiliza-se de um dos principais protocolos para IoT, o **MQTT** [Hillar 2017]. Como já mencionado, o protocolo HTTP pode ser visto como uma limitação da IoT, O motivo está relacionado a sua natureza sincrona. No entanto, através da utilização do Apache Kafka, a comunicação com outros dispositivos IoTs se torna assíncrona, mitigando a limitação evidenciada.

3.3.2 *Description Module*

O módulo de descrição pode ser considerado como o módulo principal do sistema, visto que além de ser o responsável por estabelecer a comunicação e fornecer os dados necessários para os outros módulos, também é o que possibilita a descrição de dispositivos/aplicações IoT. Além disso, fornece endpoints através de WebServices RESTful com a possibilidade de acessar as informações sintáticas, semânticas e pragmáticas armazenadas e inferidas dentro do módulo.

A descrição dos dispositivos/aplicações é feita através de um plataforma Web que conta com o apoio de informações internas e externas ao ecossistema de IoT desenvolvido como forma de apoio. O objetivo do uso desta estratégia é facilitar a descrição e diminuir o viés que pode ser gerado pelos usuários que estão realizando esta ação. A plataforma Web desenvolvida foi exposta na subseção relacionada a parte da implementação.

Para o armazenamento das informações, duas ontologias foram elaboradas de uma forma a abranger vários tipos de contextos, podendo estes estarem relacionados ou não. Como exemplo, podemos citar o contexto de *Smart Home* sendo tratado de forma independente e/ou sendo inserido em contexto mais macro, como de uma *Smart City*.

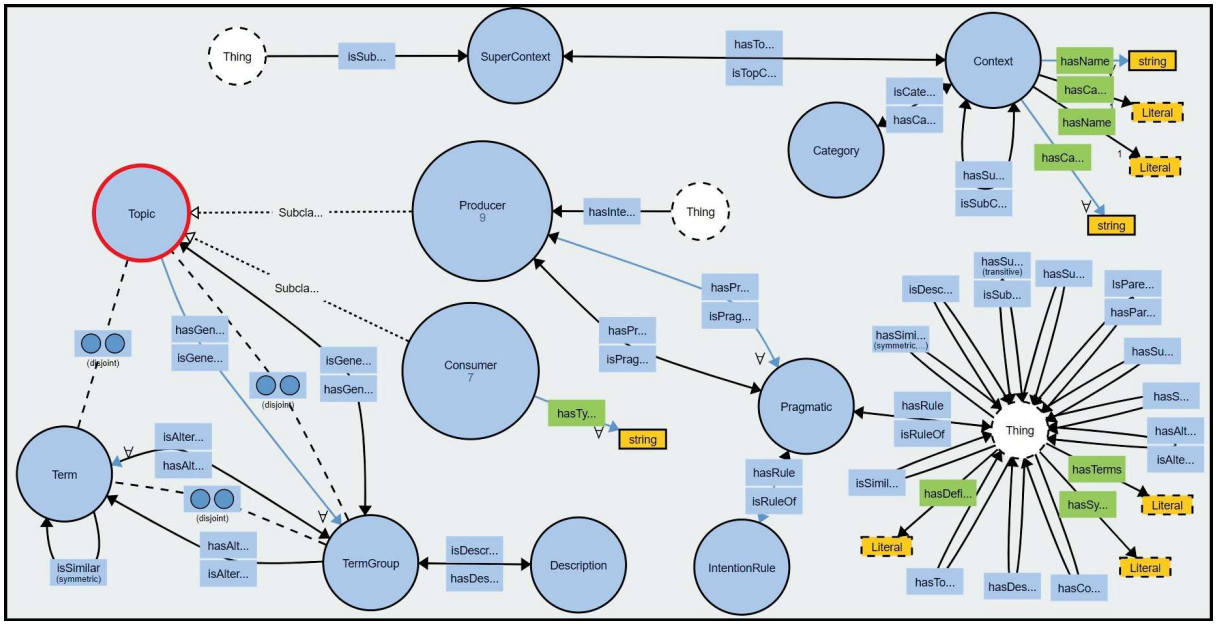
[1] https://kafka.apache.org
[2] https://zookeeper.apache.org

3.3.2.1 Ontologias

Como já relatado no capítulo de referencial teórico, as ontologias permitem uma descrição mais complexa sobre um domínio. Além disso, possibilita a descoberta de informações implícitas. Através do uso de ontologias para a descrição de dispositivos/aplicações IoT, novos relacionamentos podem ser descobertos entre estes através do uso de ***Property Chains*** e de regras sobre os relacionamentos. Além disso, o uso de ontologias permite a fácil substituição de uma mesma ontologia, só que com novas informações. Pensando nessa possibilidade, um endpoint exclusivo para o compartilhamento da ontologia, foi desenvolvido.

Dentro do domínio da IoT, várias ontologias foram desenvolvidas. Dentro de um domínio mais geral algumas se destacam por trazer soluções para os níveis sintático e semântico como, *Semantic Sensor Network* (SSN), IoT-O, IoT-*Lite*, iot-*ontology* dentre outras [Seydoux et al. 2016]. Estas ontologias visam permitir a descrição de dispositivos com o objetivo de que estes possam interoperar em uma rede. No entanto, como proposta de ontologia para a solução, torna-se necessário a utilização de uma ontologia que permita a descrição de dispositivos levando em consideração os *messages brokers*. Para isto, duas ontologias foram desenvolvidas.

Figura 4 – Primeira Ontologia



A primeira ontologia pode ser vista como a principal, pois permite uma descrição mais complexa sobre os dispositivos. Essa ontologia foi projetada como base em **tópicos**, visto que é o formato utilizado pelos **messages broker** atuais. **Tópicos** nada mais é do que um rótulo sobre o que está sendo produzido/consumido, pode ser considerado como uma descrição semântica bem superficial. Na Figura 4 é possível visualizar as relações de

forma simplificada e na Tabela 4 visualizar em forma de linguagem de descrição a primeira ontologia desenvolvida.

Tabela 4 – Primeira Ontologia: Descrição de tópicos, contextos e regras

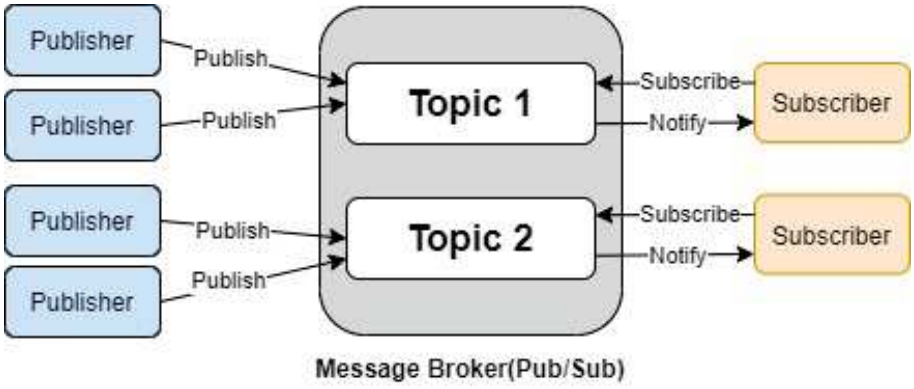
Classes	
Category	Term $\sqsubseteq \neg$ Topic
Consumer	Term $\sqsubseteq \neg$ TermGroup
Consumer $\sqsubseteq =$ hasTypeConsume	Topic $\sqsubseteq \neg$ TermGroup
Consumer $\sqsubseteq$ Topic	Term $\sqsubseteq \neg$ Topic
Consumer $\sqsubseteq \forall$ hasTypeConsume DataType: String	TermGroup $\sqsubseteq \neg$ Topic
ContextInf	**TermGroup**
ContextInf $\sqsubseteq =$ hasCategory Category	TermGroup $\sqsubseteq \forall$ hasTerm Term
ContextInf $\sqsubseteq \forall$ hasCategory Category	TermGroup $\sqsubseteq$ Tags
Description	TermGroup $\sqsubseteq \geq 1$ hasTerm Term
Description $\sqsubseteq \forall$ hasDefinition DataType: String	TermGroup $\sqsubseteq \neg$ Term
Description $\sqsubseteq \forall$ hasTerms DataType: String	Topic $\sqsubseteq \neg$ Term
Description $\sqsubseteq \forall$ hasSynonyms DataType: String	TermGroup $\sqsubseteq \neg$ Term
IntentionRule	TermGroup $\sqsubseteq \neg$ Topic
IntentionRule $\sqsubseteq \forall$ hasIntentionTopic Producer	Term $\sqsubseteq \neg$ Topic
IntentionRule $\sqsubseteq =$ hasIntentionTopic Producer	TermGroup $\sqsubseteq \neg$ Topic
Producer	**Topic**
Producer $\sqsubseteq$ Topic	Topic $\sqsubseteq \forall$ hasTermGroup TermGroup
Producer $\sqsubseteq \forall$ hasRulesGroup RulesGroup	Topic $\sqsubseteq$ Tags
Producer $\sqsubseteq =$ hasRulesGroup RulesGroup	Topic $\sqsubseteq =$ hasTermGroup TermGroup
RulesGroup	Topic $\sqsubseteq \forall$ hasContextInf ContextInf
RulesGroup $\sqsubseteq \geq 1$ hasRule IntentionRule	Topic $\sqsubseteq \geq 1$ hasContextInf ContextInf
RulesGroup $\sqsubseteq \forall$ hasRule IntentionRule	TermGroup $\sqsubseteq \neg$ Term
SuperContextInf	Topic $\sqsubseteq \neg$ Term
Tags	Term $\sqsubseteq \neg$ TermGroup
Term	Topic $\sqsubseteq \neg$ TermGroup
Term $\sqsubseteq$ Tags	Topic $\sqsubseteq \neg$ Term
Term $\sqsubseteq \neg$ TermGroup	Topic $\sqsubseteq \neg$ TermGroup
Object properties	
IsParentCategoryOf	**hasTopContextInfCategory**
IsParentCategoryOf $\equiv$ hasParentCategory$^-$	$\exists$ hasTopContextInfCategory Thing $\sqsubseteq$ SuperContextInf
$\exists$ IsParentCategoryOf Thing $\sqsubseteq$ ContextInf	$\top \sqsubseteq \forall$ hasTopContextInfCategory Category
$\top \sqsubseteq \forall$ IsParentCategoryOf Category	**isCategoryOf**
hasCategory	hasCategory $\equiv$ isCategoryOf$^-$
hasCategory $\equiv$ isCategoryOf$^-$	$\exists$ isCategoryOf Thing $\sqsubseteq$ Category
$\exists$ hasCategory Thing $\sqsubseteq$ ContextInf	$\top \sqsubseteq \forall$ isCategoryOf ContextInf
$\top \sqsubseteq \forall$ hasCategory Category	**isDescriptionOf**
hasContextInf	hasDescription $\equiv$ isDescriptionOf$^-$
$\exists$ hasContextInf Thing $\sqsubseteq$ Topic	$\exists$ isDescriptionOf Thing $\sqsubseteq$ Description
$\top \sqsubseteq \forall$ hasContextInf ContextInf	$\top \sqsubseteq \forall$ isDescriptionOf TermGroup
hasDescription	**isDescriptionSP**
hasDescription $\equiv$ isDescriptionOf$^-$	$\exists$ isDescriptionSP Thing $\sqsubseteq$ Description
$\exists$ hasDescription Thing $\sqsubseteq$ TermGroup	$\top \sqsubseteq \forall$ isDescriptionSP TermGroup
$\top \sqsubseteq \forall$ hasDescription Description	**isIntentionTopicOf**
hasIntentionTopic	hasIntentionTopic $\equiv$ isIntentionTopicOf$^-$
hasIntentionTopic $\equiv$ isIntentionTopicOf$^-$	$\exists$ isIntentionTopicOf Thing $\sqsubseteq$ Producer
$\exists$ hasIntentionTopic Thing $\sqsubseteq$ IntentionRule	$\top \sqsubseteq \forall$ isIntentionTopicOf IntentionRule
$\top \sqsubseteq \forall$ hasIntentionTopic Producer	**isRuleOf**
hasParentCategory	hasRule $\equiv$ isRuleOf$^-$
IsParentCategoryOf $\equiv$ hasParentCategory$^-$	$\exists$ isRuleOf Thing $\sqsubseteq$ IntentionRule
$\exists$ hasParentCategory Thing $\sqsubseteq$ Category	$\top \sqsubseteq \forall$ isRuleOf RulesGroup
$\top \sqsubseteq \forall$ hasParentCategory ContextInf	**isRulesGroupOf**
hasRule	hasRulesGroup $\equiv$ isRulesGroupOf$^-$

hasRule ≡ isRuleOf$^-$	∃ isRulesGroupOf Thing ⊑ RulesGroup
∃ hasRule Thing ⊑ RulesGroup	⊤ ⊑ ∀ isRulesGroupOf Producer
⊤ ⊑ ∀ hasRule IntentionRule	**isSameTermGroupOf**
hasRulesGroup	hasSameTermGroup ≡ isSameTermGroupOf$^-$
hasRulesGroup ≡ isRulesGroupOf$^-$	∃ isSameTermGroupOf Thing ⊑ Topic
∃ hasRulesGroup Thing ⊑ Producer	⊤ ⊑ ∀ isSameTermGroupOf Topic
⊤ ⊑ ∀ hasRulesGroup RulesGroup	**isSimilar**
hasSameTermGroup	isSimilar ≡ isSimilar$^-$
hasSameTermGroup ≡ isSameTermGroupOf$^-$	∃ isSimilar Thing ⊑ Term
∃ hasSameTermGroup Thing ⊑ Topic	⊤ ⊑ ∀ isSimilar Term
⊤ ⊑ ∀ hasSameTermGroup Topic	**isSubContextInfOf**
hasSimilarDescription	⊑ isSubContextInfOfWH
hasSimilarDescription ≡ hasSimilarDescription$^-$	hasSubContextInf ≡ isSubContextInfOf$^-$
TransitiveProperty hasSimilarDescription	∃ isSubContextInfOf Thing ⊑ ContextInf
∃ hasSubContextInfWH Thing ⊑ ContextInf	⊤ ⊑ ∀ isSubContextInfOf ContextInf
⊤ ⊑ ∀ hasSubContextInfWH ContextInf	**isSubContextInfOfWH**
hasSuperContextInf	hasSubContextInfWH ≡ isSubContextInfOfWH$^-$
∃ hasSuperContextInf Thing ⊑ Topic	∃ isSubContextInfOfWH Thing ⊑ ContextInf
⊤ ⊑ ∀ hasSuperContextInf SuperContextInf	⊤ ⊑ ∀ isSubContextInfOfWH ContextInf
hasSuperContextInf2	**isSubContextInfPC**
⊤ ⊑ ∀ hasSuperContextInf2 SuperContextInf	⊤ ⊑ ∀ isSubContextInfPC SuperContextInf
hasTerm	**isTermGroupOf**
hasTerm ≡ isTermOf$^-$	hasTermGroup ≡ isTermGroupOf$^-$
∃ hasTerm Thing ⊑ TermGroup	∃ isTermGroupOf Thing ⊑ TermGroup
⊤ ⊑ ∀ hasTerm Term	⊤ ⊑ ∀ isTermGroupOf Topic
hasTermForTopic	**isTermOf**
hasTermForTopic ≡ isTermOfTopic$^-$	hasTerm ≡ isTermOf$^-$
∃ hasTermForTopic Thing ⊑ Topic	∃ isTermOf Thing ⊑ Term
⊤ ⊑ ∀ hasTermForTopic Term	⊤ ⊑ ∀ isTermOf TermGroup
hasTermGroup	**isTermOfTopic**
hasTermGroup ≡ isTermGroupOf$^-$	hasTermForTopic ≡ isTermOfTopic$^-$
∃ hasTermGroup Thing ⊑ Topic	∃ isTermOfTopic Thing ⊑ Term
⊤ ⊑ ∀ hasTermGroup TermGroup	⊤ ⊑ ∀ isTermOfTopic Topic
hasTopContextInf	**isTopContextInfOf**
hasTopContextInf ≡ isTopContextInfOf$^-$	hasTopContextInf ≡ isTopContextInfOf$^-$
∃ hasTopContextInf Thing ⊑ SuperContextInf	∃ isTopContextInfOf Thing ⊑ ContextInf
⊤ ⊑ ∀ hasTopContextInf ContextInf	⊤ ⊑ ∀ isTopContextInfOf SuperContextInf
thisTopicHasRule	**TopicsPragmaticallyEquivalent**
isRuleOfThisTopic ≡ thisTopicHasRule$^-$	TopicsPragmaticallyEquivalent ≡ TopicsPragmatically-Equivalent$^-$
Data properties	
hasDefinition	**hasTerms**
hasSynonyms	**hasTypeConsume**

Para entender como a ontologia foi construída, é importante compreender o funcionamento de um *message broker* do tipo *publisher/subscriber* baseado em Tópicos. Uma representação sobre o funcionamento pode ser vista na Figura 5. Basicamente existem três tipos de entidades: (I) Um **produtor** (*Publisher*); (II) Um **consumidor** (*Subscriber*); (III) E o **intermediário** entre a comunicação. Este método funciona através de uma lista de notificação, onde os consumidores devem se inscrever em um determinado tópico e toda vez que algum produtor gerar algum dado para aquele tópico este específico, quem estiver inscrito neste receberá a informação.

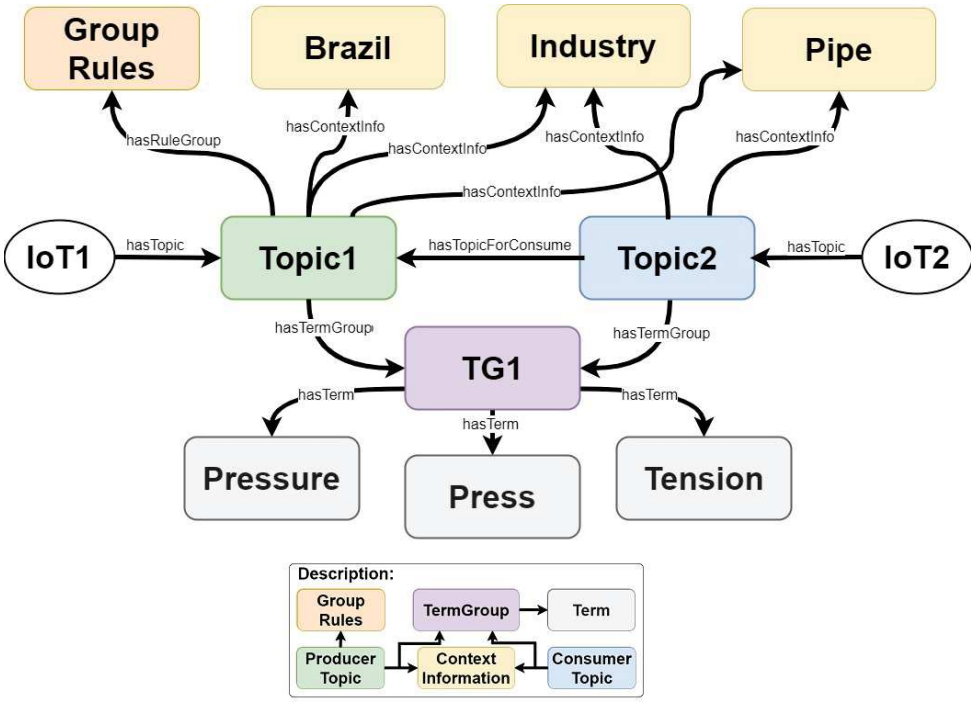
Do ponto de vista mais técnico, os *messages brokers* dão suporte à interoperabilidade de forma parcial até o nível sintático, visto que abstrai de certa forma as questões de

Figura 5 – Visão geral sobre um ***Message Broker*** (Publish/Subscribe) baseado em tópicos



protocolo de comunicação em um nível mais baixo. De certa forma, pode-se considerar que a utilização de tags de marcação (Tópicos) também está relacionado ao nível semântico, porém de forma bem superficial. Com o objetivo de enriquecer a parte semântica e alcançar o nível pragmático, a primeira ontologia foi desenvolvida com base nos tópicos.

Figura 6 – Exemplo simplificado sobre como o relacionamento entre os tópicos é traçado

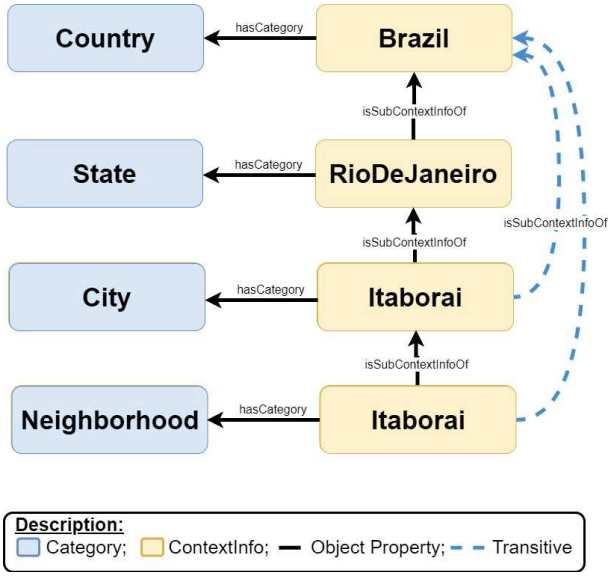


Uma exemplificação do relacionamento entre tópicos pode ser vista na Figura 6 de uma forma simplificada. O conceito de tópico pode ser visto como um termo sobre um determinado domínio. Por exemplo, caso um produtor queira produzir dados de temperatura, este pode utilizar-se do mesmo termo ou de variações destes, como ***Temp***, ***Heat***, ***DegreeHeat***. No entanto, nos métodos convencionais, para consumir dados de algum domínio, seria necessário ter ciência de todos os termos semelhantes disponíveis. Para que o consumidor não precise ter conhecimento sobre tal informação, no momento

em que o dispositivo está sendo descrito na plataforma Web, as informações passam por cálculos de similares com termos e outras informações já inseridas na plataforma e também disponíveis em bases externas a fim de traçar este relacionamento. Para isso, o sub módulo de cálculo de similaridade é utilizado em conjunto com as informações contidas na ontologia e também as informações fornecidas na base externa. Na ontologia então, todos os **termos** que podem ser tidos como semelhantes, são agrupados através de um ***TermGroup***. A partir disto, tópicos podem ser gerados destes ***TermGroup***, sendo estes diferenciados pelo o seu tipo (produtor, consumidor), informações de contexto e/ou regras de produção.

Para uma melhor representação, os tópicos foram divididos em dois grupos, produtor e consumidor. Os dois tópicos podem possuir informações de contexto, mas somente o tipo produtor pode ter regras associadas às produções. As informações de contexto são essenciais tanto para a parte semântica, por enriquecer a descrição, como para parte pragmática, visto que em conjunto com regras, apoiam a interoperabilidade pragmática [Weigand and Paschke 2012] [Wassermann and Fay 2017].

Figura 7 – Relação entre categorias e contextos

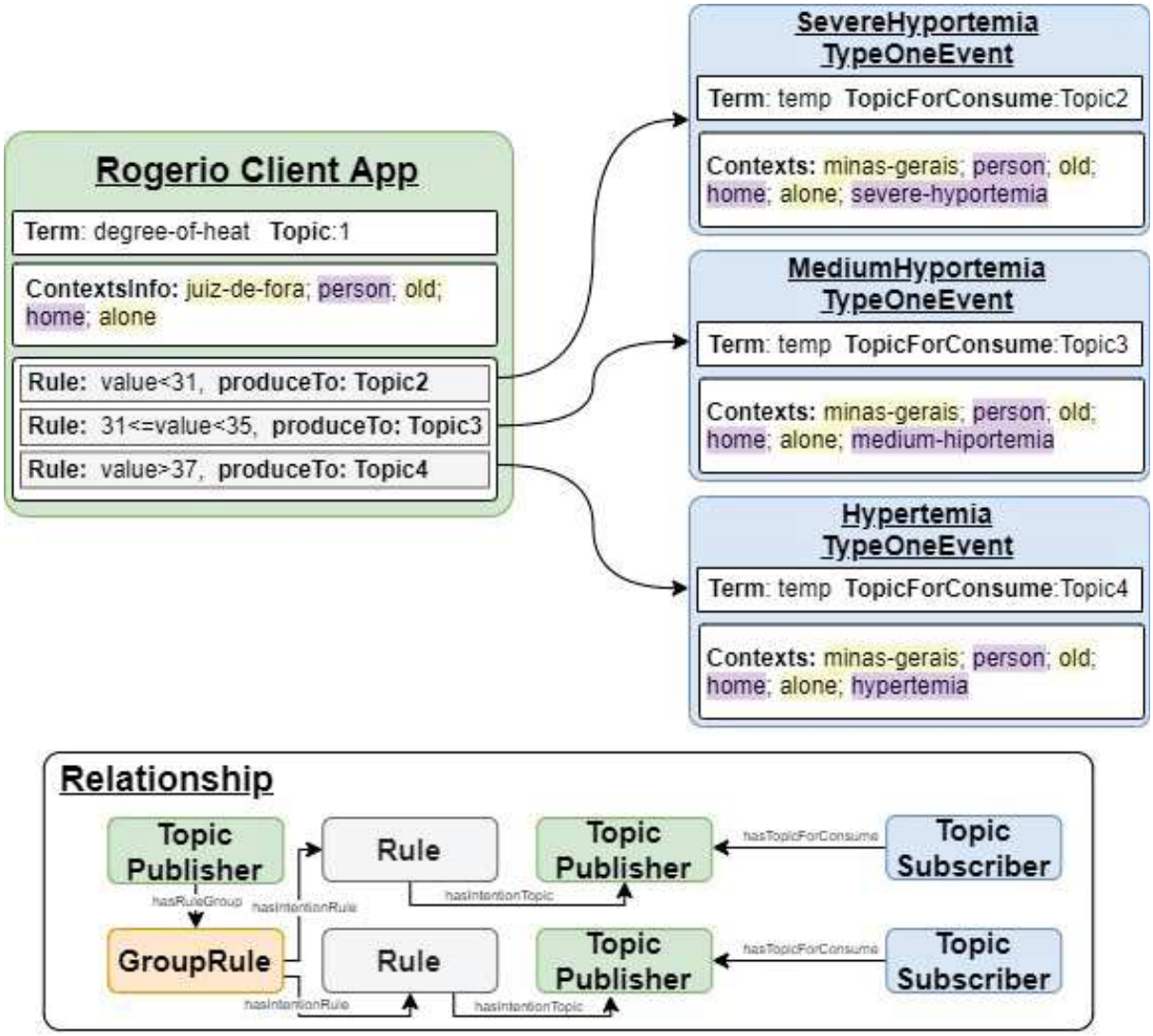


Na ontologia, as informações de contexto podem ser adicionadas em várias níveis de granularidade, como pode ser observado na Figura 7. Através deste método de descrição e também da utilização da **transitividade** e ***Property Chains***, informações antes não explícitas, podem auxiliar a tomada de decisão sobre qual tópico será consumido por quem. Por exemplo, através do modelo elaborado, é possível extrair a informação que, consumir dados provenientes de um país em específico, implica que todas as sub informações de contextos (Estados, cidades, Bairros) deste país, devem ser levados em consideração.

Como já relatado, as regras em conjunto com informações de contexto, auxiliam a alcançar o nível pragmático, uma vez que possibilita a mudança de comportamento. As regras (condicionais) foram elaboradas com base nos tópicos, como pode ser observado na Figura 8. A estrutura foi montada para que um tópico do tipo produtor possa ter uma ou

várias regras, sendo que cada uma destas regras está também vinculada a um outro tópico do tipo produtor.

Figura 8 – Relação entre **Tópicos** e **Regras**



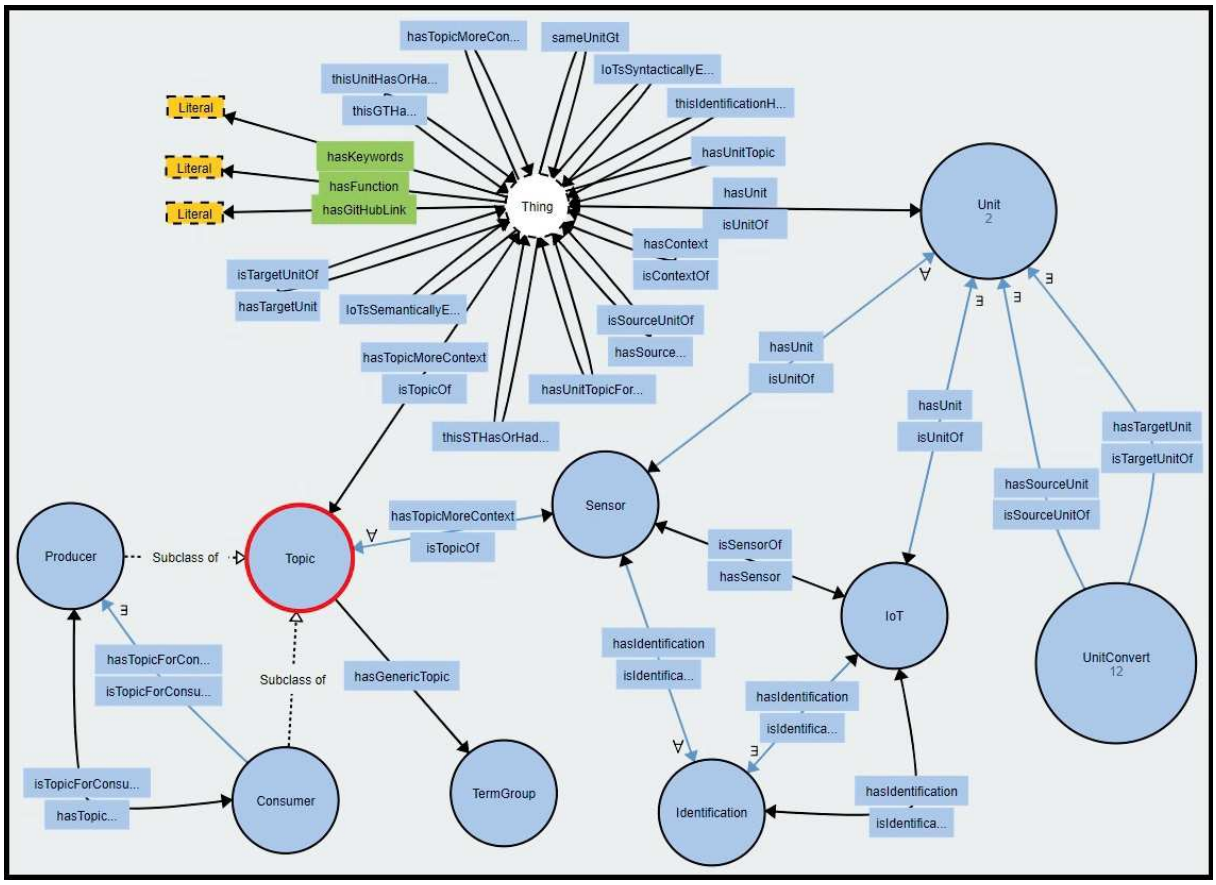
Desta forma, as produções podem ser direcionadas para um grupo específico. Na Figura 8 é possível a visualização de um exemplo de aplicação no contexto da saúde. Neste exemplo, uma aplicação coleta e produz dados para o módulo de comunicação, onde as regras vinculadas a esta aplicação são executadas. Dependendo do fluxo gerado pelas regras, aplicações distintas vinculadas a um hospital podem tomar uma decisão. Como exemplo das regras de produção citadas podemos citar, se o **valor** for **maior** que 37, então produz para o topic2. Ao mesmo tempo o tópico "*HypertemiaTypeOneEvent*"espera este tipo de evento e toma as decições necessárias.

Do ponto de vista da interoperabilidade pragmática, os tópicos contêm informações de contextos e regras para estes. O fluxo gerado pelas regras permite o direcionamento para consumidores específicos. A partir disso, ações podem ser tomadas por dispositivos que

esperam este comportamento. as **regras de produção** nesse trabalho, também podem ser consideradas como **regras pragmáticas**.

Por meio do fluxo gerado entre tópicos produtores e as regras, foi possível utilizar *Property Chains* para descobrir a ligação entre eles. Para isso, a metodologia adotada por [Neiva et al. 2015] para dizer se serviços são equivalentes sintaticamente, semanticamente e/ou pragmaticamente, foi modificada para se aderir ao contexto desta dissertação. No caso do fluxo entre os tópicos e as regras, algo que pode ser observado na Figura 8, originou a *property chain* ***TopicsPragmaticallyEquivalent***. As property chains para níveis sintático e semântico foram descritas na segunda ontologia.

Figura 9 – Segunda Ontologia



A segunda ontologia nasceu com objetivo de descrever as informações dos dispositivos. Esta foi exposta na Figura 9 para representar as relações e na Tabela 5 em forma de linguagem de descrição. Por meio desta ontologias, o relacionamento entre os IoTs é traçado a partir das informações sintetizadas da primeira ontologia. A ideia em ser duas ontologias é diminuir o processamento necessário para que a interoperabilidade possa acontecer. Dessa forma, as informações sobre os tópicos são armazenados em uma maior granularidade na primeira ontologia, enquanto que na segunda as informações possuem um menor granularidade, mas o necessário para interoperar na maior parte das vezes. Dentro

das informações sintetizadas, as regras ficaram de fora, sendo ainda necessária a consulta na primeira ontologia.

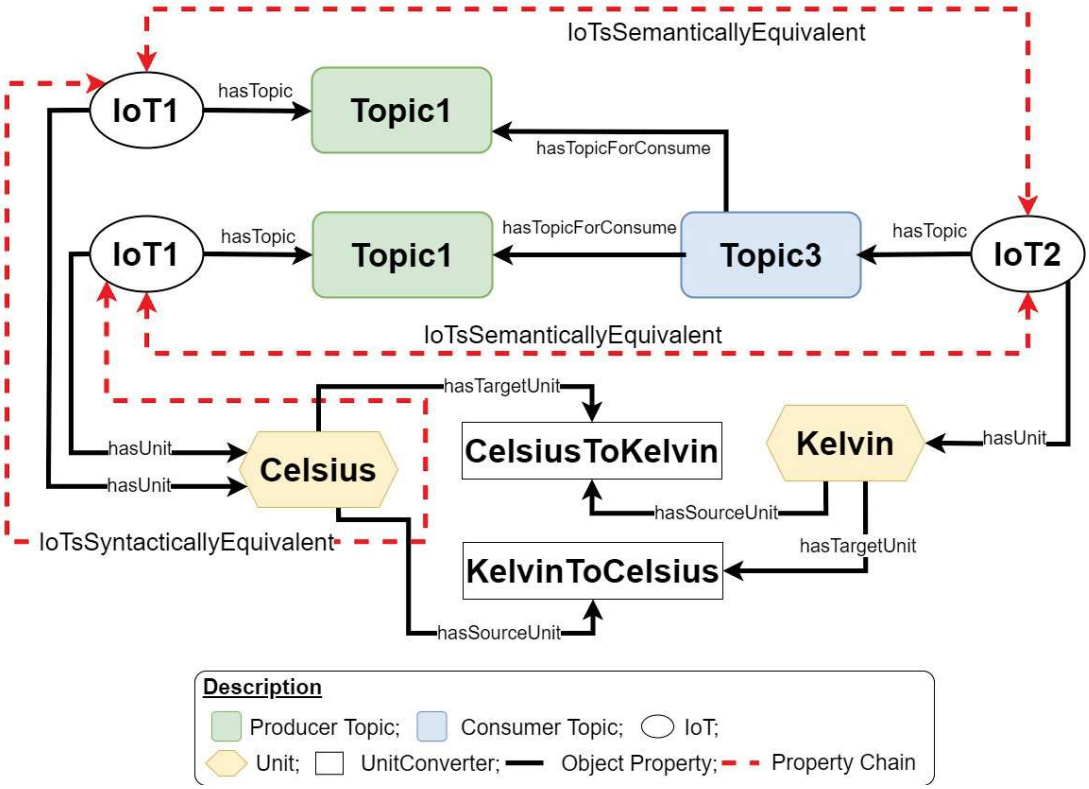
Tabela 5 – Segunda Ontologia: descrição de dispositivos

Classes	
Producer	**TopicID**
Producer $\sqsubseteq$ TopicID	TopicID $\sqsubseteq$ = hasTermGroup TermGroup
Consumer	TopicID $\sqsubseteq \forall$ hasTermGroup TermGroup
Consumer $\sqsubseteq$ TopicID	**Identification**
Consumer $\sqsubseteq \exists$ hasTopicForConsume Producer	Identification $\sqsubseteq$ = isIdentificationOf IoT
IoT	Identification $\sqsubseteq \forall$ isIdentificationOf IoT
IoT $\sqsubseteq$ = hasTopic TopicID	**Unit**
IoT $\sqsubseteq \forall$ hasUnit Unit	**UnitConvert**
IoT $\sqsubseteq$ = hasUnit Unit	UnitConvert $\sqsubseteq$ = hasSourceUnit Unit
IoT $\sqsubseteq \forall$ hasTopic TopicID	UnitConvert $\sqsubseteq \exists$ hasTargetUnit Unit
IoT $\sqsubseteq \forall$ hasIdentification Identification	UnitConvert $\sqsubseteq$ = hasTargetUnit Unit
IoT $\sqsubseteq$ = hasIdentification Identification	UnitConvert $\sqsubseteq \exists$ hasSourceUnit Unit
TermGroup	
Object properties	
IoTsSemanticallyEquivalent	**isIdentificationOf**
IoTsSyntactticallyEquivalent	hasIdentification $\equiv$ isIdentificationOf$^-$
hasContextInf	$\exists$ isIdentificationOf Thing $\sqsubseteq$ Identification
hasContextInf $\equiv$ isContextInfOf$^-$	$\top \sqsubseteq \forall$ isIdentificationOf IoT
hasIdentification	**isSensorOf**
hasIdentification $\equiv$ isIdentificationOf$^-$	hasSensor $\equiv$ isSensorOf$^-$
$\exists$ hasIdentification Thing $\sqsubseteq$ IoT	**isSourceUnitOf**
$\top \sqsubseteq \forall$ hasIdentification Identification	hasSourceUnit $\equiv$ isSourceUnitOf$^-$
hasSensor	$\exists$ isSourceUnitOf Thing $\sqsubseteq$ Unit
hasSensor $\equiv$ isSensorOf$^-$	$\top \sqsubseteq \forall$ isSourceUnitOf UnitConvert
$\exists$ hasSensor Thing $\sqsubseteq$ IoT	**isTargetUnitOf**
hasSourceUnit	hasTargetUnit $\equiv$ isTargetUnitOf$^-$
hasSourceUnit $\equiv$ isSourceUnitOf$^-$	$\exists$ isTargetUnitOf Thing $\sqsubseteq$ Unit
$\exists$ hasSourceUnit Thing $\sqsubseteq$ UnitConvert	$\top \sqsubseteq \forall$ isTargetUnitOf UnitConvert
$\top \sqsubseteq \forall$ hasSourceUnit Unit	**isTermGroupOf**
hasTargetUnit	hasTermGroup $\equiv$ isTermGroupOf$^-$
hasTargetUnit $\equiv$ isTargetUnitOf$^-$	$\exists$ isTermGroupOf Thing $\sqsubseteq$ TermGroup
$\exists$ hasTargetUnit Thing $\sqsubseteq$ UnitConvert	$\top \sqsubseteq \forall$ isTermGroupOf TopicID
$\top \sqsubseteq \forall$ hasTargetUnit Unit	**isTopicOf**
hasTermGroup	hasTopic $\equiv$ isTopicOf$^-$
hasTermGroup $\equiv$ isTermGroupOf$^-$	$\exists$ isTopicOf Thing $\sqsubseteq$ TopicID
$\exists$ hasTermGroup Thing $\sqsubseteq$ TopicID	$\top \sqsubseteq \forall$ isTopicOf IoT
$\top \sqsubseteq \forall$ hasTermGroup TermGroup	**isUnitOf**
hasTopic	hasUnit $\equiv$ isUnitOf$^-$
hasTopic $\equiv$ isTopicOf$^-$	$\exists$ isUnitOf Thing $\sqsubseteq$ Unit
$\exists$ hasTopic Thing $\sqsubseteq$ IoT	$\top \sqsubseteq \forall$ isUnitOf IoT
$\top \sqsubseteq \forall$ hasTopic TopicID	**sameUnitTermGroup**
hasTopicByID	$\exists$ sameUnitTermGroup Thing $\sqsubseteq$ Unit
hasTopicForConsume	$\top \sqsubseteq \forall$ sameUnitTermGroup Unit
$\exists$ hasTopicForConsume Thing $\sqsubseteq$ Consumer	**thisIdentificationHasUnit**
$\top \sqsubseteq \forall$ hasTopicForConsume Producer	**thisTermGroupHasOrHadAnUnit**
hasUnit	thisTermGroupHasOrHadAnUnit $\equiv$ thisUnitHasOr-HadThisTermGroup$^-$
hasUnit $\equiv$ isUnitOf$^-$	$\exists$ thisTermGroupHasOrHadAnUnit Thing $\sqsubseteq$ Term-Group
$\exists$ hasUnit Thing $\sqsubseteq$ IoT	$\top \sqsubseteq \forall$ thisTermGroupHasOrHadAnUnit Unit
$\top \sqsubseteq \forall$ hasUnit Unit	**thisTopicHasOrHadAnUnit**
hasUnitTopic	**thisUnitHasOrHadThisTermGroup**
hasUnitTopicForConsume	thisTermGroupHasOrHadAnUnit $\equiv$ thisUnitHasOr-

isContextInfOf hasContextInf ≡ isContextInfOf⁻	HadThisTermGroup⁻ ∃ thisUnitHasOrHadThisTermGroup Thing ⊑ Unit ⊤ ⊑ ∀ thisUnitHasOrHadThisTermGroup TermGroup
Data properties	
hasFunction **hasGitHubLink**	**hasKeywords**

Como já relatado, informações sobre os tópicos são sintetizadas com uma menor granularidade na segunda ontologia para possibilitar um melhor desempenho. A forma como foi elaborada esta sintetização permite que um tópico possa ser inserido ou removido da segunda ontologia sem que as informações atreladas a este se percam. Desta forma um tópico só existirá na segunda ontologia caso este esteja vinculado a um dispositivo IoT.

Figura 10 – Relação entre IoTs



Como pode ser observado na Figura 10, o relacionamento entre os dispositivos IoTs é traçado por meio da utilização da *property chain* ***IoTsSemanticallyEquivalent*** formada por três *object property* (hasTopic, isTopicForConsumeOf, isTopicOf) para dizer se um dispositivo é semanticamente equivalente e a *property chain* ***IoTsSyntacticallyEquivalent*** formada por duas *object property* (hasUnit, isUnitOf). Por meio da utilização destas *property chain* é possível determinar em quais níveis os dispositivos podem interoperar (até o nível semântico, para o pragmático é necessário acessar a outra ontologia).

Na Figura 10 também é possível visualizar uma relação entre as unidades através de uma classe para conversão de unidades. Indivíduos desta classe guardam as informações

necessárias para que a conversão possa ser feita de uma unidade para outra. O método utilizado para descrição de uma função de conversão, adotado na construção da plataforma, permite que através de uma única descrição entre duas unidades, várias outras descrições possam ser realizadas de forma automática entre as unidades do mesmo grupo. Por exemplo, Se existe uma função de conversão da unidade **A** para a **B**, uma nova descrição da unidade **C** para **A** gera também uma função de conversão da unidade **C** para a **B**. Os passos para que isso ocorra são:

- Descrição da unidade;
- Identificação das unidade semelhantes;
- Criação das funções para as outras unidade do grupo;
- Simplificação das funções de conversão;
- Criação das funções inversas;

Vale destacar, que para esta dissertação, o foco da parte sintática foi em unidades atômicas. No entanto, a forma como este nível foi tratado permite a adequação para o modelo não atômico.

3.3.2.2 Cálculo de similaridade

Para que a descrição dos tópicos possa ser mais precisa e mais clara, um módulo de cálculo de similaridade foi criado. Este módulo foi baseado no formato criado em [Neiva et al. 2015] . Este trabalho tem por objetivo apoiar a interoperabilidade pragmática em pesquisas científicas, mais especificamente na parte da experimentação. Como parte da solução, um módulo foi desenvolvido para o cálculo de similaridade entre serviços. Este consiste em utilizar-se do processamento de linguagem natural sobre as informações sintáticas, semânticas e pragmáticas a partir do resultado gerar um lista em ordem crescente dois serviços com a maior similaridade ao buscado por um usuário.

Para essa dissertação, a mesma ideia foi implementada para apoiar a descrição dos tópicos. No momento da descrição, o cálculo da similaridade é realizado entre as informações passadas pelo usuário e as informações contidas na ontologia e no endpoint externo (Este processo será melhor exemplificado no seção da implementação). Logo após, um lista com possíveis termos, ordenado pela maior similaridade, é devolvido para o usuário para que este possa optar em escolher algum dos já existentes, ou inserir algum novo. As informações passadas por este usuário são o termo e uma descrição.

Para enriquecer o cálculo de similaridade, o endpoint para sinônimos do grupo ***The STANDS4 Network***[3] foi utilizado. Este endpoint permite a pesquisa por um termo

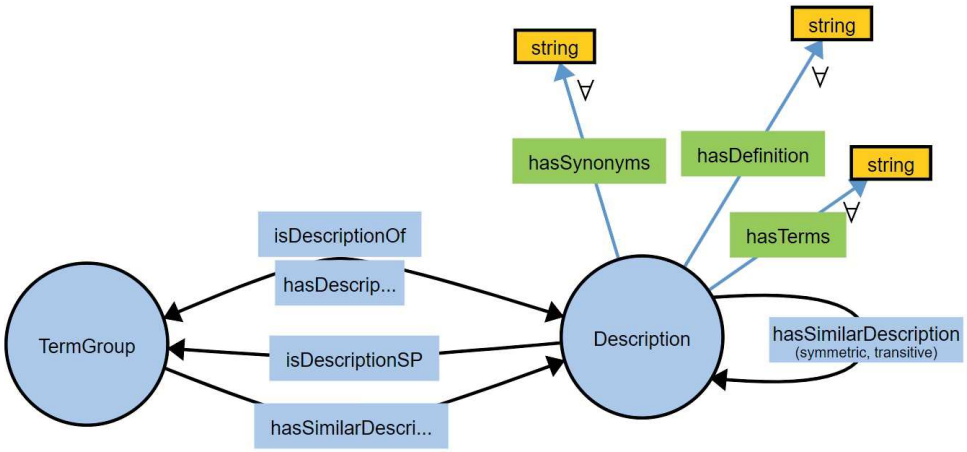
[3] https://www.synonyms.com/

qualquer e um JSON contendo várias definições em conjunto com os termos atrelados a aquela definição. A partir do momento que a plataforma web realiza a consulta de um termo específico, este fica registrado localmente para pesquisas futuras.

Os passos podem ser descritos como:

- Usuário informa um Termo e uma descrição para este;
- Uma consulta é realizada no endpoint externo (caso já exista localmente, o arquivo é consultado)
- Cálculos de similaridade são realizados entre as informações expostas pelo usuário e as informações retornadas pelo usuário;
- Uma lista é exposta para usuários com as descrições mais similares;
- Caso usuário escolha alguma das descrições, um relacionamento é feito com outros termos já existente na ontologias;

Figura 11 – Relação entre as descrições e os *TermGroups*



Os termos e as descrições são armazenados na ontologia como pode ser observado na Figura 11. A partir das similaridades entre descrições, relacionamento entres descrições também são realizados, gerando também relacionamentos indiretos entre termos de grupos distintos.

3.3.2.3 *IoT-InterArch* - Endpoints

Para dar suporte à interoperabilidade sintática, semântica e pragmática dentro do ecossistema interno e externo (ecossistema que não utiliza a estrutura a solução para realizar a comunicação, somente as informações contidas nesta), um WebService foi criado. Os endpoints criados podem ser classificados em dois grupos, interno e externo. O enpoints internos foram criados para que a comunicação entre o **módulo de comunicação** (**COM**),

módulo de **descrição** (**DESC**) e o módulo de **inteligência artificial** (**IA**) fosse possível. Basicamente, estes são para que a comunicação interna possa acontecer. A utilização deste endpoints permite que as várias possíveis instâncias do Módulo COM e IA possam estabelecer uma comunicação facilitada com o DESC. Esta comunicação poderia ser feita utilizando os próprios **message brokers**. No entanto, o fato destes serem assíncronos inviabiliza o processo de requisição e resposta. Os endpoints criados podem ser observados na Tabela 6.

Tabela 6 – *Endpoints* Internos

Module	**Endpoint**	**Path Parameter**
COM	PrepareProducer	Id
COM	UpdateRules	Id, R1-Rn
COM	CloseProducer	Id
DESC	PrepareConsumer	Id
DESC	GetFunctionsForConvert	Id
DESC	GetProducerInformation	Id
DESC	GetUnitByIot	Id
DESC	GetClassificationModuleThree	Input
DESC	SetModel	Id

O segundo grupo de endpoints, o externo, foi criado para dar um suporte à interoperabilidade sintática, semântica e pragmática em ambientes externos ao desenvolvido.Estes podem ser visualizados na Tabela 7. Estes endpoints consultam as ontologias, os módulos desenvolvidos e também outras informações armazenadas.

Tabela 7 – *Enpoints* Externos

Endpoint	**Path Parameter**
GetSameUnits	Unit, Type
GetFunction	UnitOne, UnitTwo
GetSimilarTerms	Term
GetSimilarityBetweenTwoTerms	TermOne, TermTwo
GetInternalInformationAboutTerm	Term
GetExternalInformationAboutTerm	Term
GetAllContextInfGroup	Term
GetSuperContextOfTerm	Term
GetSuperContextChildren	Term
GetModel	Term
GetOntologyFile	Type
GetPragmaticRules	Term, EC1-ECn

GetClassification	Term, EC1-ECn
GetClassificationObject	Term

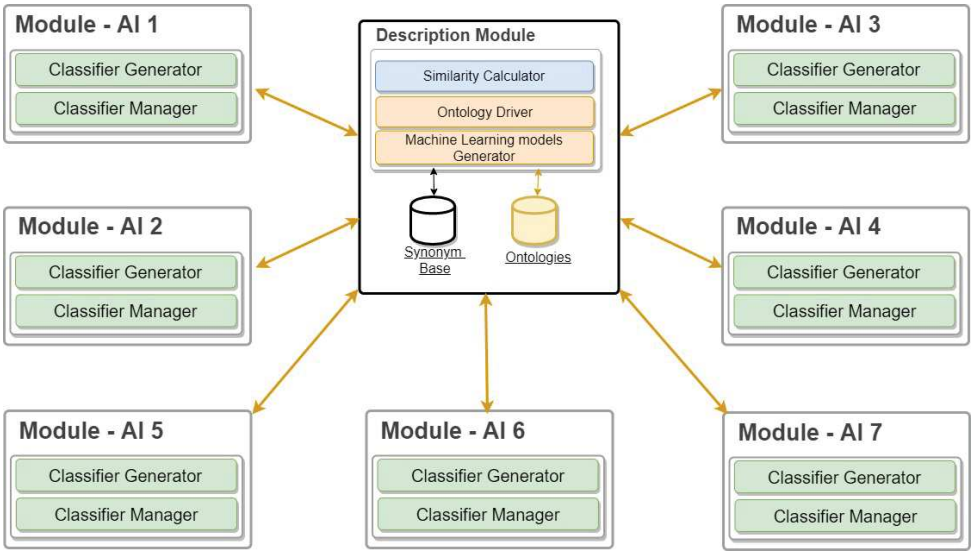
Através destes endpoints é possível (Seguindo a ordem estabelecida na Tabela 7): **(i)** Verificar unidades semelhantes; **(ii)** Requisitar funções de conversão entre duas unidade; **(iii)** Buscar termos similares; **(iv)** Calcular a similaridade entre dois termos (ou sentenças); **(v)** Buscar informações sobre um termo inserido na plataforma; **(vi)** Buscar informações sobre um termo a partir dos dados fornecidos pelo endpoint de sinônimos; **(vii)** Requerer todas as informações de contexto de um termo; **(viii)** Buscar as informações de contexto do nível mais alto de uma estrutura; **(ix)** Buscar os filhos do enpoint anterior; **(x)** Requisitar o modelo gerado para o módulo de Inteligência Artificial (o intuito é permitir que clientes possam rodar os próprios algoritmos de **machine learning** em suas máquinas); **(xi)** Expor os arquivos da ontologia; **(xii)** Buscar regras de produção, para isto o usuário deve informar o termo e as informações de contexto a fim de verificar se já existe regras para aquele contexto; **(xiii)** Buscar regras de produção também, com a diferença que este endpoint serve como opção caso o endpoint anterior não retorne regras, pois estas não existem (Na próxima subseção, isto será melhor explicado), para isto usa-se do módulo de inteligência artificial para gerar estas regras; **(xiv)** Buscar um objeto serializado (Objeto java) de uma classificador (Pronto para ser utilizado).

3.3.3 *Artificial Inteligence Module*

A utilização de regras de produção permite que fluxos possam ser criados de acordo com o que é produzido levando em consideração o contexto. No entanto, a mudança de contexto, pode acarretar na mudança também nas regras e consequentemente na intenção sobre o que está sendo produzido. Visto que, o processo de descrição de regras para todos os contextos descritos na ontologia é algo árduo, principalmente por estar em constante evolução, torna-se necessário o uso de outras tecnologias para dar este suporte. Baseado nisto, o uso da inteligência artificial, mais especificamente *machine learning* com aprendizado supervisionado (Como as informações estão classificadas em grupos, este modelo se adere bem), foi escolhido para o suporte, visto que permite a descoberta de padrões em um conjunto de dados.

Para que isto seja possível, o módulo de inteligência artificial (IA) foi criado de uma forma que permita diversas instâncias com o intuito de apoiar a escalabilidade do sistema. A ideia também em se ter diversas instância também é permitir que os diversos contextos existentes possam ser tratados de forma isolada. Por exemplo, pode-se supor a existência dos contextos de **pressão**,**glicose** e **umidade**. Cada um destes contextos, por não serem similares, serão tratados em grupos diferentes. Dessa forma, se existem três instâncias do módulo IA, então cada um dos contextos será direcionado para uma das

Figura 12 – Relação entre o módulo de descrição e as possíveis instâncias do módulo de inteligência artificial



instâncias.

Atualmente, o módulo de **descrição** (DESC) possui um sistema de gerenciamento das instâncias do módulo **IA**, como pode ser visto na Figura 12. O DESC utiliza o sistema de lista circular para direcionar os diferentes contextos para as instância Ia existentes. Além disso, o DESC gera também os modelos necessários para os classificadores do **machine learning** possam funcionar. Os classificadores são os responsáveis por gerar as inferências sobre um novo input. Como entrada, os classificadores recebem informações de contexto e tem como saída as regras. Dessa forma, novas configurações de informação de contexto podem ser submetidas aos classificadores e como resultado da análise de padrões, regras são expostas. Para que estas informações possam ser expostas, o endpoint ***GetClassification*** foi criado. A forma como o sistema foi elaborado, permite que cada contexto tenha um algoritmo diferente (dentro dos que são disponibilizados pelo Weka[4]).

Para essa dissertação o algoritmo **J48** foi escolhido com o padrão para a classificação. Este algoritmo atende diversos contextos, mas isso não significa que o mesmo seja o melhor. No entanto, o algoritmo pode facilmente ser substituído. Caso algum usuário/aplicação queira utilizar algum outro algoritmo de forma local, o *endpoint* ***GetModel*** pode ser utilizado para se obter o modelo gerado a partir da ontologia. O *endpoint* criado para gerar a classificação (***GetClassification***) também expõe as informações relacionadas a acurácia do classificador, com o objetivo em apoiar a tomada de decisão em optar pelas novas regras ou não. Caso queira, o *endpoint* ***UpdateRules*** do módulo de comunicação, pode ser utilizado para atualizar as regras de um produtor.

[4] https://www.cs.waikato.ac.nz/ml/weka/

3.3.4 Plataforma Web

Com o objetivo de facilitar a descrição destas, uma plataforma web foi desenvolvida. Por meio desta, tópicos e IoTs podem ser descritos e vinculados (através da descrição). A plataforma se utiliza do cálculo de similaridade quando novos termos estão sendo inseridos.

Na Figura 13 é possível visualizar o fluxo de criação de um tópico quando um termo (por exemplo, *pressure*) já existe na ontologia. Este fluxo contempla uma descrição geral sobre o tópico, como termo, o tipo de tópico (produtor ou consumidor) e as palavras chaves vinculadas a este (Figura 13-1.A). O próximo passo, caso se escolha um termo já existente, seria a escolha das informações de contexto deste tópico (Figura 13-1.B), sendo possível aumentar o nível de granularidade da informação, baseado no modelo exemplificado na Figura 7. O passo seguinte, o 1.C, é opcional e só é possível ir para este passo, caso o tópico seja do tipo produtor, visto que é a descrição de regras de produção (regras pragmáticas). Neste passo, regras de produção podem ser descritas no formato de condições. Como passo final, o IoT pode ser descrito e vinculado ao tópico criado ou a um existente(Figura 13-2.A).

No que se refere a descrição do tópico, para a inserção de um termo novo, um fluxo deve ser seguido entre os passos 1.A e 1.B da Figura 13. Este fluxo pode ser observado na Figura 14.

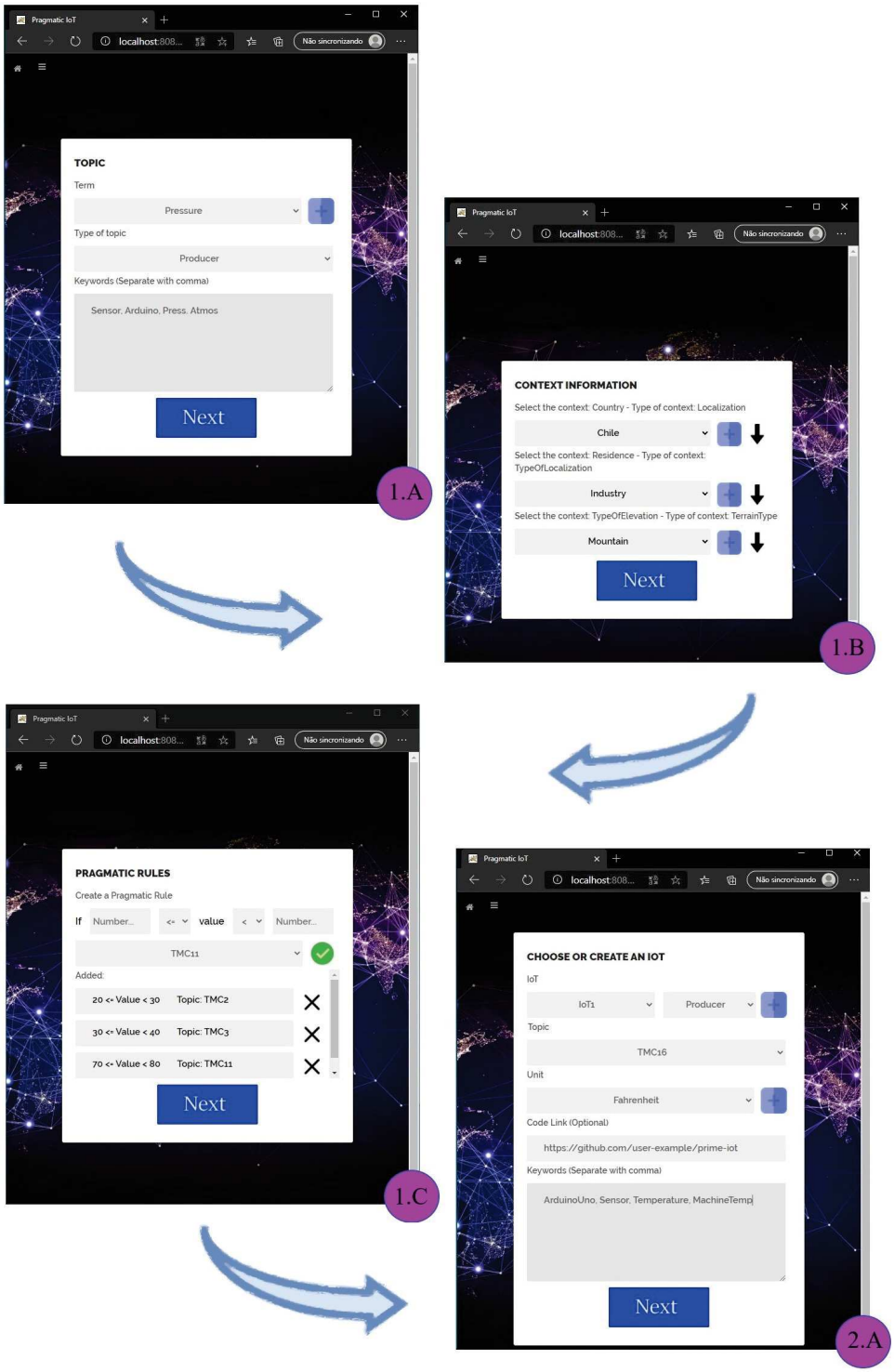
O desvio de fluxo é necessário para que possíveis relacionamentos possam ser traçados entre termos já existentes. Na tela A.1, são retornados termos, descrições e sinônimos ao novo termo inserido. Estas informações são capturadas através de uma base externa por meio de um endpoint. Através desta tela, o usuário pode escolher a opção que mais está aderente ao seu contexto. O intuito neste passo é possibilitar um cálculo de similaridade a partir de um volume maior de informação. Na próxima tela A.2, uma lista de termos já existentes, ordenados de acordo com a similaridade ao novo termo é retornada para o usuário, podendo este escolher ou não algum dos termos. Caso o usuário escolha algum termo, o fluxo original segue normalmente. Caso contrário, na tela A.3, deve-se escolher quais tipos de contexto este novo termo deverá ter. Após isto, o fluxo volta ao original (Figura 13 1.B).

3.3.5 Fluxo da interoperabilidade

A solução desenvolvida neste trabalho permite que dispositivos IoTs possam interoperar em diferentes níveis até o pragmático. Com o objetivo de esclarecer os diferentes fluxos de interoperabilidade o diagrama representado nas Figuras 15 e 16 foi elaborado (Produtor e consumidor respectivamente). Testes funcionais [5] também foram realizados como forma de testar a comunicação entre os módulos. Na Figura 17 é possível uma

[5] https://drive.google.com/file/d/1K8OzB2Mf-jfE72Ibj8BaN5p8Iz4bLaeo/view?usp=sharing

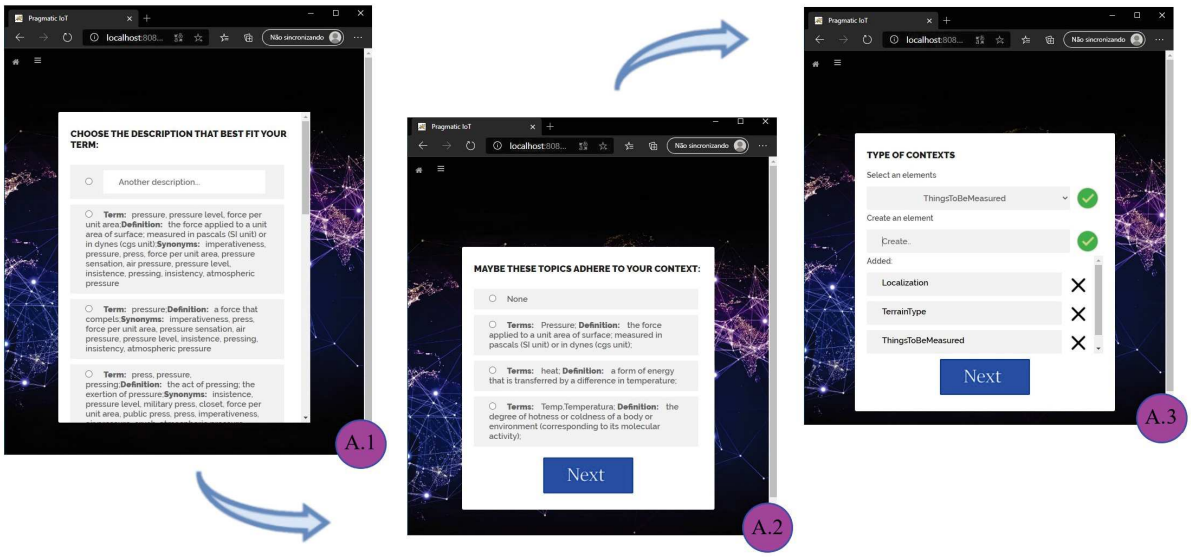
Figura 13 – Fluxo para a criação de um tópico e vínculo a um dispositivo, quando um termo já existe



visualização dos testes funcionais. Como cenário de teste, um contexto hospitalar onde se captura informações de temperatura de pacientes foi utilizado.

O primeiro fluxo considerado foi onde não existe o suporte à interoperabilidade. Neste fluxo, o uso da plataforma desenvolvida torna-se desnecessário, visto que a comunicação é direta sem nenhum tratamento da informação. Para exemplificação, nos testes

Figura 14 – Desvio de fluxo, quando o termo chave não existe, ou não é selecionado



funcionais, uma aplicação coletava dados de temperatura de um paciente e produzia para a *tag* "temperatura"utilizando o kafka. Ao mesmo tempo, uma aplicação responsável por coletar este tipo informação através da mesma tag utilizada pelos produtores citados anteriormente.

O segundo fluxo considerado foi o do nível sintático de interoperabilidade. Neste nível, uma requisição aos endpoints desenvolvidos (Localizados no **módulo de descrição**) é necessário antes que aplicações possam produzir/consumir. Após a requisição das informações, os dispositivos começam a produzir/consumir utilizando sem o auxílio da plataforma desenvolvida neste trabalho (Exceto casos de atualizações das informações). Para exemplificação, nos testes funcionais, os produtores enviaram dados sobre a *tag* "temperatura"utilizando-se de alguma unidade de temperatura (por exemplo, **celsius**) sem se preocupar com o formato, visto que os consumidores recebiam as informações necessárias para realizar a conversão entre unidades do mesmo grupo (Por exemplo, **celsius** para **kelvin**).

O terceiro e quarto fluxo são respectivamente sobre o nível semântico e pragmático. Assim como no segundo fluxo, uma requisição das informações necessárias também é realizada para o **módulo de descrição**. Como já relatado, a partir do nível semântico, tópicos foram utilizados para representar um grupo de tag (termos) em um contexto específico. Nos testes funcionais, por exemplo, as tags "temp", "temperaturagrauDeCalor"podem ser considerados de um mesmo tópico caso estes possuam o mesmo contexto (No caso dos testes, contexto hospitalar). Além disso, um tópico (tipo produtor) pode ou não ter regras de produção.

Como um tópico pode conter regras de produção, necessárias para o nível pragmático

Figura 15 – Fluxo da interoperabilidade - Produtor

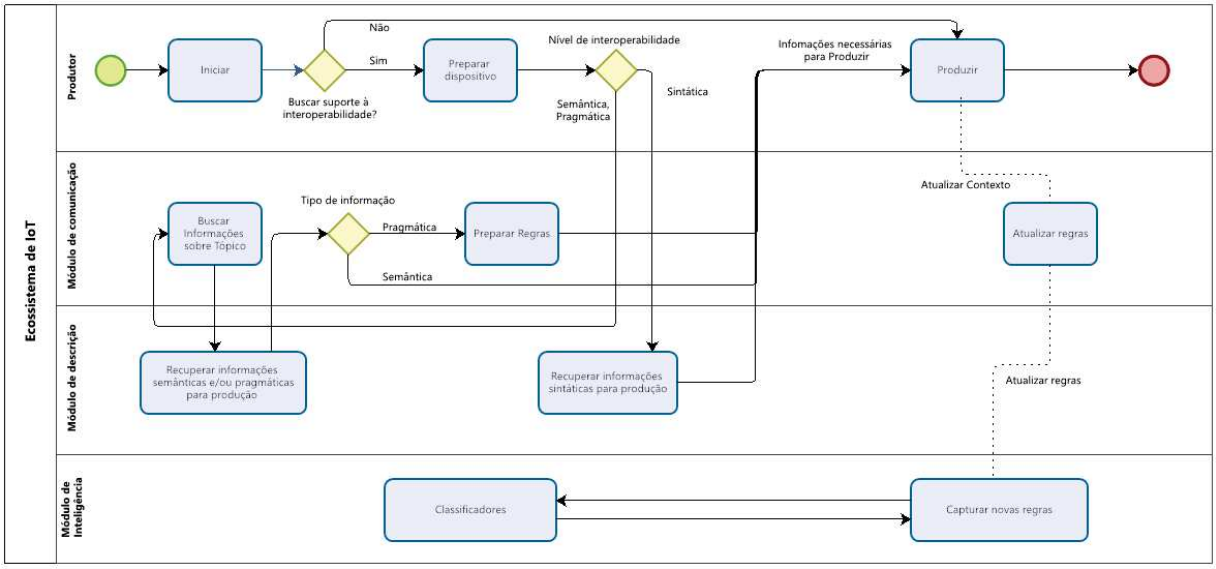
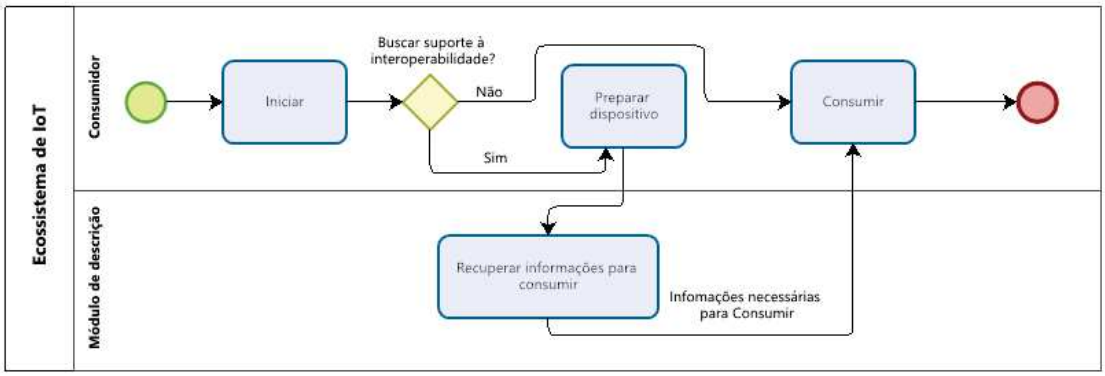


Figura 16 – Fluxo da interoperabilidade - Consumidor

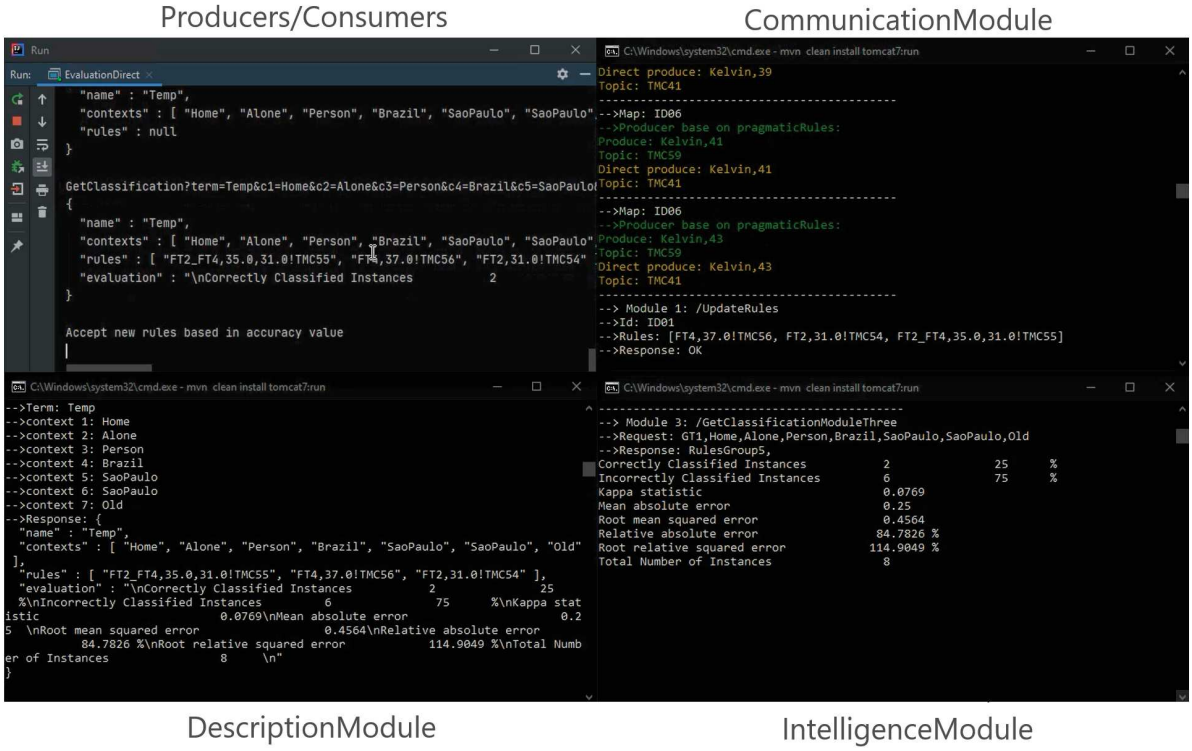


neste trabalho, a requisição ao **módulo de descrição** é feita de forma indireta. A comunicação é realizada para o **módulo de comunicação** e este realiza uma chamada para o **módulo de descrição**. O módulo de comunicação é necessário como intermediador, visto que este gera os eventos necessários para que as regras sejam executadas toda vez que houver um produção para o tópico ”pragmático”.

Nos últimos três fluxos, após a requisição das informações necessárias, a produção é realizada através de uma *message broker* (No caso deste trabalho, o Kafka). O mesmo vale para o consumo das informações. No último fluxo (nível pragmático), como já mencionado, um evento é adicionado para que a produção de um dado possa passar pelas regras (No **módulo de comunicação**). Além disso, caso exista uma mudança no contexto das informações, um novo fluxo pode ser gerado para o nível pragmático de interoperabilidade. Este fluxo consiste em passar o novo contexto para o módulo de inteligência, para que novas regras possam ser inferidas pelos classificadores. Após isso, dependendo do nível de acurácia do classificador e do nível de tolerância do dispositivo (nível mínimo de acurácia

definido pelo dispositivo), as regras podem ser atualizadas.

Figura 17 – Testes funcionais - Nível pragmático



3.4 Considerações finais do capítulo

Este capítulo apresentou uma arquitetura para dar suporte à interoperabilidade pragmática em IoT, levando em consideração as limitações da IoT. Neste capítulo os conceitos técnicos em conjunto com o projeto e a implementação foram apresentados. No capítulo seguinte, a avaliação da abordagem proposta será apresentada por meio da utilização de estudos de caso.

4 AVALIAÇÃO DA SOLUÇÃO

Este capítulo apresenta uma avaliação da solução proposta, a qual busca avaliar uma arquitetura para apoiar a interoperabilidade sintática, semântica e pragmática no Ecossistema de IoT. O objetivo é possibilitar apoiar o desenvolvimento de software IoT, levando em consideração os atributos de qualidade necessários para se construir um ecossistema de software e também os desafios e limitações presentes na IoT. A arquitetura desenvolvida neste trabalho, pode ser denotada como IoT-InterArch e a partir desta, o serviço IoT-InterServ [1] foi desenvolvido.

O capítulo começa apresentando a metodologia de avaliação, onde um estudo de caso foi escolhido como método de avaliação. Após isso, o planejamento da avaliação foi apresentado, onde a contextualização é definida para o estudo de caso. Logo após, os cenários de avaliação são definidos e expostos, assim como a preparação necessária para a avaliação. E por fim a execução e descrição do estudo de caso regular.

4.1 Definição da metodologia

De acordo com [Basili and Weiss 1984], o estabelecimento de um conjunto de objetivos deve ser um elemento primário para o processo de coleta de dados de uma metodologia investigativa. A partir deste conjunto de de objetivos podem ser derivadas questões de pesquisa. Para que o processo funcione de uma forma coerente, projetar um formulário de coleta de dados, validação e análise é algo essencial.

Os estudos de casos são utilizados em contextos reais e possuem o intuito de trazer mudanças sobre os fenômenos, além de agregar conhecimento sobre o estudo. Estes são utilizados em diversas áreas, como psicologia, medicina e negócios. Dentro da engenharia de software, estudos de casos são utilizados para avaliar situações reais em que o pesquisador possui pouco controle sobre as variáveis [Yin 2018].

De acordo com [Runeson et al. 2012], as principais etapas de uma estudo de caso podem resumidas em cinco partes: **(i) desenho do estudo de caso**, onde o planejamento, como por exemplo a contextualização do ambiente; **(ii) preparação da coleta de dados**, através preparação do ambiente e dos mecanismos para a coleta de dados e logs; **(iii) coleta das evidências**, através da execução dos procedimentos; **(iv) análise dos dados coletados**; e por fim **(v) geração de relatórios** sobre o estudo.

Com o objetivo de auxiliar a primeira etapa do estudo de caso, o método GQM (Goal / Question / Metric) [Solingen and Berghout 1999] foi utilizado para analisar a estratégia de interoperar no nível sintático, semântico e pragmático dentro do Ecossistema de IoT. Baseando-se no método GQM, o objetivo deste estudo de caso é:

[1] https://github.com/matheushs1995/IoT-InterServ

- **Analisar** a arquitetura e o serviço (instância) integrado desenvolvido, denominado IoT-InterServ.
- **Com o objetivo** de apoiar interoperabilidade.
- **Sobre a perspectiva** dos desenvolvedores.
- **No contexto** de um ecossistema de IoT.

Através do escopo da pesquisa, uma questão de pesquisa pode ser derivada: "**Como a arquitetura e o serviço originado desta apoia a interoperabilidade em ecossistemas de IoT?**". Por meio, desta questão uma hipótese nula (H0) e uma alternativa (H1) foram desenvolvida, sendo estas:

H0	A arquitetura e o serviço integrado desta **não apoia a interoperabilidade** em ecossistema de IoT.
H1	A arquitetura e o serviço integrado desta **apoia a interoperabilidade** ecossistema de IoT.

As seções que se seguem apresentam os passos seguintes do estudo de caso. Nestes passos, a questão de pesquisa será respondida e as hipóteses serão validadas. As próximas etapas relatadas são o planejamento do estudo de caso, a preparação deste, o processo de coleta de dados. Após isso, o estudo é avaliado.

4.2 Planejamento

Após definição das hipóteses e da questão de pesquisa, baseada no modelo GQM, foram definidas as atividades que foram avaliadas com o intuito de contextualizar a metodologia de avaliação.

Como teste preliminares, testes funcionais foram realizados (Seção 3.3.5) sobre o sistema e só após isso, este estudo regular foi aplicado sobre os participantes.

4.2.1 Contextualização

Para responder a questão de pesquisa elaborada na seção 4.1, cinco cenários de avaliação foram elaborados para representar os fluxos de um contexto real de utilização que pode ser agregado ao IoT Ecosystem. Estes cenários serão demonstrados mais adiante em conjunto com os possíveis fluxos que as regras podem gerar.

As atividades relatadas podem ser resumidas como a descrição através da plataforma Web dos dispositivos, das relações entre estes, e das regras de produção que serão utilizadas. Após estas atividades, outras são desempenhadas pelo próprio participante/desenvolvedor na elaboração do software, como por exemplo, a configuração dos produtores e consumidores

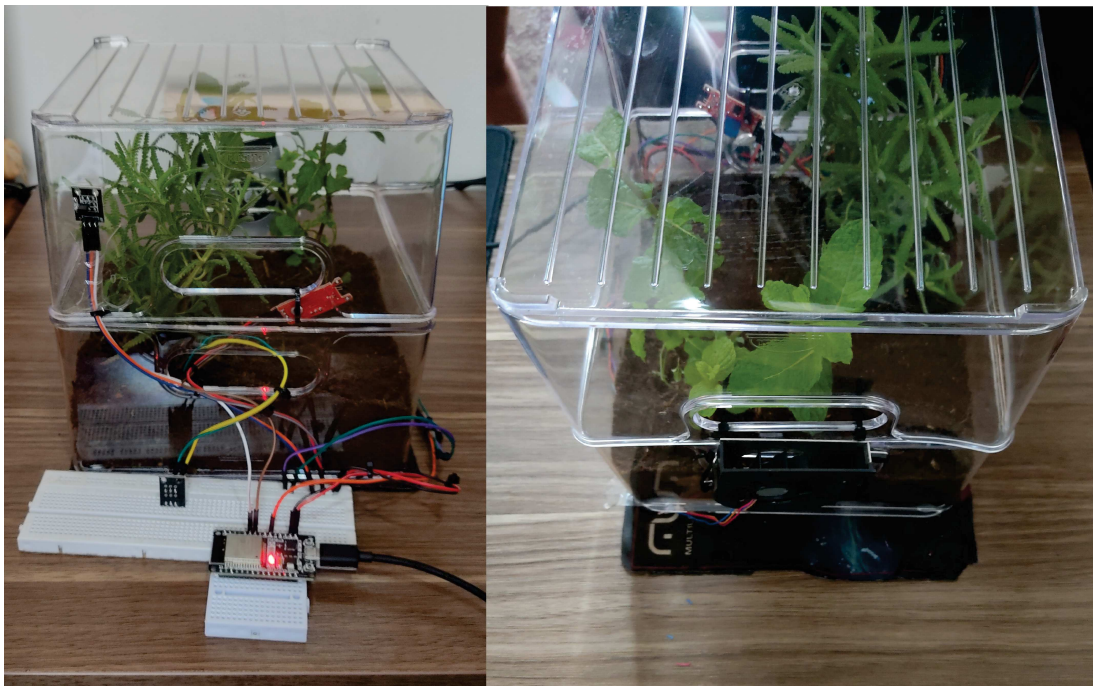
sendo que estas atividades são simplificadas em decorrência das anteriores. Como atividade relacionada à avaliação, todo dado produzido e consumido é observado através do uso de logs em ambas as partes da comunicação. Desta forma, é possível averiguar como a interoperabilidade está acontecendo.

De acordo com [Silva and Travassos 2020], a IoT poderá interligar vários dispositivos, sendo infinitas as possibilidades. No entanto, apesar de ser diversa as possibilidades de combinações, em um ambiente físico, as interações entres os produtores, consumidores e atuadores, possuem ligações e eventos finitos que podem ser representados facilmente através de cenários. Por esta razão, cenários foram elaborados para avaliar a solução proposta.

4.2.2 Preparação

Para avaliar a solução proposta, uma estufa foi construída, como pode ser visto na Figura 18. Foram utilizados três sensores inicialmente, um principal que mede a umidade do solo e dois de temperatura que servem para averiguar a temperatura interna e externa da estufa. Em um sistema comum, estes iriam produzir dados a todo momento.

Figura 18 – Foto da estufa



Como forma de monitorar a estufa sem a necessidade de utilizar vários sensores ao mesmo tempo, regras de produção (pragmáticas) foram implementadas, possibilitando a utilização de um único sensor ligado por vez. O objetivo é ativar os controladores via demanda e desta forma demonstrar como a escalabilidade, flexibilidade e extensibilidade do sistema se comportam. A partir dos possíveis fluxos, cinco cenários foram elaborados.

Para a realização desta avaliação, as aplicações foram desenvolvidas em módulos, para que este pudesse ser executado em uma mesma máquina ou em várias. Todas as aplicações foram desenvolvidas em Java (todas as aplicações que produzem ou consomem os dados) e em C no caso do microcontrolador. Em relação aos componentes físicos, os seguintes foram utilizados:

- **ESP32 NODEMCU** como controlador.
- **LM393** acoplado a um sensor genérico de umidade do solo.
- **KY-028** um sensor de temperatura digital para medir a temperatura interna.
- **Ds18b20** um sensor de temperatura digital para medir a temperatura externa.

4.2.3 Descrição dos dispositivos e das aplicações

Para que os dispositivos possam interoperar é necessário que descrições sejam realizadas nas ontologias. Como forma de facilitar este processo uma plataforma Web foi desenvolvida. As descrições dos dispositivos e das aplicações são bem semelhantes. Como forma de demonstrar o processo, será apresentado a descrição de um dos dispositivos, o responsável por produzir dados da temperatura interna da estufa. O processo pode ser visualizado na Figura 19.

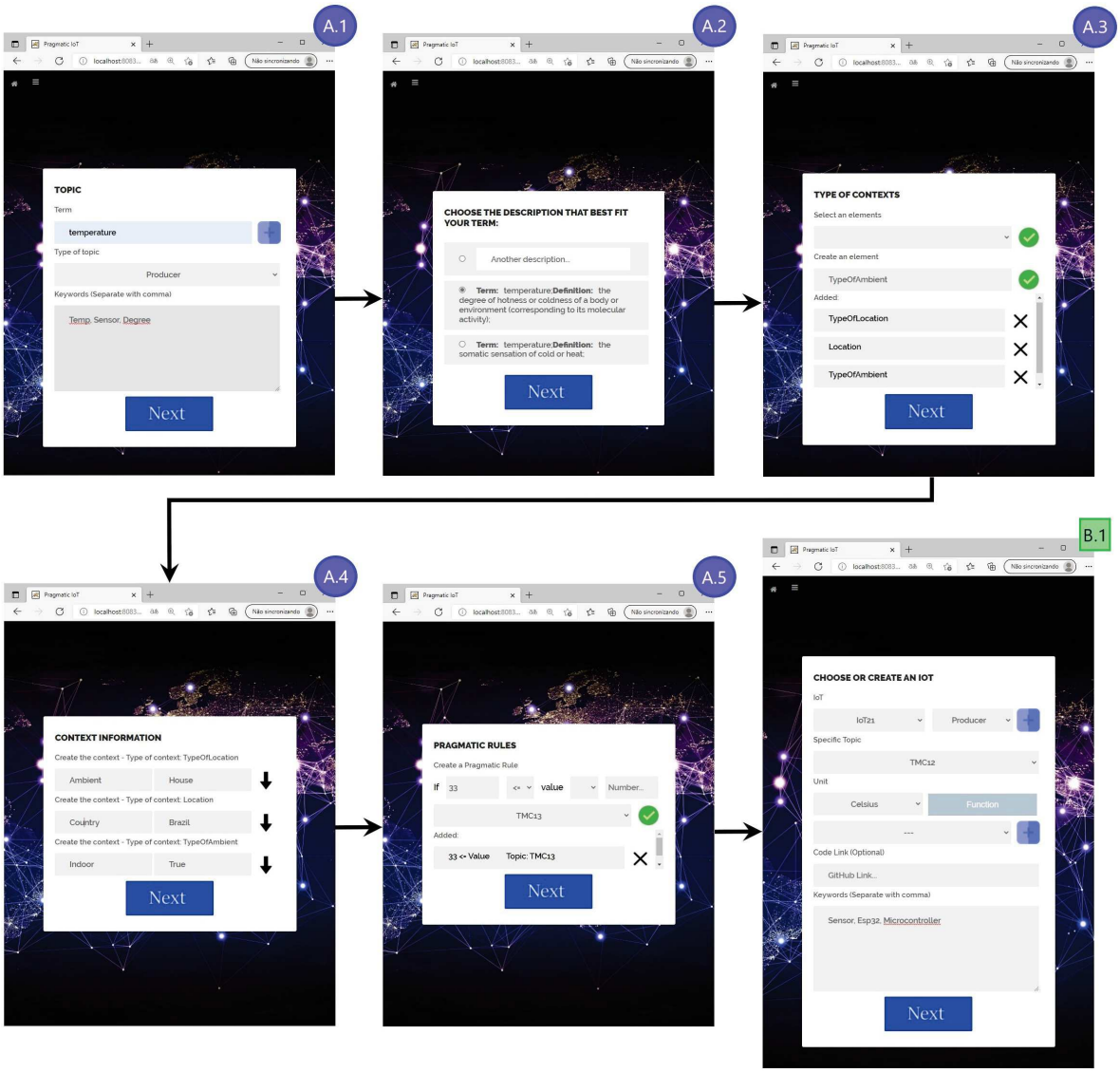
Como já mencionado, a arquitetura desenvolvida neste trabalho tem como base a utilização de tópicos. Em um único tópico é possível representar as informações sintáticas/ semânticas e pragmáticas de um dispositivo/aplicação.

Como primeiro passo da descrição dos dispositivo responsável por produzir dados da temperatura interna da estufa, o seu tópico foi descrito. Primeiramente informações básicas foram inseridas sobre o dispositivo/aplicação como pode ser observado na Figura 19 A.1. Estas informações são necessárias pois através delas buscas em uma API externa de sinônimos (*The STANDS4 Network*[2]) são realizadas e são expostas para o usuário. Após estas informações serem mostradas (Figura 19 A.2), foi possível escolher uma descrição que melhor se encaixa. Caso nenhuma das descrições estivessem dentro do esperado, uma poderia ser adicionada. Com as informações básicas e uma descrição, cálculos de similaridade são realizados entre o tópico que está sendo descrito com os que já existem na base. No caso das descrição do dispositivo responsável por produzir dados da temperatura interna da estufa, já existia um tópico com a mesma descrição selecionada, mas caso não existisse, os tópicos mais similares seriam mostrados para o usuário e este poderia escolher ou não dentre os existentes.

Após a criação da base do tópico, as informações de contexto foram adicionadas como pode ser observado na Figura 19 A.3 e A.4. Na tela A.3, as categorias são criadas para

[2] https://www.synonyms.com/

Figura 19 – Plataforma Web



o contextos, como por exemplo, ***Location***. Na tela A.4, as informações de contexto sobre as categorias são descritas, como por exemplo, ***Country/Brazil***. No caso da tela A.4, o nível de granularidade pode ser aumentado se necessário, proporcionando informações de contextos mais concisas (como por exemplo, **Country/Brazil**, **State/MinasGerais**, **City/JuizDeFora**). Até o momento, informações semânticas foram descritas. Como próximo passo as regras de produção (pragmáticas) foram adicionadas ao tópico (Figura 19 A.5). No caso do exemplo, somente uma única regra foi necessária. No entanto, não existe limite definido pela plataforma. As regras que o usuário informar, serão carregadas antes do dispositivo começar a produzir. Assim como que, para cada dispositivo um tópico é vinculado, na regra pragmática também um tópico adicionado (Este tem que ser descrito previamente). Desta forma, quando uma regra é acionada, os dados são produzidos para o tópico vinculado.

Como passo final, os dispositivos/aplicações serão descritas de fato (Figura 19 B.1), possuindo informações sintáticas como o tipo de unidade e funções de correção (caso não exista, novas podem ser adicionadas), *keywords*, link do projeto no github (se existir) e principalmente o vínculo com o tópico criado nos passos anteriores. Vale lembrar que em um grupo de unidades, a adição de uma nova, também só requer a descrição de um função. Por exemplo, se na ontologia existem várias unidades de medição (metro, centímetro, decímetro), caso uma nova fosse adicionada (milímetro), a descrição de um única é suficiente visto que o sistema gera de forma automática as outras (por exemplo, como existe a função entre metro e centímetro, descrever uma função entre milímetro e centímetro possibilita também a criação da função entre milímetro e metro através da junção e simplificação matemática entre funções).

4.2.4 Coleta dos dados

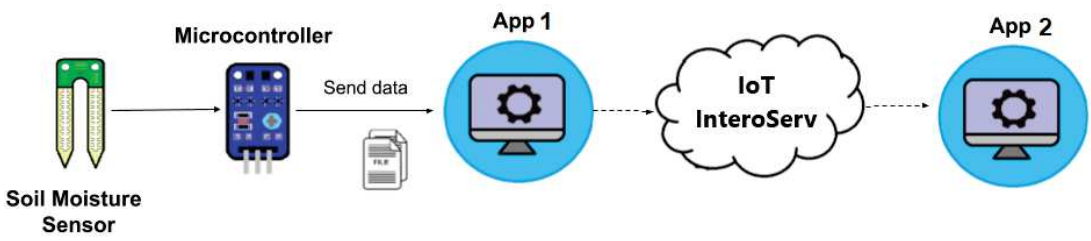
Para a coleta de dados deste cenário, todas as aplicações foram elaboradas com sistema de log. Através destes logs, foram permitidas comparações entres os dados consumidos, com os dados produzidos. Além disso, foi possível observar as aplicações sendo iniciadas e finalizadas de acordo com os eventos gerados pelas regras de produção.

4.2.5 Primeiro cenário

O primeiro fluxo é o normal (os dados gerados estão dentro da normalidade), possuindo basicamente um sensor/microcontrolador produzindo dados de umidade do solo e um consumidor recebendo os dados. Neste fluxo, a umidade do solo está normal, não sendo necessário ligar outro sensor/microcontrolador. A interação entre as partes pode ser observada na Figura 20. O fluxo se segue da seguinte forma:

> "O **microcontrolador** que está acoplado ao **sensor umidade do solo** do produz dados por meio da aplicação **App1**. O serviço **Pragmatic IoT Service** identifica que o valor produzido está dentro da normalidade. Como evento, o dado é direcionado apenas para a aplicação **App2** (Aplicação que recebe todos os dados que são enviados pelo **App1**)."

Figura 20 – Cenário - Fluxo normal

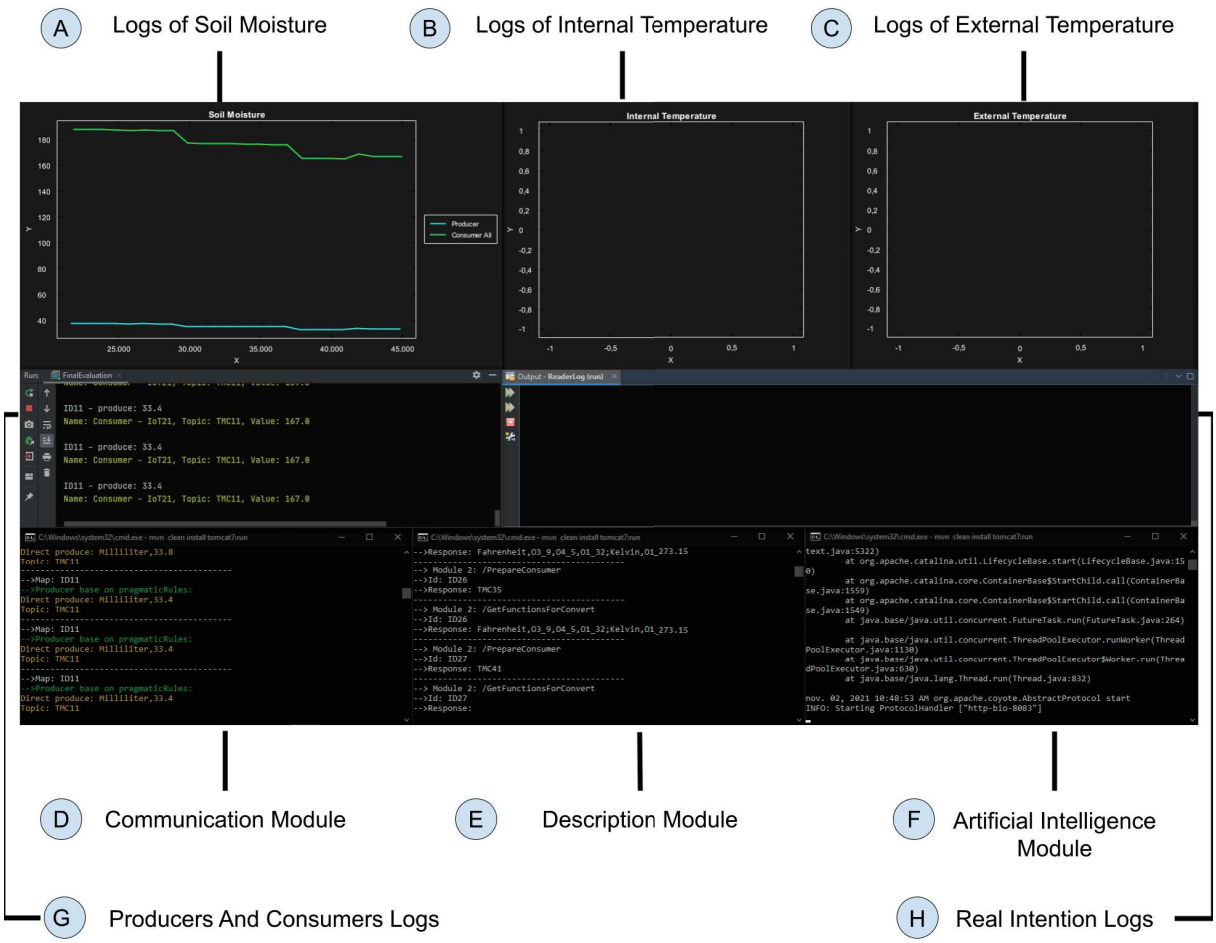


A aplicação **App1** possui um contexto diferente e produz para uma tipo de unidade diferente da aplicação **App2**. No entanto, por meio do **Pragmatic IoT Service** inferências são feitas e, por meio disso, é identificado que apesar da descrição dos dispositivos serem diferentes, o serviço consegue interpretar que as duas possuem o mesmo significado (**Semântica**) e que existe a necessidade de se realizar a conversão das unidades (**Sintática**). A aplicação **App2** dados da **App1**, pois estes dados passaram pelas regras de produção (**Pragmática**).

4.2.5.1 Execução

Para a realização da execução deste cenário, os módulos do Pragmatic IoT Service foram iniciados (Figura 21 D,E,F). Após isto, todas as aplicações e microcontroladores foram iniciadas (Os Logs das aplicações podem ser visto na Figura 21 G). Após a inicialização, a **App1** e o seu microcontrolador que coleta dados da umidade do solo permaneceram ativos (Figura 21 A), enquanto que os demais entraram em modo de hibernação (Figura 21 B,C).

Figura 21 – Gráficos das interações entre as aplicações do cenário 1



Como início do ciclo, dados de umidade do solo em porcentagem (Quanto maior, mais úmido) começaram a ser coletados através do microcontrolador e do sensor. Os

valores coletados iniciaram em 37.6 % e no decorrer do tempo, os valores foram caindo até atingir o valor de 33.4 % (Figura 21 A). Os valores produzidos passaram pelas regras de produção. Nas regras de produção da aplicação **App1**, caso o valor seja inferior a 10 %, o valor seria direcionado para um aplicação que é responsável por iniciar o processo de irrigação inteligente da estufa. Como o menor valor produzido foi 33.4 %, valor este superior ao descrito nas regras, o processo de irrigação inteligente não foi iniciado. Também nas regras de produção da aplicação **App1**, foi definido que qualquer valor produzido seria direcionado para a aplicação **App2**. Os valores produzidos, foram recebidos pela aplicação **App2** e como esta utiliza um outro tipo de unidade para interpretação, os valores também foram convertidos (informações inferidas através do módulo de descrição, como pode ser observado na Figura 21 E). A **App2** utiliza um tipo de unidade genérica.

Através da Figura 21 é possível observar as interações entres os produtores e consumidores como foi descrito nesta execução. Como o processo de irrigação inteligente não foi iniciado, os microcontroladores que coletam dados de temperatura interna e externa continuam em modo de espera, como pode ser observado nos gráficos da Figura 21. Nesse cenário, um microcontrolador, um produtor e um consumidor foram utilizados, um cenário relativamente fácil de ser replicado. No entanto, o cenário se limitou à aplicações/microcontrolador de forma fixa, sem possibilidade de estender ou acionar outras aplicações. Como forma de validar a extensibilidade e escalabilidade do sistema com adição de novas aplicações e sensores e validar a flexibilidade devido a adição de novos fluxos, um segundo cenário foi elaborado.

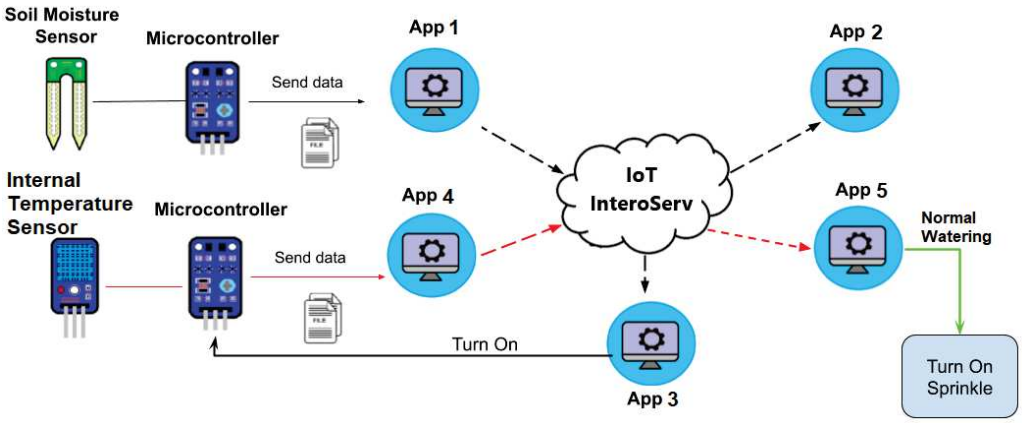
4.2.6 Segundo cenário

No segundo fluxo, a umidade do solo está baixa e como evento associado, o sensor/-microcontrolador responsável por observar a temperatura interna é ativado. Neste fluxo a temperatura interna está dentro da normalidade registradas nas regras pragmáticas.A interação entre as partes pode ser observada na Figura 22. O fluxo segue da seguinte forma:

> "O **microcontrolador** que está acoplado ao **sensor umidade do solo** do produz dados por meio da aplicação **App1**. O serviço **Pragmatic IoT Service** identifica que o valor produzido é inferior ao descrito nas regras. Como evento, o dado é direcionado para a aplicação **App2**, pois recebe qualquer dado produzido pela aplicação **App1**. Como evento, também produz para a aplicação **App3**, pois esta é responsável por iniciar o processo de irrigação inteligente. A aplicação **App3** liga o microcontrolador responsável por coletar dados de temperatura interna da estufa. A aplicação **App4** recebe os dados de temperatura interna e produz estes dados. O serviço **Pragmatic IoT Service** identifica que o valor produzido está dentro da normalidade. Como evento, o

dado é direcionado para a aplicação **App5**, pois esta é responsável por realizar um irrigação normal, visto que a temperatura está dentro do normal."

Figura 22 – Cenário - Baixa umidade do solo, temperatura interna baixa



Neste cenário, as regras de produção (**Pragmática**) identificam que os dados produzidos pela aplicação **App1** devem ser direcionados para aplicação **App2**, mas também deve ser gerado um evento para aplicação **App 3**. A aplicação **App3** também possui uma descrição diferente da aplicação **App1**, mas como mencionado as inferências do **Pragmatic IoT Service** permitem compreender os dados de forma **Semântica** e **Sintática**.

4.2.6.1 Execução

Para a realização da execução deste cenário, os módulos do Pragmatic IoT Service foram iniciados (Figura 23 D,E,F). Após isto, todas as aplicações e microcontroladores foram iniciadas (Os Logs das aplicações podem ser visto na Figura 23 G). Após a inicialização, a **App1** e o seu microcontrolador que coleta dados da umidade do solo permaneceram ativos (Figura 23 A), enquanto que os demais entraram em modo de hibernação (Figura 23 B,C).

Como início do ciclo, dados de umidade do solo em porcentagem (Quanto maior, mais úmido) começaram a ser coletados através do microcontrolador e do sensor. Os valores coletados iniciaram em 38.9 % e após um longo tempo, os valores foram caindo até atingir o valor de 8.7 % (Figura 23 A). Os valores produzidos passaram pelas regras de produção. Nas regras de produção da aplicação **App1**, caso o valor seja inferior a 10 %, o valor seria direcionado para um aplicação que é responsável por iniciar o processo de irrigação inteligente da estufa. Como valores menores que 10 % foram coletados o processo de irrigação inteligente foi iniciado através do direcionamento para a aplicação **App3** (como pode ser observado na Figura 23 A.1). Também nas regras de produção da aplicação **App1**, foi definido que qualquer valor produzido seria direcionado para a aplicação **App2**. Os valores produzidos, foram recebidos pela aplicação **App2** e como esta utiliza um outro tipo de unidade para interpretação, os valores também foram convertidos

Figura 23 – Gráficos das interações entre as aplicações do cenário 2

(informações inferidas através do módulo de descrição, como pode ser observado na Figura 23 E). A **App2** utiliza um tipo de unidade genérica.

O início do processo de irrigação a partir da aplicação **App3**, teve como evento o acionamento do microcontrolador responsável por produzir dados de temperatura interna. O microcontrolador ficou por um breve momento ligado e teve como produção (por meio da aplicação **App4**) valores entre 20 e 22 graus celsius. Os valores produzidos passaram pelas regras de produção. Nas regras de produção da aplicação **App4** caso o valor seja inferior a 33 graus celsius, o valor seria direcionado para a aplicação **App5** que é responsável por iniciar a irrigação normal. Como pode ser visto na Figura 23 (H), o log demonstra exatamente este fluxo. A aplicação **App5** recebeu os dados produzidos pela aplicação **App3** com as devidas formatações, visto que a aplicação **App5** espera dados de temperatura em Kelvin (informações inferidas através do módulo de descrição, como pode ser observado na Figura 23 E).

Através da Figura 23 é possível observar as interações entres os produtores e consumidores como foi descrito nesta execução. Como o processo de irrigação inteligente foi iniciado, o microcontrolador que coleta dados de temperatura externa continuou em

modo de espera, como pode ser observado nos gráficos da Figura 23.

Neste cenário, três aplicações e um sensor foram adicionados. A metodologia de ativar e desativar aplicações e sensores (de acordo com o uso) demonstrou ser eficiente do ponto de vista da escalabilidade, visto que a adição de novos sensores/aplicações na rede não diminui a disponibilidade da mesma. No que tange à flexibilidade, as regras de produção proporcionaram desvios de fluxos eficientes, sendo estas gerenciadas pelo módulo de comunicação (Figura 23 D). Por meio da utilização do *message broker* Apache kafka e do projeto Apache Zookeeper, a escalabilidade, a flexibilidade e a extensibilidade foram apoiadas minimamente mesmo com o acréscimo no número de aplicações/microcontroladores. O Apache Kafka permite inúmeras instâncias, sendo estas gerenciadas de forma automática pelo Apache Zookeeper, o que permite a abstração de aplicações, podendo estas estarem em regiões distintas. Como todo o acesso às informações sintáticas, semânticas, pragmática do serviço desenvolvido nesse trabalho é feito por meio de endpoints, foi possível alocar também várias instâncias de um mesmo serviço o que garante redundância no sistema e uma melhor escalabilidade horizontal do sistema.

A adição de novas aplicações e sensores demonstrou não comprometer os atributos de qualidade do sistema, visto que o mesmo foi construído baseados no Apache Kafka e Zookeeper. Além disso, o direcionamento via fluxo permitiu a utilização de aplicações/sensores apenas em tempo de execução, possibilitando a utilização de várias aplicações/sensores sem comprometer a escalabilidade do sistema. No entanto, apenas um único fluxo foi adicionado. Como forma de continuar a validação da extensibilidade e da escalabilidade do sistema com adição de novas aplicações e sensores e validar a flexibilidade devido a adição de novos fluxos, um terceiro cenário foi elaborado.

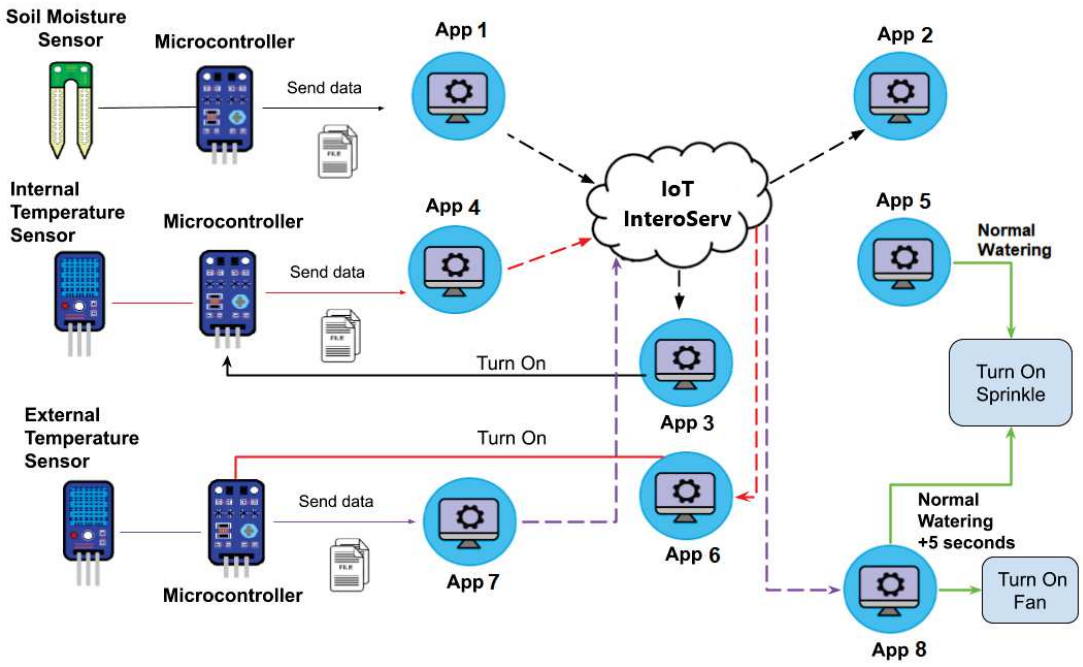
4.2.7 Terceiro cenário

No terceiro fluxo, diferente do fluxo anterior, a temperatura interna se encontra acima da superior definida nas regras. Como evento associado às regras, o sensor/microcontrolador responsável por observar a temperatura externa é ativo. Neste fluxo a temperatura externa está dentro da normalidade registradas nas regras pragmáticas. A interação entre as partes pode ser observada na Figura 24. O fluxo acontece da seguinte forma:

> "O **microcontrolador** que está acoplado ao **sensor umidade do solo** do produz dados por meio da aplicação **App1**. O serviço **Pragmatic IoT Service** identifica que o valor produzido é inferior ao descrito nas regras. Como evento, o dado é direcionado para a aplicação **App2**, pois recebe qualquer dado produzido pela aplicação **App1**. Como evento, também produz para a aplicação **App3**, pois esta é responsável por iniciar o processo de irrigação inteligente. A aplicação **App3** liga o microcontrolador responsável por coletar

dados de temperatura interna da estufa. A aplicação **App4** recebe os dados de temperatura interna e produz estes dados. O serviço **Pragmatic IoT Service** identifica que o valor produzido está fora da normalidade (acima do normal). Como evento, o dado é direcionado para a aplicação **App6**, pois esta é responsável por dar continuidade no processo de irrigação inteligente, quando a temperatura interna estiver acima do normal. A aplicação **App6** liga o microcontrolador responsável por coletar dados de temperatura externa da estufa, para verificar se existe uma diferença significativa com a interna. A aplicação **App7** recebe os dados de temperatura externa e produz estes dados. O serviço **Pragmatic IoT Service** identifica que o valor produzido está dentro da normalidade. Como evento, o dado é direcionado para a aplicação **App8**, pois esta além de realizar uma irrigação com um tempo maior que o segundo cenário (Devido a temperatura interna alta), também liga o circulador de ar para igualar a temperatura interna com a externa."

Figura 24 – Cenário - Temperatura interna alta e externa baixa



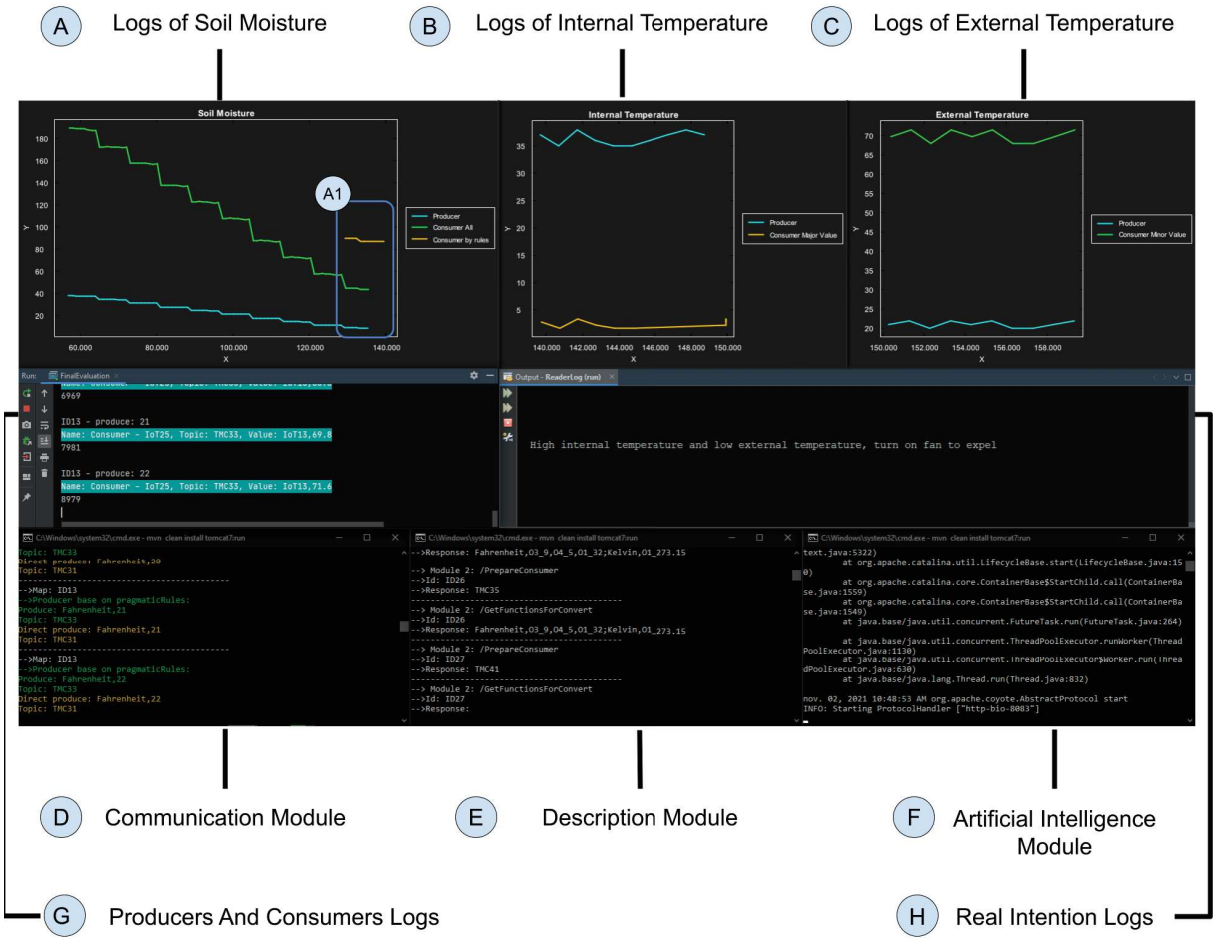
Neste cenário, devido a baixa **temperatura externa**, a aplicação **App 8** também é acionada via regras de produção (**Pragmática**). As aplicações **App 7** e **App 8** conseguem estabelecer uma comunicação efetiva graças às regras de produção (**Pragmática**), as inferências sobre o significado (**Semântica**) e a conversão das unidades (**Sintática**).

4.2.7.1 Execução

Para a realização da execução deste cenário, os módulos do Pragmatic IoT Service foram iniciados (Figura 25 D,E,F). Após isto, todas as aplicações e microcontroladores foram

iniciadas (Os Logs das aplicações podem ser visto na Figura 25 G). Após a inicialização, a **App1** e o seu microcontrolador que coleta dados da umidade do solo permaneceram ativos (Figura 25 A), enquanto que os demais entraram em modo de hibernação (Figura 25 B,C).

Figura 25 – Gráficos das interações entre as aplicações do cenário 3



Como início do ciclo, dados de umidade do solo em porcentagem (Quanto maior, mais úmido) começaram a ser coletados através do microcontrolador e do sensor. Os valores coletados iniciaram em 37.9 % e após um longo tempo, os valores foram caindo até atingir o valor de 8.7 % (Figura 25 A. Os valores produzidos passaram pelas regras de produção. Nas regras de produção da aplicação **App1**, caso o valor seja inferior a 10 %, o valor seria direcionado para um aplicação que é responsável por iniciar o processo de irrigação inteligente da estufa. Como valores menores que 10 % foram coletados o processo de irrigação inteligente foi iniciado através do direcionamento para a aplicação **App3** (como pode ser observado na Figura 25 A.1). Também nas regras de produção da aplicação **App1**, foi definido que qualquer valor produzido seria direcionado para a aplicação **App2**. Os valores produzidos, foram recebidos pela aplicação **App2** e como esta utiliza um outro tipo de unidade para interpretação, os valores também foram convertidos (informações inferidas através do módulo de descrição, como pode ser observado na Figura

25 E). A **App2** utiliza um tipo de unidade genérica.

O início do processo de irrigação a partir da aplicação **App3**, teve como evento o acionamento do microcontrolador responsável por produzir dados de temperatura interna. O microcontrolador ficou por um breve momento ligado e teve como produção (por meio da aplicação **App4**) valores entre 35 e 38 graus celsius (Figura 25 B). Os valores produzidos passaram pelas regras de produção. Nas regras de produção da aplicação **App4**, caso o valor seja superior a 33 graus celsius, o valor seria direcionado para a aplicação **App6** para ligar o controlador responsável por coletar dados de temperatura interna. O microcontrolador ficou por um breve momento ligado e teve como produção (por meio da aplicação **App7**) valores entre 20 e 22 graus celsius (Figura 25 C). Os valores produzidos passaram pelas regras de produção. Nas regras de produção da aplicação **App7**, caso o valor seja inferior a 33 graus celsius, o valor seria direcionado para a aplicação **App8** que é responsável por iniciar a irrigação com um tempo maior e ligar o circulador de ar. Como pode ser visto na Figura 25 (H), o log demonstra exatamente este fluxo. A aplicação **App8** recebeu os dados produzidos pela aplicação **App7** com as devidas formatações, visto que a aplicação **App8** espera dados de temperatura em Fahrenheit (informações inferidas através do módulo de descrição, como pode ser observado na Figura 21 E).

Através da Figura 25 é possível observar as interações entres os produtores e consumidores como foi descrito nesta execução. Neste cenário, todos os microcontroladores foram ativados como pode ser observado nos gráficos da Figura 25.

Neste cenário, três aplicações e um sensor foram adicionados. A metodologia de ativar e desativar aplicações e sensores (de acordo com o uso) demonstrou ser eficiente do ponto de vista da escalabilidade, visto que a adição de novos sensores/aplicações na rede não diminui a disponibilidade da mesma. No que tange à flexibilidade, as regras de produção proporcionaram desvios de fluxos eficientes, sendo estas gerenciadas pelo módulo de comunicação (Figura 25 D). Por meio da utilização do *message broker* Apache kafka e do projeto Apache Zookeeper, a escalabilidade, a flexibilidade e a extensibilidade foram apoiadas mesmo com o acréscimo no número de aplicações/microcontroladores. O Apache Kafka permite inúmeras instâncias, sendo estas gerenciadas de forma automática pelo Apache Zookeeper, o que permite a abstração de aplicações, podendo estas estarem em regiões distintas. Como todo o acesso às informações sintáticas, semânticas, pragmática do serviço desenvolvido nesse trabalho é feito por meio de endpoints, foi possível alocar também várias instâncias de um mesmo serviço o que garante redundância no sistema e uma melhor escalabilidade horizontal do sistema.

A adição de novas aplicações e sensores demonstrou não comprometer os atributos de qualidade do sistema, visto que o mesmo foi construído baseados no Apache Kafka e Zookeeper. Além disso, a utilização de regras pragmáticas possibilitou a utilização de aplicações/sensores apenas em tempo de execução através do direcionamento de fluxo,

possibilitando a utilização de inúmeras aplicações/sensores sem comprometer a escalabilidade do sistema. Em comparação com o segundo cenário, este teve a adição de mais um fluxo (novas regras) para averiguar se a adição de novos fluxos, não iria comprometer os atributos de qualidades abordados neste trabalho. Um quarto cenário foi desenvolvido, a fim de continuar esta validação que implica em verificar a adição de novos fluxos e como isto impacta nos atributos de qualidade abordados neste trabalho.

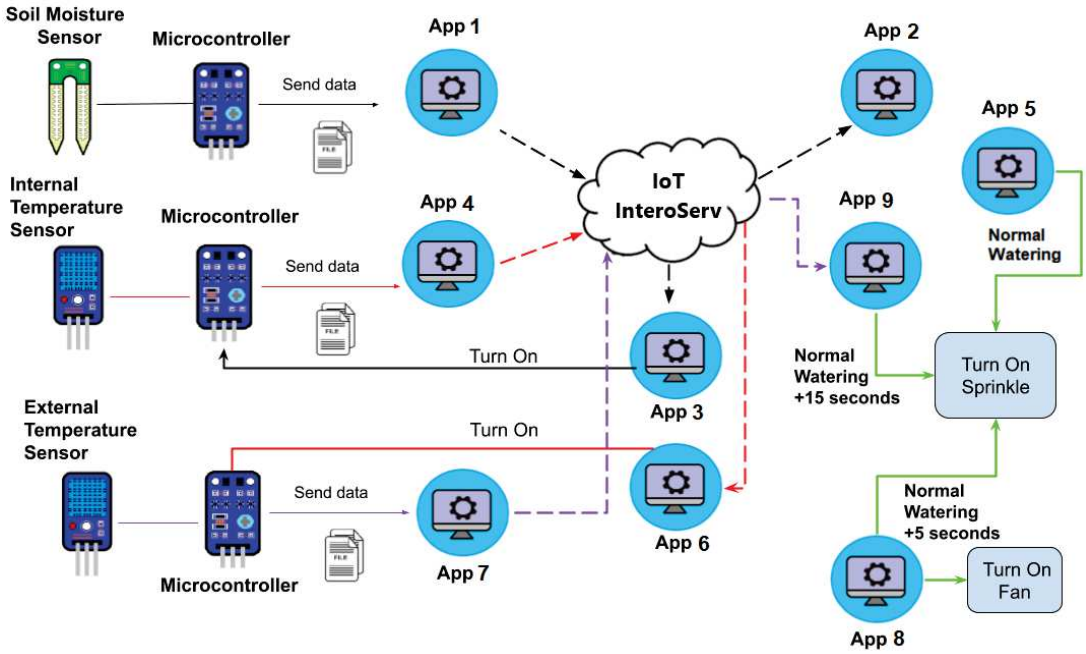
4.2.8 Quarto cenário

No quarto fluxo, diferente do fluxo anterior, a temperatura externa se encontra acima da superior definida nas regras. Neste fluxo a temperatura externa está dentro da normalidade registradas nas regras pragmáticas. A interação entre as partes pode ser observada na Figura 26. O fluxo se segue da seguinte forma:

> "O **microcontrolador** que está acoplado ao **sensor umidade do solo** do produz dados por meio da aplicação **App1**. O serviço **Pragmatic IoT Service** identifica que o valor produzido é inferior ao descrito nas regras. Como evento, o dado é direcionado para a aplicação **App2**, pois recebe qualquer dado produzido pela aplicação **App1**. Como evento, também produz para a aplicação **App3**, pois esta é responsável por iniciar o processo de irrigação inteligente. A aplicação **App3** liga o microcontrolador responsável por coletar dados de temperatura interna da estufa. A aplicação **App4** recebe os dados de temperatura interna e produz estes dados. O serviço **Pragmatic IoT Service** identifica que o valor produzido está fora da normalidade (acima do normal). Como evento, o dado é direcionado para a aplicação **App6**, pois esta é responsável por dar continuidade no processo de irrigação inteligente, quando a temperatura interna estiver acima do normal. A aplicação **App6** liga o microcontrolador responsável por coletar dados de temperatura externa da estufa, para verificar se existe uma diferença significativa com a interna. A aplicação **App7** recebe os dados de temperatura externa e produz estes dados. O serviço **Pragmatic IoT Service** identifica que o valor produzido está fora da normalidade (acima do normal). Como evento, o dado é direcionado para a aplicação **App9**, visto que esta realiza a irrigação com o maior tempo entre todos os cenários, visto que não é possível regular a temperatura com o ambiente externo."

Neste cenário, devido a alta **temperatura externa**, a aplicação **App 9** também é acionada via regras de produção (**Pragmática**). As aplicações **App 7** e **App 9** conseguem estabelecer uma comunicação efetiva graças às regras de produção (**Pragmática**), as inferências sobre o significado (**Semântica**) e a conversão das unidades (**Sintática**).

Figura 26 – Cenário - Temperatura interna e externa alta



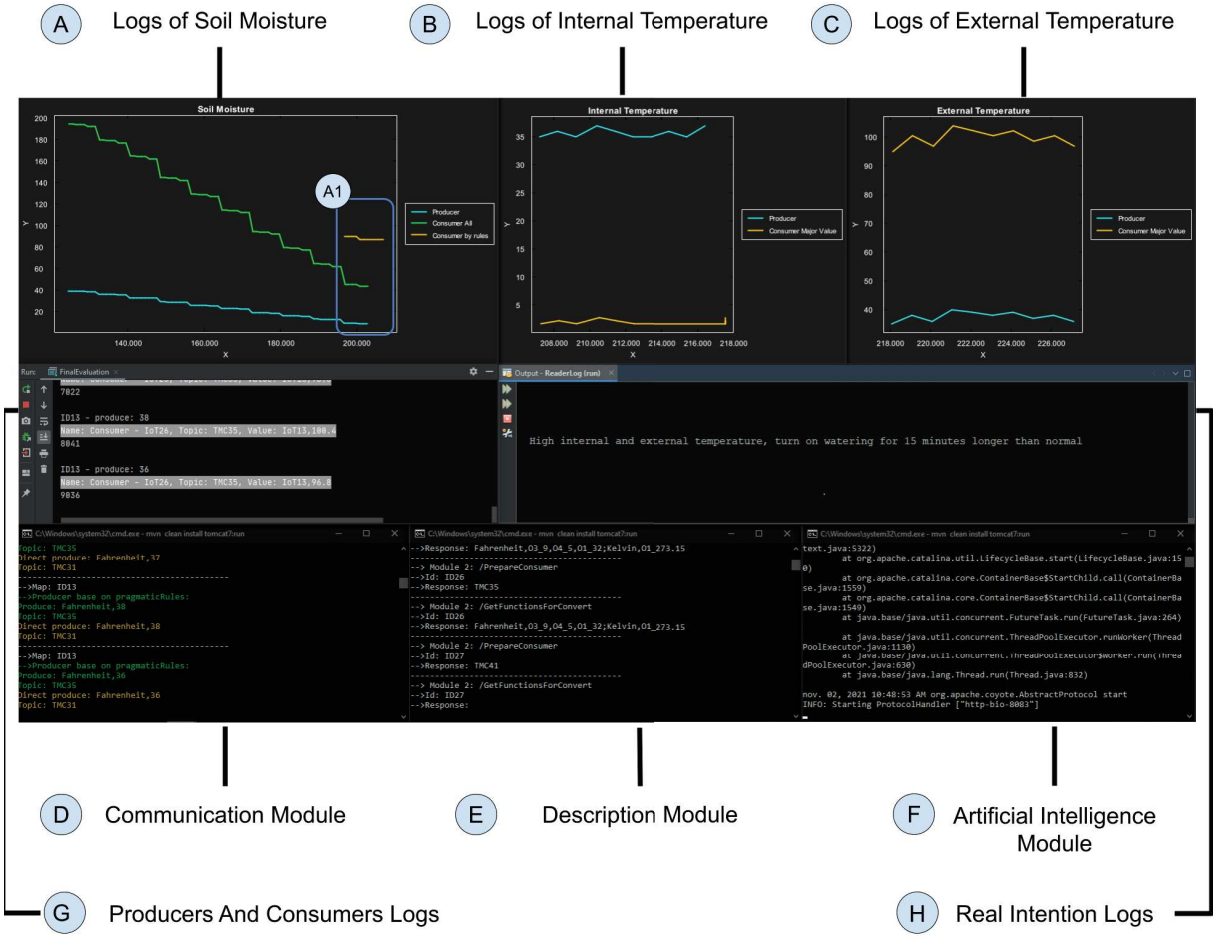
4.2.8.1 Execução

Para a realização da execução deste cenário, os módulos do Pragmatic IoT Service foram iniciados (Figura 27 D,E,F). Após isto, todas as aplicações e microcontroladores foram iniciadas (Os Logs das aplicações podem ser visto na Figura 27 G). Após a inicialização, a **App1** e o seu microcontrolador que coleta dados da umidade do solo permaneceram ativos (Figura 27 A), enquanto que os demais entraram em modo de hibernação (Figura 27 B,C).

Como início do ciclo, dados de umidade do solo em porcentagem (Quanto maior, mais úmido) começaram a ser coletados através do microcontrolador e do sensor. Os valores coletados iniciaram em 38.9 % e após um longo tempo, os valores foram caindo até atingir o valor de 8.7 % (Figura 27 A). Os valores produzidos passaram pelas regras de produção. Nas regras de produção da aplicação **App1**, caso o valor seja inferior a 10 %, o valor seria direcionado para um aplicação que é responsável por iniciar o processo de irrigação inteligente da estufa. Como valores menores que 10 % foram coletados o processo de irrigação inteligente foi iniciado através do direcionamento para a aplicação **App3** (como pode ser visto na Figura 27 A.1). Também nas regras de produção da aplicação **App1**, foi definido que qualquer valor produzido seria direcionado para a aplicação **App2**. Os valores produzidos, foram recebidos pela aplicação **App2** e como esta utiliza um outro tipo de unidade para interpretação, os valores também foram convertidos (informações inferidas através do módulo de descrição, como pode ser observado na Figura 27 E). A **App2** utiliza um tipo de unidade genérica.

O início do processo de irrigação a partir da aplicação **App3**, teve como evento o acionamento do microcontrolador responsável por produzir dados de temperatura interna.

Figura 27 – Gráficos das interações entre as aplicações do cenário 4



O microcontrolador ficou por um breve momento ligado e teve como produção (por meio da aplicação **App4**) valores entre 35 e 37 graus celsius (Figura 27 B). Os valores produzidos passaram pelas regras de produção. Nas regras de produção da aplicação **App4**, caso o valor seja superior a 33 graus celsius, o valor seria direcionado para a aplicação **App6** para ligar o controlador responsável por coletar dados de temperatura interna. O microcontrolador ficou por um breve momento ligado e teve como produção (por meio da aplicação **App7**) valores entre 35 e 37 graus celsius (Figura 27 C). Os valores produzidos passaram pelas regras de produção. Nas regras de produção da aplicação **App7**, caso o valor seja superior a 33 graus celsius, o valor seria direcionado para a aplicação **App9** que é responsável por iniciar a irrigação com maior tempo. Como pode ser visto na Figura 27 (H), o log demonstra exatamente este fluxo. A aplicação **App9** recebeu os dados produzidos pela aplicação **App7** com as devidas formatações, visto que a aplicação **App9** espera dados de temperatura em Fahrenheit (informações inferidas através do módulo de descrição, como pode ser observado na Figura 27 E).

Através da Figura 27 é possível observar as interações entres os produtores e consumidores como foi descrito nesta execução. Neste cenário, todos os microcontroladores

foram ativados como pode ser observado nos gráficos da Figura 27.

Neste cenário, três aplicações e um sensor foram adicionados. A metodologia de ativar e desativar aplicações e sensores (de acordo com o uso) demonstrou ser eficiente do ponto de vista da escalabilidade, visto que a adição de novos sensores/aplicações na rede não diminui a disponibilidade da mesma. No que tange à flexibilidade, as regras de produção proporcionaram desvios de fluxos eficientes, sendo estas gerenciadas pelo módulo de comunicação (Figura 27 D). Por meio da utilização do *message broker* Apache kafka e do projeto Apache Zookeeper, a escalabilidade, a flexibilidade e a extensibilidade foram apoiadas mesmo com o acréscimo no número de aplicações/microcontroladores. O Apache Kafka permite inúmeras instâncias, sendo estas gerenciadas de forma automática pelo Apache Zookeeper, o que permite a abstração de aplicações, podendo estas estarem em regiões distintas. Como todo o acesso às informações sintáticas, semânticas, pragmática do serviço desenvolvido nesse trabalho é feito por meio de endpoints, foi possível alocar também várias instâncias de um mesmo serviço o que garante redundância no sistema e uma melhor escalabilidade horizontal do sistema.

A adição de novas aplicações e sensores demonstrou não comprometer os atributos de qualidade do sistema, visto que o mesmo foi construído baseados no Apache Kafka e Zookeeper. Além disso, a utilização de regras pragmáticas possibilitou a utilização de aplicações/sensores apenas em tempo de execução através do direcionamento de fluxo, possibilitando a utilização de inúmeras aplicações/sensores sem comprometer a escalabilidade do sistema. Em relação ao segundo e terceiro cenário, o quarto cenário possui integrado todos as aplicações/microcontrolador e sensores e todos os fluxos descritos na ontologia. Até o momento, somente a extensibilidade e escalabilidade foram avaliadas com a adição de novos fluxos, aplicações e sensores. Um quinto cenário foi desenvolvido para validar a flexibilidade do sistema frente a mudanças no mesmo.

4.2.9 Quinto cenário

Para a realização do quinto cenário, uma mudança do contexto do cenário foi realizada em tempo de execução. A estufa foi inserida em ambiente mais quente do que o anterior. Desta forma, antes de começar a execução do cenário, a aplicação **App1** foi forçada a atualizar as regras de produção, uma vez que o contexto também foi alterado. Para isto, o módulo de inteligência artificial foi acessado a fim de buscar novas regras de produção para a aplicação **App1**.

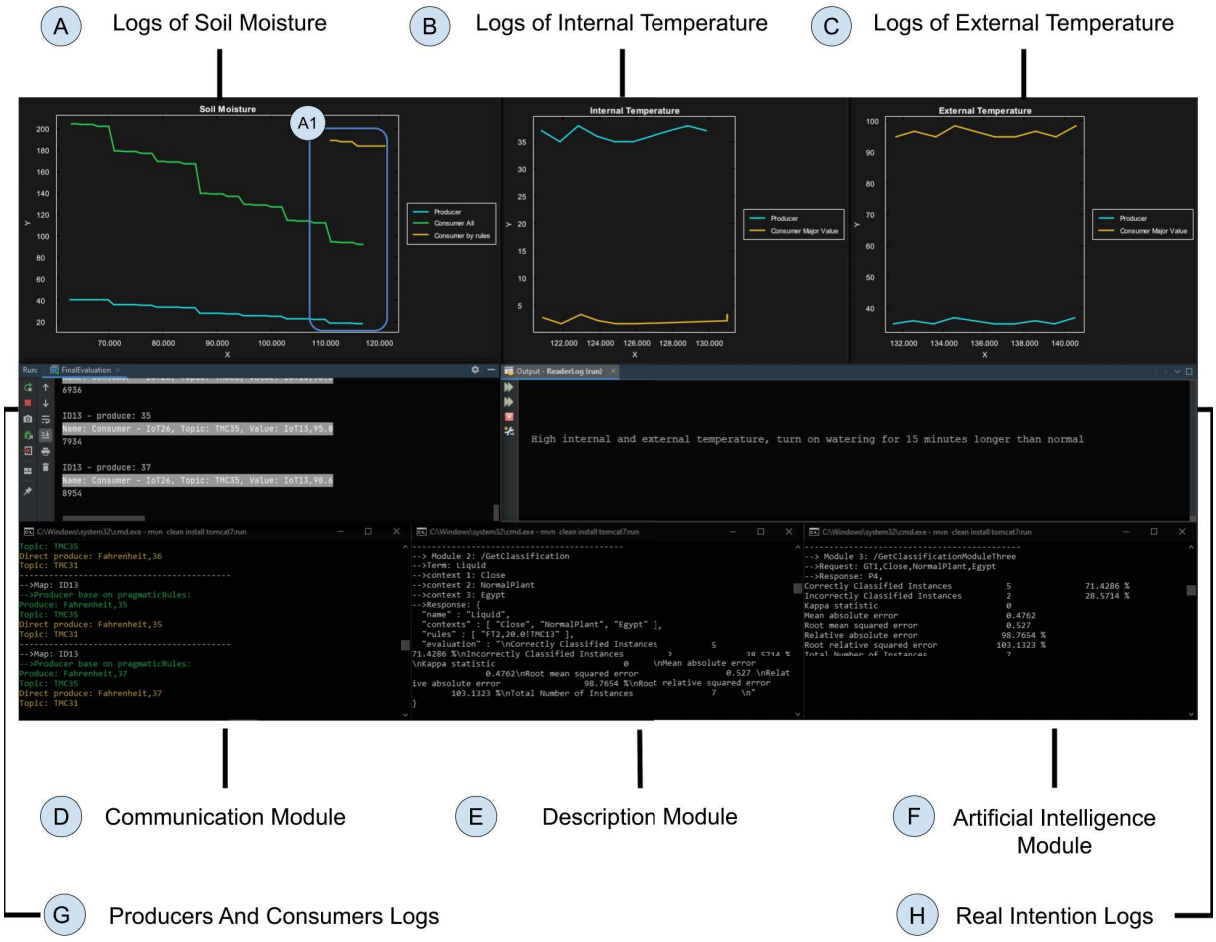
4.2.9.1 Execução

Para a realização da execução deste cenário, os módulos do Pragmatic IoT Service foram iniciados (Figura 28 D,E,F). Após isto, todas as aplicações e microcontroladores foram iniciadas (Os Logs das aplicações podem ser visto na Figura 28 G). Após a inicialização, a

App1 e o seu microcontrolador que coleta dados da umidade do solo permaneceram ativos (Figura 28 A), enquanto que os demais entraram em modo de hibernação (Figura 28 B,C).

Como início do ciclo, a aplicação **App1** acessa o endpoint que gera novas regras de produção por meio da classificação do machine learning, visto que o contexto mudou. Primeiramente o módulo de descrição é acessado (Figura 28 E) e logo em seguida o módulo de inteligência (Figura 28 F). Novas regras então são carregadas no servidor. Novas regras foram necessárias, visto que a estufa foi deslocada para ambiente mais quente do que o original. Como resposta do módulo de inteligência artificial (Figura 28 F), foi inferido que o valor mínimo de umidade de solo deveria ser maior que o original. Dessa forma, menos irrigações seriam necessárias.

Figura 28 – Gráficos das interações entre as aplicações do cenário 5



Após isto, os dados de umidade do solo em porcentagem (Quanto maior, mais úmido) começaram a ser coletados através do microcontrolador e do sensor. Os valores coletados iniciaram em 40.8 % e após um longo tempo, os valores foram caindo até atingir o valor de 18.4 % (Figura 28 A). Os valores produzidos passaram pelas regras de produção. Nas novas regras de produção da aplicação **App1**, caso o valor seja inferior a 20 % (anteriormente era 20 %), o valor seria direcionado para um aplicação que é responsável por

iniciar o processo de irrigação inteligente da estufa. Como valores menores que 20 % foram coletados o processo de irrigação inteligente foi iniciado através do direcionamento para a aplicação **App3** (como pode ser visto na Figura 28 A.1). Também nas regras de produção da aplicação **App1**, foi definido que qualquer valor produzido seria direcionado para a aplicação **App2**. Os valores produzidos, foram recebidos pela aplicação **App2** e como esta utiliza um outro tipo de unidade para interpretação, os valores também foram convertidos (informações inferidas atráves do módulo de descrição, como pode ser observado na Figura 28 E). A aplicação **App2** utiliza um tipo de unidade genérica.

O início do processo de irrigação a partir da aplicação **App3**, teve como evento o acionamento do microcontrolador responsável por produzir dados de temperatura interna. O microcontrolador ficou por um breve momento ligado e teve como produção (por meio da aplicação **App4**) valores entre 35 e 38 graus celsius (Figura 28 B). Os valores produzidos passaram pelas regras de produção. Nas regras de produção da aplicação **App4**, caso o valor seja superior a 33 graus celsius, o valor seria direcionado para a aplicação **App6** para ligar o controlador responsável por coletar dados de temperatura interna. O microcontrolador ficou por um breve momento ligado e teve como produção (por meio da aplicação **App7**) valores entre 35 e 37 graus celsius (Figura 28 C). Os valores produzidos passaram pelas regras de produção. Nas regras de produção da aplicação **App7**, caso o valor seja superior a 33 graus celsius, o valor seria direcionado para a aplicação **App9** que é responsável por iniciar a irrigação com maior tempo. Como pode ser visto na Figura 28 (H), o log demonstra exatamente este fluxo. A aplicação **App9** recebeu os dados produzidos pela aplicação **App7** com as devidas formatações, visto que a aplicação **App9** espera dados de temperatura em Fahrenheit (informações inferidas através do módulo de descrição, como pode ser observado na Figura 28 E).

Através da Figura 28 é possível observar as interações entres os produtores e consumidores como foi descrito nesta execução. Neste cenário, todos os microcontroladores foram ativados como pode ser observado nos gráficos da Figura 28.

Neste cenário, novas aplicações e sensores foram adicionados. A metodologia de ativar e desativar aplicações e sensores (de acordo com o uso) demonstrou ser eficiente do ponto de vista da escalabilidade, visto que a adição de novos sensores/aplicações na rede não diminui a disponibilidade da mesma. No que tange à flexibilidade, as regras de produção proporcionaram desvios de fluxos eficientes, sendo estas gerenciadas pelo módulo de comunicação (Figura 23 D). Por meio da utilização do *message broker* Apache kafka e do projeto Apache Zookeeper, a escalabilidade, a flexibilidade e a extensibilidade foram apoiadas mesmo com o acréscimo no número de aplicações/microcontroladores. O Apache Kafka permite inúmeras instâncias, sendo estas gerenciadas de forma automática pelo Apache Zookeeper, o que permite a abstração de aplicações, podendo estas estarem em regiões distintas. Como todo o acesso às informações sintáticas, semânticas, pragmática do serviço desenvolvido nesse trabalho é feito por meio de endpoints, foi possível alocar

também várias instâncias de um mesmo serviço o que garante redundância no sistema e uma melhor escalabilidade horizontal do sistema. A mudança de contexto presente neste cenário, demonstrou ser eficiente do ponto de vista dos atributos de qualidade de software, visto que a captura de novas regras permitiu que novos fluxos surjam, sem comprometer os fluxos já existentes.

O estudo regular, utilizando cenários, evidenciou que a hipótese alternativa (a arquitetura e o serviço integrado apoia a interoperabilidade em ecossistema de IoT) é verdadeira sobre a questão de pesquisa "**Como a arquitetura e o serviço originado desta apoia a interoperabilidade em Ecossistemas de IoT?**", visto que os atributos de qualidade de software abordados são atendidos, como apresentado nas execuções dos cenários. Através da execução dos cenários, foi possível observar que a arquitetura e o serviço desenvolvido apoia à interoperabilidade em ecossistemas de IoT de forma macro e micro, ou seja, o serviço derivado da arquitetura desenvolvida neste trabalho é escalável, flexível, extensível e o mesmo pode ser dito das aplicações que interromperam através deste serviço.

4.2.10 Ameaças à validade

O estudo realizado possui algumas ameaças à validade. No que se refere ao mapeamento realizado, especialistas foram utilizados, artigos de controle e trabalhos conceituados, mas nem todas as bases foram pesquisadas por estas serem fechadas e nem todos os artigos estavam disponíveis para leitura. Com isso, estudos que poderiam agregar mais conhecimento a este trabalho não foram lidos. Além disso, a execução do mapeamento foi feita por um único pesquisador. Um segundo pesquisador, poderia reproduzir os passos do mapeamento a fim aumentar a sua veracidade. No que tange a avaliação do trabalho, o estudo de caso não foi executado em ambiente real de utilização, o que pode ser considerado como uma ameaça. No entanto, a utilização de sensores e microcontroladores reais contribuiu para mitigar esta ameaça à validade. Dadas as limitações da avaliação, poucos foram os sensores utilizados no estudo de caso. A utilização de um número significativo se sensores, poderia demonstrar pontos de melhoria na arquitetura/serviço desenvolvido. Uma outra ameaça está atrelada ao fato da área ser muito pouco explorada ainda (interoperabilidade pragmática em ecossistemas de IoT), o que gera pouco conhecimento sobre tecnologias e modelos que poderiam contribuir ainda mais na elaboração da solução. Outra ameaça está relacionada ao módulo de inteligência artificial, visto que necessita de uma quantidade significativa de dados para que a acurácia seja relevante (isso pode variar de contexto para contexto). Além disso, um modelo de algoritmo genérico que foi utilizado para o *machine learning* pode afetar negativamente na acurácia da inferência. No entanto, o sistema permite uma fácil troca nos algoritmos de *machine learning* permitindo que o algoritmo possa ser alterado de acordo com o contexto.

4.2.11 Considerações Finais do capítulo

Esse capítulo teve por objetivo avaliar a solução proposta que consiste em, apoiar a interoperabilidade sintática, semântica e pragmática em ecossistema de IoT levando em consideração os atributos de qualidade. Um estudo regular foi realizado utilizando cenários para verificar as hipóteses levantadas a partir da questão de pesquisa. Através da execução dos cenários, a hipótese que a solução proposta apoia a interoperabilidade em ecossistemas de IoT foi evidenciada, principalmente quando o objeto de estudo eram os atributos de qualidade de software. A partir das evidências coletadas, outros cenários também poderão ser conduzidos sobre a solução proposta em outros trabalhos.

5 CONSIDERAÇÕES FINAIS

A crescente utilização de dispositivos inteligentes em rede (IoT) pode ser considerada como a internet do futuro, visto que por meio da coleta, processamento e distribuição de dados, tarefas diárias pessoais e industriais poderão ser simplificadas. No entanto, a maior parte dos dispositivos inteligentes são provenientes de indústrias distintas, que possuem os seus próprios padrões de comunicação gerando silos verticais de dados, o que gera uma barreira para interoperar na IoT.

Na literatura, uma gama de soluções foram sugeridas e/ou implementadas para tratar os problemas de interoperabilidade. No entanto, em um contexto geral e na IoT, o foco da maioria está no nível sintático e semântico.

Com o objetivo de observar os trabalhos que tratem a interoperabilidade em ecossistemas de IoT, considerando o nível pragmático, um estudo foi realizado e publicado. Como evidência principal, foi constatado que poucos são os trabalhos que abordam este tema e em sua maioria, estes não foram validados em um contexto real de utilização e/ou não levaram os atributos de qualidade de software, principalmente do ponto de vista de dispositivos IoT. Com o propósito de mitigar estas lacunas, a arquitetura IoT-InterArch foi elaborada.

A arquitetura desenvolvida neste trabalho foi construída para ser modular e trabalhar em conjunto com *messsage brokers*. A solução permite que diversos dispositivos possam conectar-se na rede, através das descrições desses dispositivos nas ontologias desenvolvidas. Como forma de simplificar a descrição, uma plataforma Web foi desenvolvida e acoplada a ela, um sistema de cálculo de similaridade foi inserido para trazer uma maior entendimento sobre o que está sendo descrito. Através da solução desenvolvida os dispositivos podem utilizar o serviço (IoT-InterServ) originado da arquitetura, sendo necessário apenas uma implementação simples de software, visto que o processamento fica a IoT-InterServ. A solução foi avaliada utilizando cenários em um contexto real de utilização de uma estufa. Por meio das evidências foi possível observar que a solução apoia a interoperabilidade em ecossistemas de IoT.

As contribuições deste trabalho foram as seguintes:

- Um mapeamento sistemático que observa os estudos primários que apoiam a interoperabilidade pragmática em IoT
- Uma arquitetura modular que apoia a interoperabilidade em ecossistema de IoT, mas não se limita a este tipo de ecossistema;
- Construção de duas ontologias que descrevem dispositivos e tópicos para apoiar a interoperabilidade sintática, semântica e pragmática da IoT;

- Um sistema de cálculo de similaridade que considera, além do termo em si, a descrição e os sinônimos do mesmo;
- Uma plataforma Web que visa facilitar a descrição na ontologia desenvolvida;
- Construção de endpoints para apoiar a interoperabilidade em ecossistemas Externos, não sendo limitados pela a IoT;
- Construção de um módulo inicial que permite que as regras de um contexto possam ser alteradas quando o contexto for mudado, mesmo quando não existe descrição para isto;

A solução deste trabalho foi baseada na utilização de uma *message broker* como base do sistema e por isso as implementação originas a partir da arquitetura terão que utilizar este tipo de tecnologia e agregado a isto deverão utilizar as ontologias desenvolvidas, não sendo possível utilizar outras ontologias de IoT.

A arquitetura foi desenvolvida para ser modular com várias instâncias para o módulo de comunicação e módulo de inteligência artificial. No que tange ao módulo de conhecimento, este só poderá ter uma única instância. No entanto esta limitação é mitigada, visto que para interoperar só é necessário um primeiro acesso ao módulo de conhecimento (exceto em casos de atualizações periódicas).

O módulo de inteligência artificial foi desenvolvido para que novas informações possam ser inferidas além daquelas obtidas através da ontologia. No entanto, não foi o foco deste trabalho escolher o melhor algoritmo de *machine learning*. Por outro ponto de vista, a natureza modular do sistema, permite inúmeras instâncias desse módulo, sendo possível utilizar diversos algoritmos disponíveis, ou seja, algoritmos diferentes podem ser utilizados em contextos distintos de forma independente. No que tange a avaliação, a utilização de um único desenvolvedor pode ser considerada como uma limitação.

Como trabalhos futuros podemos citar:

- Evolução do módulo de conhecimento para permitir inúmeras instâncias e proporcionar uma escalabilidade maior do sistema;
- Evolução das ontologias para permitir que estas possam ser distribuídas na rede que é formada pelo serviço (a distribuição das ontologias permite que as mesmas possam ser mais leves e de fácil substituição).
- Explorar a funcionalidade de troca de contexto em ecossistemas de IoT (A arquitetura desenvolvida permite a mudança nas regras a partir da mudança de contexto. No entanto, a mudança de contexto está sendo feito de uma forma estática);

- Evolução do módulo de inteligência artificial para permitir a troca de algoritmo de forma dinâmica ;
- Focar nos outros atributos de qualidade de software, como a questão da segurança dos dados;
- Agregar outras bases de conhecimento externas como forma de enriquecer ainda mais a interoperabilidade, principalmente para quando não houver dados iniciais;

REFERÊNCIAS

[Asuncion and Sinderen 2010]ASUNCION, C. H.; SINDEREN, M. J. V. Pragmatic interoperability: A systematic review of published definitions. In: SPRINGER. *IFIP International Conference on Enterprise Architecture, Integration and Interoperability*. [S.l.], 2010. p. 164–175.

[Basili and Weiss 1984]BASILI, V. R.; WEISS, D. M. A methodology for collecting valid software engineering data. *IEEE Transactions on software engineering*, IEEE, n. 6, p. 728–738, 1984.

[Berk, Jansen and Luinenburg 2010]BERK, I. V. D.; JANSEN, S.; LUINENBURG, L. Software ecosystems: a software ecosystem strategy assessment model. In: *Proceedings of the Fourth European Conference on Software Architecture: Companion Volume*. [S.l.: s.n.], 2010. p. 127–134.

[Berners-Lee, Hendler and Lassila 2001]BERNERS-LEE, T.; HENDLER, J.; LASSILA, O. The semantic web. *Scientific american*, JSTOR, v. 284, n. 5, p. 34–43, 2001.

[Bosch 2009]BOSCH, J. From software product lines to software ecosystems. In: *SPLC*. [S.l.: s.n.], 2009. v. 9, p. 111–119.

[Boucharas, Jansen and Brinkkemper 2009]BOUCHARAS, V.; JANSEN, S.; BRINKKEMPER, S. Formalizing software ecosystem modeling. In: *Proceedings of the 1st international workshop on Open component ecosystems*. [S.l.: s.n.], 2009. p. 41–50.

[Bravo and Alvarado 2008]BRAVO, M.; ALVARADO, M. On the pragmatic similarity between agent communication protocols: Modeling and measuring. *On the Move to Meaningful Internet Systems: OTM*, p. 128–137, 2008.

[Brezolin et al. 2018]BREZOLIN, F. L. et al. Desenvolvimento de um dispositivo iot lúdico para monitoramento de variáveis ambientais. Universidade de Passo Fundo, 2018.

[Delicato et al. 2013]DELICATO, F. C. et al. Towards an iot ecosystem. In: *Proceedings of the first international workshop on software engineering for systems-of-systems*. [S.l.: s.n.], 2013. p. 25–28.

[Deng, Liu and Li 2016]DENG, L.; LIU, N.; LI, G. Importance computing-based swot ontology summarization system design. In: SPRINGER. *Proceedings of the 2015 International Conference on Electrical and Information Technologies for Rail Transportation*. [S.l.], 2016. p. 631–644.

[Dhivya, Parameswaran et al. 2020]DHIVYA, M.; PARAMESWARAN, T. et al. Smart scheduling on cloud for iot-based sprinkler irrigation. *International Journal of Pervasive Computing and Communications*, Emerald Publishing Limited, 2020.

[Doan, Halevy and Ives 2012]DOAN, A.; HALEVY, A.; IVES, Z. *Principles of data integration*. [S.l.]: Elsevier, 2012.

[D'silva et al. 2017]D'SILVA, G. M. et al. Real-time processing of iot events with historic data using apache kafka and apache spark with dashing framework. In: IEEE. *2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*. [S.l.], 2017. p. 1804–1809.

[Falbo et al. 2004]FALBO, R. A. et al. Ontologias e ambientes de desenvolvimento de software semânticos. In: *4th Ibero-American Symposium on Software Engineering and Knowledge Engineering, JIISIC*. [S.l.: s.n.], 2004. v. 1, p. 277–292.

[Ferreira 2019]FERREIRA, J. S. Ontology-based data integration for an internet-of-things (iot) ecosystem to support the collaborative research. dissertação, universidade federal de juiz de fora, 2019, 97p. 2019.

[Forum 2019]FORUM, W. E. *World Economic Forum. Fourth Industrial Revolution Beacons of Technology and Innovation in Manufacturing.e*. 2019. `https://www.weforum.org/whitepapers/fourth-industrial-revolution-beacons-of-technology-and-innovation-in-manufacturing` [Online; accessed 16-April-2019].

[Gruber 1995]GRUBER, T. R. Toward principles for the design of ontologies used for knowledge sharing? *International journal of human-computer studies*, Elsevier, v. 43, n. 5-6, p. 907–928, 1995.

[Guner, Kurtel and Celikkan 2017]GUNER, A.; KURTEL, K.; CELIKKAN, U. A message broker based architecture for context aware iot application development. In: IEEE. *2017 International Conference on Computer Science and Engineering (UBMK)*. [S.l.], 2017. p. 233–238.

[Hanssen 2012]HANSSEN, G. K. A longitudinal case study of an emerging software ecosystem: Implications for practice and theory. *Journal of Systems and Software*, Elsevier, v. 85, n. 7, p. 1455–1466, 2012.

[Hillar 2017]HILLAR, G. C. *MQTT Essentials-A lightweight IoT protocol*. [S.l.]: Packt Publishing Ltd, 2017.

[Hooi, Hassan and Sharifff 2018]HOOI, Y. K.; HASSAN, M. F.; SHARIFFF, M. A. Pragmatic ontology design pattern for contextual integration. In: IEEE. *2018 4th International Conference on Computer and Information Sciences (ICCOINS)*. [S.l.], 2018. p. 1–6.

[Issarny et al. 2016]ISSARNY, V. et al. Revisiting service-oriented architecture for the iot: a middleware perspective. In: SPRINGER. *International conference on service-oriented computing*. [S.l.], 2016. p. 3–17.

[Jones 2020]JONES, R. H. The rise of the pragmatic web: Implications for rethinking meaning and interaction. *Message and Medium: English Language Practices across Old and New Media. Amsterdam: Mouton de Gruyter*, p. 17–37, 2020.

[Kitchenham 2004]KITCHENHAM, B. Procedures for performing systematic reviews. *Keele, UK, Keele University*, v. 33, n. 2004, p. 1–26, 2004.

[Kjær 2007]KJÆR, K. E. A survey of context-aware middleware. In: CITESEER. *Proceedings of the 25th conference on IASTED International Multi-Conference: Software Engineering*. [S.l.], 2007. p. 148–155.

[Koop et al. 2008]KOOP, C. E. et al. Future delivery of health care: Cybercare. *IEEE Engineering in Medicine and Biology Magazine*, IEEE, v. 27, n. 6, p. 29–38, 2008.

[Krytska, Skarga-Bandurova and Velykzhanin 2017]KRYTSKA, Y.; SKARGA-BANDUROVA, I.; VELYKZHANIN, A. Iot-based situation awareness support system for real-time emergency management. In: IEEE. *2017 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*. [S.l.], 2017. v. 2, p. 955–960.

[Lee and Lee 2015]LEE, I.; LEE, K. The internet of things (iot): Applications, investments, and challenges for enterprises. *Business Horizons*, Elsevier, v. 58, n. 4, p. 431–440, 2015.

[Lee and Lee 2015]LEE, I.; LEE, K. The internet of things (iot): Applications, investments, and challenges for enterprises. *Business Horizons*, Elsevier, v. 58, n. 4, p. 431–440, 2015.

[Leminen et al. 2012]LEMINEN, S. et al. Towards iot ecosystems and business models. In: *Internet of things, smart spaces, and next generation networking*. [S.l.]: Springer, 2012. p. 15–26.

[Maciel et al. 2017]MACIEL, R. S. P. et al. Full interoperability: Challenges and opportunities for future information systems. *Sociedade Brasileira de Computação*, 2017.

[Mazhelis, Luoma and Warma 2012]MAZHELIS, O.; LUOMA, E.; WARMA, H. Defining an internet-of-things ecosystem. In: *Internet of Things, Smart Spaces, and Next Generation Networking*. [S.l.]: Springer, 2012. p. 1–14.

[Moore 1993]MOORE, J. F. A new ecology of competition. *Harvard Business Review*, v. 71, n. 3, p. 75–86, 1993.

[Muniz et al. 2019]MUNIZ, M. H. et al. Pragmatic interoperability in iot: a systematic mapping study. In: *Proceedings of the 25th Brazillian Symposium on Multimedia and the Web*. [S.l.: s.n.], 2019. p. 73–80.

[Neiva et al. 2015]NEIVA, F. W. et al. Prime: Pragmatic interoperability architecture to support collaborative development of scientific workflows. In: IEEE. *2015 IX Brazilian Symposium on Components, Architectures and Reuse Software*. [S.l.], 2015. p. 50–59.

[Neiva et al. 2016]NEIVA, F. W. et al. Towards pragmatic interoperability to support collaboration: A systematic review and mapping of the literature. *Information and Software Technology*, Elsevier, v. 72, p. 137–150, 2016.

[Noura, Atiquzzaman and Gaedke 2019]NOURA, M.; ATIQUZZAMAN, M.; GAEDKE, M. Interoperability in internet of things: Taxonomies and open challenges. *Mobile Networks and Applications*, Springer, v. 24, n. 3, p. 796–809, 2019.

[Ogawa et al. 2019]OGAWA, K. et al. Iot device virtualization for efficient resource utilization in smart city iot platform. In: IEEE. *2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. [S.l.], 2019. p. 419–422.

[Oliveira et al. 2016]OLIVEIRA, C. C. et al. Pratical introduction to internet of things: Practice using arduino and node. js. In: *Proceedings of the 22nd Brazilian Symposium on Multimedia and the Web*. [S.l.: s.n.], 2016. p. 17–18.

[Petticrew and Roberts 2008]PETTICREW, M.; ROBERTS, H. *Systematic reviews in the social sciences: A practical guide*. [S.l.]: John Wiley & Sons, 2008.

[Pickler 2007]PICKLER, M. E. V. Web semântica: ontologias como ferramentas de representação do conhecimento. *Perspectivas em Ciência da Informação*, SciELO Brasil, v. 12, n. 1, p. 65–83, 2007.

[Purnomosidi et al. 2014]PURNOMOSIDI, B. et al. Pragmatic web as a service provider for the internet of things. In: IEEE. *2014 2nd International Conference on Information and Communication Technology (ICoICT)*. [S.l.], 2014. p. 308–313.

[Rahmani et al. 2015]RAHMANI, A.-M. et al. Smart e-health gateway: Bringing intelligence to internet-of-things based ubiquitous healthcare systems. In: IEEE. *2015 12th Annual IEEE Consumer Communications and Networking Conference (CCNC)*. [S.l.], 2015. p. 826–834.

[Rees 2003]REES, R. V. Clarity in the usage of the terms ontology, taxonomy and classification. *CIB REPORT*, CIB, v. 284, n. 432, p. 1–8, 2003.

[Rezaei et al. 2014]REZAEI, R. et al. Interoperability evaluation models: A systematic review. *Computers in Industry*, Elsevier, v. 65, n. 1, p. 1–23, 2014.

[Ribeiro et al. 2021]RIBEIRO, E. L. F. et al. Towards a pragmatic interoperability on the midas middleware. In: *Proceedings of the Brazilian Symposium on Multimedia and the Web*. [S.l.: s.n.], 2021. p. 161–168.

[Rivera and Meulen 2014]RIVERA, J.; MEULEN, R. van der. Gartner says the internet of things will transform the data center. *Retrieved August*, v. 5, p. 2014, 2014.

[Robles, Narendra and Kiviranta 2020]ROBLES, M. I.; NARENDRA, N. C.; KIVIRANTA, S. M. Pragmatic distance in iot devices. *IEEE Transactions on Network and Service Management*, IEEE, v. 17, n. 4, p. 2731–2743, 2020.

[Runeson et al. 2012]RUNESON, P. et al. *Case study research in software engineering: Guidelines and examples*. [S.l.]: John Wiley & Sons, 2012.

[Saxena et al. 2017]SAXENA, N. et al. Efficient iot gateway over 5g wireless: A new design with prototype and implementation results. *IEEE Communications Magazine*, IEEE, v. 55, n. 2, p. 97–105, 2017.

[Seydoux et al. 2016]SEYDOUX, N. et al. Iot-o, a core-domain iot ontology to represent connected devices networks. In: SPRINGER. *European Knowledge Acquisition Workshop*. [S.l.], 2016. p. 561–576.

[Silva and Travassos 2020]SILVA, V. M. D.; TRAVASSOS, G. H. Scenariot: Support for scenario specification of internet of things-based software systems. In: SBC. *Anais Estendidos do XI Congresso Brasileiro de Software: Teoria e Prática*. [S.l.], 2020. p. 195–209.

[Skouby and Lynggaard 2014]SKOUBY, K. E.; LYNGGAARD, P. Smart home and smart city solutions enabled by 5g, iot, aai and cot services. In: IEEE. *2014 International Conference on Contemporary Computing and Informatics (IC3I)*. [S.l.], 2014. p. 874–878.

[Solingen and Berghout 1999]SOLINGEN, R. V.; BERGHOUT, E. W. *The Goal/Question/Metric Method: a practical guide for quality improvement of software development*. [S.l.]: McGraw-Hill, 1999.

[Spaccapietra, Parent and Dupont 1992]SPACCAPIETRA, S.; PARENT, C.; DUPONT, Y. Model independent assertions for integration of heterogeneous schemas. *The VLDB Journal*, Springer, v. 1, n. 1, p. 81–126, 1992.

[Syafrudin et al. 2018]SYAFRUDIN, M. et al. Performance analysis of iot-based sensor, big data processing, and machine learning model for real-time monitoring system in automotive manufacturing. *Sensors*, Multidisciplinary Digital Publishing Institute, v. 18, n. 9, p. 2946, 2018.

[Tan and Wang 2010]TAN, L.; WANG, N. Future internet: The internet of things. In: IEEE. *2010 3rd international conference on advanced computer theory and engineering (ICACTE)*. [S.l.], 2010. v. 5, p. V5–376.

[Thein 2014]THEIN, K. M. M. Apache kafka: Next generation distributed messaging system. *International Journal of Scientific Engineering and Technology Research*, v. 3, n. 47, p. 9478–9483, 2014.

[Thoma et al. 2013]THOMA, M. et al. Linked services for enabling interoperability in the sensing enterprise. In: SPRINGER. *International IFIP Working Conference on Enterprise Interoperability*. [S.l.], 2013. p. 131–144.

[Tolk and Muguira 2003]TOLK, A.; MUGUIRA, J. A. The levels of conceptual interoperability model. In: CITESEER. *Proceedings of the 2003 fall simulation interoperability workshop*. [S.l.], 2003. v. 7, p. 1–11.

[Veiga et al. 2018]VEIGA, E. F. et al. A lightweight mobile service for context representation through an iot-oriented ontology. In: *Proceedings of the 24th Brazilian Symposium on Multimedia and the Web*. [S.l.: s.n.], 2018. p. 299–306.

[Venceslau et al. 2019]VENCESLAU, A. D. P. et al. Iot semantic interoperability: A systematic mapping study. In: *ICEIS (1)*. [S.l.: s.n.], 2019. p. 535–544.

[Wassermann and Fay 2017]WASSERMANN, E.; FAY, A. Interoperability rules for heterogenous multi-agent systems: Levels of conceptual interoperability model applied for multi-agent systems. In: IEEE. *2017 IEEE 15th International Conference on Industrial Informatics (INDIN)*. [S.l.], 2017. p. 89–95.

[Weigand and Paschke 2012]WEIGAND, H.; PASCHKE, A. The pragmatic web: Putting rules in context. In: SPRINGER. *International Workshop on Rules and Rule Markup Languages for the Semantic Web*. [S.l.], 2012. p. 182–192.

[Wortmann and Flüchter 2015]WORTMANN, F.; FLÜCHTER, K. Internet of things. *Business & Information Systems Engineering*, Springer, v. 57, n. 3, p. 221–224, 2015.

[Xia et al. 2012]XIA, F. et al. Internet of things. *International journal of communication systems*, v. 25, n. 9, p. 1101, 2012.

[Yin 2018]YIN, R. K. *Case study research and applications*. [S.l.]: Sage, 2018.

[Zörrer et al. 2018]ZÖRRER, H. et al. Chatting roles: A pragmatic service resolution infrastructure for service choreography based on publish/subscribe. *IFAC-PapersOnLine*, Elsevier, v. 51, n. 11, p. 1379–1384, 2018.