UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

CHARLES STEINMETZ

Uma Abordagem para a Integração de Sistemas Industriais Aplicando o Conceito de Internet das Coisas e de Modelos Semânticos no Contexto da Indústria 4.0

Dissertação apresentada como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação

Orientador: Prof. Dr. Carlos Eduardo Pereira

Porto Alegre
2018

CIP — CATALOGAÇÃO NA PUBLICAÇÃO

Steinmetz, Charles

Uma Abordagem para a Integração de Sistemas Industriais Aplicando o Conceito de Internet das Coisas e de Modelos Semânticos no Contexto da Indústria 4.0 / Charles Steinmetz. – Porto Alegre: PPGC da UFRGS, 2018.

105 f.: il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2018. Orientador: Carlos Eduardo Pereira.

1. Internet das Coisas. 2. Indústria 4.0. 3. Ontologia. 4. Integração. I. Pereira, Carlos Eduardo. II. Título.

“A persistência é o menor caminho do êxito”

— CHARLES CHAPLIN

AGRADECIMENTOS

Aos meus pais, pelo apoio e por serem minha principal fonte de inspiração. Desde cedo mostraram-me que as conquistas acontecem com dedicação e persistência.

Ao Programa de Pós-Graduação Computação - PPGC pela oportunidade de estudo e pesquisa.

Ao orientador Dr. Carlos Eduardo Pereira pelas diversas oportunidades oferecidas, pela confiança depositada e pelas importantes sugestões dadas.

Ao CNPq pela concessão da bolsa de estudos no Brasil e à CAPES pela bolsa de estudos na Alemanha.

Ao Professor Reiner Franthesco Perozzo, que incentivou-me a seguir meus estudos.

Aos colegas e amigos do Laboratório de Sistemas Embarcados - LSE e do PPGEE. Aos colegas e amigos da Alemanha que ajudaram a construir uma das melhores experiências que já tive, em especial o Luis Filipe Araujo Pessoa que se tornou um grande amigo.

À Maiara Bortolussi de Oliveira, pela paciência, incentivo e compreensão durante as fases difíceis que passei, desde a graduação até agora. Obrigado por estar ao meu lado.

À Fabíola Gonçalves Coelho Ribeiro, pessoa a qual em um curto prazo tornou-se uma fonte de apoio e inspiração. Te agradeço pelas horas dedicadas a ajudar-me, pelas discussões e pelas palavras de incentivo.

À Greyce Schroeder que me apoiou desde o início do mestrado até o final, sendo uma pessoa que trouxe-me calma e sempre incentivou-me a continuar.

Aos meus amigos pela paciência com minhas ausências e preocupação com o meu bem estar.

RESUMO

Com a chegada da era da computação ubíqua, o número de dispositivos com poder computacional vêm crescendo de forma acelerada. A conexão desses dispositivos em uma rede de comunicação traz novas possibilidades e serve como base para o conceito chamado Internet das Coisas (*Internet of Things* - IoT). Uma das aplicações desse conceito é no domínio industrial e está impulsionando uma nova revolução industrial, a chamada Indústria 4.0. Esta pesquisa apresenta uma abordagem de integração de componentes de forma automatizada, no contexto da Indústria 4.0, utilizando ontologias para representar os elementos do sistema e um *middleware* IoT para servir de meio de integração.

Essa abordagem utiliza conceitos propostos em trabalhos relacionados, com características inovadoras na criação de sistemas IoT. Dentre estas está a integração automatizada a partir de modelos semânticos, que possibilita que usuários modelem sistemas em alto nível. A partir desse modelo, as interfaces de comunicação são criadas automaticamente, trazendo uma garantia de consistência sintática nas chamadas de métodos ou funções. Além disso, a pesquisa proposta traz a possibilidade de usar esse mesmo modelo semântico para apresentar as informações ao usuário final. Como resultados deste trabalho podem ser ressaltados o desenvolvimento de uma ontologia para modelar esses elementos industriais e também o de uma extensão para um *middleware* IoT a fim de poder-se trabalhar com esses modelos de forma automatizada.

Palavras-chave: Internet das Coisas. Indústria 4.0. Ontologia. Integração.

An Approach for the Integration of Industrial Systems through the Adoption of the Internet of Things and Semantic Model Concepts in the Industry 4.0 Context

ABSTRACT

With the advent of the ubiquitous computing era, the number of devices with computing power is growing rapidly. The connection of these devices onto a communication network brings new possibilities and serves as the basis for the concept called the Internet of Things (IoT). One of the applications of this concept is in the industrial domain where it is driving a new industrial revolution, usually designated as "Industry 4.0".

This research study presents a automated integration approach in the context of Industry 4.0, using ontologies to represent elements of the system and an IoT middleware to provide a means for its integration.

This approach uses concepts proposed in related works with innovative features regarding the creation of IoT systems. Among these features, an automated integration based on semantic models is proposed, which enables users to model their systems at a high level. From this model, communication interfaces are created automatically, bringing a guarantee of syntactic consistency in calling methods or functions. Another advantage that this work brings is the possibility of using this same semantic model to present information to the end user. As result of this work, an ontology was developed to model industrial elements, and an extension for an IoT middleware was developed to enable to work with these models.

Keywords: Internet of Things, Industry 4.0, Ontology, Integration.

LISTA DE ABREVIATURAS E SIGLAS

IoT	Internet of Things
RFID	Radio frequency Identification
IP	Internet Protocol
IIoT	Industrial Internet of Things
IMS	Intelligent Maintenance Systems
SPSC	Spare Parts Supply Chain
I2MS2C	Integrating Intelligent Maintenance Systems and Spare Parts Supply Chains
SoA	Service-oriented Architecture
MDA	Model Driven Architecture
UML	Unified Modeling Language
B2C	Business-to-consumer
C2C	Consumer-to-consumer
B2B	Business-to-business
B2G	Business-to-government
API	Application Programming Interfaces
M2M	Machine to Machine
PLM	Product Lifecycle Management

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

As interações humanas têm passado por um processo de transformação nos últimos anos. Dispositivos inteligentes têm alterado a maneira com que as pessoas interagem entre si e como as mesmas realizam suas atividades. A tecnologia está conectando cada vez mais pessoas(MIRANDA et al., 2015), serviços, cidades, casas, empresas e muito mais (FARAHZADI et al., 2017) (QIN et al., 2016). Dada a alta integração e conexão entre os componentes físicos e virtuais, vindos de diferentes fontes, inúmeras informações podem ser capturadas e processadas a fim de contribuir para o processo de tomada de decisões nas mais diferentes corporações.

Internet das Coisas (*Internet of Things* - IoT), tem sido amplamente utilizada em problemas em que é necessário interagir com dispositivos físicos, como sensores, atuadores, máquinas, entre outros. A adoção da IoT possibilita a "união" entre mundo físico e o virtual de forma que exista uma profunda interação, cooperação, compartilhamento de informações e um melhor "compartilhamento de conhecimento"entre os objetos desses dois mundos (YAO; LIN, 2016).

O conceito de IoT foi proposto em 1999 por Ashton na área de logística (GUBBI et al., 2013). As primeiras aplicações de IoT consistiam na identificação e rastreabilidade de objetos em aplicações de logística através do uso de RFIDs (Radio Frequency Identification) (SHENG et al., 2013). Atualmente as aplicações que empregam o conceito de IoT em seus processos têm se expandido mais e mais, assim IoT também refere-se à conexão de objetos através do protocolo *Internet Protocol* (IP) (DEERING, 1998), o que permite que mais dispositivos possam conectar-se de forma nativa, ou seja, sem a necessidade de criar adaptações para que haja comunicação entre eles (MAINETTI; PATRONO; VILEI, 2011).

De acordo com (ATZORI; IERA; MORABITO, 2010) o conceito de IoT é classificado de acordo com três visões: a primeira visão seria orientada à *Internet*, a segunda orientada aos objetos genéricos (aptos a integração futura) e a terceira é a visão semântica (que fornece suporte à representação e ao armazenamento das informações).

Dentre as principais característica da IoT destaca-se a ubiquidade. Desta maneira, a tecnologia das "coisas" e objetos interconectados não é percebida e permite que tais objetos trabalhem conjuntamente em prol de um objetivo em comum (GIUSTO et al., 2010). Existem previsões que em 2020 o número de objetivos inteligentes e conectados à *Internet* chegue aos 50 bilhões (VESTBURG, 2010) (EVANS, 2011). Sendo assim,

inúmeros desafios ao desenvolvimento e implantação desta tecnologia têm surgido como, por exemplo, a interoperabilidade entre objetos heterogêneos (LIU et al., 2017) (ATZORI; IERA; MORABITO, 2010) (GUBBI et al., 2013).

No contexto industrial, existe uma variedade de sistemas que precisam trocar informações, como sistemas de engenharia, sistema de compras, sistemas de controle de processos, de controle de produção entre outros. A interoperabilidade nesse domínio é algo buscado há muito tempo e, atualmente, modelos semânticos são vistos como uma potencial solução (PLOENNIGS et al., 2017).

Ainda na indústria, o conceito de IoT tem trazido vantagens como eficácia nos processos produtivos, a possibilidade de compartilhar informações entre empresas, o aumento do lucro, a redução dos custos e, consequentemente, uma maior competitividade (FAN; ZHOU, 2011). Essas informações são usadas em tempo de execução para tomada de decisões possibilitando, assim, sistemas reativos e com maior performance (SPIESS et al., 2009).

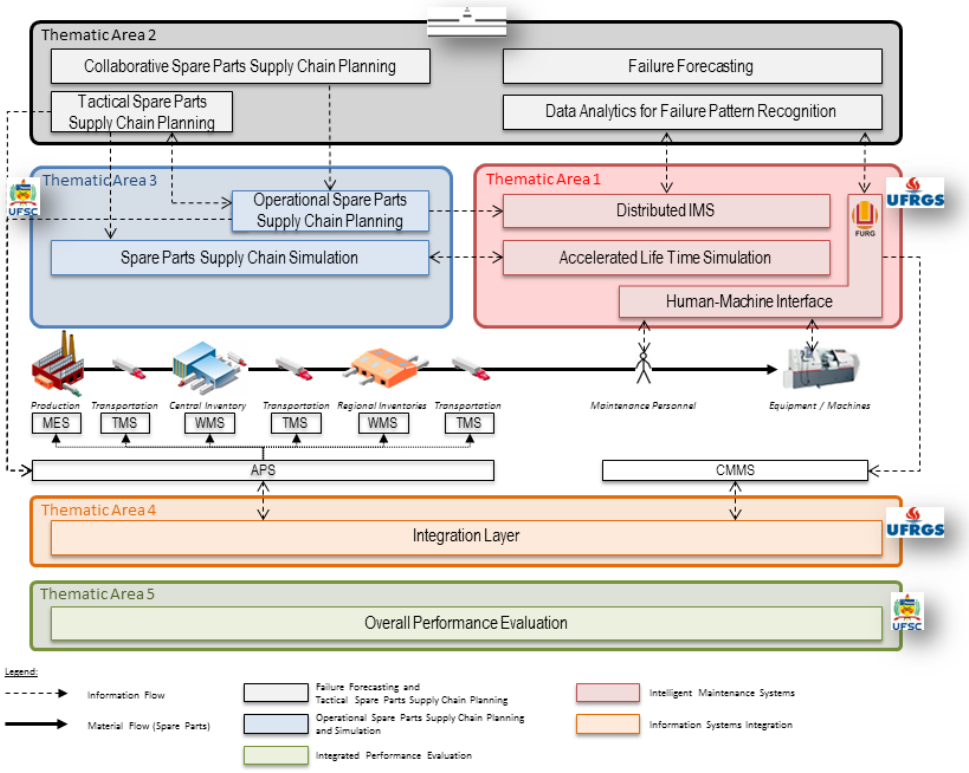
O conceito de IoT é aplicado, por exemplo, em Sistemas de Manutenção Inteligentes (*Intelligent Maintenance Systems* - IMS) (LEE et al., 2006). Estes sistemas permitem, através da análise dos dados lidos em sensores, prever diferentes status de um equipamento. Neste caso, é possível prever quando um equipamento irá falhar e a partir disto traçar estratégias de contenção e/ou contingência para a manutenção planejada do mesmo (ESPÍNDOLA et al., 2012). Isso evita que a produção fique parada por atrasos em reparos não planejados. Pode ser citado como benefício da IoT no domínio dos IMS o fato de que a produção poderá funcionar interruptamente ou com melhor previsão/contenção de falhas (BANDYOPADHYAY; SEN, 2011).

IoT pode ser aplicada também em Cadeias de Suprimentos de Peças de Reposição (*Spare Parts Supply Chain* - SPSC), onde possibilita um planejamento baseado em informações coletadas de dispositivos em campo, por exemplo, ao monitorar-se uma planta industrial, através do conceito de IMS, a partir da coleta de informações sobre os dispositivos, tais como temperatura, vibração, torque, entre outras, pode-se estimar o tempo remanescente de operação sem falha e assim criar um planejamento mais preciso (BANDYOPADHYAY; SEN, 2011).

O projeto *Integrating Intelligent Maintenance Systems and Spare Parts Supply Chains* (I2MS2C) (ESPÍNDOLA et al., 2012) é parte do programa *Brazilian-German Collaborative Research Initiative on Manufacturing Technologies* (BRAGECRIM). Este projeto busca contribuir para a efetividade e a eficiência das operações de gerenciamento

de serviços para sistemas técnicos complexos, unindo pesquisas entre as áreas de IMS e SPSC. O projeto foi especificado em cinco áreas temáticas que seguem as perspectivas IMS e SPSC como pode ser observado na Figura 1.1.

Figura 1.1: Visão geral do projeto $I2MS2C$



Fonte: (ESPÍNDOLA et al., 2012)

Uma das áreas temáticas do projeto $I2MS2C$ é responsável pela integração dos elementos que envolvem esses dois domínios e é a área com a qual a proposta deste trabalho está mais relacionada. Dentro desse contexto, o trabalho de Silva and Pereira (2014) propõe uma ontologia para descrever o conhecimento e as relações entre os domínios envolvidos na integração entre IMS e SPSC. Esse conhecimento vai desde itens encontrados em uma planta industrial, como dispositivos físicos e suas partes até componentes mais abstratos como processos, atores ou regras que são encontrados em SPSC. Essa descrição foi desenvolvida para servir de base para futuras arquiteturas de integração. Conceitos apresentados nesse trabalho foram usados na presente proposta, entretanto este trabalho não cobre alguns pontos importantes como conceitos de IoT e outros relacionados à sistemas baseados em *middlewares*.

Ainda no escopo do projeto I2MS2C foi desenvolvido o trabalho de Rodrigues (2016) que apresenta um *middleware* para integração dos níveis de gerenciamento indus-

trial, com base nos conceitos dos Sistemas Ciberfísicos de Produção (CPPS).

Nesse contexto, o presente trabalho busca evoluir os trabalhos já desenvolvidos no âmbito do projeto $I2MS2C$ como parte de sua contribuição. É proposto uma abordagem que possibilite a integração entre componentes industriais através de uma modelagem semântica e do uso de IoT para possibilitar essa conexão entre os elementos envolvidos.

Essa abordagem segue a lógica de trabalhos relacionados já existentes na literatura, entretanto, existem características inovadoras que trazem algumas vantagens na criação de sistemas IoT, como por exemplo, a integração automatizada a partir de modelos semânticos, que possibilita que usuários modelem seu sistemas em alto nível, com o nível de detalhes necessário para cada projeto. A partir desse modelo, as interfaces de comunicação são criadas automaticamente, trazendo uma garantia de consistência sintática na chamada dessas funções. Outro ponto que este trabalho traz é a possibilidade de usar esse mesmo modelo semântico para representar a forma de como os dados referentes aos dispositivos devem ser exibidos aos usuários finais. Foram desenvolvidas uma ontologia para modelar esses elementos industriais e também uma extensão de um *middleware* IoT para que fosse possível trabalhar com esse tipo de modelos.

1.1 Motivação

Este trabalho está motivado pela ocorrência de uma nova revolução industrial que exigirá novas abordagens para esse patamar ser atingido. Diversos conceitos estão sendo estudados e propostos para suprir necessidades não apenas da indústria, mas para problemas em geral, como por exemplo o conceito de IoT. Já no contexto da Indústria 4.0, o uso de tecnologias relacionadas a semântica está crescendo, pois provê uma forma de compartilhar informações entre máquinas (PLOENNIGS et al., 2017).

Um dos pilares da Indústria 4.0 é o uso da internet não apenas para conectar máquinas, dispositivos, sensores e pessoas, mas também para criar novas funcionalidades. Dentre essas funcionalidade, pode ser citada o uso da internet como fonte de dados/informações para os produtos criados (BASSI, 2017).

De acordo (LIU et al., 2017) as seguintes questões devem ser consideradas na integração e interação de componentes inteligentes:

1. Como os dados de manufatura devem ser codificados para suportarem a análise, a troca, o processamento e o compartilhamento de dados em um ambiente IoT?

2. Como a massa de dados, capturada pelos dispositivos heterogêneos deve ser transferida e integrada?
3. Qual metodologia deve ser usada para agregar valor às informações e, também, para dar suporte à tomada de decisões da empresa?
4. Como se consegue otimização para o processo de manufatura, baseado em informações em tempo real?

Uma possível solução para esse problema é o uso de modelos semânticos para representar informações que serão trocadas entre os componentes inteligentes. Tais modelos semânticos permitem criar uma linguagem comum aos diversos componentes e, ainda, garante que as partes envolvidas possam entender as informações trocadas. Esses modelos permitem ainda que máquinas interajam umas com as outras e com humanos em uma linguagem comum.

Na literatura, são definidos alguns modelos semânticos desenvolvidos para esse domínio, porém existe a necessidade de evoluir esses modelos, agregando novos conceitos conforme o sistema e as tecnologias evoluem. Isso se faz importante para garantir que se tenha um maior aproveitamento do que está disponível no mercado, pois máquinas, processos e outros fatores evoluem de forma muito rápida. Nesse sentido, um item importante para garantir uma integração completa é manter uma representação fidedigna do sistema e uma interoperabilidade entre os componentes.

Segundo (DIEP; ALEXAKOS; WAGNER, 2007) existem três requisitos para ter-se a interoperabilidade nesse novo modelo de indústria:

- toda informação compartilhada entre as diferentes entidades deve ser estruturada em um modelo de referência comum;
- todo o conhecimento deve ser reunido em uma ontologia comum;
- o acesso a essa ontologia deve ser garantido com o uso de uma linguagem comum, adaptada para as capacidades de cada componente.

Visto o grande número de possibilidades que a integração de sistemas pode trazer para uma indústria, existe uma necessidade de criar meios para garantir que esses sistemas possam comunicar-se e mais que tudo, que possam ter a mesma compreensão de uma certa informação.

Nesse sentido, o uso de *middlewares*, agregados com modelos semânticos, podem contribuir como uma solução para se garantir a interoperabilidade. Apesar de existirem ferramentas de integração que utilizam modelos semânticos, ainda existe uma carência

de uma plataforma que utiliza estes modelos desde as camadas físicas (dispositivos) até nas camadas mais altas que interagem com o usuário. Além disso, para adoção de uma abordagem por parte de uma organização, é importante que essa abordagem seja viável e que traga benefícios reais para o negócio. Para isso, a presente proposta busca contribuir ainda com um método de geração automática de interfaces para a integração de sistemas. Essas interfaces podem ser usadas para a implementação da integração, padronizando etapas no desenvolvimento e evitando possíveis erros humanos nesse processo. Essas contribuições podem alavancar a adoção de conceitos emergentes e inovadores, levando indústrias com métodos tradicionais a usarem tecnologias atuais e tornando-as ainda mais competitivas no mercado.

1.2 Objetivos

O principal objetivo deste trabalho é propor uma abordagem de integração de componentes de forma automatizada, no contexto da Indústria 4.0, utilizando ontologias para representar os elementos do sistema e um *middleware* IoT para servir de meio de integração.

Os objetivos específicos são:

- Prover uma integração de componentes industriais utilizando conceitos de IoT com 4 camadas.
- Criar um modelo semântico com base em trabalhos encontrados na literatura que são relacionados à integração dos domínios de IMS e SPSC.
- Criar uma maneira de gerar interfaces de comunicação de forma automática com base nos modelos do sistema.
- Criar uma extensão de um *middleware* consolidado para que seja possível trabalhar com modelos semânticos através de uma API.
- Possibilitar a automatização de processos com geração de código fonte a partir de um modelo semântico.

1.3 Organização da dissertação

O presente trabalho se divide em sete capítulos, sendo o primeiro a Introdução. Nesse capítulo tem-se uma contextualização do tema bem como os objetivos do trabalho.

O Capítulo 2 apresenta conceitos importantes para o desenvolvimento e entendimento da pesquisa, como por exemplo o conceito de IoT, Indústria 4.0, Web Semântica e outros derivados.

O Capítulo 3 traz os trabalhos já existentes na literatura e que são relacionados ao tema da pesquisa. Ao final do capitulo e feita uma discussão dos trabalhos relacionados e apresentadas lacunas que o presente trabalho abordará.

O Capítulo 4 apresenta a proposta de integração baseada em IoT e no uso de modelos semânticos no contexto da Indústria 4.0.

O Capítulo 5 descreve a implementação da proposta.

O Capítulo 6 apresenta os estudos de caso realizados com uma breve discussão sobre eles.

O Capítulo 7 conclui o trabalho ressaltando as contribuições e os trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

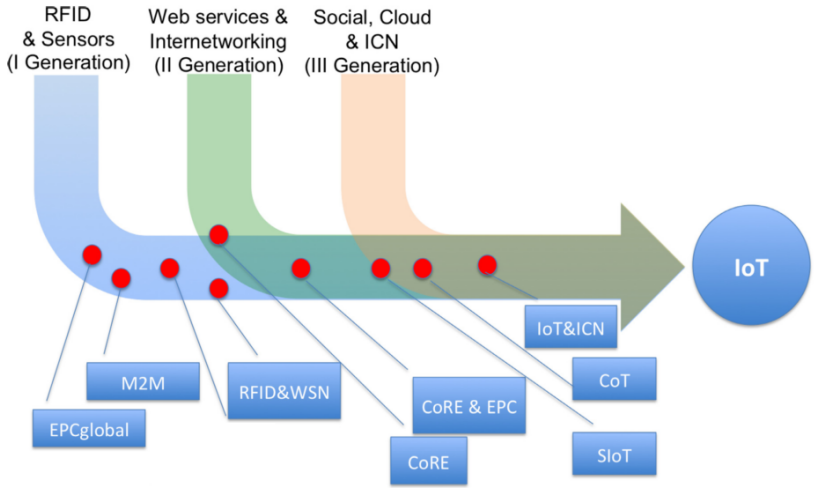
Este capítulo contém a revisão dos principais conceitos de Internet das Coisas, Indústria 4.0, Ontologias, OWL, além de dar uma visão sobre a ferramenta Protégé.

2.1 Internet das Coisas

O termo Internet das Coisas (*Internet of Things*) foi usado pela primeira vez em 1999 em uma apresentação sobre o tema Cadeia de Suprimentos (*Supply Chain*)(ASHTON, 2009). Esse termo tornou-se popular pelo seu sucesso com o uso de RFID (*Radio Frequency Identification*) para a identificação de objetos (KORTUEM et al., 2010).

Desde a criação deste conceito até os dias atuais, houveram evoluções que segundo (ATZORI; IERA; MORABITO, 2017) podem ser representadas em três gerações, como mostrado na Figura 2.1.

Figura 2.1: Gerações do conceito de IoT



Fonte: Adaptado de (ATZORI; IERA; MORABITO, 2017)

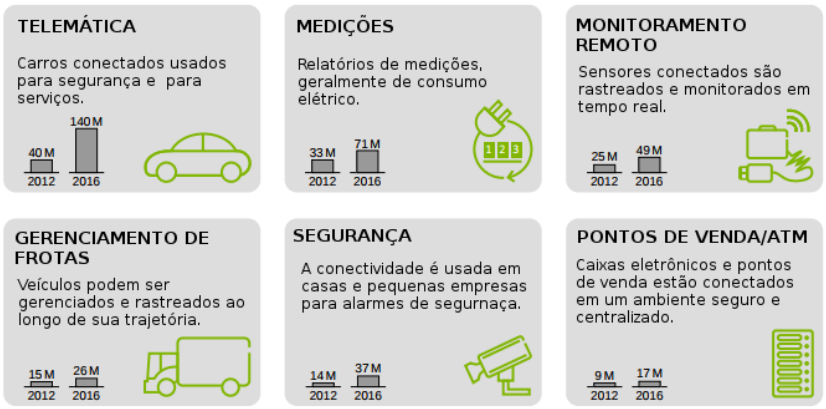
A primeira geração é marcada pelos objetos identificados com *tags*(ATZORI; IERA; MORABITO, 2017). Nesse contexto, destaca-se a marcação com *tags* RFID. A ideia de se ter objetos unicamente identificados e conectados cresceu e o conceito de IoT começou a ser utilizado para referir-se à conexão de objetos através do protocolo *Internet Protocol* (IP) (DEERING, 1998) o que permite que mais dispositivos possam conectar-se de forma nativa, ou seja, sem a necessidade de criar adaptações para que haja comunicação entre eles(MAINETTI; PATRONO; VILEI, 2011). Assim, espera-se que a internet possa possibilitar a comunicação não só de pessoas com pessoas, mas também de pessoas

com objetos ou até objetos com objetos, este último, chamado de comunicação máquina a máquina (ou em inglês, *Machine to Machine* - M2M) (HOLLER et al., 2014).

M2M refere-se às soluções que permitem a comunicação entre dispositivos em uma aplicação específica, via comunicação cabeada ou *wireless*. É uma tecnologia que permite que máquinas comuniquem-se sem intervenção humana, automatizando a transferência de dados entre elementos inteligentes, produtores e consumidores de dados (HOLLER et al., 2014).

M2M pode ser aplicada em diferentes cenários e, no setor empresarial, por exemplo, geralmente é usada para atingir uma maior produção, redução de custos e aumentar a segurança (HOLLER et al., 2014). Entre 2012 e 2016 os principais domínios de aplicação desse conceito podem ser vistos na Figura 2.2. O segmento que mais se destaca é o de telemática (*telematics*) em carros e veículos, como nos sistemas de navegação, controle remoto de diagnósticos, cobrança automática de pedágios ou, ainda, sistema anti furto.

Figura 2.2: Principais cenários de aplicação de M2M entre 2012 e 2016



Fonte: Adaptado de (HOLLER et al., 2014)

Podem ser encontrados ainda entre os principais segmentos que utilizam M2M, aplicações de medição de consumo de gás, água e energia, monitoramento remoto, gerenciamento de frotas de veículos e aplicações de segurança (HOLLER et al., 2014).

IoT traz a ideia de "coisas"conectadas através da internet e isso vem de encontro com o conceito de M2M. Porém, o conceito de IoT refere-se à conexão desses sistemas e sensores além do uso de tecnologias relacionadas à Internet. IoT busca conectar objetos reais para que esses sejam capazes de se conectarem, comunicarem e interagirem como uma pessoa utiliza a web (HOLLER et al., 2014).

A segunda geração traz a possibilidade de fornecer a objetos simples a capacidade de conectar-se à internet como quaquer outro *host*(ATZORI; IERA; MORABITO, 2017). Surge o conceito de *Web of Things* (GUINARD; TRIFA, 2009), onde dispositivos es-

tão conectados à *World Wide Web* (WWW) (BERNERS-LEE; FISCHETTI, 2001) como recursos. Os dispositivos são modelados como serviços com uma identificação única, semelhante a páginas de internet que possuem um endereço único.

Ainda nessa geração, surge o conceito de Serviços de Redes Sociais (*Social Networks Services* - SNS) (RICHTER; KOCH, 2008), que possibilita que dispositivos possam interagir com redes sociais, ficando mais próximos aos usuários (ATZORI; IERA; MORABITO, 2017). Por exemplo, sensores de uma casa podem publicar no *Facebook* alguma alteração nas variáveis monitoradas nesse local e o proprietário terá uma via de acesso mais fácil a essa informação.

A terceira geração engloba conceitos mais modernos e aproxima ainda mais as pessoas aos objetos envolvidos em um sistema IoT. Essa geração é geralmente chamada de Internet do Futuro (*Future Internet*), que vai explorar a tecnologia de computação em nuvem (*cloud computing*) e será centralizada nas pessoas, no conteúdo e nos serviços (ATZORI; IERA; MORABITO, 2017).

Nessa geração, surge a ideia de Social Internet of Things (ATZORI et al., 2012), onde adiciona-se a capacidade de objetos participarem de comunidades, com grupos de interesse em comum e colaborando entre si em suas ações.

Para viabilizar esses conceitos, é necessário um padrão de comunicação e uma forma de representar as informações entre sistemas, máquinas e pessoas, para que todos consigam receber e transmitir informações sem que hajam interpretações ambíguas. O uso de semântica pode ser uma solução viável para esse problema. Nesse sentido, o modelo semântico *Semantic Sensor Network Ontology* (SSN) (COMPTON et al., 2012) destaca-se na literatura por trazer representações genéricas que podem ser usadas para diferentes domínios IoT.

Nesse sentido, para possibilitar que haja essa capacidade de comunicação e interação por parte de objetos do mundo real, e que geralmente são heterogêneos, é necessário ter-se uma arquitetura flexível.

Atualmente existe um grande número de arquiteturas propostas, porém nenhuma é tida como um modelo de referência. Existem projetos, como a IoT-A (KRCO; POKRIC; CARREZ, 2014), que buscam fornecer uma arquitetura comum baseada nas necessidades encontradas por pesquisadores e pela indústria (AL-FUQAHA et al., 2015).

O trabalho de (YANG et al., 2011) apresenta uma arquitetura de três camadas: camada de percepção; camada de rede; e camada de aplicação. A principal função da camada de percepção é reconhecer e perceber “coisas", coletando as informações desses

objetos. Nessa camada existe uma preocupação de utilizar dispositivos de baixo custo e de baixo consumo de energia. A camada de rede é responsável por enviar as informações coletadas através da *internet*. A camada de aplicação é responsável por prover um meio de compartilhar as informações entre os usuários e ainda manter essas informações seguras.

Já trabalho de (WU et al., 2010) propõe uma arquitetura dividida em cinco camadas, como pode ser observado na Figura 2.3.

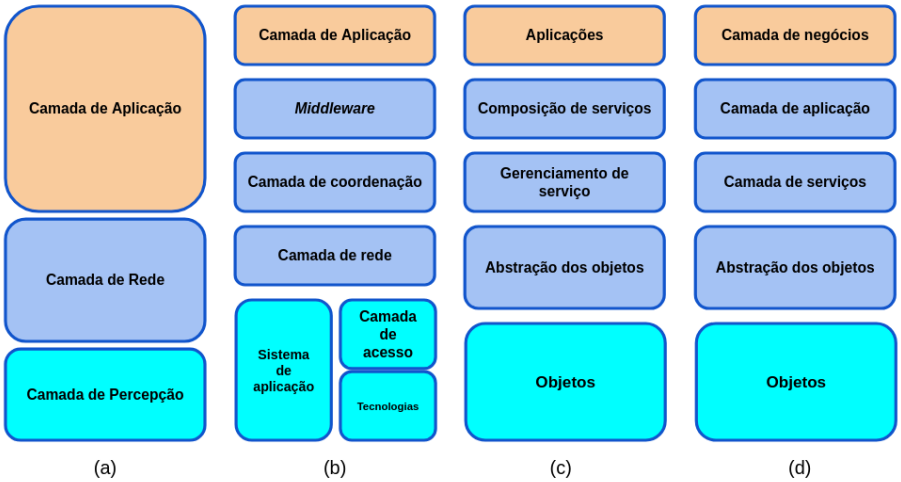
Figura 2.3: Arquitetura de cinco camadas

Fonte: (WU et al., 2010)

A camada de percepção (*Perception Layer*) é responsável por perceber as variáveis físicas de objetos, como temperatura, pressão, entre outros. Ela é composta por diversos sensores e converte essas informações em sinais digitais, os quais são transportados pela camada superior, chamada de camada de transporte (*Transport Layer*). Essa camada é responsável por transmitir os dados da camada de percepção para a camada de processamento (*Processing Layer*) através de uma rede sem fio ou cabeada. A camada de processamento é responsável por armazenar, analisar e processar esses dados que são recebidos. A camada de aplicação (*Application Layer*) tem função de fornecer todos tipos de aplicações para cada indústria. Por último, a camada de negócio (*Business Layer*) traz a possibilidade de adicionar aplicações de gerenciamento de aplicações incluindo o modelo de negócios da empresa.

Em (AL-FUQAHA et al., 2015) são apresentadas arquiteturas com diferentes níveis, como pode ser observado na Figura 2.4.

Figura 2.4: Diferentes arquiteturas IoT



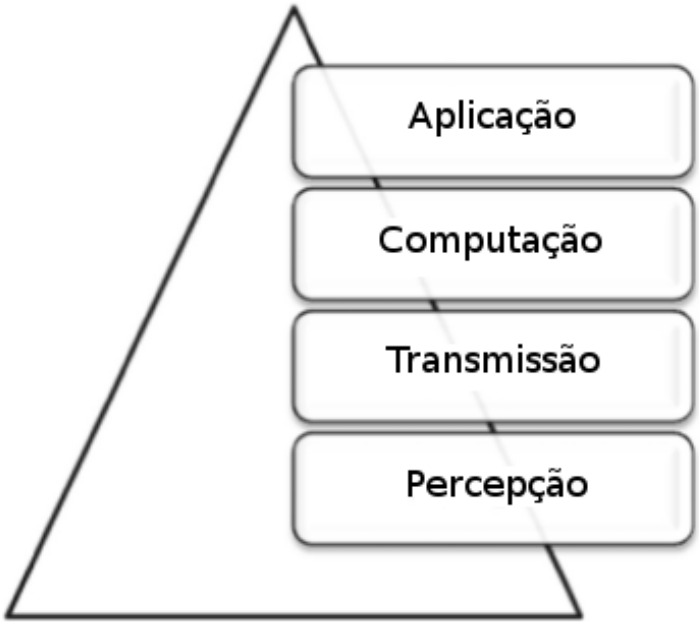
Fonte: Adaptado de (AL-FUQAHA et al., 2015)

Pode-se observar que existem diferentes níveis de abstração nas arquiteturas encontradas na literatura. Em (a) tem-se uma arquitetura de três níveis; em (b) uma arquitetura baseada em *middleware*; em (c) uma arquitetura baseada em serviços (SoA) e em (d) é apresentada uma arquitetura de cinco níveis.

Apesar de existirem diferentes propostas nesse tópico, é possível perceber que as camadas possuem semelhanças na maioria dos trabalhos encontrados na literatura. O trabalho de (TRAPPEY et al., 2017) apresenta uma revisão sobre padrões relacionados à IoT e, dentro de um conjunto de arquiteturas de três camadas até de nove camadas, o autor conclui que os elementos podem ser “logicamente e intuitivamente"divididos entre as camadas de percepção (*Perception*), transmissão (*Transmission*), computação (*Computation*) e aplicação (*Application*). A Figura 2.5 apresenta a arquitetura apresentada em (TRAPPEY et al., 2017).

Esta arquitetura de quatro níveis pode ser comparada com a arquitetura proposta por (YANG et al., 2011) e percebe-se que, com a evolução dos sistemas IoT, adicionou-se uma camada (a de computação) que recebe, armazena e disponibiliza os dados para as aplicações clientes. Já se comparada com a proposta de (WU et al., 2010) (com cinco camadas) observa-se que a camada de negócio está incluída na camada de aplicação.

Figura 2.5: Arquitetura de quatro camadas



Fonte: (TRAPPEY et al., 2017)

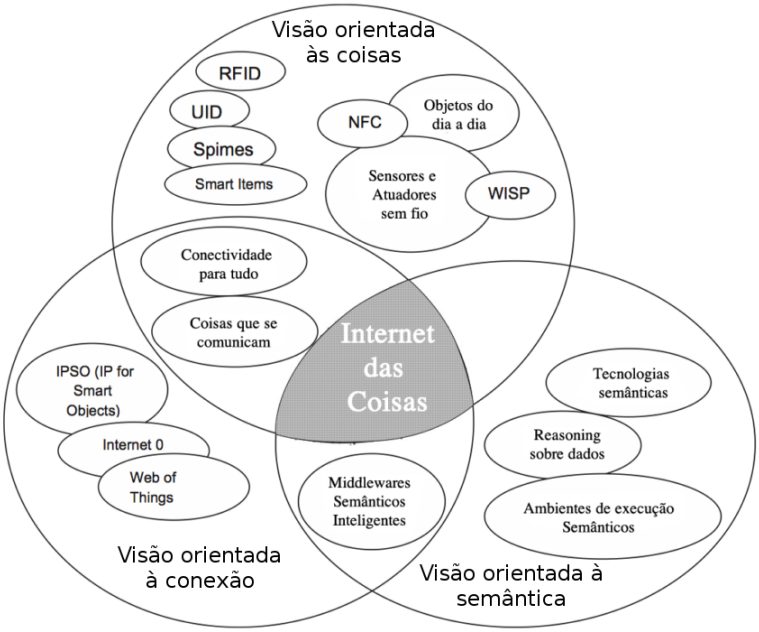
2.1.1 Perspectivas da IoT

A IoT pode ser pode ser compreendida como uma convergência de três principais visões: a) uma visão orientada às coisas; b) uma visão orientada à internet e; c) uma visão orientada à semântica (ATZORI; IERA; MORABITO, 2010).

A Figura 2.6 apresenta essas visões e alguns elementos pertencentes a cada uma.

- **Visão orientada às "coisas"**: Essa visão traduz dados entre o mundo real e o virtual, em como identificar um objeto unicamente, em como aplicar ações e ler estados de dispositivos físicos. É nessa área que é adicionada a sensibilidade ao contexto em que o dispositivo está inserido. Acredita-se que os humanos serão a minoria no grupo de fornecedores e consumidores de dados trafegados na internet, ou seja, a rede será mais usada por máquinas e dispositivos.
- **Visão orientada à internet**: Com o surgimento de tecnologias como o TCP/IP, que padronizaram a maioria das comunicações entre computadores utilizadas na internet, é possível criar conectar dispositivos em uma mesma rede, todos com uma identificação única. Ainda nessa visão, existe uma uma exigência maior quanto ao consumo de energia por parte dos dispositivos, visto que estes ficaram mais baratos e menores e, consequentemente, com menor autonomia de energia.
- **Visão orientada à semântica**: Existe uma necessidade de se ter um padrão para

Figura 2.6: Paradigma de IoT como uma convergência de diferentes visões



Fonte: (ATZORI; IERA; MORABITO, 2010)

abstrair os dados relativos a dispositivos. A interoperabilidade é um conceito chave na IoT, visto que diferentes dispositivos e sistemas podem estar conectados trocando informações.

Ontologias podem contribuir para esse ponto fornecendo práticas, como *reasoning* e buscas ontológicas. Esse tipo de modelo utiliza tecnologias de representação como o XML(BRAY et al., 1997) e o JSON(CROCKFORD, 2006) o que facilita a optenção de interoperabilidade e permite a reutilização desses modelos de dados (ATZORI; IERA; MORABITO, 2010).

2.1.2 Middlewares e plataformas IoT

Um dos problemas da IoT é a questão da heterogeneidade dos elementos que compõem o sistema, pois esse conceito propõe justamente a integração de diferentes "coisas" que podem ter implementações diferentes. Esse problema pode ser resolvido através do uso de *middlewares* entre essas "coisas" e as aplicações, provendo uma interface de comunicação entre dispositivos, sistemas operacionais e arquiteturas (FARAHZADI et al., 2017).

2.1.2.1 FIWARE

Atualmente existem muitas soluções para o problema de integração de sistemas. Grandes empresas como Google, Microsoft, Amazon ou IBM disponibilizam ferramentas e serviços para armazenamento de dados, servidores, *Big Data* e inclusive soluções para IoT. Por exemplo, o Watson IoT (IBM, 2017) oferece serviços para construção de sistemas IoT utilizando a infraestrutura da IBM. A maioria dessas soluções oferecidas são genéricas e podem ser feitas algumas customizações conforme a necessidade de cada problema, contudo, existem poucas possibilidades em que se tenha serviços específicos para um problema, como por exemplo para uma planta industrial.

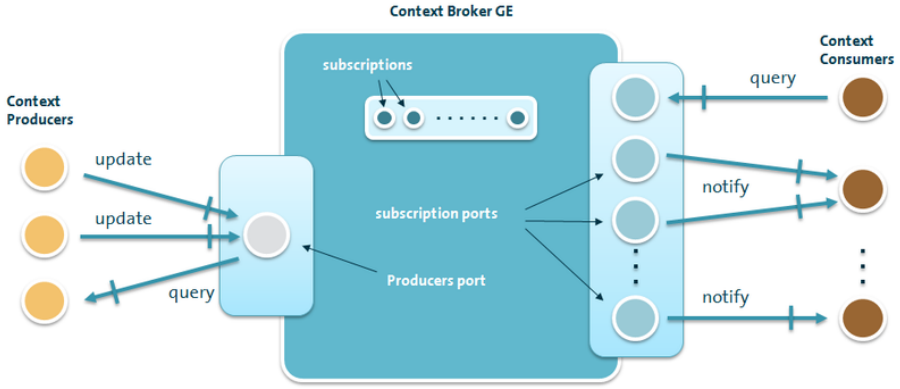
Além disso, os recursos gratuitos oferecidos por essas companhias são bastante limitados e ainda impõem cotas de uso desses recursos. Isso pode tornar-se um problema à medida que o sistema cresce, pois o custo aumentará também. A escalabilidade é um dos pilares de sistemas IoT justamente pelo fato de se poder conectar qualquer objeto que possa fornecer dados importantes, então o crescimento da maioria desses sistemas é inevitável.

Considerando esses pontos levantados, para esse trabalho optou-se por utilizar a plataforma FIWARE que pode ser definida como uma ferramenta que "fornece um conjunto de API (*Application Programming Interfaces*) bastante simples e potente que facilitam o desenvolvimento de aplicações inteligentes em vários setores". As descrições dessas APIs são públicas e gratuitas além de o projeto ser de código aberto, possibilitando que novas funcionalidades ou correções sejam feitas por desenvolvedores de diferentes projetos (FIWARE, 2017a).

O projeto FIWARE oferece um espaço chamado *FIspace*(BUZIN; FOURNIER; ARCUSHIN, 2013) que é uma plataforma para sistemas de colaboração. Essa plataforma atende tanto soluções genéricas como as soluções para problemas específicos e possibilita que desenvolvedores modelem, implementem, e ainda façam a implantação de seus sistemas. Essa é mais uma vantagem que a maioria das demais soluções encontradas na literatura não fornecem.

FIWARE possiu uma arquitetura modular e centralizada que permite que elementos troquem informações através de um módulo chamado *Context Broker*. Através desse módulo, os componentes de um sistema podem produzir informações e enviá-las ao *Context Broker* e ao mesmo tempo pode-se ter aplicações consumindo essas informações. Existem duas formas de aplicações consumirem informações: a) a primeira é onde uma aplicação faz uma consulta para um determinado elemento e; b) a segunda onde uma

Figura 2.7: Arquitetura genérica do FIWARE



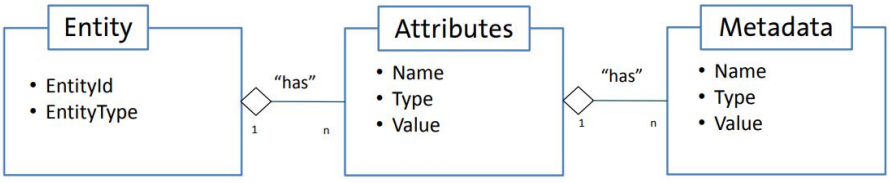
(FIWARE, 2017b)

aplicação está registrada para receber notificações de mudanças em um determinado elemento. Essa funcionalidade é chamada de *Publish/Subscribe* (FIWARE, 2017b). A Figura 2.7 apresenta uma forma genérica e simplificada da arquitetura dessa plataforma.

Existe, ainda, um recurso baseado em componentes chamado *Generic Enabler* que oferece um conjunto de funções compartilhadas que podem ser utilizadas por aplicações de diversos setores. Esses componentes podem ser usados em conjunto conforme a necessidade de cada projeto, sendo acoplados à aplicação.

Um elemento do sistema IoT no FIWARE é chamado de Entidade (*Entity*). Cada Entidade possui um identificador e um tipo, como por exemplo um estudante possui um número de matrícula que o identifica e ele possui uma classificação "Aluno" que significa que essa pessoa pertence à classe de alunos. Cada Entidade ainda pode possuir atributos que são informações referentes a ela e cada atributo pode possuir um conjunto de metadados que fazem essa informação ainda mais rica em termos de conteúdo e significado. A Figura 2.8 apresenta a estrutura de dados básica do FIWARE.

Figura 2.8: Estrutura das entidades e atributos



Fonte: (FIWARE, 2017a)

2.2 Indústria 4.0

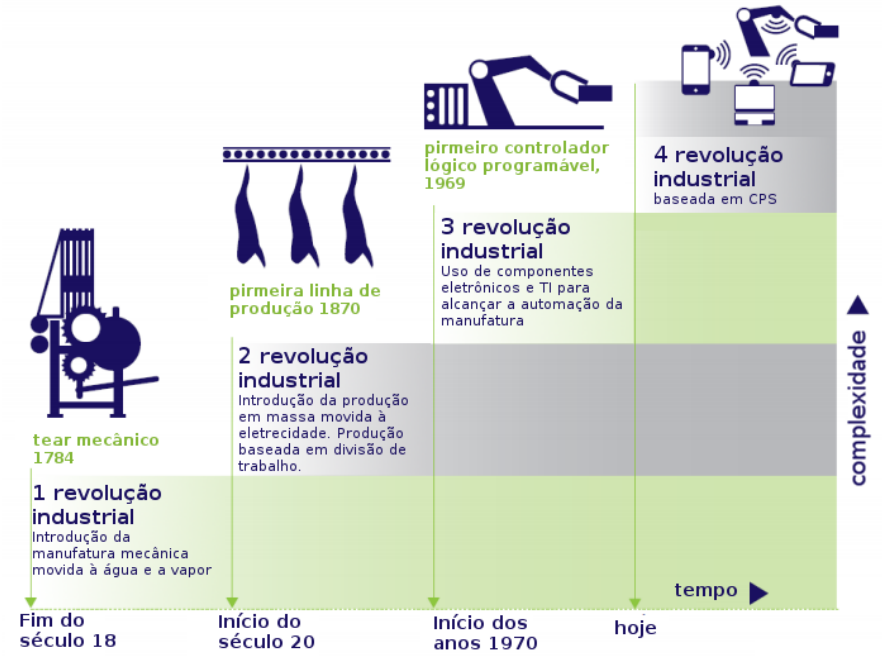
Os avanços tecnológicos trouxeram um aumento da produtividade a cada nova era industrial. Esses avanços marcaram 4 revoluções industriais (Figura 2.9):

- **Primeira revolução industrial (1712-1913)**: começou com o aperfeiçoamento da máquina a vapor por James Watt, colocando a indústria têxtil como símbolo da produção excedente, gerando a riqueza da época, criando um novo modelo econômico.
- **Segunda revolução industrial (1913-1969)**: Henry Ford foi o idealizador desta revolução. Ele criou a linha de produção em massa, reduzindo custos de produção e popularizando o produto.
- **Terceira revolução industrial (1969-2010)**: chamamos de a Era da automação, com a implantação de computadores no chão de fábrica, trazendo controles eletrônicos, sensores e atuadores capazes de otimizar a produção e torná-la mais rápida e com controle de qualidade.
- **Quarta revolução industrial (2012-?)**: inicia com a Era da internet, sendo colocada dentro da indústria e adaptada para as máquinas e equipamentos. Sendo assim devemos pensar que estes equipamentos estão conectados e trocam informações entre si.

A Indústria 4.0 está sendo motivada por três grandes mudanças no mundo industrial produtivo: avanço exponencial da capacidade dos computadores; imensa quantidade de informação digitalizada; e novas estratégias de inovação. Com isso pretende-se tornar a manufatura um processo flexível, decentralizado, com eficiência de recursos, produtos individualizados com períodos curtos de desenvolvimento da produção (LASI et al., 2014).

Portanto, alguns benefícios da Industria 4.0 são: redução de custos, aumento da segurança, personalização, economia de energia, redução de erros, aumento da qualidade, etc. A tecnologia base para toda esta revolução é o uso de IoT e M2M (*Machine to Machine*). Além disso, a Industria 4.0 engloba outros conceitos como *Cyber Physical Systems* e *Smart Factory*.

Figura 2.9: Revolução Industrial



Fonte: Adapatado de (HENNING, 2013)

2.2.1 Cyber Physical Systems

Um sistema ciber-físico (Cyber-Physical Systems ou CPS) é um sistema que combina e coordena elementos computacionais e físicos. O CPS integra a capacidade de atuação do mundo físico e a inteligência do mundo cibernético para adicionar novos recursos aos sistemas físicos do mundo real (LEE; BAGHERI; KAO, 2015). Nesses sistemas, as operações são monitoradas, coordenadas, controladas e integradas por um núcleo de computação e comunicação.

Um CPS agrega recursos de computação, comunicação e armazenamento para monitorar e controlar entidades do mundo físico, de forma confiável, segura, eficiente e em tempo real (SHA et al., 2009). CPS é uma tecnologia emergente e capacitadora para enfrentar os desafios futuros em relação à automação industrial. Neste domínio, CPS compreendem máquinas inteligentes, sistemas de armazenamento e instalações de produção capazes de realizar troca de informações de forma autônoma, desencadeando ações e controle uns dos outros de forma independente. Essa evolução tecnológica é descrita como a quarta revolução industrial (JAZDI, 2014).

2.2.2 Smart Factory

Smart Factory ou Fábrica Inteligente pode ser entendida como a sumarização dos benefícios vindos da integração da Indústria 4.0 e de sistemas ciberfísicos(LEE, 2015). Ela pode transformar a forma de comunicação entre sistemas de engenharia assim como a *internet* mudou a forma com que as pessoas se comunicam (LEE, 2015).

A Fábrica Inteligente é um sistema ciberfísico de manufatura que integra objetos físicos, como máquinas, transportadores e produtos, com sistemas de informação como sistemas de gerenciamento das atividades de produção (ou em inglês, *Manufacturing Execution Systems* - MES) e sistemas de Planejamento dos Recursos da Empresa (ou em inglês, *Enterprise Resource Planning*- ERP) para implementar uma produção flexível e ágil (WANG et al., 2016).

A Fábrica Inteligente é uma característica importante da *Industry 4.0* que aborda a integração vertical e os sistemas de fabricação em rede para a produção inteligente (HENNING, 2013). Para que a Fábrica Inteligente seja implementada, ela deve combinar os objetos inteligentes com análise de grande quantidade de dados. Os objetos inteligentes podem se reconfigurar dinamicamente para alcançar uma alta flexibilidade, enquanto a grande análise de dados pode fornecer *feedback* e coordenação global para alcançar alta eficiência. Portanto, a fábrica inteligente pode produzir produtos personalizados e de pequeno lote com eficiência e lucratividade.

Segundo (SHROUF; ORDIERES; MIRAGLIOTTA, 2014), algumas características das Fábricas Inteligentes podem ser destacadas, como por exemplo:

- Produção em massa: o processo de produção deve atender aos requisitos de produção, incluindo o cliente na especificação do produto permitindo mudanças de última hora.
- Flexibilidade: o processo de produção deve seguir diferentes aspectos como tempo, qualidade, custo e aspectos ecológicos.
- Criação de novos serviços: IoT abre caminhos para criação de novos serviços antes e depois da compra dos produtos por parte dos clientes.
- Monitoramento remoto: a IoT também permite que sistemas externos sejam conectados, como por exemplo, fornecedores podem monitorar equipamentos.
- Manutenção proativa: o monitoramento da produção, coletando dados de performance dos equipamentos, trazem um impacto positivo na previsão de manutenções.

2.3 Ontologias

A palavra ontologia vem da Filosofia onde significa uma explicação sistemática da Existência(GÓMEZ-PÉREZ; BENJAMINS, 1999). Ela pode ser entendida, ainda, como uma especificação formal e explícita de uma conceptualização compartilhada(GRUBER et al., 1993). Ontologias provêm uma forma genérica de representar conhecimento de domínio para um entendimento comum entre aplicações (CHANDRASEKARAN; JOSEPHSON; BENJAMINS, 1999). Assim, através desses modelos é possível criar um vocabulário comum de uma área, com os significados de cada termo e as relações entre eles (GÓMEZ-PÉREZ; BENJAMINS, 1999).

De acordo com (WANG et al., 2004) uma ontologia provê um vocabulário comum para pessoas que necessitam compartilhar informação em um determinado domínio, possibilitando ainda, que os conceitos básicos do domínio e as relações entre eles sejam interpretáveis por máquinas. Ter-se um vocabulário comum é um requisito básico para a comunicação de máquinas com outras máquinas ou com humanos (FIORINI et al., 2017).

Modelos baseados em ontologias podem trazer diversos benefícios para o desenvolvimento de um sistema, como por exemplo(WANG et al., 2004):

- **Compartilhamento de conhecimento**: pode-se usar ontologias para que entidades computacionais, como, por exemplo, máquinas de uma indústria, tenham um conjunto comum de conceitos relacionados a esse domínio.
- **Reuso de conhecimento**: é possível unir ontologias de diferentes domínios para se ter uma maior envolvendo todos os conceitos. Um exemplo é o uso de uma ontologia com conceitos relacionados a unidades de medida, pois esses conceitos podem ser usados em diversos domínios de aplicação.
- **Inferência lógica**: é possível fazer inferências lógicas com base nos dados obtidos de um contexto de baixo nível, isto é, dados fornecidos pelos sensores, pode-se verificar e corrigir inconsistências no contexto devido a falhas na aquisição desses dados.

2.3.1 OWL - *Web Ontology Language*

O uso de ontologias no contexto de Web Semântica requer uma linguagem que se enquadre em alguns requisitos básicos: sintaxe e semântica bem definida, suporte a

reasoning, e conveniência de expressão (ANTONIOU; HARMELEN, 2004).

Existem diferentes linguagens de ontologias, porém a W3C (CONSORTIUM et al., 2001) desenvolveu, a partir de linguagens e padrões existentes, a linguagem OWL (*Web Ontology Language*) (ANTONIOU; HARMELEN, 2004). Os elementos básicos para a construção de ontologias com OWL são classes, indivíduos e propriedades.

Classes em ontologias podem ser relacionadas ao conceito de classe na programação orientada a objetos onde objetos no mundo real podem ser agrupados em grupos ou conjuntos com características similares (LACY, 2005).

Uma propriedade é uma relação entre o indivíduo e um valor. Esse valor pode ser um dado ou outro indivíduo. As propriedades fornecem atributos para os indivíduos, o que é semelhante à programação orientada a objetos onde se tem métodos para acessar os dados de objetos, porém a ontologia fornece uma reusabilidade maior podendo relacionar uma propriedade com múltiplas classes (LACY, 2005).

Os indivíduos representam instâncias das classes descritas na ontologia. Essas instâncias são similares aos objetos na programação orientada a objetos, diferenciando-se por não terem métodos associados. Indivíduos podem representar ambos conceitos virtuais e objetos físicos (LACY, 2005).

2.3.2 Protégé

Protégé (STANFORD, 2013) é uma ferramenta consolidada, de código aberto, que permite a visualização e edição de ontologias. Ela foi desenvolvida na Universidade de Stanford. Essa ferramenta pode ajudar usuários a criar outras aplicações para aquisição de conhecimento por parte de especialistas em determinadas áreas, ou seja, é possível que se desenvolva uma extensão ou uma nova ferramenta, para uso específico, com base no Protégé (MUSEN, 1989).

Outra funcionalidade interessante dessa ferramenta é a possibilidade de realizar consultas SPARQL (PRUD; SEABORNE et al., 2006), que é uma linguagem que fornece meios para expressar padrões a serem consultados em uma ontologia.

Por fim, essa ferramenta ainda permite a criação de *plugins* que são geralmente úteis para aplicações com propósito específico. No desenvolvimento desse trabalho, foi criado um *plugin* capaz de extrair informações de modelos e enviar essas informações para um *middleware* IoT. No decorrer do trabalho, percebeu-se que esse modelo de dados seria melhor explorado se acoplado junto ao *middleware*, assim permitindo que outras

aplicações recebem informações desse modelo que estão relacionadas aos dados do sistema.

3 ANÁLISE DO ESTADO DA ARTE

Este capítulo apresenta trabalhos relacionados que vem de encontro com a proposta deste trabalho, nas áreas de IoT, modelos semânticos e trabalhos sobre a quarta revolução industrial.

3.1 *Middlewares* para Internet of Things

O conceito de IoT está cada vez mais presente em diversas áreas possibilitando a conexão de diferentes objetos e sistemas. Nesse contexto, conectar empresas a consumidores (business-to-customer - B2C) é algo procurado por empresas, uma vez que o modelo que conecta empresas a empresas (business-to-business - B2B) já está em estágio avançado no contexto de soluções IoT no mercado atual (LEMINEN et al., 2012).

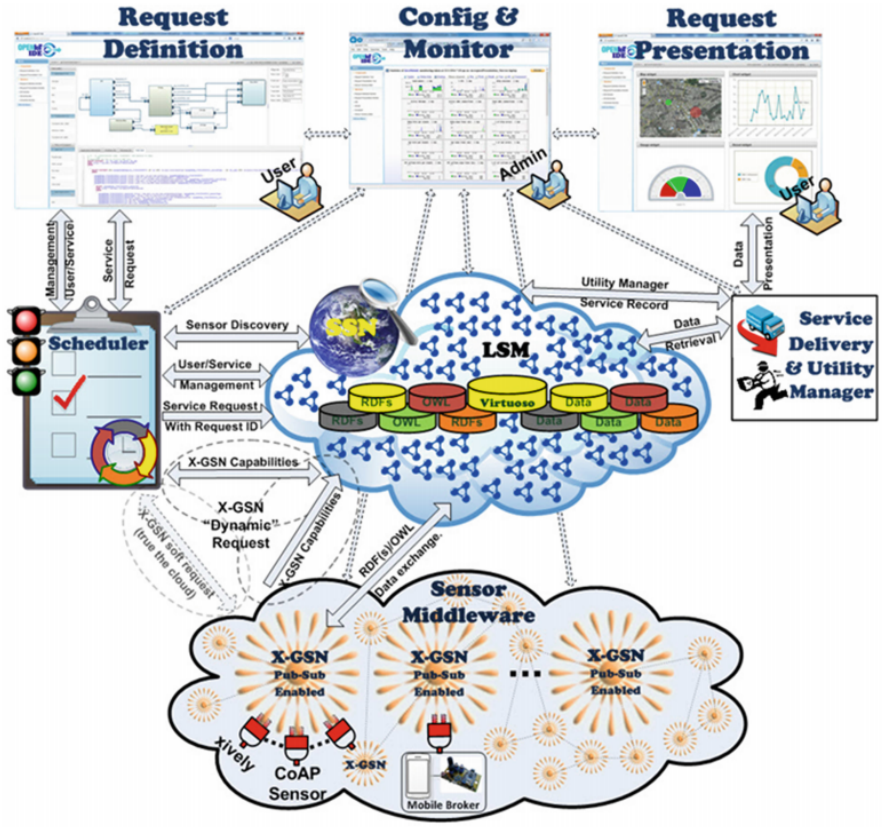
Para que seja possível implementar essa conexão entre diferenes elementos, é necessário ter-se ferramentas que suportem esse conceito inovador. Nesse sentido, o *framework* OpenIoT (SOLDATOS et al., 2015) é uma plataforma aberta que possui sete principais elementos que podem ser encaixados nesse contexto. A Figura 3.1 apresenta esses elementos e suas relações entre si.

O *Sensor Middleware* é responsável por coletar, filtrar e combinar dados vindos de sensores virtuais ou dispositivos físicos. Ele pode ser utilizado com um ou mais instâncias, possibilitando a criação de sub redes de objetos IoT. *Cloud Data Storage* trabalha no armazenamento dos dados recebidos dos sensores. O *Scheduler* processa as requisições e garante que os recursos requisitados estejam acessíveis. O *Service Delivery & Utility Manager* faz a conexão dos dados com o sistema OpenIoT a fim de disponibilizar os serviços requisitados pelos clientes do sistema. Os últimos três elementos são Request *Definition*, que permite a especificação de requisições em tempo de execução, *Request Presentation*, que fornece uma visualização das respostas geradas pelos serviços, e o *Configuration and Monitoring*, que fornece uma forma de gerenciamento e configuração das funcionalidades do sistema.

OpenIoT foi construído para aplicações de cidades inteligentes (*Smart Cities*) e já foi aplicado em diversos projetos desse domínio, o que mostra que é um projeto com um grau de amadurecimento maior que a maioria das plataformas encontradas na literatura.

Ainda nesse contexto de IoT e a possibilidade de diferentes tipos de conexão, o trabalho apresentado por (KIM; LEE, 2014), também busca prover uma forma de co-

Figura 3.1: Visão geral do OpenIoT e seus componentes principais



Fonte: (SOLDATOS et al., 2015)

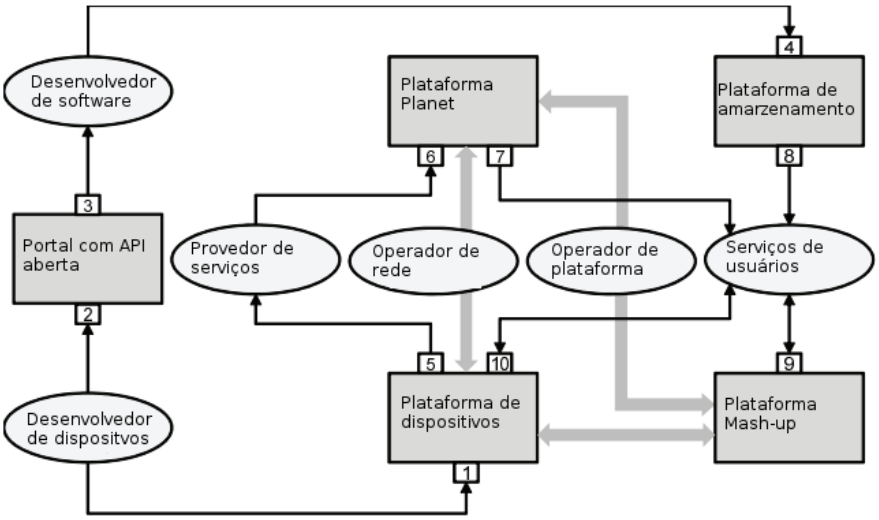
nectar esses mais variados domínios, uma vez que ele fornece um *link* entre empresas e consumidores (B2C), consumidores e consumidores (C2C), empresas e empresas e ainda empresas e entidades governamentais (B2G).

A arquitetura desse trabalho consiste basicamente em três plataformas que constituem a parte do servidor e uma plataforma que compõe o lado dos dispositivos. Todas as plataformas possuem APIs abertas que possibilitam a comunicação entre si. A Figura 3.2 apresenta a arquitetura desse trabalho.

As plataformas encontradas na arquitetura proposta são: a) *Device Platform* é um software embarcado que facilita a conexão das “coisas"; b) *Planet Platform* está localizada no servidor e é responsável por registrar, gerenciar e monitorar dispositivos; c) *Mashup Platform* é capaz de fornecer novos serviços integrados baseados em dados coletados; d) *Store Platform* contém aplicativos de armazenamento e que permitem serviços aos usuários.

Essa arquitetura é interessante pois aborda a possibilidade de usuários consumidores interagirem com sistemas. Isso possibilita que esses usuários participem do processo de produção de seus produtos, o que é uma das características das Fábricas Inteligentes,

Figura 3.2: Serviços e plataformas propostas para o *framework* IoT



Fonte: Adaptado de (KIM; LEE, 2014)

já abordada no capítulo anterior.

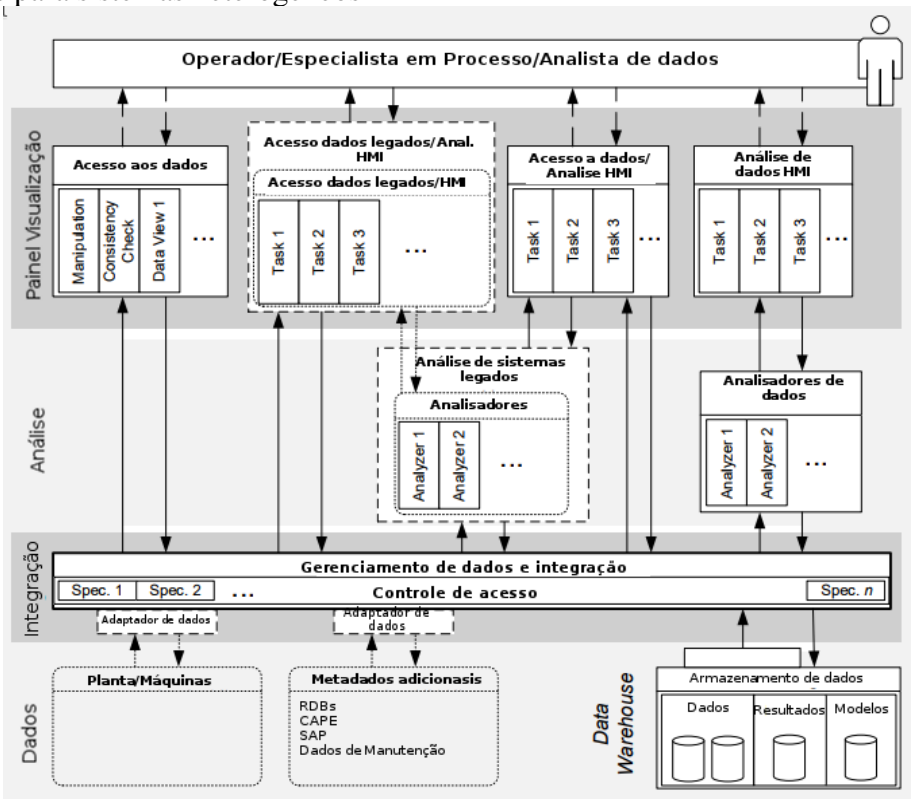
Visando à integração de dados no domínio de sistemas de automação, o trabalho de (TRUNZER et al., 2017) apresenta uma arquitetura flexível que engloba desde as fontes de dados até a análise desses dados e visualização pelo usuário. Essa proposta busca facilitar a integração, o recebimento e o compartilhamento de dados entre diferentes organizações.

A Figura 3.3 apresenta a arquitetura proposta pelo trabalho, onde pode-se observar que a integração está baseada em um componente chamado Gerenciamento de dados e Integração. Esse componente é responsável por conectar as camadas da arquitetura o que vem de encontro à proposta do presente trabalho. Entretanto, ainda existe uma lacuna no sentido de se ter um modelo dos dados que circulam no sistema. Isso se faz importante para garantir-se uma interoperabilidade e um entendimento único de certa informação entre todos os envolvidos no sistema.

Já o trabalho de (UNGUREAN; GAITAN; GAITAN, 2014) traz uma arquitetura que pode ser utilizada tanto no contexto industrial quanto para a automação residencial. Em termos tecnológicos, o trabalho utiliza o conceito de *middleware*, através da união de um sistema SCADA (BOYER, 2009) juntamente com o conceito de IoT. A comunicação é baseada no padrão OPC.NET, o que pode trazer muitas limitações para o uso dessa proposta. Adicionalmente, o trabalho não utiliza modelos, o que pode prejudicar o gerenciamento e a compreensão do sistema quando este começar a tomar proporções maiores.

Nesse contexto, ainda pode ser citado o trabalho de (EISENHAUER; ROSEN-

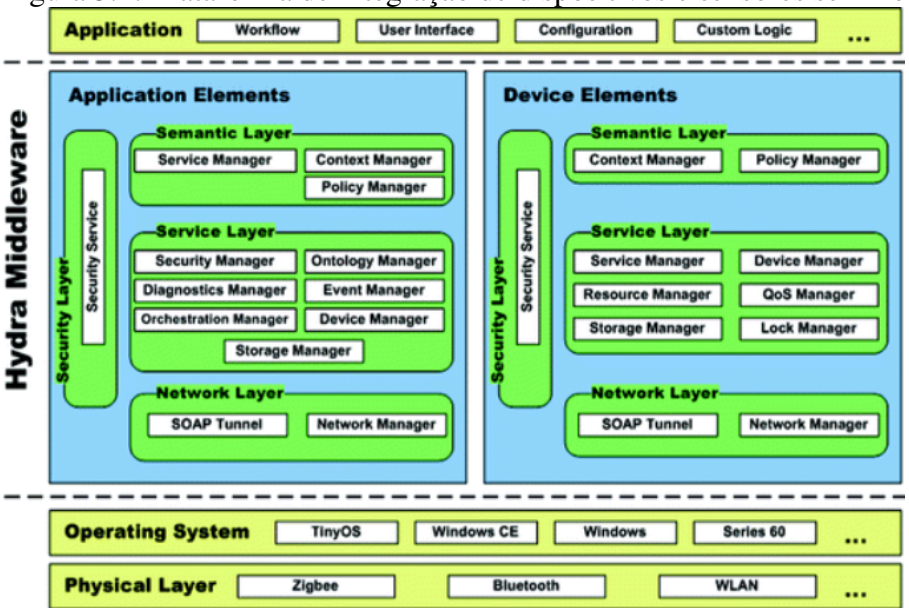
Figura 3.3: Arquitetura genérica para aquisição, agregação, integração e armazenamento de dados para sistemas heterogêneos



Fonte: Adaptado de (TRUNZER et al., 2017)

GREN; ANTOLIN, 2010) que é baseado na combinação de uma arquitetura baseada em serviços (SoA) com uma baseada em modelos (MDA) o que possibilita a criação de serviços genéricos para dispositivos heterogêneos. A Figura 3.4 apresenta um visão geral da plataforma de integração proposta no trabalho.

Figura 3.4: Plataforma de integração de dispositivos e sensores sem fio



Fonte: (EISENHAUER; ROSENGREN; ANTOLIN, 2010)

O *middleware* está localizado entre o sistema operacional (que faz a comunicação com os dispositivos físicos) e a camada de aplicação (que faz interface com aplicações externas e usuários). Essa plataforma contém um grande número de componentes de software rodando para atender as demandas necessárias para prover a comunicação entre os dispositivos.

A plataforma Hydra é dividida em duas partes parecidas, uma destinada a elementos de aplicação e outra para os dispositivos. Ambas partes possuem quatro camadas: a) de comunicação, baseado no padrão SOA; b) de segurança; c) serviços, que contém a maior parte da implementação do *middleware*, pois trata dos dispositivos, eventos, armazenamento dos dados, diagnósticos e ainda manipula a última camada, que é a d) camada semântica, onde estão contidas informações sobre serviços, regras de negócio e informações relacionadas ao gerenciamento de contexto.

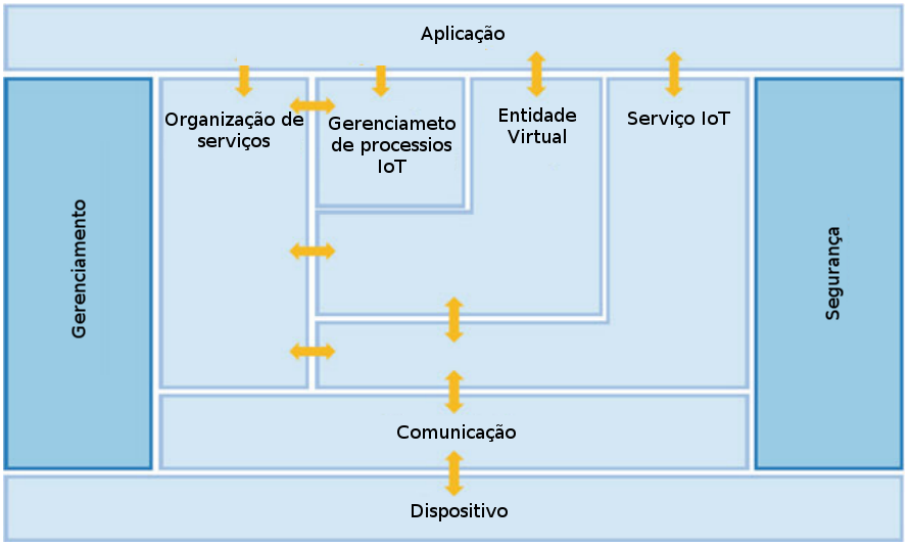
O principal foco desse trabalho é nas áreas de agricultura, saúde e automação residencial, entretanto traz abordagens como o uso de uma camada semântica, que pode ser utilizada em contextos diferentes, como por exemplo, na automação industrial.

Os trabalhos de (SOTIRIADIS; STRAVOSKOUFOS; PETRAKIS, 2017) e de (LÓPEZ-RIQUELME et al., 2017) utilizam o *middleware* FIWARE no contexto de agricultura de precisão. Os autores apresentam estudos de caso reais onde fazem sensoriamento de variáveis em uma lavoura. Essas informações são adquiridas e enviadas por um equipamento de *hardware* desenvolvido no âmbito dos trabalhos. Esses trabalhos são importantes, pois mostram que o conceito de IoT pode ser aplicado em áreas totalmente diferentes e que a ferramenta utilizada também atende a diversos domínios.

Para o desenvolvimento da proposta, os autores basearam-se na arquitetura IoT-A, como pode ser observado na Figura 3.5.

A plataforma foi implementada com módulos independentes e em conjunto com o *middleware* FIWARE objetivando obter-se uma solução que suporte conexões de larga escala. O módulo de gerenciamento de processos (*Management*) provê um mecanismo para conectar os dispositivos IoT com processos de negócio empresariais. O módulo de organização (*Organization*) é responsável por organizar e compor os serviços do sistema. Através desse módulo é possível que aplicações e dispositivos se comuniquem. Ainda, existe o módulo de segurança que é responsável por garantir acesso ao sistema apenas para os autorizados.

Figura 3.5: Módulo funcional da arquitetura IoT-A



Fonte: Adaptado de (SOTIRIADIS; STRAVOSKOUFOS; PETRAKIS, 2017)

3.2 Uso de semântica em projetos IoT

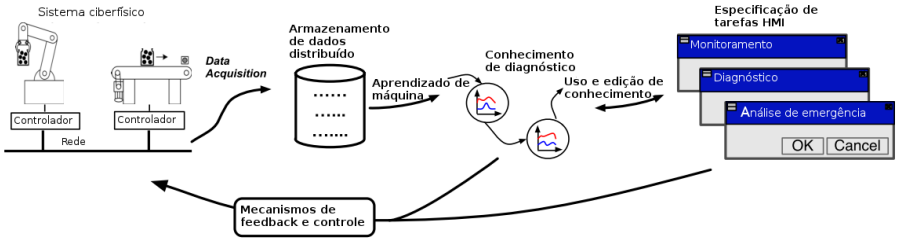
Em (KATASONOV et al., 2008) é proposto um *middleware* para IoT que tem como objetivo permitir o autogerenciamento de sistemas complexos, em especial, sistemas industriais, distribuídos, heterogêneos, compartilhados e que tenham componentes reusáveis de diferentes naturezas (sensores, atuadores, máquinas, aplicações, humanos...).

Foi utilizado o conceito de agentes para obter-se uma comunicação flexível e uma coordenação dos componentes, além de possibilitar o uso de mecanismos de descoberta de serviços, a comunicação baseada no protocolo FIPA(BELLIFEMINE; POGGI; RIMASSA, 1999) e, ainda, a negociação baseada na composição de serviços. O uso de agentes para representar os dispositivos do sistema IoT traz benefícios, visto que existem tecnologias que implementam esse conceito e já trazem toda a infraestrutura de comunicação. Em termos de segurança, o autor ressalta que ainda há uma necessidade de maiores testes relacionados a tecnologias baseadas em agentes. Para desenvolver-se uma integração com sistemas de terceiros, pode-se precisar de um esforço a mais, sendo que será necessário utilizar o protocolo FIPA e o conceito de agentes.

Esse trabalho sugere que todos esses componentes sejam tratados de forma semelhante e que cada um possua funcionalidades que possam ser mapeadas em modelos semânticos. Ontologias são usadas, nesse trabalho, não só para facilitar a descoberta de novos componentes mas também para descrever os comportamentos e a coordenação desses elementos.

Existem também abordagens de modelagem baseadas no padrão *Unified Modeling*

Figura 3.6: Ilustração dos desafios encontrados para uma solução baseada em modelos de dados.



Fonte: Adaptado de (NIGGEMANN et al., 2015)

Language (UML) que permitem representar sistemas de automação em uma notação de alto nível. Por exemplo, o trabalho de (THRAMBOULIDIS; CHRISTOULAKIS, 2016) apresenta a UML4IoT, que é um padrão de modelagem baseado na UML, porém voltado para a IoT. O autor destaca a necessidade de se ter formas de modelar esse tipo de sistema visto o grande crescimento na implantação desse conceito em projetos de diversas áreas.

Em um sistema de CPS, através da UML4IoT é possível mapear dispositivos e objetos físicos e gerar interfaces para a camada cibernética. Essa característica vem de encontro a uma das contribuições desse trabalho, onde através de modelos se obtém interfaces para integração dos dispositivos físicos. Entretanto, o presente trabalho explora o uso de ontologias para essa modelagem, permitindo que se faça consultas no modelo semântico que estará atualizado com os dispositivos físicos.

O trabalho de (NIGGEMANN et al., 2015) apresenta uma proposta de análise de um monitoramento de CPS orientado a dados. São usados os conceitos de *Big Data* e IoT para prover diagnósticos e controle para a planta industrial. É proposto ainda que o usuário possa auxiliar na adequação do conhecimento gerado pela análise dos dados para então fornecer um *feedback* mais preciso à planta.

Para aplicar-se uma solução baseada em modelos de dados para o monitoramento, diagnóstico e controle dos CPS existem alguns desafios que devem ser superados. A Figura 3.6 ilustra os principais desafios destacados por (NIGGEMANN et al., 2015).

Na aquisição de dados pode-se destacar o problema de que existem sensores e atuadores que trabalham em diferentes taxas e isso deve ser tratado. Ainda, existe a necessidade de modelar-se os dados para uma futura análise de dados. Os dados precisam ser armazenados para que estejam disponíveis posteriormente às aplicações, entretanto é necessário criar-se uma infraestrutura para armazenar esses dados e isso reforça a ideia de se ter um modelo onde é possível mapear características de cada dado ou informação coletada.

Após ter-se esses dados salvos, é possível fazer-se uma análise para obtenção de informações mais precisas em relação à planta industrial. Nessa etapa existem diversos algoritmos que podem ser aplicados para se obter informações. Após esse processamento, as informações são mostradas ao usuário/especialistas para que esse avalie as informações obtidas e ainda acrescente seu conhecimento. Nessa etapa é muito importante que exista uma linguagem comum, sem ambiguidades entre humanos e as máquinas. Portanto, os modelos semânticos também podem ser utilizados nessa fase de análise. Ao final, o trabalho propõe uma forma de *feedback* à planta para que essa opere normalmente.

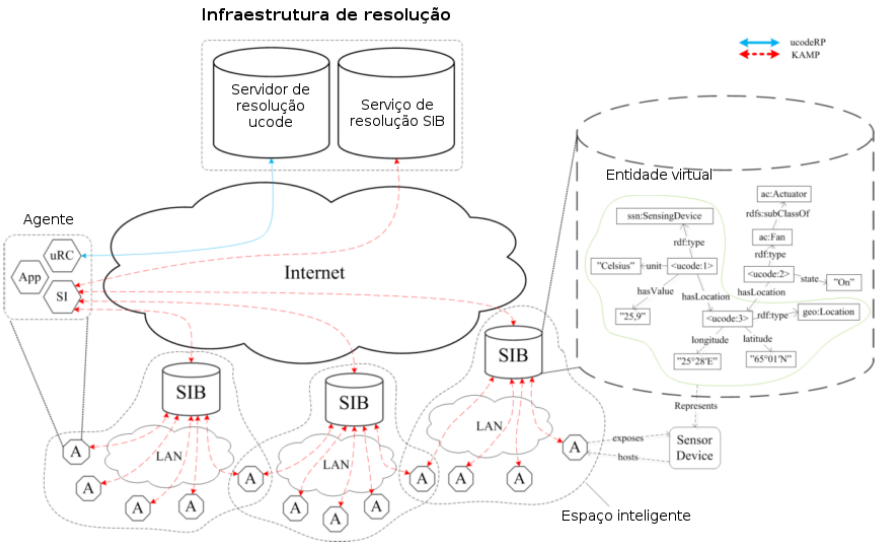
O trabalho de (ALAM; CHOWDHURY; NOLL, 2010) introduz um *framework* de virtualização, com uma arquitetura orientada a serviços, para suportar eventos de sensores conectados. Através dessa ferramenta é possível realizar *reasoning* com a camada semântica que existe sobre o *framework*. Os autores mostram, através de um estudo de caso, que com o uso de SoA(NEWCOMER; LOMOW, 2005), modelos semânticos e virtualização é possível garantir conectividade, segurança e monitoramento, respectivamente.

Uma arquitetura para integração em nível semântico de dispositivos heterogêneos para computação pervasiva e IoT é proposta em (KILJANDER et al., 2014). É proposto um *Broker* que faz a intermediação da comunicação dos dispositivos e os modelos semânticos. A arquitetura proposta foi construída conforme o modelo de referência IoT-A, onde são aproveitados elementos principais desse modelo de referência.

A Figura 3.7 descreve a arquitetura divida em quatro entidades principais: a) *Semantic Information Broker* (SIB) fornece um meio de comunicação semântica entre os agentes possibilitando a comunicação entre eles; b) *Agents* monitoram o ambiente e compartilham informações através do SIB; c) *Virtual Entity* pode representar qualquer elemento do mundo real, como sensores, atuadores, produtos, pessoas. Ou seja, é uma representação virtual de uma entidade física; d) *Resolution Infrastructure* tem a função de fornecer informações aos agentes relacionadas à infraestrutura do sistema, como por exemplo, encontrar um SIB do interesse de um agente.

Os trabalhos de (KILJANDER et al., 2014) vem de encontro à presente proposta no sentido de usar modelos semânticos para realizar a integração de dispositivos, além de fornecer uma interface para acesso ao modelo semântico por parte dos dispositivos. Entretanto, esse trabalho não aborda o uso de modelos para usuários finais, o que pode trazer um enriquecimento no entendimento das informações do sistema e, consequentemente, uma melhor tomada de decisões.

Figura 3.7: Visão geral da arquitetura em nível semântico



Fonte: Adaptado de (KILJANDER et al., 2014)

3.3 Indústria 4.0

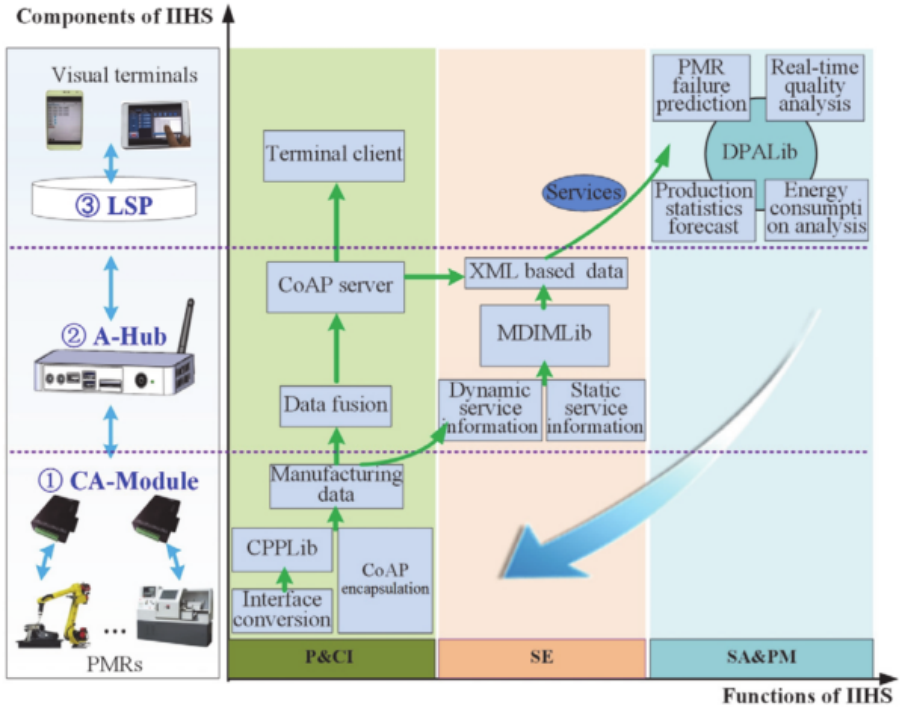
Dentro do contexto da Indústria 4.0 existem pesquisas na área de modelagem de dados visando à integração de elementos desse domínio. Isso porque essa nova revolução industrial une elementos físicos e cibernéticos, garantindo um ambiente de manufatura ideal (STRANG; ANDERL, 2014). Nesse sentido, existe uma necessidade de se ter um meio comum de representar os dados e os elementos do sistema, desde a montagem e produção de produtos (STRANG; ANDERL, 2014) até os processos de alto nível como modelo de negócios e estratégias da empresa (PETRASCH; HENTSCHKE, 2016).

Considerando os desafios de se ter uma conexão inteligente no contexto industrial, o trabalho de (TAO; CHENG; QI, 2017) propõe uma ferramenta para esse propósito chamada IIHub (*Industrial Internet of Things Hub*). IIHub é divido em três módulos: a) módulo customizado de acesso que é onde os dispositivos físicos estão localizados; b) módulo *hub*, onde está a infraestrutura de integração e; c) módulo de serviços locais que contém os terminais de visualização dos dados. Em cada um dos módulos existem funções relacionadas que são necessárias para a integração, como pode ser observado na Figura 3.8.

Este trabalho possibilita a integração de dispositivos com diferentes interfaces de comunicação, bibliotecas para facilitar o desenvolvimento de aplicações que possam ser integradas a essa ferramenta além de a possibilidade de mapear serviços de manufatura através de uma linguagem de descrição.

O trabalho de (JIRKOVSKÝ; OBITKO, 2017) introduz um modelo de dados hí-

Figura 3.8: Arquitetura e visão das funções do IIHub



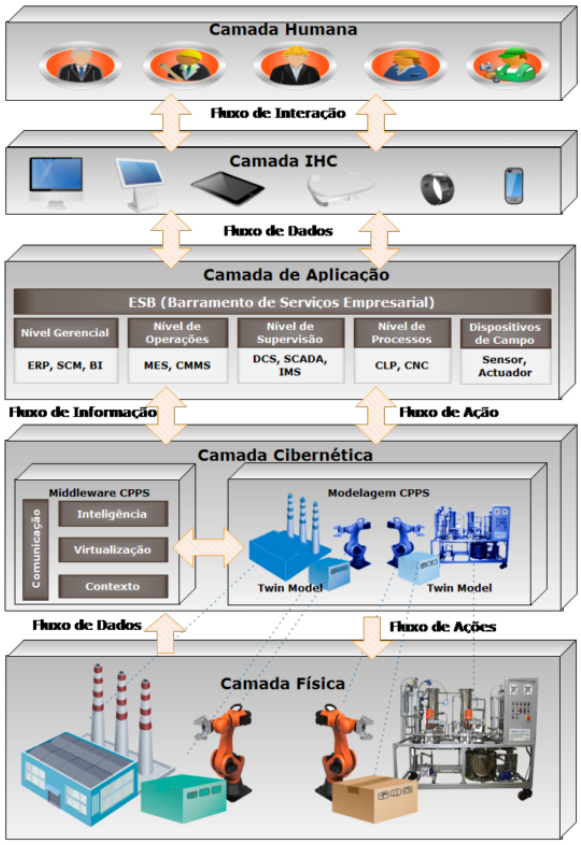
(TAO; CHENG; QI, 2017)

brido voltado para o armazenamento de dados de sensores. É proposta um arquitetura de quatro camadas que engloba desde a aquisição de dados até o armazenamento. A primeira camada é a de aquisição de dados, que é composta por sensores, sistemas e fontes de dados externas, como por exemplo informações de tráfego ou sobre o clima. A segunda camada é a de transformação, que é responsável por transformar os dados recebidos e relacioná-los ao modelo semântico. A heterogeneidade é tratada nessa camada. A terceira camada é a de armazenamento, onde os dados são salvos e por fim, a quarta camada é a de análise de dados, onde é fornecida uma interface para ferramentes de análise.

O trabalho de (RODRIGUES, 2016) propõe um *middleware* para integração de sistemas industriais baseados no conceito de Sistemas Ciberfísicos de Produção (CPPS). O objetivo é prover uma forma de integração de informações de diferentes níveis de gerenciamento industrial através de uma interface comum. O trabalho apresente uma arquitetura de cinco camadas: a) camada humana; b) camada IHC; c) camada de aplicação; d) camada cibernética; e) camada física.

A proposta apresenta as camadas e suas interações entre si, como os fluxos de dados, de informação, interação com o usuário e ações. A Figura 3.9 apresenta a arquitetura proposta pelo trabalho. Também foi desenvolvido um *middleware* CPPS que foi aplicado em diferentes elementos em uma planta industrial, envolvendo diversos tipos de sensores, atuadores e um sistema SCADA. Essa arquitetura traz uma abordagem que vem de encon-

Figura 3.9: Arquitetura CPPS



Fonte: (RODRIGUES, 2016)

tro à proposta do presente trabalho, no sentido de usar camadas parecidas para projetar o sistema.

3.4 Discussão

Neste capítulo foram analisados trabalhos relacionados visando encontrar contribuições que possam ser adotadas e, principalmente, lacunas para possíveis contribuições.

A Tabela 6.1 apresenta os uma comparação entre os principais trabalhos relacionados sob a perspectiva de três principais pontos: Formalismo semântico, Integração e Automatização.

O Formalismo semântico indica se o trabalho usou algum padrão para representar os dados, como por exemplo, UML, ontologias, $AutomationML$, $SysML$ ou outro. A coluna Integração identifica o tipo de estratégia de integração utilizada, como por exemplo, *middlewares*, arquitetura distribuída ou outra. Por fim, a Automatização indica se existe alguma estratégia que faça essa integração mais automatizada, como geração de código a partir de modelos ou disponibilização de bibliotecas para o desenvolvimento.

Pode-se perceber que já existem diversos trabalhos que buscam contribuir no domínio de integração de sistemas, inclusive utilizando conceitos que a presente proposta aborda. Entretanto, observa-se que é uma área que necessita de pesquisa, pois apesar de existirem diversas abordagens, não existe uma que seja absoluta, uma vez que trata-se de um problema novo e em constante evolução. As abordagens encontradas apresentam soluções com o uso de *middlewares* e modelos semânticos, assim como a presente proposta, porém existem poucas soluções de como essa integração pode ser automatizada e ainda como esses modelos podem ser usados desde a representação de objetos físicos até os aplicativos que fazem interface para os usuários finais. Nesse sentido, o presente trabalho busca contribuir exatamente em apresentar uma abordagem de integração no contexto da Indústria 4.0 utilizando modelo semânticos e conceito de IoT, bem como prover uma forma automática de geração de interfaces para que essa integração possa ocorrer.

Tabela 3.1: Comparativo dos principais trabalhos

Trabalho	*Descrição*	*Formalismo semântico*	*Integração*	*Automatização*
Kim and Lee (2014)	*Framework* IoT para múltiplos propósitos. Voltado para conectar empresas, produtos e consumidores.	Não	*middleware*	Não
Trunzer et al. (2017)	Integração de dados no domínio de sistemas de automação, desde as fontes de dados, englobando a análise e processamento, até a visualização pelo usuário.	UML	*middleware*	Não
Katasonov et al. (2008)	Plataforma IoT para integração de dispositvos wireless e sensores em sistemas industriais.	Ontologias	*middleware*	parcial - não para novos dispositivos.
Sotiriadis, Stravoskoufos and Petrakis (2017)	Trabalho com FIWARE voltado para agricultura de precisão destacando a opção de se trabalhar com *Generic Enablers* dessa plataforma IoT.	Não	*middleware*	Não
Niggemann et al. (2015)	Esse trabalho mostra a importância de modelos de dados através de uma proposta de análise de um monitoramento de CPS orientado a dados. Os conceitos de *Big Data* e IoT são aplicados.	Ontologias	*middleware*	Não
Kiljander et al. (2014)	Uma arquitetura para integração em nível semântico de dispositivos heterogêneos para computação pervasiva e IoT	Ontologias	*broker* distribuído	Não
Tao, Cheng and Qi (2017)	Ferramenta chamada IIHub para prover uma conexão inteligente no contexto da Indústria 4.0.	Ontologias	*middleware*	Não - apenas bibliotecas
Rodrigues (2016)	*Middleware* para integração de CPPS. Trabalho está inserido no projeto $I2MS2C$.	Ontologias	*middleware*	Não
Prensente proposta	Abordagem para integrar elementos industriais utilizando ontologias e *middleware* IoT. Apresenta uma ontologia para esse fim e uma extensão de um *middleware* para trabalhar com esse modelo de forma automática.	Ontologias	*middleware*	Sim

Fonte: O Autor

4 DIFERENTES ETAPAS DO CICLO DE VIDA DE UM PRODUTO EM SISTEMAS DE AUTOMAÇÃO

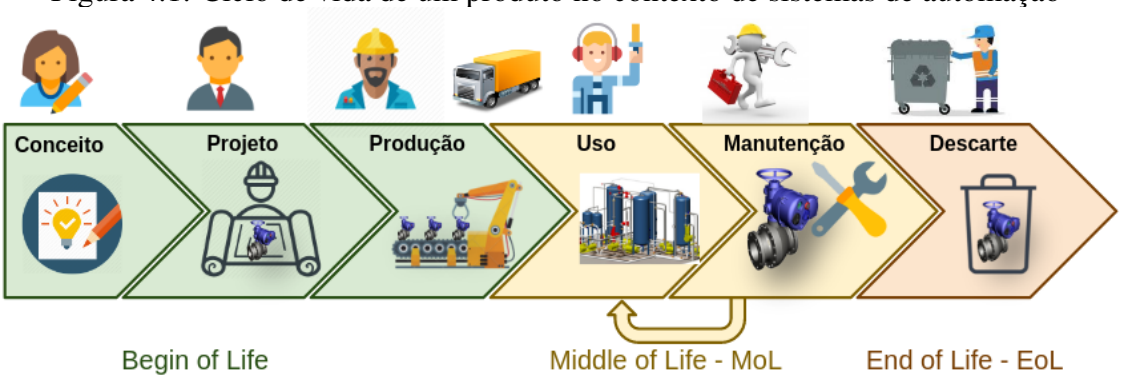
Este capítulo objetiva contextualizar a abordagem de pesquisa no ciclo de vida de um produto e, também, relacionar a proposta de pesquisa com a integração de elementos industriais no contexto dos Sistemas de Automação. Um *workflow* é detalhado com objetivo de descrever um possível cenário de aplicação da pesquisa desenvolvida.

4.1 Ciclo de vida do produto

O gerenciamento do ciclo de vida do produto (*Product Lifecycle Management* - PLM) visa rastrear produto em seu ciclo de vida e gerenciar o conjunto de atividades relacionadas a sua produção e desenvolvimento. Tais atividades contemplam os estágios iniciais (desde sua concepção), passando pela sua produção até a entrega ou descarte do mesmo (STARK, 2015).

Conforme destacado em (STARK, 2015), o ciclo de vida de um produto é composto, inicialmente, pelas fases de **conceito**, **projeto** e **produção** (*Begin of Life*). São consideradas, ainda, como fases pertencentes ao ciclo de vida do produto as fases de **uso** e **manutenção** (*Middle of Life*) e, também, a **fase de descarte** (*End of Life*). A Figura 4.1 ilustra as etapas do ciclo de vida do produto.

Figura 4.1: Ciclo de vida de um produto no contexto de sistemas de automação



Fonte: Adaptado de (STARK, 2015)

Segundo (STARK, 2015), em cada fase do ciclo de vida, o produto está em um estado diferente. Durante a fase de conceito, o produto é apenas uma ideia. Na fase de projeto, essas ideias são convertidas em uma descrição detalhada do produto. Na produção, o produto existe em sua forma final (por exemplo, uma válvula industrial) que

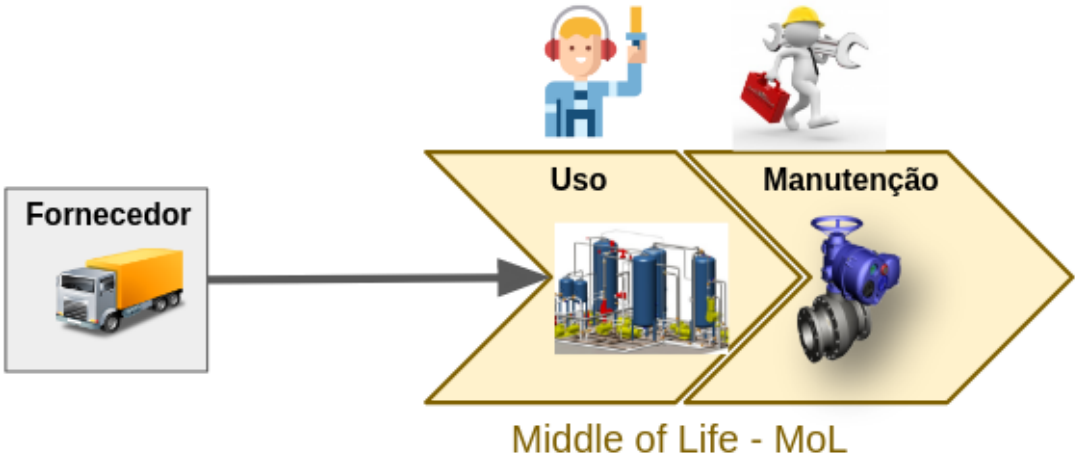
já pode ser utilizada pelos usuários finais. Na fase de uso, o produto já foi entregue ao consumidor que possivelmente estará utilizando-o.

Durante a utilização do produto, há a necessidade de manutenções corretivas, adaptativas ou evolutivas uma vez que o produto pode apresentar problemas durante a sua utilização ou necessitar de aprimoramentos. No contexto de manutenção, a saúde ou a condição em que o dispositivo encontra-se pode ser monitorada com os Sistemas de Manutenção Inteligentes (LEE et al., 2009).

Existem sistemas PLM que integram atividades técnicas, gerenciais e organizacionais executadas por diferentes atores no ciclo de vida do produto (LEE et al., 2009). Esse monitoramento é feito através do sensoriamento, análise e processamento de sinais vindos de diferentes dispositivos em uma cadeia de produção (ESPÍNDOLA, 2011).

O cenário de empregabilidade e adoção desta pesquisa relaciona-se com o estágio de uso e manutenção dos produtos (MoL), bem como com a relação da manutenção de um produto no contexto dos sistemas relacionados a cadeias de suprimentos. A Figura 4.2 mostra os componentes e o fluxo deste cenário e a sua contextualização, perante a proposta de pesquisa, é detalhada na próxima seção.

Figura 4.2: Etapa MoL do produto relacionada com cadeias de suprimentos



Fonte: O Autor

4.2 Cenário da Proposta

Durante o uso de um equipamento, inúmeros sinais são lidos e mapeados para serem processados e analisados posteriormente. Pode-se citar como exemplo as válvulas industriais que, neste caso, funcionam juntamente com um atuador elétrico. Este atuador é responsável por controlar o fluxo de fluido (ex: água, petróleo, etc) que passa pela

válvula. A abertura e o fechamento dos dutos são feitos através do acionamento por motores elétricos que movimentam uma haste.

Devido à natureza do equipamento, existe uma predisposição à degradação desse dispositivo, uma vez que desgastes podem ocorrer devido a condições de uso e operação ou desgastes ao longo do tempo. Conforme o nível de degradação, pode-se realizar predições para uma possível falha e mensurar estratégias de contingência para as mesmas. Por exemplo, através dos valores de torque a cada fechamento ou abertura de uma válvula, pode-se realizar processamentos e análises desses dados através de uma ferramenta como o *Watchdog Agent* (DJURDJANOVIC; LEE; NI, 2003) para identificar possíveis falhas (LEE, 2003).

Com isso, é possível, por exemplo, prever quando uma válvula, similar à descrita acima, apresentará problemas no seu funcionamento e não será capaz de desempenhar as suas funções. Quando não monitorada ou prevista, essa falha poderá trazer diversos problemas para a indústria. Por exemplo, caso ela quebre inesperadamente a planta industrial pode ficar parada até a equipe da manutenção vir e reparar o equipamento.

O planejamento da manutenção é importante, pois tem-se um prejuízo, enquanto a planta está parada, que está relacionado ao tempo gasto para realizar a manutenção desse equipamento, ou seja, quanto maior o tempo de manutenção, maiores serão os custos. De maneira geral, as análises e melhorias do processo de manutenção destes equipamentos contribuem para a contínua operação destes dispositivos.

Nesse cenário existem pelo menos dois tipos de usuários. Na fase do uso da válvula industrial, um operador que trabalha com esse equipamento e é responsável por toda sua operação. Na fase de manutenção, existe um outro ator que é o técnico de manutenção. Isso traz uma característica diferenciada para esse tipo de sistema, visto que podem existir variados tipos de usuários e estes podem precisar de informações diferentes relacionadas ao equipamento. Por exemplo, um operador poderá necessitar das informações sobre o estado atual do atuador. Em contrapartida, um técnico de manutenção precisará saber todo o histórico daquele equipamento para poder entender melhor as causas de um possível problema.

O uso de ontologias pode ser adequado para esse tipo de aplicação, visto que pode-se criar modelos semânticos para serem usados como formalismos de representação de conhecimento. A adoção de ontologias neste contexto torna possível a criação de um meio de comunicação entre máquinas e humanos através desses modelos. Ainda, o uso de ontologias pode permitir a modelagem diferenciada de uma informação para ser

apresentada para diferentes tipos de usuários, por exemplo, na fase MoL do ciclo de vida existem, pelo menos, dois tipos de usuários: um operador e um técnico de manutenção.

O cenário proposto considera a integração de IMS e SPSC como proposto no projeto I2MS2C. Esse projeto busca unir pesquisas de IMS e SPSC com intuito de melhorar a eficiência e efetividade de sistemas de cadeia de suprimentos de peças de reposição. Isso é alcançado através da integração de dados de sistemas IMS com sistemas de reposição. Sendo assim, o planejamento da manutenção é melhorado com base no aprimoramento da disponibilidade e capacidade dos sistemas de reposição.

Para essa integração, o conceito de IoT traz diferentes benefícios, visto que é um conceito que visa prover um meio de comunicação entre diferentes elementos. A presente proposta é aplicada nesse cenário da etapa MoL do ciclo de vida do produto e visa criar uma forma de integrar os componentes industriais através do conceito de IoT juntamente com modelos semânticos. A proposta é descrita em detalhes no próximo capítulo.

5 PROPOSTA DE INTEGRAÇÃO USANDO SEMÂNTICA E IOT NA INDÚSTRIA 4.0

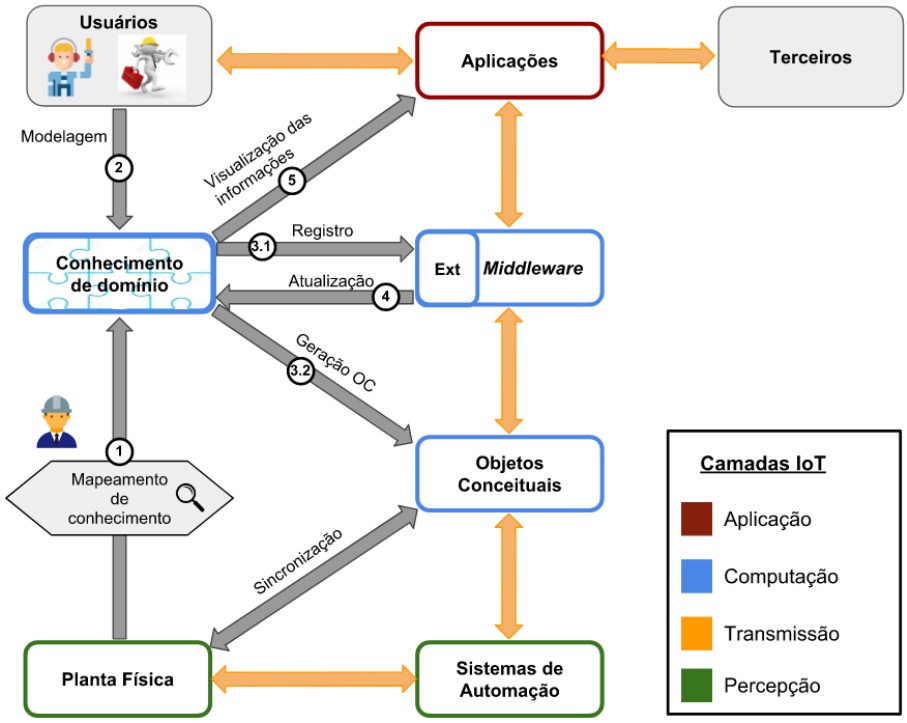
Este capítulo objetiva descrever a abordagem de pesquisa relacionada à integração de elementos industriais utilizando modelos semânticos e o conceito de IoT baseado na Indústria 4.0.

5.1 Visão geral do trabalho

Neste trabalho é apresentada uma abordagem para integração de sistemas industriais utilizando o conceito de Internet das Coisas e de modelos semânticos no contexto da Indústria 4.0.

A Figura 5.1 apresenta a proposta deste trabalho juntamente com os principais elementos envolvidos na integração de objetos no contexto da Indústria 4.0. Pode-se observar que essa proposta está baseada em uma arquitetura IoT de quatro camadas englobando todos os agentes envolvidos no sistema.

Figura 5.1: Proposta de integração com elementos classificados por cor nas camadas da arquitetura IoT



Fonte: O Autor

A arquitetura apresenta os principais elementos envolvidos na proposta deste tra-

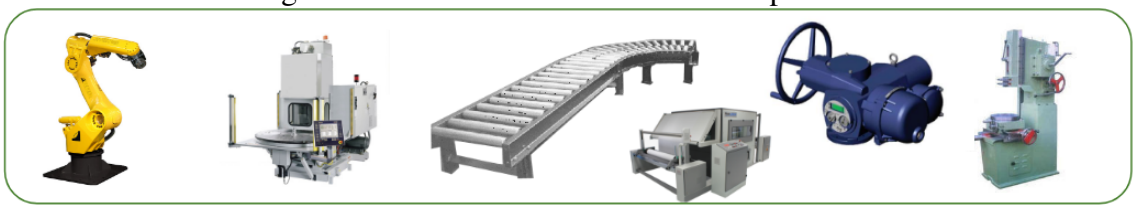
balho.

A proposta pode ser dividida em duas fases principais: a primeira é a **fase da modelagem**, onde os componentes do sistema são modelados em alto nível e as interfaces de comunicação são criadas; a segunda fase é a **fase de execução do sistema** propriamente dita, ou seja, após ter-se as interfaces criadas o sistema poderá executar o seu comportamento esperado em conjunto com a integração dos dispositivos.

5.2 Fase de modelagem

A fase de modelagem baseia-se na planta industrial, que é composta pelos dispositivos físicos do sistema, como máquinas, equipamentos, materiais e produtos. Os sensores são responsáveis por perceber o ambiente e as suas variáveis fornecendo, assim, os dados para o sistema de automação. Além disso, os atuadores são responsáveis por receber comandos e executá-los, alterando os processos físicos. A Figura 5.2 mostra alguns exemplos de componentes físicos encontrados em uma planta industrial. Esses componentes possuem diferentes capacidades e podem executar serviços que geralmente são controlados por sistemas responsáveis por automatizarem processos. Esses sistemas são chamados de Sistemas de Automação.

Figura 5.2: Planta Industrial e seus Componentes



Fonte: O Autor

A primeira etapa consiste em mapear os elementos encontrados na planta física (1) para um modelo semântico. Essa atividade pode ser realizada por um especialista do equipamento e com conhecimento mais aprofundado do dispositivo que será modelado. Em seguida (2), e de forma complementar, os usuários do domínio da aplicação contribuem com esse modelo, inserindo informações relacionadas com os dados gerados pelo sistema, como por exemplo, a forma de visualização de cada dado. Existe a possibilidade de configurar variáveis do sistema para facilitar a sua visualização. Por exemplo, pode-se descrever as possíveis formas de visualização da temperatura de um componente, sendo que para o técnico em manutenção essa variável deve ser apresentada em forma de gráfico, com um histórico de valores, e para o operador do equipamento essa informação

descreve sempre o valor atual obtido do sensor.

Esse mapeamento é feito em uma ontologia, onde criam-se classes para representar um conjunto de objetos e indivíduos, instâncias dessas classes, que representam os dispositivos físicos da planta industrial. Esse conceito é semelhante ao do paradigma de orientação a objetos. Existe uma classe chamada "IoT" que é a classe "mãe" de todos os dispositivos que serão mapeados e extraídos posteriormente para a geração das interfaces (passo 3). Essa classe possui propriedades relacionadas ao acesso ao servidor, como por exemplo o endereço e a porta do servidor, dados de autenticação e ainda uma propriedade ID que será utilizada para identificar todos os indivíduos das classes "filhas" de IoT. Por exemplo, caso exista uma válvula na planta e se queira incluí-la ao sistema, pode-se criar uma classe "Valvula" para representar um novo conjunto daquele dispositivo e, posteriormente, cria-se um indivíduo que represente uma válvula específica.

O próximo passo é a criação dos objetos conceituais (3) a partir do modelo já construído. Considerando que a IoT conta com um grande número de dispositivos heterogêneos, cada um provendo funções específicas através de sua interface de comunicação, faz-se necessário uma camada de abstração que seja capaz de promover e padronizar o acesso a esses elementos através de uma linguagem e de procedimentos comuns. Assim, esses objetos são representados na camada de Objetos Conceituais e deve ser fornecida uma forma de comunicação comum entre eles. Esses Objetos Conceituais são gerados com métodos que possibilitam o acesso ao *middleware* de forma automática, com a lógica das requisições já implementadas. Isso possibilita que se tenha uma padronização na forma de comunicação e também a minimização de possíveis erros humanos no desenvolvimento dessas interfaces.

Os passos anteriormente descritos (1, 2 e 3) são executados apenas para o primeiro dispositivo a ser adicionado no sistema, ou seja, caso existam 10 válvulas semelhantes, pode-se usar a representação gerada para o primeiro dispositivo e o sistema criará um novo indivíduo no modelo semântico (4) para os 9 elementos restantes.

Em paralelo à criação dos objetos conceituais (3.2) também são cadastrados representações dos dispositivos físicos no *middleware* o que possibilita a comunicação dos objetos conceituais com o mundo físico. Esse registro dos dispositivos é feito através de requisições, a partir das mesmas são enviadas as informações de cada indivíduo. Nesse momento é feita uma conexão entre o indivíduo do modelo semântico e a representação do dispositivo no *middleware*, ou seja, ao criar-se um registro no *middleware* é retornado um identificador (ID) que é salvo no modelo semântico. Isso serve para possibilitar uma

possível consulta a esse elemento específico, além de permitir a atualização das propriedades do indivíduo (4), ou seja, conforme o *middleware* recebe dados do dispositivo físico, esses dados são atualizados no modelo semântico com base nesse identificador.

A adoção de modelos semânticos permite mapear os elementos físicos e suas características através de representações de alto nível. Isso permite uma representação mais próxima aos analistas e aos desenvolvedores. Os modelos semânticos também permitem a criação de regras através da linguagem SWRL (*Semantic Web Rule Language*) e provêm um maior formalismo semântico em relação aos demais modelos.

Como visto em (2), existe uma possibilidade de usuários auxiliarem na criação dos modelos semânticos. Essa etapa é importante para que os usuários finais possam adicionar informações que sejam úteis para o funcionamento do sistema para posterior utilização. Essas informações, inseridas pelos usuários do domínio, são utilizadas por aplicações como, por exemplo, em aplicativos de *smartphones* ou aplicações de *dashboards*.

5.3 Fase de execução

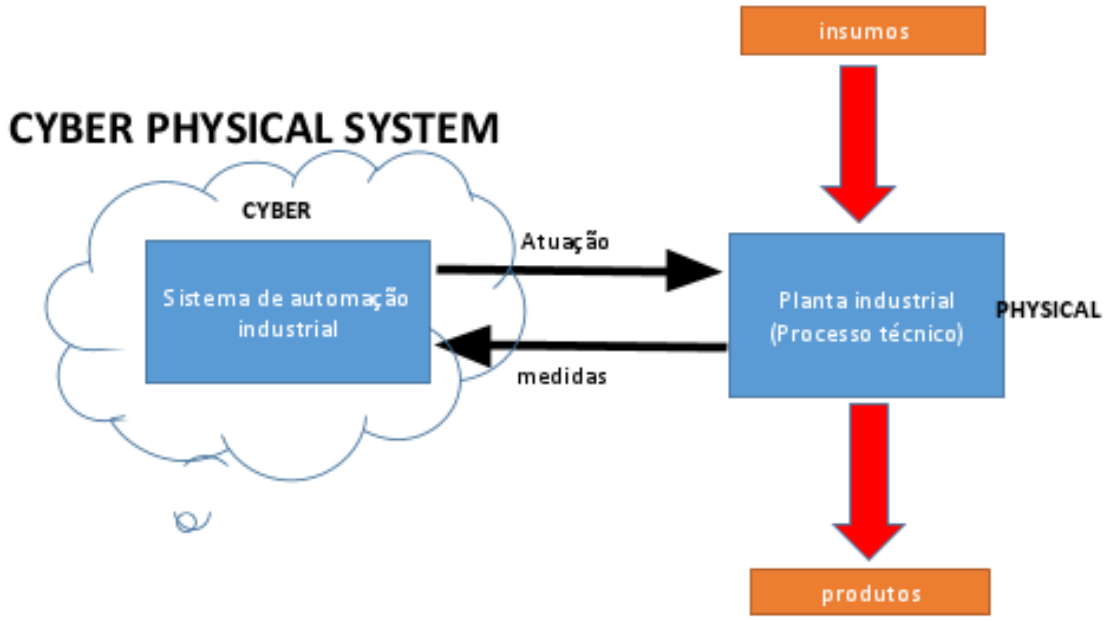
A fase de execução descreve as atividades relacionadas ao sistema em funcionamento. Nesta fase, a integração é realizada e os dispositivos começam efetivamente a interagir e comunicar-se.

A Planta Física é onde os dispositivos físicos, como sensores e atuadores, estão localizados. Os sensores têm a função de detectar mudanças no ambiente e transformá-las em sinais elétricos que serão lidos e processados posteriormente. Os atuadores recebem instruções de controle baseadas no processamento dos dados dos sensores. Por exemplo, uma válvula e um atuador elétrico estariam situados na planta industrial. Os sensores como o de temperatura e pressão seriam os produtores de dados. O atuador elétrico (que faz a válvula abrir ou fechar) recebe instruções para abrir ou fechar a válvula.

Um Sistema de Automação é um componente apto a receber medidas de uma planta industrial e enviar comandos de atuação e controle para mesma. É possível comparar essa arquitetura com CPS onde a planta industrial representa a parte física e o sistema de automação representa a parte cibernética. A aplicação de CPS nas práticas industriais atuais é um fator que transformaria a indústria de hoje na Indústria 4.0 (LEE; BAGHERI; KAO, 2015). A Figura 5.3 apresenta uma estrutura do sistema de automação juntamente com o conceito de CPS.

Conforme pode ser observado na Figura 5.3, os componentes do sistema *cyber*

Figura 5.3: Relação entre Sistema de Automação e o CPS



Fonte: O Autor

physical estão relacionados ao sistema de automação e à planta industrial, ou seja, a parte *cyber* está representada pelo sistema de automação e a parte *physical* é composta pela planta industrial. Por conseguinte, o Sistema de Automação interage com a Planta Industrial através de sinais de atuação e de medidas. Para a Planta Industrial, os insumos são consideradas entradas (*inputs*) enquanto os produtos são considerados saídas (*outputs*).

Os Objetos Conceituais, gerados na fase de modelagem, são sincronizados com os elementos da Planta Física e, como possuem interfaces com o *middleware*, eles são responsáveis por transmitirem os dados para essa plataforma.

Visto que este trabalho emprega o conceito de IoT como forma de integração entre os elementos de um sistema, é possível que dispositivos heterogêneos com interface e com implementações distintas possam trocar informações e colaborar por um objetivo comum. Nesse contexto, *middlewares* são uma opção comumente adotada em projetos de IoT, pois eles proveem funções que possibilitam a comunicação entre elementos heterogêneos e com isso contribuem para a concepção de projetos IoT (FARAHZADI et al., 2017).

Apesar de os Objetos Conceituais já serem gerados de uma forma padronizada, o uso de um *middleware* permite que aplicações externas possam se conectar a ele, aumentando o conjunto de aplicações passíveis de serem criadas.

Atualmente existem diversos *middlewares* na literatura que podem atender os requisitos da integração de sistemas no contexto da Indústria 4.0. Dessa forma, optou-se

por utilizar uma plataforma conceituada e previamente padronizada ao invés de criar uma nova.

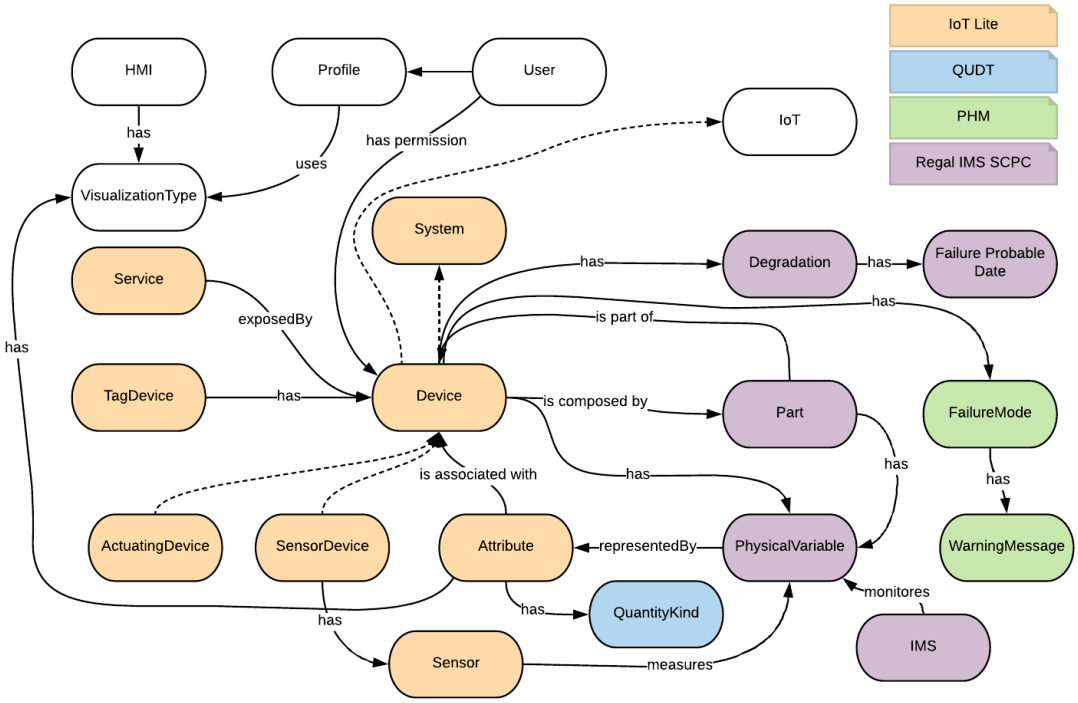
Finalmente, a última camada da proposta é a Aplicação. Essa camada serve de interface para os usuários, além de permitir uma conexão com sistemas de terceiros, como por exemplo, os sistemas de cadeias de suprimentos, ERPs, entre outros.

5.4 Modelo Semântico desenvolvido

Este trabalho utiliza algumas ontologias já definidas na literatura de domínios específicos, como por exemplo, a IoT-Lite (BERMUDEZ-EDO et al., 2016) e a QUDT (RIJGERSBERG; ASSEM; TOP, 2013), visto que estes modelos já estão padronizados e já são usados por outros projetos. Além dessa união, a proposta adiciona conceitos relacionados à Interface Humano Computador, bem como a modelagem de componentes industriais que compõem o sistema. Também criou-se uma classe para identificar os dispositivos que serão extraídos desse modelo para uma futura geração de código.

A Figura 5.4 ilustra como as ontologias foram agrupadas e a partir desse agrupamento foram adicionadas algumas classes necessárias para o sistema.

Figura 5.4: Proposta de estrutura de ontologias



Fonte: O Autor

Conforme pode ser observado na Figura 5.4 diferentes ontologias podem ser cor-

relacionadas e agrupadas em uma estrutura comum para fundamentar o desenvolvimento de sistemas.

A classe “IoT” foi adicionada para identificar todos os dispositivos que serão extraídos do modelo para a geração das interfaces de comunicação. Existem classes como a “*User*” que é utilizada para fazer o mapeamento dos usuários que usarão o sistema. Ainda, cada perfil de usuário terá uma forma de visualizar um atributo de um dispositivo. Essas informações são modeladas através da classe “*VisualizationType*” que está relacionada com a classe “*Attribute*”, ou seja, cada atributo possui uma forma de visualização.

A Tabela 5.1 lista as ontologias utilizadas para geração do modelo da proposta.

Tabela 5.1: Ontologias utilizadas para a construção do modelo da proposta

Nome	*Descrição*	*Referência*
IoT Lite	É uma ontologia baseada na SSN (COMPTON et al., 2012) voltada para a modelagem de sistemas IoT. Essa ontologia possui conceitos como Sensores e Atuadores que são indispensáveis em sistemas desse domínio.	(BERMUDEZ-EDO et al., 2016)
IMS & SPSC	É uma ontologia que foi criada para unir conceitos de IMS e SPSC. Ela possui conceitos importantes relacionados à cadeia de suprimentos, como processos, recursos, ator, papel e conceitos relacionados à planta industrial com peça, dispositivo e degradação.	(SILVA; PEREIRA, 2014)
PHM	É uma ontologia voltada para conceitos relacionados à saúde de uma máquina industrial. Ela aprofunda mais o conceito de degradação e traz outros conceitos relacionados aos dispositivos físicos.	(NUÑEZ; BORSATO, 2017)
QUDT	Esta ontologia apresenta conceitos relacionados às quantidades e unidades de um objeto/elemento.	(RIJGERSBERG; ASSEM; TOP, 2013)
Extensão	Além dessas ontologias citadas, foi necessário mapear conceitos relacionados à integração utilizando o conceito de IoT.	(STEINMETZ et al., 2017)

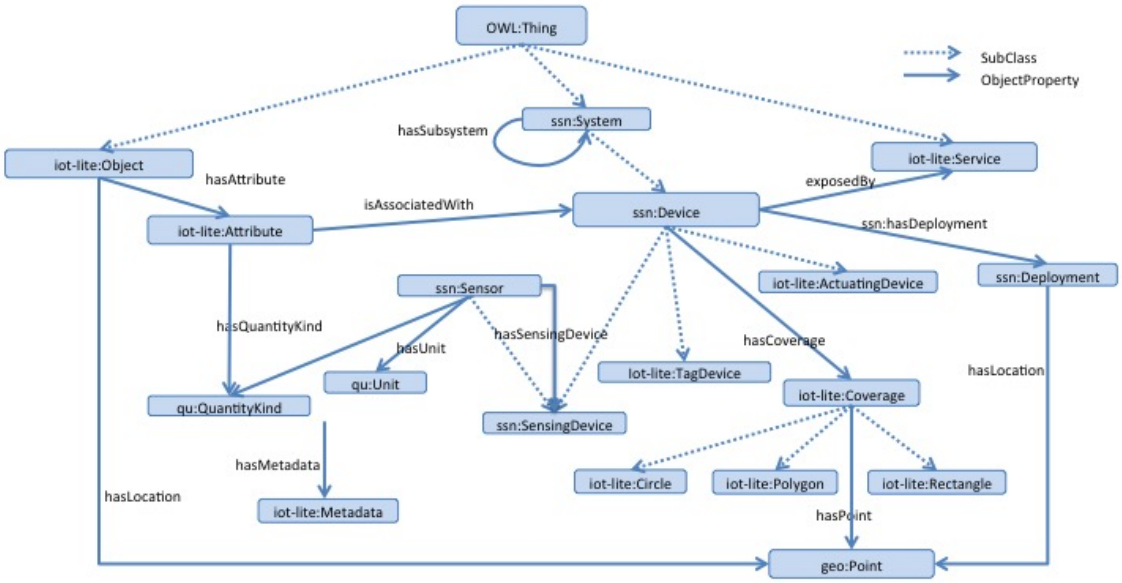
Fonte: O Autor

A ontologia IoT-Lite contém conceitos no domínio da IoT, como pode ser observado na Figura 5.5.

Essa ontologia foi escolhida devido ao fato de representar de forma simplificada alguns conceitos que vem de encontro com esta proposta. As principais elementos dessa ontologia estão listadas abaixo:

- **Device**: classe para representar os dispositivos presentes no sistema.

Figura 5.5: Ontologia IoT-Lite



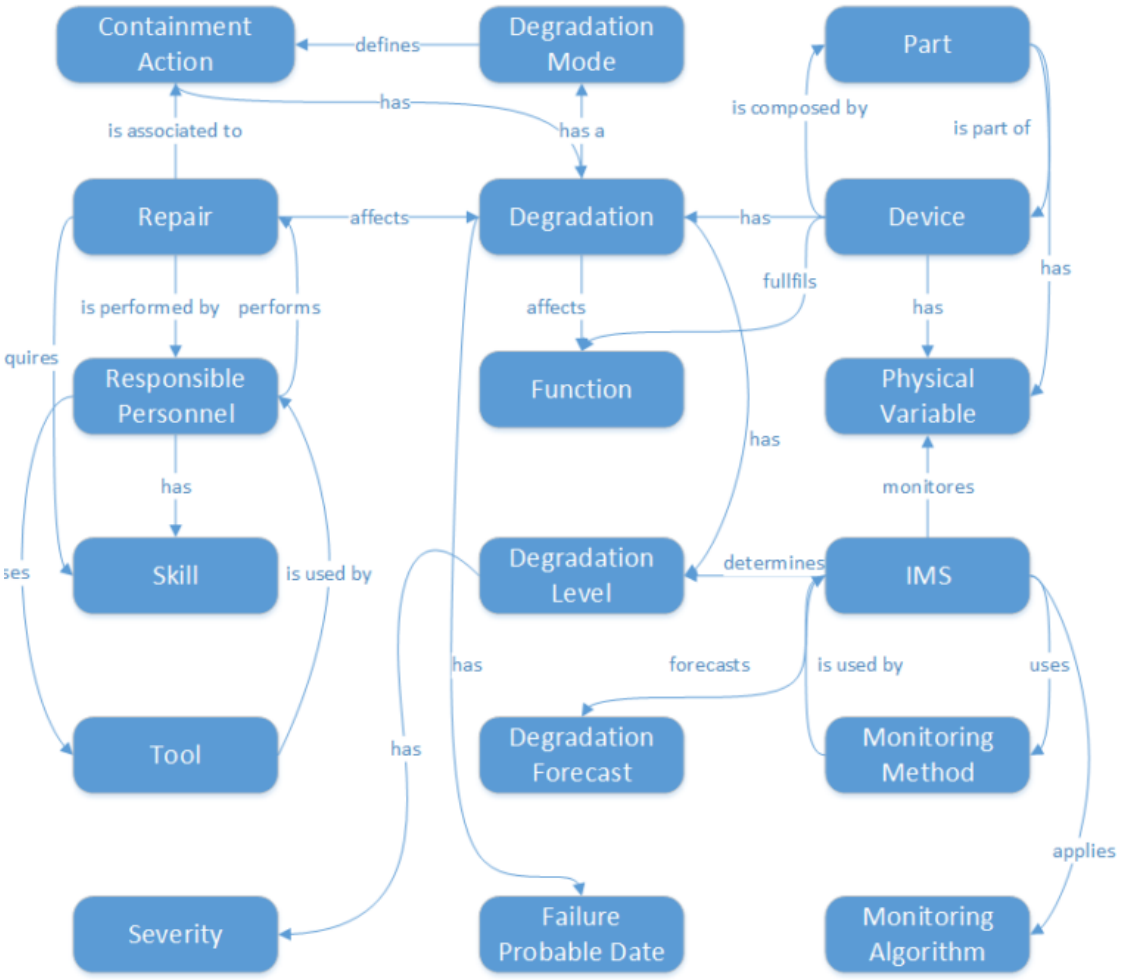
Fonte: (BERMUDEZ-EDO et al., 2016)

- **ActuatingDevice**: classe para representar um tipo de *device* com características de atuação. No contexto industrial, uma válvula poderia ser vinculada a essa classe, visto que ela possui capacidade de atuação (abrir e fechar).
- **TagDevice**: uma característica de sistemas IoT é que cada dispositivo tenha uma identificação única. Assim, essa classe traz justamente esse conceito vinculado ao dispositivo.
- **SensingDevice**: é uma classe para representar um tipo de *device* que possui característica de "sentir" ou "perceber" variáveis do ambiente, como temperatura, umidade, pressão entre outros.
- **Attribute**: essa classe está associada aos dispositivos, pois eles podem possuir características ou estados que são representados por atributos. Atributos possuem um tipo de quantidade, por exemplo, metros, graus, quilos entre outros. Nesse sentido, a ontologia QUDT traz diversas classes para representação de unidades e quantidades.

De forma complementar, a ontologia de (SILVA, 2015) traz conceitos do contexto de IMS e SPSC que são importantes no âmbito desta proposta. Uma visão geral dessa ontologia pode ser observada na Figura 5.6.

IMS propõe um conceito em que é possível monitorar variáveis físicas vinculadas a uma peça ou dispositivo a fim de identificar uma possível degradação que possa afetar o funcionamento e então prover ações programadas de manutenção. Assim, a ontologia

Figura 5.6: Visão geral da camada IMS



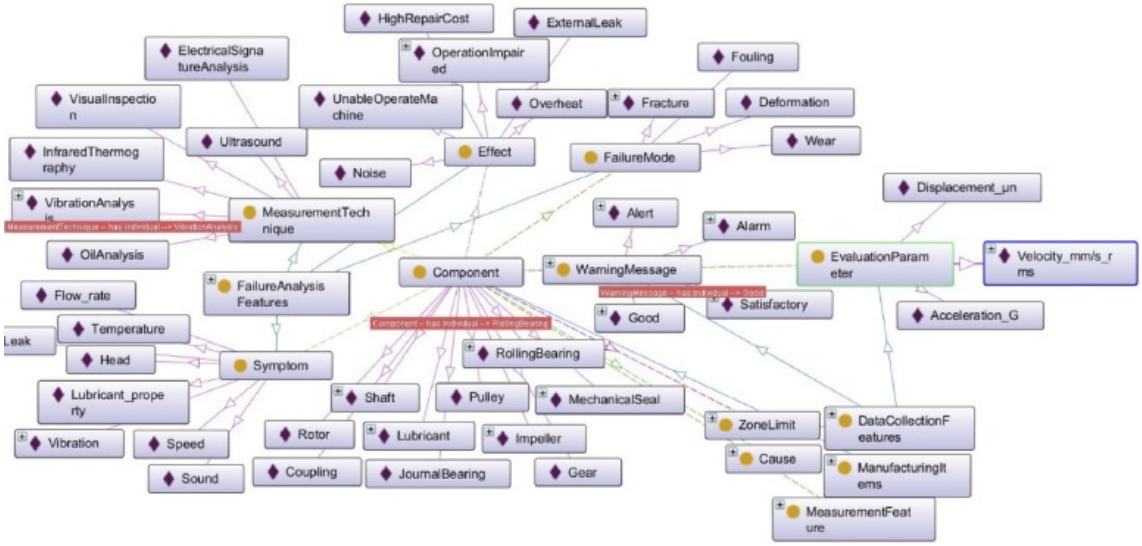
Fonte: (SILVA; PEREIRA, 2014)

de (SILVA; PEREIRA, 2014) traz conceitos relacionados a esse domínio, onde existem dispositivos (*Device*) que são compostos por partes (Part), possuem variáveis físicas (*Physical Variable*) que são monitoradas por sistemas IMS através de algum método (*Monitoring Method*). Os dispositivos sofrem degradação (*Degradation*) e essa afeta as funções do dispositivo. A degradação é afetada por reparos (*Repair*) executados por uma pessoa responsável (*Responsible Personel*) que possui habilidades (*Skill*) para essa tarefa. Através da degradação ainda pode-se determinar uma previsão de uma possível falha (*Failure probable date*).

Adicionalmente, foi utilizada a ontologia *Prognostics and Health Management* (PHM), proposta por (NUÑEZ; BORSATO, 2017). Nela podem ser encontrados conceitos relacionados à saúde de uma máquina industrial, como por exemplo, a degradação.

Uma parte da ontologia pode ser observada na Figura 5.7, onde estão modelados conceitos relacionados a falhas em um componente industrial.

Figura 5.7: Parte da ontologia PHM com classes e instâncias



Fonte: (NUÑEZ; BORSATO, 2017)

Pode-se observar que existe uma classe para componentes (*Component*) e algumas instâncias dessa classe que representam os componentes, como por exemplo, engrenagens (*Gear*), polia (*Pulley*), lubrificante (*Lubrificant*), entre outros.

Existem, ainda, diversas classes relacionadas com falhas em componentes, como por exemplo a classe Sintoma (*Symptom*) a qual possui instâncias como temperatura (*temperature*), vibração (*vibration*), velocidade (*Speed*), som/barulho (*Sound*), entre outros. Outra classe relacionada é a de mensagens (*WarningMessage*), a qual possui instâncias como alerta (*Alert*), alarme (*Alarm*), bom (*Good*) e satisfatório (*Satisfactory*). A classe efeitos (*Effect*) possui indivíduos como superaquecimento (*Overheat*), vazamento externo (*ExternalLeak*), alto custo de reparo (*HighRepairCost*), barulho (*Noise*), entre outros, os quais representam os possíveis efeitos de uma falha.

A proposta utiliza as ontologias acima descritas, porém é possível que se adicionem outras ontologias que possam vir a ser necessárias em um outro domínio de aplicação.

5.5 Aplicação do conceito de IoT

Para a presente proposta, foi adotado o modelo de quatro camadas de (TRAPPEY et al., 2017) por entender-se que este traz uma maior clareza na organização do sistema. Essa forma de organização é bastante usada em projetos IoT (SUO et al., 2012) o que ajuda no entendimento das inovações que esse conceito pode trazer (TRAPPEY et al.,

2017).

O conceito IoT une o físico e o cibernético e, no contexto industrial, pode proporcionar um uma produção inteligente (YAO; LIN, 2016).

5.5.1 Arquitetura de quatro camadas

As camadas e os elementos da arquitetura que as compõem são importantes, pois contribuem para a diminuição de problemas de interoperabilidade em processos industriais. Além disso, esta abordagem contribui, ainda, a entender as tecnologias envolvidas no sistema bem como os possíveis desafios ou limitações (TRAPPEY et al., 2017). As quatro camadas utilizadas na proposta deste trabalho estão descritas nas próximas subseções.

5.5.2 Camada de Percepção

A camada de percepção é composta pelos sensores e atuadores de um sistema. O principal objetivo dessa camada é, através desses dispositivos físicos, perceber o ambiente que pode trazer informações úteis. Ela faz com que objetos possam ver, sentir, tocar, ouvir e ainda pensar (LEXINNOVA, 2015). A percepção é usada como fonte de dados para as camadas superiores, como por exemplo para a camada de aplicação em que a mesma é responsável por processar os dados enviados pela camada de percepção.

Na presente proposta a camada de percepção é representada pelo Sistema de Automação juntamente com a Planta Industrial, onde encontram-se os componentes físicos do sistema. Tais componentes, como por exemplo sensores de temperatura, pressão ou torque, em válvulas industriais, são os principais fornecedores de dados.

5.5.3 Camada de Transmissão

A camada de transmissão está "posicionada" acima da camada de percepção e sua função é conectar o conjunto de sensores e atuadores com as camadas superiores. Existem diversas tecnologias e arquiteturas padronizadas para cumprir essa função, como por exemplo conexão cabeada (USB, RJ45...), Wireless (Bluetooth, wifi, NFC...) e ainda os diferentes protocolos (IPV4, IPV6, HTTP, MQTT...) (TRAPPEY et al., 2017).

O conjunto de regras e restrições arquiteturais chamado REST (*Representational*

state transfer) busca diminuir a latência e a comunicação na rede, bem como maximizar a escalabilidade do sistema. REST fornece um acesso direto à informação e permite que a própria requisição a um serviço já contenha todas as ações ao determinado elemento que se acessa (FIELDING; TAYLOR, 2002). Esse padrão é geralmente utilizado para aplicações comunicarem com servidores.

5.5.4 Camada de Computação

A camada de computação é responsável por receber e processar os dados, tomar decisões e ainda disponibilizar esses dados para a camada de aplicação. Essa camada é composta por componentes de hardware, software, algoritmos, segurança de informação e por uma plataforma de integração (TRAPPEY et al., 2017).

Podem ser citados como componentes desta camada os Objetos conceituais, os modelos de conhecimento de domínio e o *middleware* usado para realizar a integração.

Esses componentes representam a maior parte da proposta, pois engloba os modelos que contêm as representações dos dispositivos físicos, os quais são a base para a camada superior que é a de aplicação. Os modelos são utilizados para mapear as informações referentes à planta industrial, posteriormente são usados para o desenvolvimento da integração e por fim são utilizados para apresentar as informações aos usuários.

Outro componente importante dessa camada é o *middleware*. Ele é o responsável por prover uma interface padrão de acesso através da qual todos os dispositivos possam enviar e receber informações.

Ainda, nessa camada estão localizados os componentes de software que interagem com os dispositivos físicos, como os Objetos Conceituais. Esses elementos criam uma abstração dos detalhes de comunicação com o mundo físico, facilitando a interação com as camadas superiores.

5.5.5 Camada de Aplicação

A última camada é denominada como Camada de Aplicação. Nessa camada é provida uma interface para os usuários visualizarem informações vindas das camadas inferiores. A aplicação consiste em uma relação do sistema com os consumidores (IoT2C) e com empresas (IoT2B) (TRAPPEY et al., 2017).

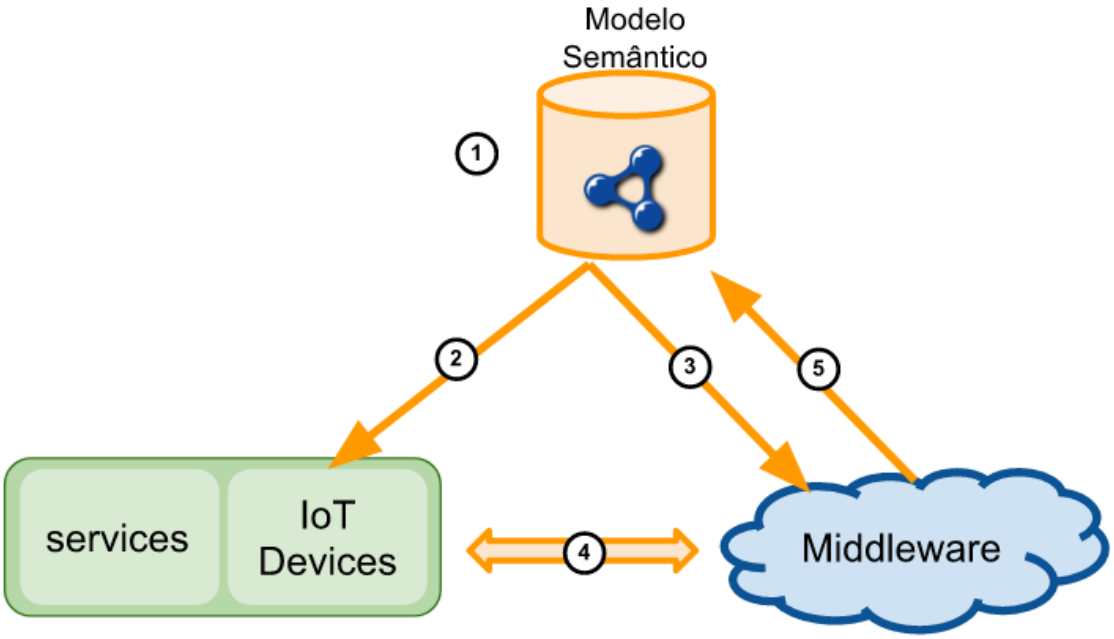
Atualmente existem diversas possibilidades para criar-se interfaces com os usuários, como, por exemplo, aplicativos para dispositivos móveis, sistemas para Internet, aplicações convencionais para computadores ou ainda aplicações de realidade aumentada que podem ser acessadas em dispositivos inteligentes, como relógios, óculos, entre outros.

Essa camada é composta basicamente por software e protocolos que padronizam o acesso às informações. Ainda, são providas formas de autenticação para garantir a segurança das informações.

5.6 Ferramentas desenvolvidas para extração das informações da ontologia

Uma das grandes diferenças desta proposta em relação às demais encontradas na literatura é a possibilidade de gerar automaticamente interfaces de conexão dos dispositivos com o *middleware* IoT. A Figura 5.8 ilustra essa ideia, onde o modelo poderá ser utilizado em todo o processo de integração.

Figura 5.8: Proposta de geração de interfaces



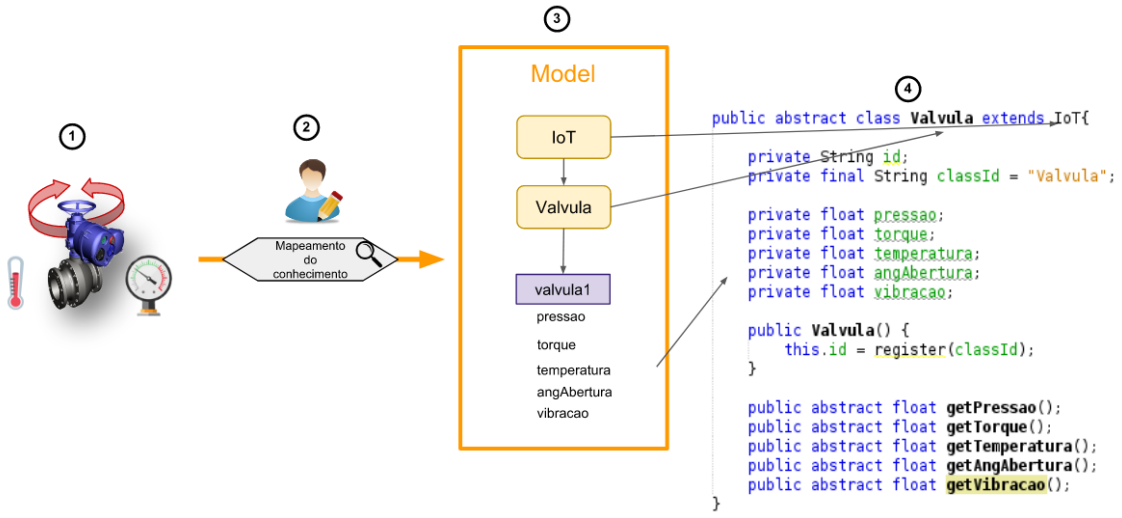
Fonte: O Autor

Após ter-se um modelo que descreve os componentes físicos e suas características, passo (1) (que corresponde aos passos 1 e 2 da Figura 5.1), é possível extrair essas informações do modelo e então criar interfaces automaticamente, passos 2 e 3 (que correspondem ao passo 3 da Figura 5.1) que servirão para facilitar o processo de comunicação, passo 4 (que corresponde às setas em laranja na Figura 5.1).

Ontologias possuem conceito de classes e indivíduos e através desses conceitos pode-se representar objetos de um sistema real bem como suas características. Ao extrair essas informações, as mesmas podem ser usadas para a criação de código para ser usado em serviços e até mesmo em dispositivos IoT. De forma semelhante, essas informações podem ser registradas no *middleware*, criando-se uma instância para cada indivíduo da ontologia.

A Figura 5.9 apresenta um exemplo de como as informações são modeladas em uma ontologia e após extraídas para geração das interfaces de comunicação.

Figura 5.9: Ilustração mostrando como os indivíduos são extraídos



Fonte: O Autor

Considerando que se tenha uma válvula industrial (Planta Física) com sensores de temperatura, torque e pressão que geram sinais relacionadas a essas grandezas (1). O próximo passo é uma tarefa humana por parte de alguém que tenha conhecimento desse dispositivo e esse conhecimento é extraído (2) e representado em um formalismo semântico (3).

Pode-se observar que a classe “IoT” está no topo, pois ela indica os indivíduos que devem fazer parte do sistema, ou seja, os indivíduos que são “filhos"ou instâncias de “IoT” representam um componente físico.

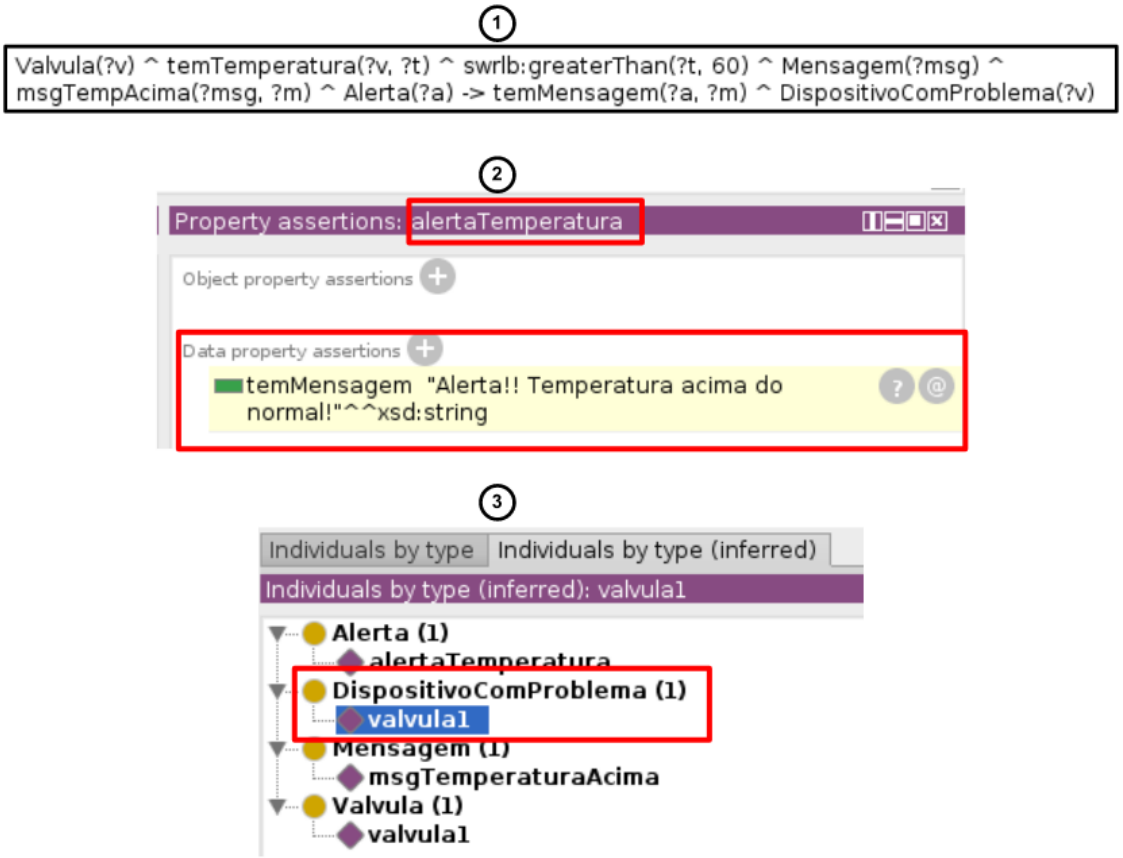
Ao final (4) essas informações são lidas do modelo semântico e então são criadas as interfaces que facilitam a integração desses dispositivos, fornecendo métodos padrões e implementados para envio de dados ao *middleware*.

Ainda, é possível inserir regras semânticas escritas em SWRL o que torna o modelo ainda mais completo. As regras funcionam da seguinte forma: Sempre que as condições especificadas no antecedente forem verdadeiras, então a regra é ativada e as con-

dições do consequente também tornam-se verdadeiras (PRENTZAS; HATZILYGEROUDIS, 2007).

A Figura 5.10 apresenta um exemplo do uso de regras e escritas em SWRL. Esse tipo de regras podem ser adicionadas no modelo proposto neste trabalho.

Figura 5.10: Exemplo de regra SWRL para um alerta de temperatura.



Fonte: O Autor

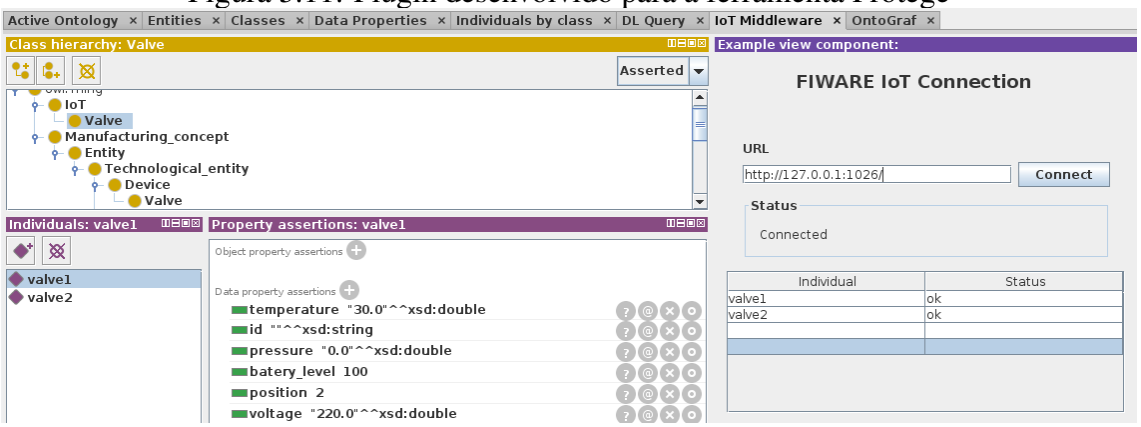
Em (1) a regra é propriamente escrita. Percebe-se que existe uma seleção dos indivíduos da classe “Valvula” e são filtrados os elementos que tiverem a temperatura maior que 60. Esse valor foi usado de forma hipotética, apenas para mostrar a possibilidade de escrever regras. Caso essa regra seja verdadeira, o indivíduo “alertaTemperatura” (2) receberá uma propriedade com uma mensagem de alerta, indicando esse problema. Ainda, o indivíduo que se encaixar nessa regra é vinculado à classe “DispositivoComProblema” (3), facilitando uma possível consulta nesse modelo.

Para essa tarefa, inicialmente foi desenvolvido um *plugin* (STEINMETZ et al., 2017) para a ferramenta Protégé. Posteriormente, também foi desenvolvida uma extensão para um *middleware* IoT que possibilitasse trabalhar com esses modelos. As subseções abaixo descrevem essas ferramentas.

5.6.1 Plugin para o Protégé

Essa ferramenta facilita a criação de sistemas IoT a partir de um modelo de alto nível e, como está acoplada a um software consolidado, torna ainda mais trivial essa tarefa. A Figura 5.11 apresenta a interface gráfica do *plugin* desenvolvido.

Figura 5.11: Plugin desenvolvido para a ferramenta Protégé



Fonte: O Autor

Esse *plugin* foi desenvolvido pensando em auxiliar arquitetos de software ou usuários de um domínio a criarem sistemas IoT baseada em modelos. Essa ferramenta foi desenvolvida para funcionar com um *middleware* específico, chamado FIWARE (FIWARE, 2017a).

Essa plataforma foi escolhida pelo fato de ela fornecer um conjunto de APIs simples que facilitam o desenvolvimento de aplicações para diferentes setores. Essas APIs são públicas, gratuitas e contam com uma documentação completa para consulta. Além disso, FIWARE é um projeto de código aberto, o que possibilita que novas funcionalidades sejam desenvolvidas de acordo com as necessidades de cada projeto. Outra vantagem dessa plataforma é o *FIspace*(BUZIN; FOURNIER; ARCUSHIN, 2013) que é uma plataforma para sistemas de colaboração onde é possível modelar, implementar e implantar soluções para diferentes problemas.

Existe um campo de texto para ser inserido o endereço e a porta do servidor que contém a plataforma FIWARE instalada. O botão “Connect” faz a conexão com o servidor e abaixo apresenta-se uma mensagem para indicar se a conexão foi feita com êxito ou não. Ainda é possível observar uma lista de indivíduos que lista esses elementos da ontologia.

Considerando que tenha-se carregado um arquivo de uma ontologia no Protege, ao clicar-se no botão “Connect” o *plugin* desenvolvido executa uma varredura na ontologia em busca de indivíduos, filhos da classe “IoT”. Ao encontrá-los, suas propriedades são

identificadas e é criada uma *string* no formato $JSON$ que é transmitida ao *middleware*. Esse *middleware* cria uma instância desse indivíduo e retorna um identificador o qual é salvo no modelo semântico.

Uma grande dificuldade no desenvolvimento de sistemas complexos é em manter modelos atualizados conforme as atualizações e modificações ocorrem (FRANCE; RUMPE, 2007). Especialmente em sistemas IoT essa adaptação de modelos também está bastante presente, sendo que dispositivos podem entrar e sair do sistema a qualquer momento.

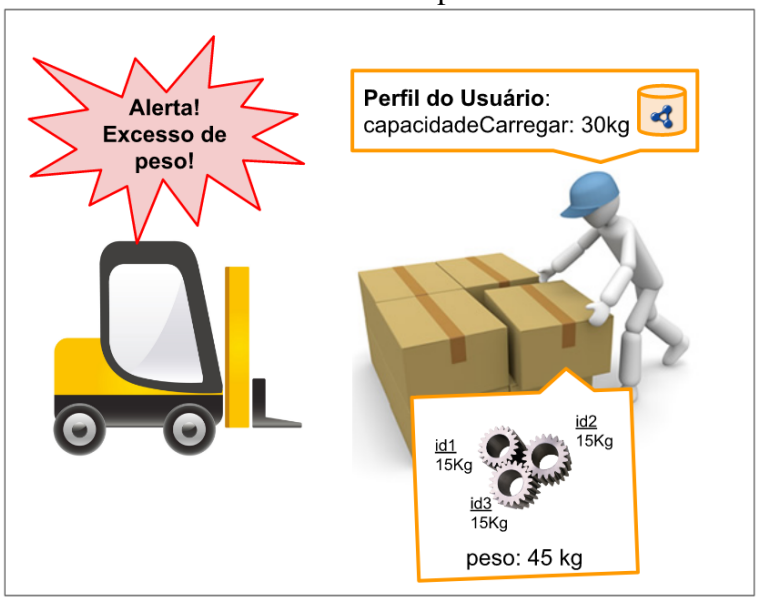
Nesse sentido, faz-se necessária uma estratégia para manter os modelos atualizados. Para essa proposta, criou-se uma extensão do *middleware* que possibilita uma atualização contínua dos modelos. Assim, a cada atualização no sistema, a extensão é notificada e o modelo é atualizado, tendo-se um modelo atualizado dinamicamente e em tempo de execução. Com isso, obteve-se uma nova perspectiva em relação à apresentada com o desenvolvimento do *plugin* para o Protégé, tendo-se um meio comum de acesso ao modelo via a plataforma de integração.

O modelo atualizado permite que se façam consultas SPARQL muito precisas em relação ao estado atual do sistema. Isso é extremamente importante em situações em que se precise tomar uma decisão de forma mais ágil. O trabalho de Mayer et al. (2017) apresenta justamente essa necessidade, através de uma proposta em que o sistema é capaz de criar alertas aos usuários no desenvolvimento de suas atividades dentro da indústria. Um exemplo de uso dessa proposta é como avisar o usuário que ele deve usar luvas anti vibração para usar uma furadeira. Nesse caso, o sistema pode identificar que o usuário não está com as luvas através de *tags* e sensores que detectam essas marcações, ou ainda, o sistema pode gerar mensagens de aviso que determinada atividade necessita de luvas, independente se o usuário está com luvas ou não.

Pode ser citado como exemplo, nesse contexto industrial, a tarefa de carregar objetos pesados. Existem leis que regulam a quantia de peso que funcionários podem levantar no exercício de suas atividades. Caso essas informações estejam mapeadas em um modelo, e esse modelo esteja em constante atualização com os processos de uma planta industrial, pode-se criar mecanismos que ajudem os usuários. Neste caso, diferentes tipos de alertas podem ser desenvolvidos indicando possíveis auxílios na execução de suas atividades. A Figura 5.12 ilustra o cenário mencionado acima, onde uma pessoa que tem capacidade de carregar 30kg irá erguer uma caixa contendo 3 engrenagens de 15kg cada. Como cada item pode ser identificado, o sistema pode identificar que o peso daquela caixa

está acima do peso que o usuário pode operar e assim emitir um alerta.

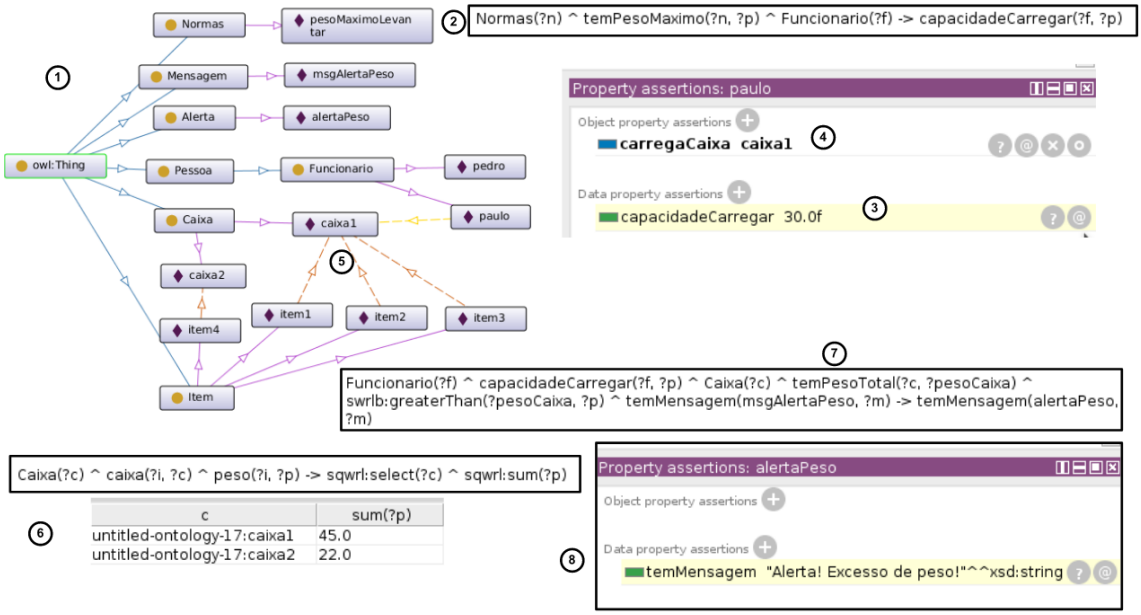
Figura 5.12: Possível cenário industrial para uso de modelos semânticos



Fonte: O Autor

Esse controle pode ser adicionado à ontologia em forma de uma regra em SWRL e assim uma mensagem ser gerada a partir dessa regra. A Figura 5.13 apresenta uma possível forma de modelar esse problema através de uma ontologia.

Figura 5.13: Possível modelo para o cenário industrial



Fonte: O Autor

Em (1) pode-se observar o modelo semântico com as classes, suas relações e indivíduos. A classe “Normas” possui uma instância chamada “pesoMaximoLevantar” que nesse cenário representa uma norma em que funcionários só podem levantar 30kg nas

suas atividades na indústria. Essa norma é vinculada aos funcionários através de uma regra descrita em (2) e aplicada em (3). Supondo que o funcionário Paulo está encarregado de carregar uma caixa (4) e essa caixa (5) possui três itens de 15kg cada, conforme ilustrado na Figura 5.12, é possível consultar e identificar o peso da caixa, como mostrado em (6). Nesse cenário, o sistema pode, através desse modelo semântico, alertar o que usuário necessita de ajuda para trabalhar com essa caixa. Assim, em (7) pode-se observar uma regra que busca a capacidade do usuário de exercer essa atividade, o peso total da caixa e então compara esses valores. Caso o peso da caixa for maior que a capacidade do usuário, então é vinculada uma mensagem de alerta (8) e o sistema pode processá-la e apresentá-la através de uma interface de comunicação.

5.6.2 Extensão para o *middleware* IoT

Alternativamente, foi desenvolvida uma extensão para um *middleware* IoT que possibilitou trabalhar com os modelos semânticos de uma forma integrada a essa plataforma. Dessa forma, o modelo fica disponível às demais aplicações que se comunicam com o sistema.

A extensão possui todas as funcionalidades que o *plugin* desenvolvido para o Protégé, porém, ao estar junto ao *middleware*, esse modelo pode ser usado para disponibilizar os dados com informações referentes a cada atributo bem como existe a possibilidade de criar-se restrições de acesso a esses dados. As regras SWRL são executadas pela extensão, o que permite que alertas sejam gerados a partir do modelo atualizado.

Como o *middleware* utilizado trabalha com *publish/subscribe*, foi possível criar uma conexão em que a extensão é notificada a cada alteração no sistema. Assim, o modelo é mantido atualizado a cada notificação recebida pelo *middleware*.

Foram criadas interfaces no padrão REST para que dispositivos possam acessar diretamente a extensão desenvolvida. Essas interfaces possibilitam a criação de um dispositivo novo e, quando sua classe já estiver modelada na ontologia, a extensão é capaz de criar uma instância no modelo de forma automática.

5.7 Aplicações com base nos modelos semânticos

Modelos semânticos são amplamente usados em diversos domínios e para diferentes propósitos, sendo que esses modelos têm sido fortemente usados para prover interoperabilidade entre sistemas. Entretanto, tais modelos também podem ser usados para outros fins, como por exemplo, serem base para a geração de aplicações finais para usuários.

Interfaces baseadas nesse tipo de modelos possibilitam que os dados sejam apresentados de uma forma mais próxima ao nível de abstração do usuário, pois esses modelos são criados visando representar as informações que o sistema pode fornecer. Neste modelo, os usuários auxiliam na definição da maneira com que os dados devem ser apresentados pelas aplicações. Assim, não existe essa preocupação no desenvolvimento desses sistemas, visto que nessa fase algumas características do sistema não estão definidas (ENDERT et al., 2015). Isso pode ser visto como uma vantagem, pois os usuários participarão de processos que sejam mais próximos aos seus conhecimentos e terão algo mais concreto do sistema em relação às fases iniciais do desenvolvimento.

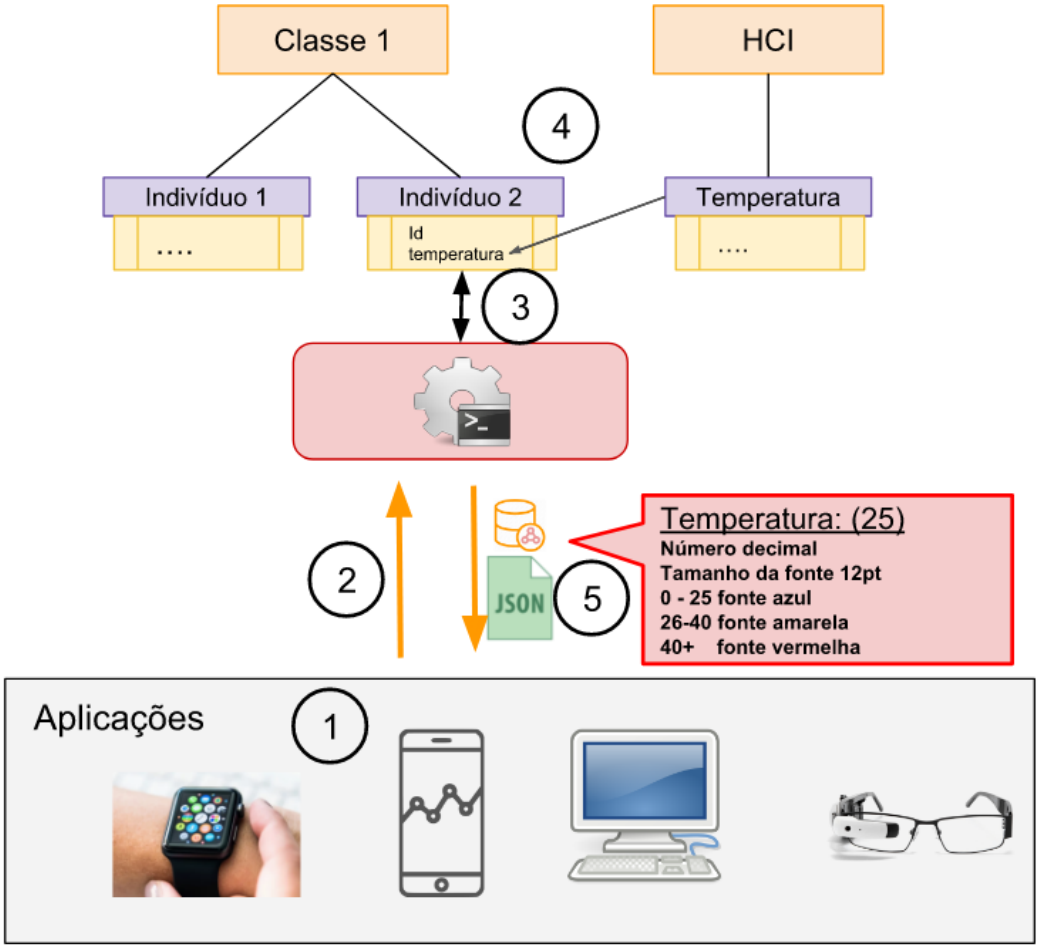
Nesse sentido, esta proposta busca adotar esse modelo de desenvolvimento, uma vez que utiliza modelos semânticos para contribuir com a visualização das informações.

Na abordagem proposta, existe a possibilidade de os dados estarem relacionados com informações semânticas, as quais dizem respeito a como eles devem ser visualizados para um melhor entendimento pelo usuário.

A Figura 5.14 apresenta um cenário em que modelos semânticos estão juntos aos dados e às aplicações e, ao fazerem consultas, recebem esses dados juntamente com informações extras que serão utilizadas pelas aplicações para gerar as interfaces para o usuário.

Nesse cenário, aplicações (1) que estão sendo executadas nos dispositivos dos usuários fazem requisições (2) de informações sobre um elemento da planta industrial. Quando a extensão do *middleware* receber as requisições, é feita uma consulta no modelo semântico (3) para buscar informações extras que não estão em base de dados convencionais. Posteriormente (4), todas as informações relacionadas a um determinado elemento são coletadas e agrupadas juntamente com as informações referentes ao estado do dispositivo. Pode-se perceber que nessa etapa existem informações relacionadas a um determinado dado, no exemplo, temperatura. Esse modelo permite que se tenham definições ilimitadas relacionadas a uma informação, entretanto isso está vinculado à performance do sistema. Após ter-se todas essas informações unidas, elas são disponibilizadas (5) em um formato padrão para que as aplicações possam consumi-las. Por fim, esse conjunto

Figura 5.14: Exemplo de aplicações recebendo dados com informações semânticas



Fonte: O Autor

de informações é processado e exibido ao usuário conforme modelado em alto nível no modelo semântico.

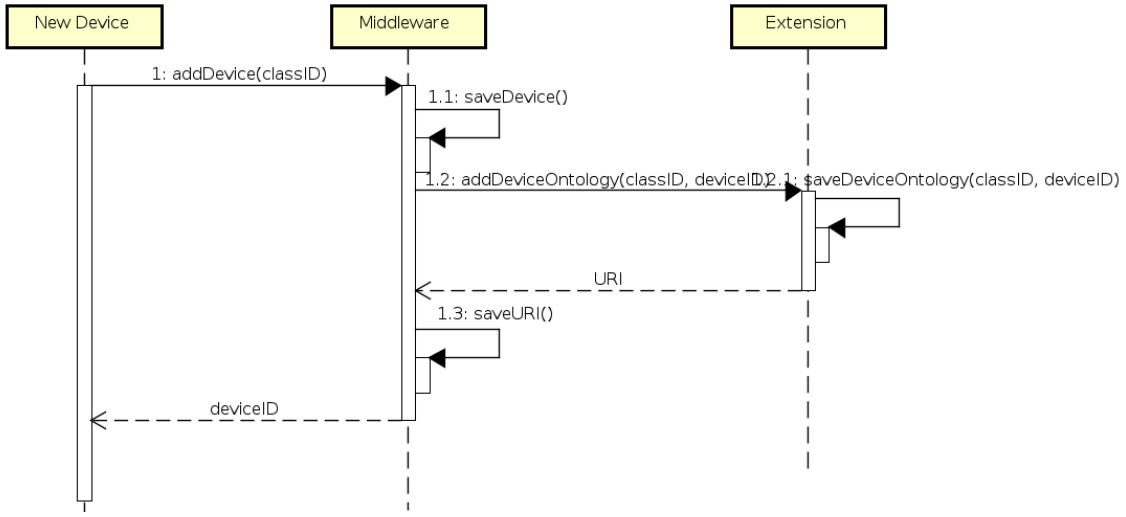
5.8 Fluxos do sistema

A fim de descrever o funcionamento do sistema, foram criados dois diagramas de sequência que mostram duas situações comuns no âmbito da proposta.

A Figura 5.15 apresenta a sequência de passos necessários para adicionar um dispositivo físico ao sistema.

O primeiro ator nesse caso é a representação lógica do dispositivo novo que será adicionado ao sistema. Ele fará uma requisição ao *middleware*, através das interfaces previamente geradas, enviando um código identificador da classe a qual ele pertence. Por exemplo, caso seja adicionada uma nova válvula industrial a qual foi modelada na onto-

Figura 5.15: Adicionando um novo dispositivo



Fonte: O Autor

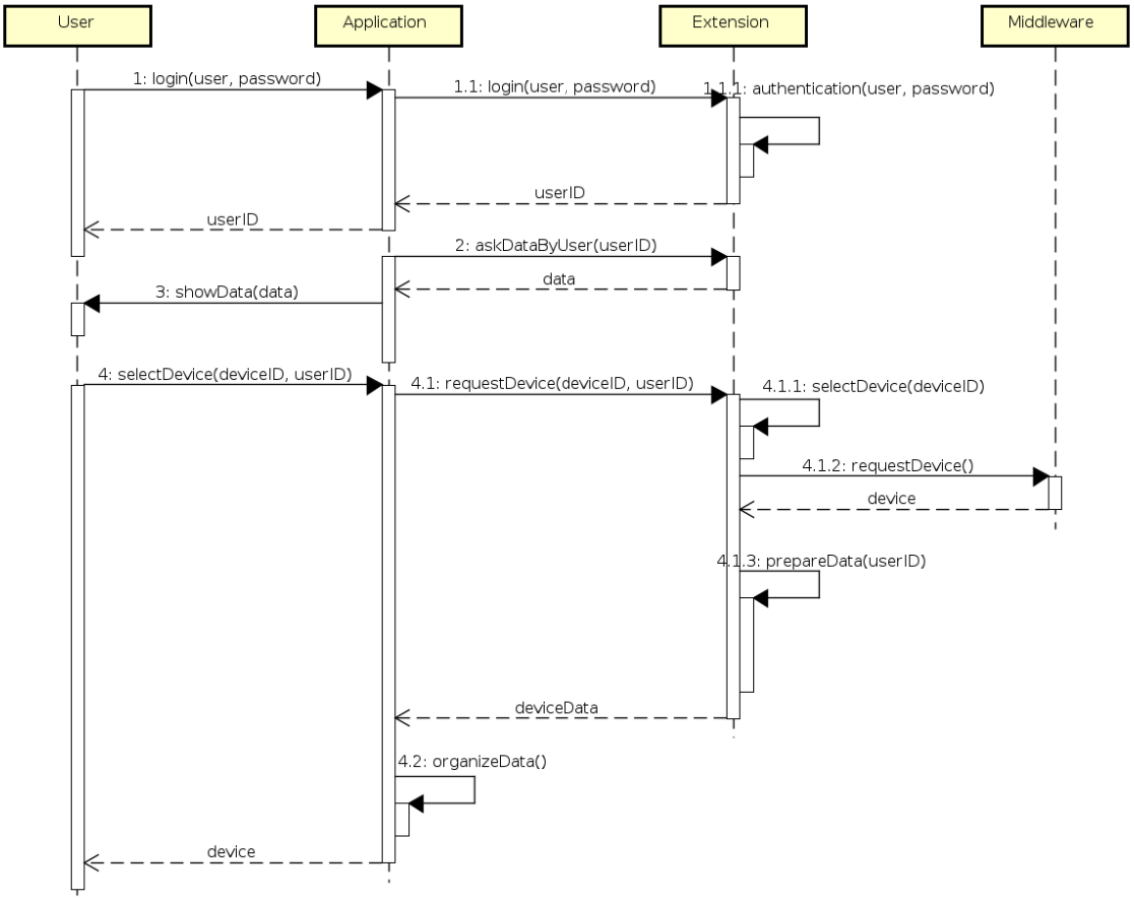
logia como sendo um elemento pertencente à classe “ControlValve”, então será enviado o identificador dessa classe para indicar que o novo dispositivo pertence a ela e então que seja possível encontrar essa classe na ontologia. O FIWARE adiciona esse dispositivo à sua base dados e notifica a extensão criada neste trabalho, e esta atualiza o modelo semântico com o novo dispositivo. Por fim, o dispositivo recebe uma resposta com um código que será seu identificador (deviceID).

Outro fluxo importante é onde o usuário final busca informações sobre um determinado dispositivo no sistema. A Figura 5.16 apresenta esse fluxo entre os atores envolvidos.

Nesse caso, o usuário é o ator a iniciar o fluxo do sistema ao inserir suas credenciais na aplicação. Após feita a autenticação, a aplicação receberá todos os dados dos dispositivos disponíveis para o usuário autenticado e este pode selecionar um dos dispositivos para visualizar os detalhes. Ao selecionar o dispositivo, a aplicação faz uma requisição à extensão criada a qual seleciona os dados relacionados que estão no modelo semântico juntamente com os dados que estão salvos na base de dados do *middleware*. Após isso, os dados são preparados e retornados à aplicação que os mostrará ao usuário. Como descrito nas sessões anteriores, a aplicação apresenta os dados conforme as informações descritas no modelo semântico.

Esse fluxo representa o primeiro acesso ao sistema, ou seja, ele será executado na primeira vez que o usuário buscará visualizar as informações dos dispositivos físicos. Após autenticado, é possível fazer consultas repetidas para o mesmo ou para outros dispositivos sem a necessidade de autenticar-se novamente.

Figura 5.16: Usuário consultando informações dos dispositivos físicos



Fonte: O Autor

5.9 Conclusão

Nesse capítulo foi apresentada uma proposta de arquitetura para integração que possibilite a integração entre componentes industriais através de uma modelagem semântica e do uso de IoT onde é buscado fornecer uma maneira automatizada de integração, dentro do contexto da Indústria 4.0.

A proposta é dividida em camadas que foram pensadas no conceito de IoT. As camadas são as seguintes: Percepção (onde encontram-se os elementos físicos do sistema que serão os fornecedores de dados); Transmissão (representa a maneira e o meio em que esses dados serão transmitidos); Computação (engloba a parte cibernética do sistema, onde se tem o *middleware*, o modelo de dados e os elementos de software que interagem com os dispositivos físicos) e por último a camada de Aplicação (que é responsável por criar um meio de comunicação com os usuários finais). Essa divisão ajuda a se ter um entendimento do sistema, o que é imprescindível para se ter um melhor aproveitamento dele. Mesmo que não exista uma arquitetura padrão e única para IoT, a arquitetura em

quatro camadas é amplamente utilizada na literatura, o que pode ser entendido como um ponto positivo, visto que essa proposta está em sintonia com a maioria dos trabalhos relacionados à essa área.

No decorrer do trabalho, percebeu-se que, em termos arquiteturais, o modelo de dados poderia ser melhor explorado se estiver no mesmo nível dos dados propriamente dito, ou seja, se o modelo de dados estiver junto ao *middleware*, onde os dados são recebidos e disponibilizados, esse modelo poderia ser usado tanto para a geração de interfaces de integração, quanto para aplicações que fazem interface com os usuários. Isso facilitou também manter o modelo atualizado conforme alterações nos dispositivos.

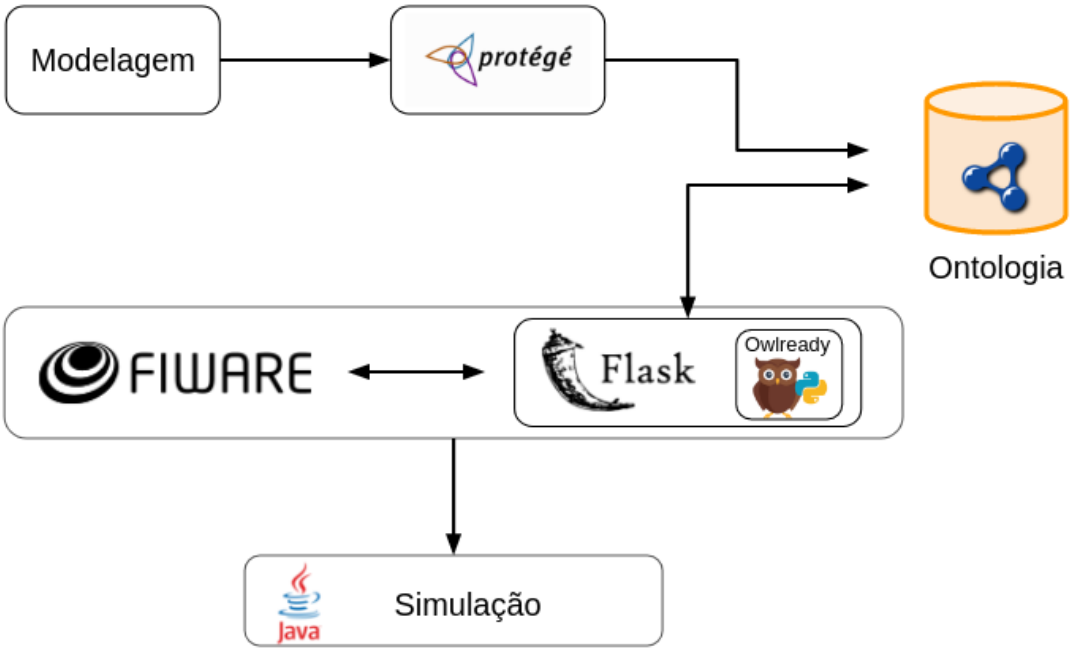
No próximo capítulo será apresentada a implementação da proposta, ou seja, toda a infraestrutura necessária, bem como as tecnologias utilizadas.

6 IMPLEMENTAÇÃO

Este capítulo apresenta a etapa de implementação do sistema de integração de componentes industriais proposto. A implementação foi divida em duas partes. Primeiro, foi definido o modelo semântico baseado em modelos já existentes na literatura. Para a fase de implementação, foi utilizada a ferramenta Protégé para a manipulação do modelo. Posteriormente, foi desenvolvida uma extensão para o *middleware* IoT FIWARE, capaz de trabalhar com esse modelo semântico. Para essa extensão foi utilizado o *framework* Flask (GRINBERG, 2014) juntamente com a biblioteca Owlready2(LAMY, 2017) para a leitura e manipulação do modelo semântico.

A Figura 6.1 ilustra as principais tecnologias utilizadas na implementação da proposta. Inicialmente, tem-se a modelagem da ontologia e posteriormente tem-se as tecnologias que, através do modelo, possibilitam a integração dos componentes. Esses componentes foram simulados com uma aplicação desenvolvida em Java.

Figura 6.1: Tecnologias utilizadas na implementação



Fonte: O Autor

6.1 Implementação da Ontologia

Esta seção apresenta a etapa de implementação da ontologia que compõe a proposta. Foram utilizadas ontologias existentes na literatura, adicionando algumas adapta-

ções para encaixar-se à proposta.

6.1.1 Criação de classes

A hierarquia de classes da ontologia do modelo de usuário foi criada utilizando a interface gráfica fornecida pelo *software* Protégé. Após todas as informações pertinentes às classes serem inseridas, a ferramenta gera o código OWL de maneira automática, como pode ser observado no código que define a classe "Valve", que é sub classe de "IoT", a seguir:

```
<owl:Class rdf:about="http://www.semanticweb.org/charles/ontologies/2017/0/untitled-ontology-5#Valve">
    <rdfs:subClassOf rdf:resource="http://www.semanticweb.org/charles/ontologies/2017/0/untitled-ontology-5#Device"/>
    <rdfs:subClassOf rdf:resource="http://www.semanticweb.org/charles/ontologies/2017/0/untitled-ontology-5#IoT"/>
</owl:Class>
```

6.1.2 Criação de indivíduos

A ferramenta Protégé permite também a criação de indivíduos e propriedades. Como explicado na proposta, Capítulo 4, os indivíduos que são instâncias da classe "IoT" são representações de objetos do mundo real, como um dispositivo físico. A seguir é apresentado um trecho de código que representa um dispositivo válvula.

```
<owl:NamedIndividual rdf:about="http://www.semanticweb.org/charles/ontologies/2017/0/untitled-ontology-5#valve1">
    <rdf:type rdf:resource="http://www.semanticweb.org/charles/ontologies/2017/0/untitled-ontology-5#Valve"/>
    <id rdf:datatype="http://www.w3.org/2001/XMLSchema#string"></id>
</owl:NamedIndividual>
```

6.2 Implementação da arquitetura baseada no *middleware* IoT

O conceito de IoT está em constante crescimento e evolução, entretanto muitas de suas vantagens já são conhecidas e ele tem sido aplicado em diversos projetos de domínios variados. Para este trabalho, utilizou-se o *middleware* IoT FIWARE, o qual foi instalado e configurado em um servidor para que estivesse acessível aos demais componentes do sistema. As subseções abaixo descrevem o processo de implementação da parte responsável pela integração do sistema.

6.2.1 Preparação do ambiente de trabalho

Para a implementação desta proposta, foi necessário criar-se um ambiente em que fosse possível instalar as ferramentas necessárias, bem como adicionar as implementações provenientes da proposta.

Para isso, utilizou-se um computador que serviu de servidor, com o sistema operacional CentOS 6.7 pelo fato de o *middleware* FIWARE ter sido desenvolvido para funcionar nesse sistema. Outro requisito para instalação do FIWARE é que se tenha instalado o banco de dados MongoDB. Este é um banco de dados NoSQL de código aberto base escrito em C++.

Afim de reproduzir a proposta deste trabalho, desenvolveu-se uma extensão do FIWARE que fosse capaz de trabalhar com modelos semânticos, fornecendo benefícios já citados na proposta. Para isso, optou-se por utilizar o *framework* Flask, que é implementado na linguagem de programação Python e fornece uma forma de se criar meios de comunicação via uma API REST com poucas linhas de código. Assim, considera-se que o desenvolvimento desse tipo de aplicações, com Flask, é de baixa complexidade e alta agilidade.

6.2.2 Instalação do *middleware* FIWARE

A instalação do FIWARE pode ser feita por duas maneiras, as quais estão descritas na documentação da plataforma. Um possibilidade é baixar o código fonte em um computador, compilá-lo e instalá-lo manualmente. A outra opção, e mais prática, é adicionar o repositório do FIWARE nos repositórios do sistema operacional que se está usando e então instalar via o gerenciador de pacotes do próprio Linux. Abaixo pode ser observado como instalar essa plataforma em um sistema operacional CentOS.

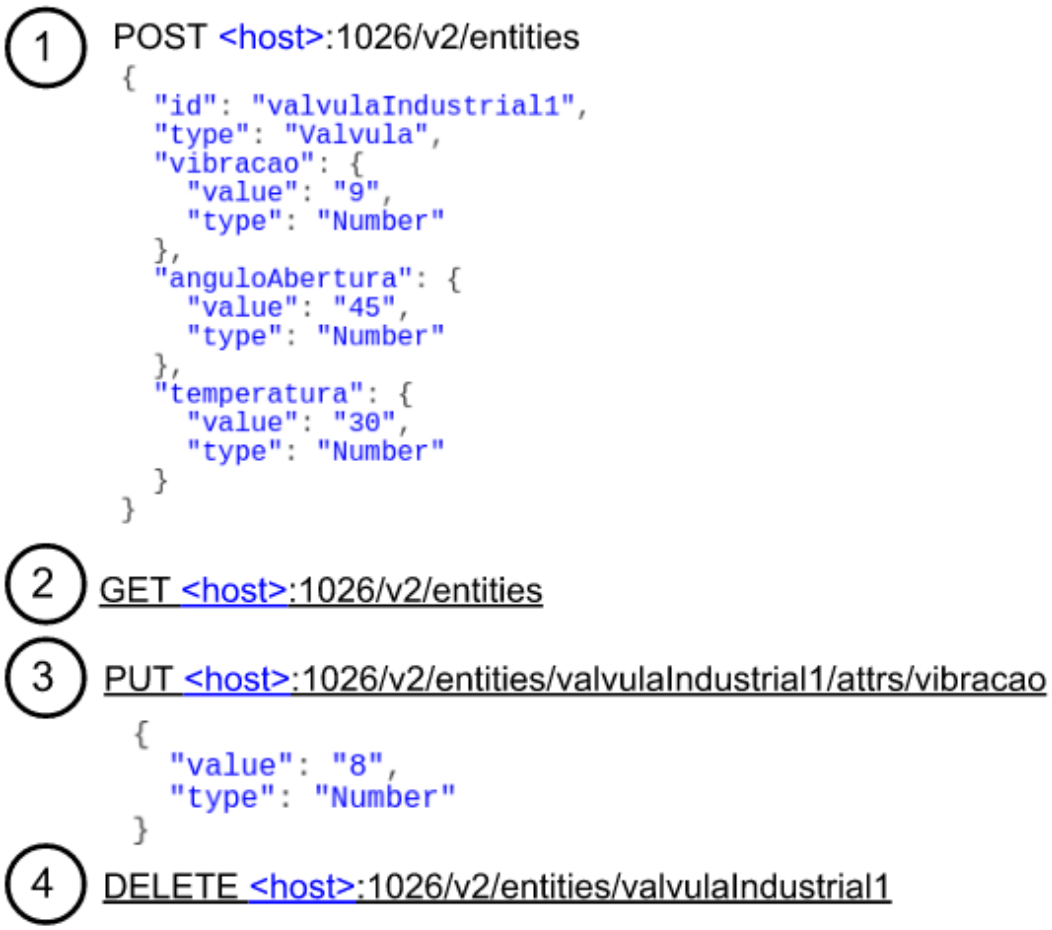
```
yum install contextBroker
```

Após o sistema estiver instalado, é importante configurar as credenciais de acesso para evitar que o acesso fique aberto para clientes não autorizados.

A Figura 6.2 apresenta as formas de acesso ao *middleware* FIWARE em funcio-

namento. No exemplo, existe um elemento chamado “valvulaIndustrial1”, que representa uma entidade física, que pode ser encontrada em sistemas de automação industrial.

Figura 6.2: Métodos de acesso ao FIWARE



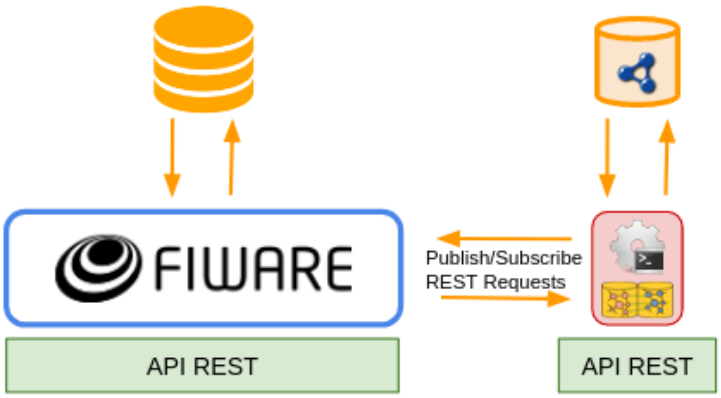
Fonte: O Autor

Como pode ser observado, existem quatro ações básicas que podem ser executadas sobre as entidades. Em (1) é apresentada a criação da entidade, juntamente com seus atributos e um identificador. A consulta a essa entidade é representada em (2). Para alterar um atributo de uma entidade, utiliza-se o método PUT e envia-se as informações a serem atualizadas, como ilustrado em (3). Ainda, existe a possibilidade de deletar uma entidade (4), caso essa não tenha mais interesse ao sistema ou não exista mais.

6.2.3 Implementação da extensão do *middleware*

Foi implementada uma extensão para o *middleware* FIWARE a fim de possibilitar que se trabalhe com as ontologias. Assim, essa extensão foi criada utilizando o *framework* Flask, pois este oferece métodos que facilitam a criação de interfaces *web* no padrão REST. A Figura 6.3 apresenta os fluxos de comunicação entre a extensão e o FIWARE.

Figura 6.3: Fluxo de comunicação entre a extensão e o FIWARE



Fonte: O autor

A extensão está sendo executada no mesmo servidor em que o *middleware* está instalado, porém em uma porta distinta. O FIWARE trabalha com o paradigma *Publish/Subscribe*, o que facilita a criação de novas funcionalidades que precisam ser atualizadas a cada vez que ocorre uma alteração no sistema, como por exemplo, a adição de um novo dispositivo físico na planta industrial, o que acarreta a criação de uma nova entidade no *middleware* e, consequentemente, um novo indivíduo na ontologia.

Assim, foi possível manter os modelos atualizados a cada modificação na camada física, que representa os dispositivos reais.

A extensão também possibilita a geração de interfaces, na linguagem Java, que servem para serem implementadas pelos dispositivos. Essas interfaces proporcionam uma integração automatizada, visto que são gerados métodos que fazem o acesso às APIs do *middleware*.

6.2.4 Manipulação da ontologia

Após finalizar o modelo com a ferramenta Protégé, foi necessário utilizar uma biblioteca para a manipulação da ontologia via programação, ou seja, a atualização auto-

mática do modelo foi implementada utilizando a biblioteca Owlready2 (LAMY, 2017). Essa biblioteca traz métodos que facilitam a manipulação desse tipo de modelos como pode ser observado na Figura 6.4.

Figura 6.4: Utilizando a biblioteca Owlready2

```
from owlready2 import *

onto = get_ontology("file:///home/charles/diss/diss.owl").load()

cl = onto.search(iri="http://www.semanticweb.org/charles/ontologies/diss/diss#Valve")

instance = cl("MyID")

instance.is_a
[diss.Valve]
```

Fonte: O autor

Apesar de essa biblioteca suprir as necessidades desta proposta, ela apresentou problemas com modelos muito grandes, pois os modelos são carregados inteiramente na memória, limitando a escalabilidade do sistema.

6.3 Interfaces criadas para acesso ao *middleware* FIWARE através da extensão criada

Para a comunicação foi necessário criar novas interfaces no servidor a fim de prover meios de acessar as informações contidas no modelo semântico.

O acesso ao *middleware* é realizado através do protocolo HTTP/REST, o que permite o acesso à informação e à execução de comandos em componentes, por meio da semântica direta de acesso ao serviço.

As interfaces criadas e suas descrições podem ser observadas na Tabela 6.1.

Todos os dados que trafegam no sistema estão formatados no formato *JavaScript Object Notation* (JSON) que é um formato de texto para serialização de dados estruturados (CROCKFORD, 2006).

6.4 Desenvolvimento de Interface Humano-Computador

Existem diversas formas de criar-se interfaces de sistemas para usuários. Atualmente, o uso de *smartphones* vem se popularizando em diversos contextos, pois traz uma capacidade de processamento comparadas à de um computador e a portabilidade de um celular móvel.

Tabela 6.1: Interfaces da API REST da extensão criada

Requisição	*URL*	*Descrição*
POST	/auth	Autenticação no sistema. É retornado um *token* para futuras requisições.
POST	/sparql	Executa uma consulta na ontologia.
POST	/devices	Cria uma nova instância de um dispositivo.
POST	/supply/<deviceid>	Interface para receber resposta do sistema de gerenciamento de cadeias de suprimentos.
GET	/devices	Listagem de todos os dispositivos
GET	/devices/<id>	Informações de um dispositivo específico
DELETE	/devices/<id>	Remove um indivíduo do sistema.

Fonte: O Autor

Esses dispositivos ainda possibilitam a criação de aplicativos completos, como por exemplo com acesso à internet ou com realidade aumentada, fazendo uso da câmera.

Entre os dispositivos disponíveis no mercado, uma das plataformas mais populares é o Android (GANDHEWAR; SHEIKH, 2010). O Android é um sistema operacional aberto para dispositivos móveis que possui um conjunto de aplicações que facilitam a sua utilização por parte dos usuários.

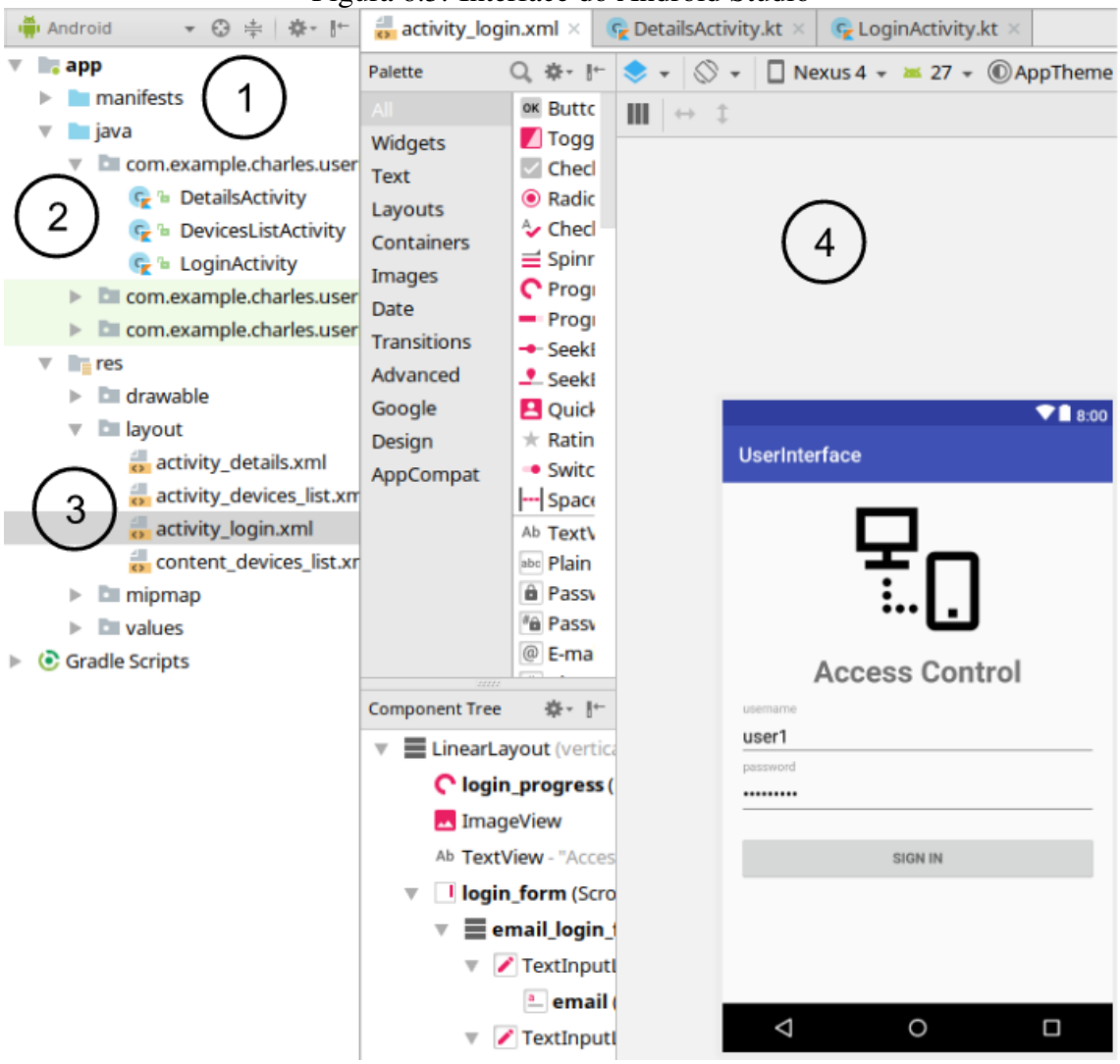
Assim, implementou-se um aplicativo para Android a fim de criar um meio de interação com o usuário final. O aplicativo foi desenvolvido de forma genérica para visualização de informações, ou seja, a interface do usuário era gerada conforme as informações contidas no modelo semânticos e que eram disponibilizadas pelo *middleware*.

Para o desenvolvimento do aplicativo, foi utilizada a IDE (*Integrated Development Environment*) Android Studio (SMYTH, 2017), que é a ferramenta oficial para desenvolvimento de programas para essa plataforma e que está disponível gratuitamente na Internet.

A Figura 6.5 apresenta a tela inicial do Android Studio. Pode-se observar que é uma ferramenta completa que disponibiliza diversos recursos para o desenvolvimento.

No desenvolvimento de aplicativos nativos para Android, existem duas principais partes. Em (1) encontra-se o arquivo de manifesto *AndroidManifest.xml* onde estão des-

Figura 6.5: Interface do Android Studio



Fonte: O Autor

critas as informações essenciais sobre o aplicativo e que são necessárias para o sistema executar os aplicativos. O código que contém a lógica de programação é encontrado em (2) e os arquivos que descrevem os *layouts* da aplicação encontram-se em (3). Geralmente esses dois elementos trabalham em conjunto, onde os arquivos de *layout* descrevem os elementos que serão exibidos na tela e os códigos de programação manipulam esses elementos conforme os eventos que acontecem na aplicação. Em (4) é possível visualizar de forma gráfica a construção de um layout de uma aplicação Android.

7 ESTUDOS DE CASO

A fim de aplicar os conceitos e validar a proposta, foram desenvolvidos dois estudos de caso envolvendo componentes industriais. Os estudos de caso estão relacionados ao uso de atuadores elétricos de válvulas no contexto de sistemas de transporte de óleo.

O primeiro estudo de caso é baseado em atuador e uma válvula industrial e envolve a visualização das informações do sistema por parte dos diferentes tipos de usuários.

O segundo estudo de caso aborda a integração com sistemas externos, como sistema de cadeia de suprimentos. Ainda, é feita uma análise de desempenho da estrutura da desenvolvida na proposta.

Por fim, ainda são apresentados resultados de testes de stress da proposta, através de gráficos que mostram dados para serem analisados e se avaliar até que ponto a proposta pode ser aplicada em sistemas reais.

7.1 Estudo de caso 1: Válvulas Industriais

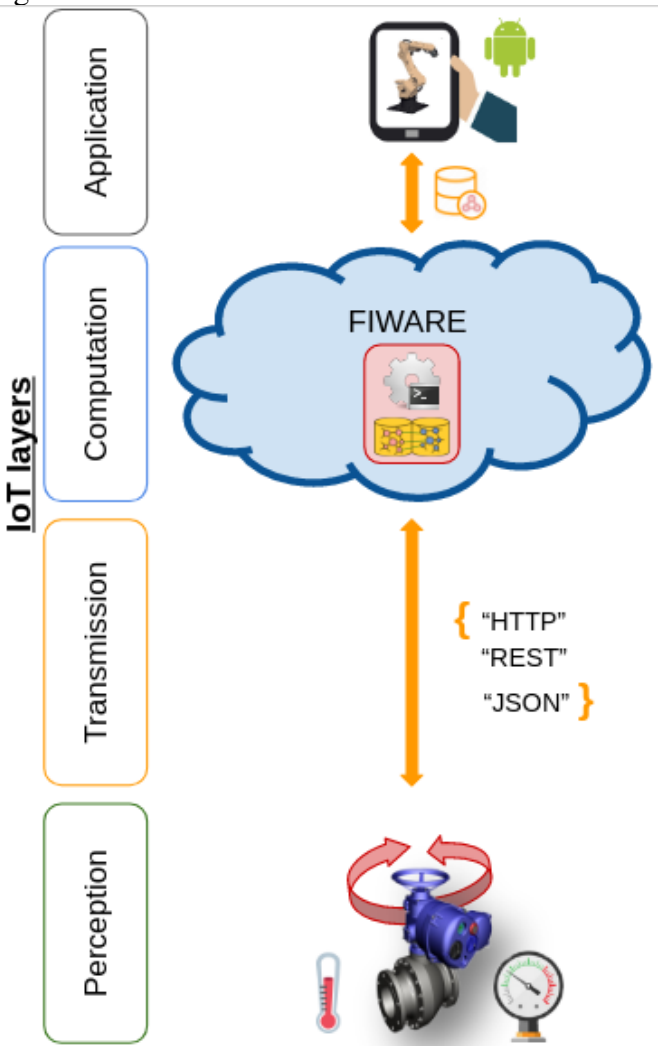
A Figura 7.1 apresenta a estrutura do primeiro estudo de caso. A ideia é que se tenham componentes físicos, no caso uma válvula industrial, com sensores que monitoram seu estado e esses dados são transmitidos ao *middleware* FIWARE que recebe esses dados, os salva e ainda os disponibiliza para as aplicações. Junto ao *middleware*, na camada de Computação, está a extensão desenvolvida nessa proposta, que serve de intermediário com o modelo semântico. Essa extensão, além de manter o modelo atualizado, disponibiliza os dados com informações semânticas para os aplicativos.

O estudo de caso segue a arquitetura proposta neste trabalho. Portanto, a primeira etapa é identificar os componentes físicos do sistema, que pertencem à camada de percepção. Esses componentes são os produtores de dados do sistema.

Para isso, utilizou-se uma válvula industrial de uma empresa brasileira. A válvula pode abrir e fechar, o que permite que líquidos possam passar ou não por ela. Ela possui sensores para monitorar a sua saúde, como por exemplo, sensor de temperatura, sensor de pressão e ainda um sensor de torque, que mede esse atributo quando a válvula abre ou fecha. Esse valor é importante para prever possíveis problemas com o equipamento.

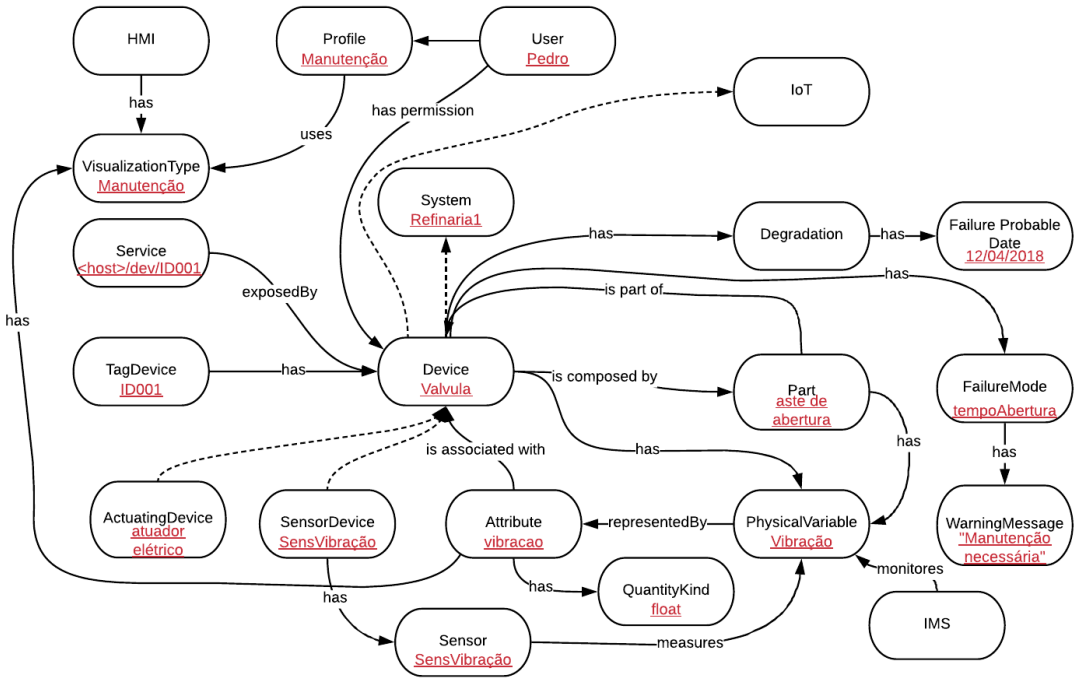
Foi criado um modelo para representar essa válvula, como pode ser observado na Figura 7.2. Pode-se observar que foram utilizadas ontologias já disponibilizadas na literatura, como a IoT-Lite (BERMUDEZ-EDO et al., 2016) e a ontologia de integração

Figura 7.1: Estudo de caso com válvulas industriais



Fonte: O Autor

Figura 7.2: Modelo semântico utilizado no estudo de caso



Fonte: O Autor

entre IMS e SPSC (SILVA; PEREIRA, 2014). As classes são representadas em preto e os indivíduos são representados em vermelho. As classes pertencentes à IHC referem-se a questões de interface com o usuário. No exemplo, criou-se uma forma de representar que o atributo "vibração" do dispositivo "Valvula" deve ser mostrado em forma de gráfico (chart) nas aplicações utilizadas pelos usuários.

Uma das contribuições deste trabalho é a possibilidade de gerar interfaces automaticamente para dispositivos e registrá-los no *middleware* de forma automática também. Isso facilita no desenvolvimento da integração além de evitar falhas humanas.

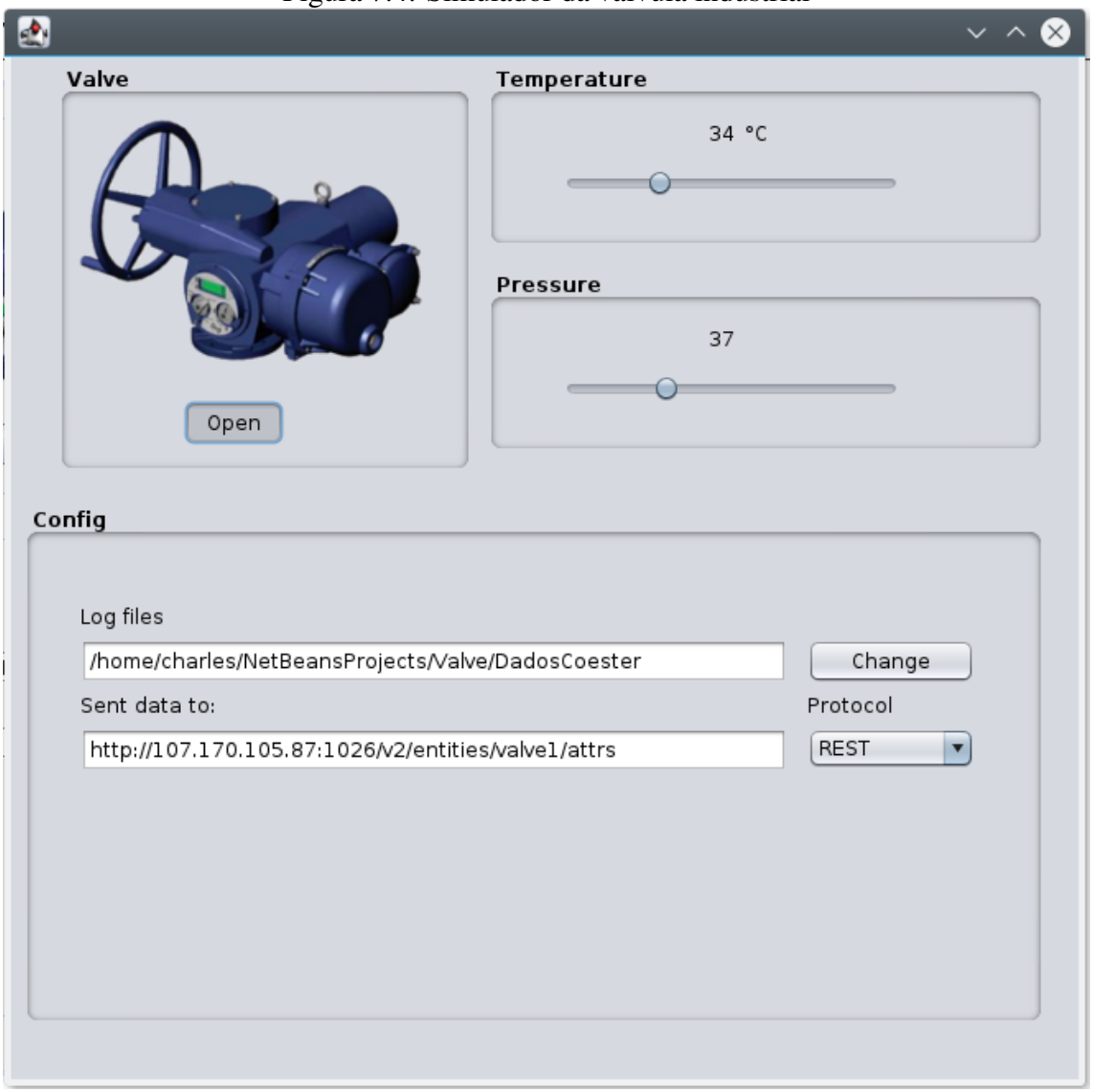
A ideia desenvolvida para essa função é extrair todos os indivíduos da ontologia que são "filhos" da classe "IoT", isso é um indicativo de que esses indivíduos representam um dispositivo físico do sistema. Após identificados esses indivíduos, as propriedades são coletadas e usadas para gerar as interfaces e ainda cadastrar esse elemento no *middleware* IoT.

O arquivo de modelo que é gerado é em formato XML, o que possibilita a leitura e processamento por *softwares* externos ou de terceiros. Entretanto, existem bibliotecas que facilitam a leitura desse tipo de arquivo. A Figura 7.3 apresenta um trecho de código onde em 1 observa-se um indivíduo "valve1" modelado na ontologia, juntamente com suas propriedades. Em 2 está o código que poderá ser utilizado para implementação dos dispositivos. Por fim, em 3 é apresentado o indivíduo registrado ao *middleware* FIWARE.

Figura 7.3: Trecho do arquivo XML com o indivíduo "valve1"e as interfaces geradas para o dispositivo e para o *middleware*

Fonte: O Autor

Figura 7.4: Simulador da válvula industrial



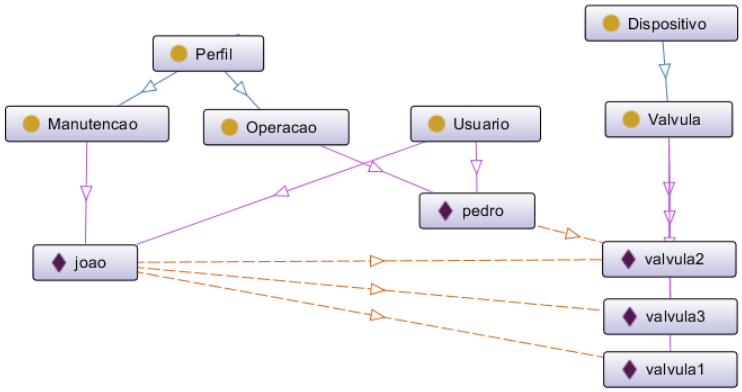
Fonte: O Autor

Os testes do estudo de caso foram executados utilizando dados capturados de uma válvula elétrica modelo CS06 da empresa Coester, o que possibilitou trazer essa validação mais próxima à um cenário real. Foi desenvolvido um simulador da válvula para que os testes pudessem ser executados diversas vezes. A Figura 7.4 apresenta o simulador desenvolvido e usado no estudo de caso.

Por fim, após os dados estarem salvos na nuvem, é possível consumir esses dados através de aplicativos que servem de interface para os usuários finais. Para esse fim, foi desenvolvido um aplicativo para a plataforma Android em que é possível consultar as informações referentes ao dispositivo físico, no caso, a válvula industrial.

O aplicativo possui uma tela de login onde o usuário entra com as credenciais e é autenticado pelo sistema. Após o usuário autenticado, o sistema identifica seu perfil e

Figura 7.5: Modelo com diferentes acessos às informações



Fonte: O Autor

apresenta as informações disponíveis para esse perfil de usuário. Essas informações são descritas no modelo semântico, como pode ser observado na Figura 7.5.

Percebe-se que existem dois tipos de usuários, um de Operação e outro de Manutenção. O usuário de manutenção está vinculado às três válvulas do modelo enquanto o de operação está vinculado apenas a uma delas. Essa vinculação significa que o usuário terá acesse às informações desses dispositivos. Essa relação foi feita através da vinculação de propriedades desses indivíduos.

A Figura 7.6 apresenta o aplicativo desenvolvido com um fluxo simples de login e da tela de detalhes do dispositivo industrial.

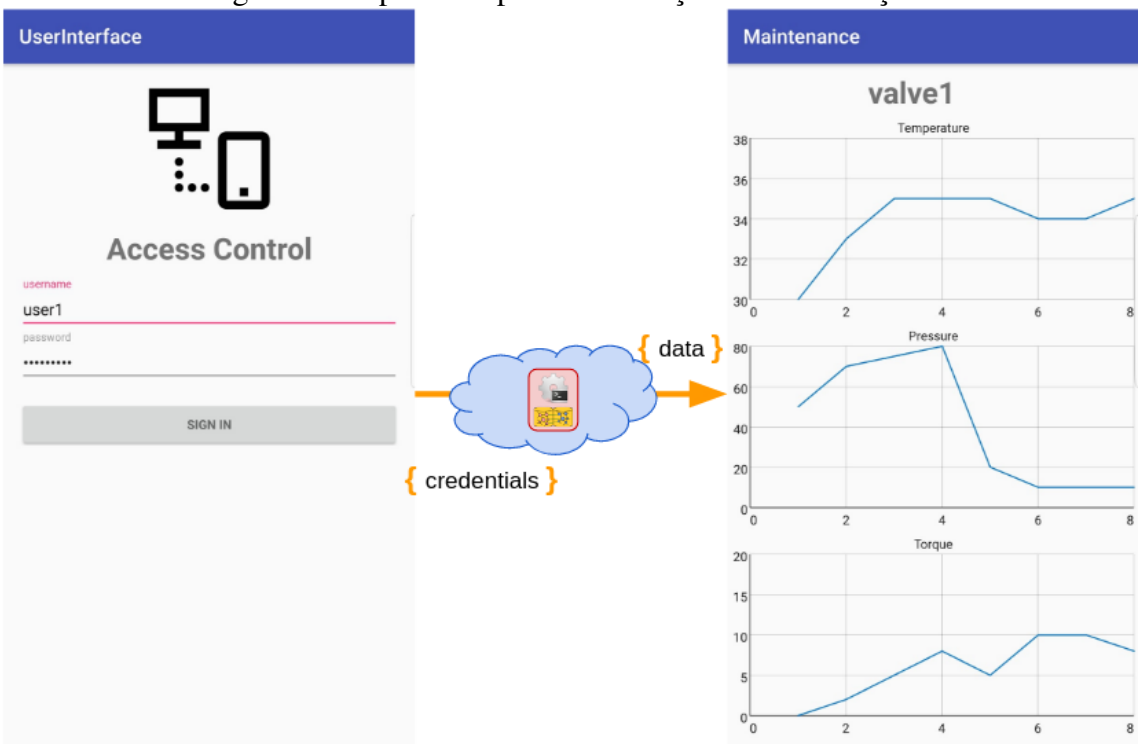
Um diferencial dessa proposta é justamente a possibilidade de vincular informações semânticas aos dados, por exemplo, foi modelado no modelo semântico que as informações devem ser mostradas em forma de gráfico para o usuário com perfil de "Manutenção", pois para esse tipo de usuário pode ser relevante ver um histórico dessas variáveis a fim de identificar anomalias.

Com este estudo de caso é possível verificar que a proposta é aplicável desde os dispositivos físicos até as aplicações de alto nível. A utilização do *middleware* FIWARE possibilita que a proposta seja aplicada em projetos já em andamento sem uma reimplementação de todo o sistema.

7.2 Estudo de caso 2: Integração entre IMS e SPSC

O segundo estudo de caso é importante para mostrar que a proposta possibilita a integração com sistemas de cadeias de suprimentos. Como visto em (ESPÍNDOLA et al., 2012), a integração entre IMS e SPSC traz benefícios para ambos os domínios, onde, por

Figura 7.6: Aplicativo para visualização das informações

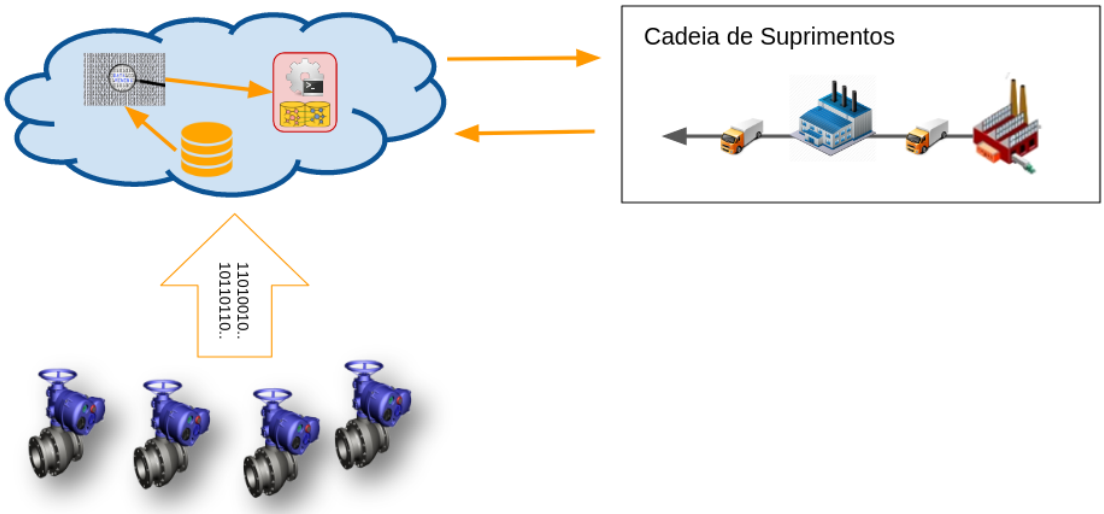


Fonte: O Autor

um lado pretende-se estar apto a estimar as necessidades de manutenção com antecedência, evitando interrupções na produção. Por outro lado, os sistemas de planejamento nas cadeias de suprimentos podem tomar decisões mais precisas com base nas informações recebidas do IMS.

A Figura 7.7 apresenta o estudo de caso onde é constatada uma possibilidade de comunicação entre dois domínios diferentes utilizando a proposta do presente trabalho.

Figura 7.7: Estudo de caso 2: Integração entre IMS e SPSC



Fonte: O Autor

A principal ideia desse estudo de caso é que, a partir de dados recebidos dos dispositivos físicos, o *middleware* armazene esses dados e algoritmos de análise de dados processem e gerem informações que possam auxiliar na previsão de falhas. A extensão proposta neste trabalho monitora o "Degradation Forecast", ontologia de (SILVA; PEREIRA, 2014), que possui a data provável de uma falha, que é baseada no nível de degradação da peça. Assim que uma data de possível falha é atribuída a uma peça, o sistema dispara uma mensagem ao sistema de gerenciamento de cadeia de suprimentos o qual recebe essas informações e as apresenta ao usuário.

A regra que dispara um alarme é descrita na Figura 7.8. Essa regra verifica o tempo de abertura de um dispositivo. Caso esse tempo passe de 10 segundos, é atribuída uma mensagem de alerta a esse dispositivo que é mostrada para o operador de manutenção. O dispositivo também é classificado como "DispositivoComProblema" através de uma classe com esse nome.

Figura 7.8: Regra que monitora o tempo de abertura da válvula

```
Valvula(?v) ^ temTempoAbertura(?v, ?t) ^ swrlb:greaterThan(?t, 10) ^ Mensagem(?msg) ^
msgTempAbertura(?msg, ?m) ^ Alerta(?a) -> temMensagem(?a, ?m) ^ DispositivoComProblema(?v)
```

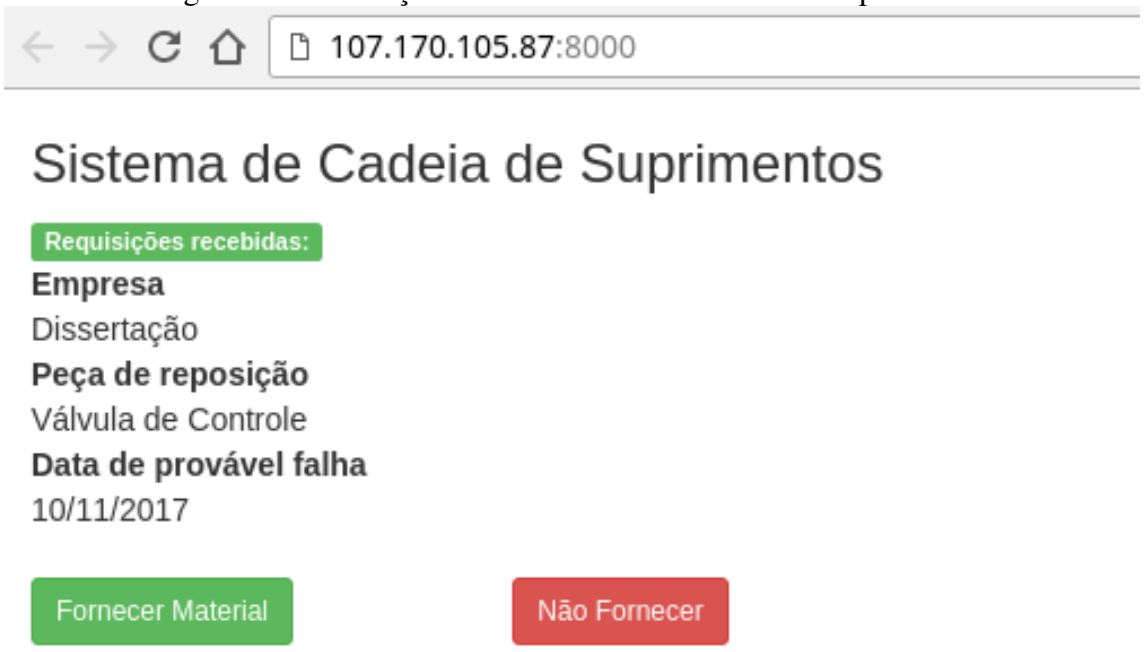
Fonte: O Autor

Ao encontrar dispositivos com problema, o sistema é capaz de fazer uma requisição a sistemas externos, como por exemplo, para sistemas de reposição.

Para fins de simulação de um sistema de gerenciamento de cadeia de suprimentos, desenvolveu-se um sistema *web*, também baseado no *framework* Flask, que é capaz de receber requisições REST e apresentá-las na tela para o usuário. Esse sistema lista as requisições recebidas e possibilita que os usuários possam tomar duas ações: "Fornecer Material", que significa que seria fornecido o material de reposição para a manutenção ser efetuada, ou o usuário poderia optar por "Não Fornecer", o que como o nome já diz, o material não poderia ser fornecido por essa empresa. A Figura 7.9 apresenta a interface gráfica desenvolvida para esse sistema.

É possível observar que existe uma requisição que foi recebida pelo sistema, com algumas informações referentes ao componente industrial que está apresentando indícios que irá estragar. Para este estudo de caso, as informações fornecidas na hora da requisição à cadeia de suprimentos foram: "Empresa", que é o nome da empresa ou indústria que está solicitando uma peça de reposição; "Peça de reposição", que representa a peça em si que será substituída, e ainda a "Data de provável falha", que indica uma previsão de possível falha. Essa informação é discutida em outros trabalhos dentro do escopo do

Figura 7.9: Simulação de um sistema de Cadeias de Suprimentos



Fonte: O Autor

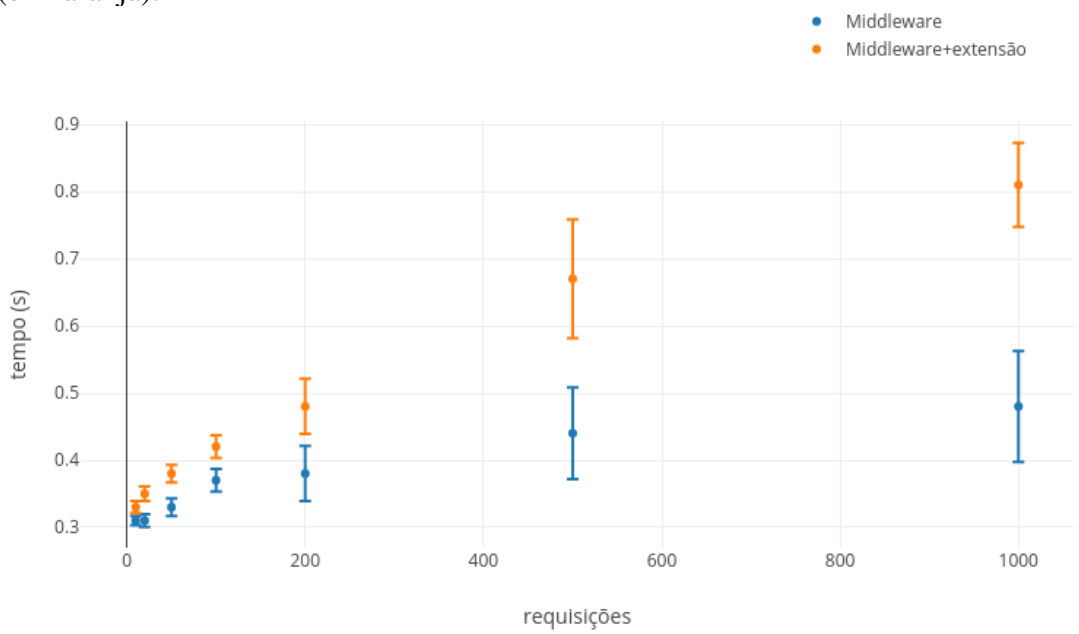
Projeto I2MS2C, porém, para este estudo de caso, foi feita uma simulação de uma data aleatória.

Quando o usuário clicar em uma das duas possíveis opções, o sistema envia uma mensagem para o *middleware*, onde essa informação pode ser processada e trabalhada para o planejamento da manutenção.

7.3 Avaliação de desempenho

A fim de verificar o impacto que a presente proposta tem no desempenho do sistema IoT, foram executados testes com diferentes números de dispositivos (simulados) enviando e requisitando informações do servidor. O testes foram realizados da seguinte maneira: primeiro, colocou-se um número X de dispositivos transmitindo informações ao servidor em paralelo em um intervalo aleatório de 10 a 100 milissegundos. Posteriormente, criou-se um novo dispositivo que transmitia informações por 50 vezes e mediu-se o tempo gasto a cada requisição. Após ter-se o tempo de todas as requisições, calculou-se o tempo médio gasto para cada número X de dispositivos transmitindo em paralelo. Nesse caso, X varia de 10 até 1000.

Figura 7.10: Gráfico comparando medida de tempo de requisições apenas com o *middleware* FIWARE (em azul) versus o *middleware* jutamente com a extensão desenvolvida (em laranja).



Fonte: O Autor

Percebe-se que o desvio padrão do tempo de resposta com 10 requisições é menor do que com 200 a 1000 requisições. Isso indica que conforme o sistema cresce, a plataforma pode variar no seu tempo de resposta. Para esse problema, acredita-se que seja necessário trabalhar o paralelismo da plataforma utilizada.

Comparando o tempo médio de resposta apenas com *middleware* e com a extensão implementada, percebe-se que a extensão gera um atraso médio de 0.1 segundo em relação ao uso de apenas o *middleware*.

Acredita-se que esse aumento no tempo de resposta está vinculado à biblioteca utilizada para a manipulação da ontologia, pois esse modelo é carregado integralmente na memória a cada vez que é feita uma requisição.

Uma possível solução para esse problema é criar camadas na ontologia, ou até mesmo dividi-la em partes para que informações desnecessárias não sejam carregadas a cada requisição. Por exemplo, pode-se criar um arquivo de ontologia apenas para representar as válvulas de uma planta industrial e esse arquivo é pertencente a um modelo maior que engloba toda uma planta. Nessa caso, cria-se uma estratégia para identificar qual arquivo de ontologia deve ser carregado a cada requisição feita ao servidor.

7.4 Discussão

Este capítulo apresentou dois estudos de caso que exploram as contribuições descritas na proposta. O primeiro estudo de caso é focado em um Sistemas de Manutenção Inteligente, onde dispositivos físicos podem ser modelados em uma ontologia e esse modelo poderá ser usado para gerar interfaces de integração automaticamente. Já o segundo estudo de caso apresenta a possibilidade de integração com sistemas de gerenciamento de cadeias de suprimentos.

Ambos estudos de caso mostram que a proposta pode ser aplicada em sistemas desse domínio e é possível concluir que essa pode ser uma opção viável para tornar o processo de integração mais fácil e mais compreensível a todos os usuários.

Os testes de desempenho do sistema mostram que a proposta apresenta uma fragilidade quanto a questão de escalabilidade, pois a biblioteca utilizada para leitura do modelo semântico carrega toda a ontologia na memória a cada acesso. Isso tornou-se uma limitação do sistema. Esse ponto também abre uma discussão se ontologias são um modelo apropriado para sistemas com muito fluxo de dados.

O próximo capítulo apresenta as conclusões relacionadas ao trabalho e ainda discute sobre possíveis trabalhos futuros.

8 CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho propôs uma abordagem que possibilite a integração entre componentes industriais através de uma modelagem semântica e do uso de IoT onde o objetivo principal é fornecer uma maneira automatizada de integração, no contexto da Indústria 4.0. Essa abordagem está fundamentada em conceitos atuais e que estão em crescimento, tanto na pesquisa como na indústria. Para possibilitar essa integração, foi proposta uma arquitetura com componentes chaves que envolvem um sistema de integração. Os principais elementos envolvidos são o domínio de conhecimento (que contém as informações necessárias para a construção do sistema), o sistema de automação (que é o domínio onde esta proposta foi aplicada), o *middleware* IoT juntamente com a extensão desenvolvida neste trabalho (que é responsável pela integração) e, por fim, a aplicação (que faz a comunicação com os usuários ou sistemas externos).

A abordagem utilizada neste trabalho segue a lógica de trabalhos relacionados já existentes na literatura, entretanto, existem características inovadoras que trazem algumas vantagens na criação de sistemas IoT, como por exemplo, a integração automatizada a partir de modelos semânticos, que possibilita que usuários modelem seu sistemas em alto nível, com o nível de detalhes necessário para cada projeto. A partir desse modelo, as interfaces de comunicação são criadas automaticamente, trazendo uma garantia de consistência semântica. Outro ponto que este trabalho traz é a possibilidade de usar esse mesmo modelo semântico para apresentar as informações ao usuário, ou seja, as aplicações que fazem interface com os usuários se adaptam conforme o modelo que possui informações referentes a cada informação.

A validação foi feita através da implementação de dois estudos de caso que mostram que é possível aplicar a proposta nesse contexto da Indústria 4.0 e que modelos semânticos trazem uma abstração que facilita o entendimento das informações por parte de usuários. Ainda, foram executados testes de estresse para identificar possíveis limitações da proposta. Esses testes mostraram que a presente proposta suporta pelo menos mil requisições simultâneas, ou seja, o servidor continuou respondendo às requisições, tornando-se viável para aplicações em sistemas desse tamanho.

Pode-se perceber que já existem diversos trabalhos que buscam contribuir no domínio de integração de sistemas, inclusive, utilizando conceitos que a presente proposta aborda. Entretanto, observa-se que é uma área que necessita de pesquisa, pois apesar de existirem diversas abordagens, não existe uma que seja absoluta, uma vez que trata-se de

um problema novo e em constante evolução. As abordagens encontradas apresentam soluções com o uso de *middlewares* e modelos semânticos, assim como a presente proposta, porém existem poucas soluções de como essa integração pode ser automatizada e ainda como esses modelos podem ser usados desde a representação de objetos físicos até os aplicativos que fazem interface para os usuários finais. Nesse sentido, o presente trabalho busca contribuir exatamente em apresentar uma abordagem de integração no contexto da Indústria 4.0 utilizando modelo semânticos e conceito de IoT, bem como prover uma forma automática de geração de interfaces para que essa integração possa ocorrer.

Portanto, acredita-se que as contribuições deste trabalho podem diminuir o esforço de implantação de conceitos como IoT dentro do contexto industrial, levando indústrias com métodos tradicionais a usarem tecnologias atuais e tornando-as ainda mais competitivas no mercado.

O uso de conceitos e tecnologias atuais abre diversas oportunidades para trabalhos futuros, como por exemplo, podem ser observadas:

- Adicionar o conceito de *Digital Twin* à arquitetura proposta. Será necessário fazer alterações no modelo semântico para adicionar conceitos relacionados a esse domínio. Pretende-se também criar um mecanismo de simulação para os objetos "*twins*" bem como para simular variáveis externas ao sistema.
- Executar testes com outros *middlewares* para verificar se o uso de outras tecnologias pode interferir no desempenho e na escalabilidade do sistema.
- O modelo semântico traz diversos benefícios para o desenvolvimento desse tipo de sistemas, porém existe uma limitação de escalabilidade. Pode-se criar estratégias para processar o modelo em paralelo, alocando um processador para cada conjunto de informações. Outra estratégia é trabalhar com camadas de abstração e apenas carregar para a memória as camadas que estão em uso.
- Implementar a proposta em outros domínios, como por exemplo na agricultura de precisão, que está em crescente evolução no uso de tecnologias. Pode-se modelas sensores e máquinas e possibilitar a integração desses dispositivos através da presente proposta.

REFERÊNCIAS

AL-FUQAHA, A. et al. Internet of things: A survey on enabling technologies, protocols, and applications. **IEEE Communications Surveys & Tutorials**, IEEE, v. 17, n. 4, p. 2347–2376, 2015.

ALAM, S.; CHOWDHURY, M. M.; NOLL, J. SenaaS: An event-driven sensor virtualization approach for internet of things cloud. **2010 IEEE International Conference on Networked Embedded Systems for Enterprise Applications, NESEA 2010**, p. 1–6, 2010. ISSN 1089-7798.

ANTONIOU, G.; HARMELEN, F. V. Web ontology language: Owl. In: **Handbook on ontologies**. [S.l.]: Springer, 2004. p. 67–92.

ASHTON, K. That 'internet of things' thing. **RFiD Journal**, v. 22, n. 7, p. 97–114, 2009.

ATZORI, L.; IERA, A.; MORABITO, G. The internet of things: A survey. **Computer Networks**, v. 54, n. 15, p. 2787 – 2805, 2010. ISSN 1389-1286. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S1389128610001568>.

ATZORI, L.; IERA, A.; MORABITO, G. Understanding the internet of things: definition, potentials, and societal role of a fast evolving paradigm. **Ad Hoc Networks**, Elsevier, v. 56, p. 122–140, 2017.

ATZORI, L. et al. The social internet of things (siot)–when social networks meet the internet of things: Concept, architecture and network characterization. **Computer networks**, Elsevier, v. 56, n. 16, p. 3594–3608, 2012.

BANDYOPADHYAY, D.; SEN, J. Internet of things: Applications and challenges in technology and standardization. **Wireless Personal Communications**, Springer, v. 58, n. 1, p. 49–69, 2011.

BASSI, L. Industry 4.0: Hope, hype or revolution? In: IEEE. **Research and Technologies for Society and Industry (RTSI), 2017 IEEE 3rd International Forum on**. [S.l.], 2017. p. 1–6.

BELLIFEMINE, F.; POGGI, A.; RIMASSA, G. Jade–a fipa-compliant agent framework. In: LONDON. **Proceedings of PAAM**. [S.l.], 1999. v. 99, n. 97-108, p. 33.

BERMUDEZ-EDO, M. et al. Iot-lite: a lightweight semantic model for the internet of things. In: IEEE. **Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld), 2016 Intl IEEE Conferences**. [S.l.], 2016. p. 90–97.

BERNERS-LEE, T.; FISCHETTI, M. **Weaving the Web: The original design and ultimate destiny of the World Wide Web by its inventor**. [S.l.]: DIANE Publishing Company, 2001.

BOYER, S. A. **SCADA: supervisory control and data acquisition**. [S.l.]: International Society of Automation, 2009.

BRAY, T. et al. Extensible markup language (xml). **World Wide Web Journal**, v. 2, n. 4, p. 27–66, 1997.

BUZIN, M. S.; FOURNIER, F.; ARCUSHIN, S. Fispace design and release plan. 2013.

CHANDRASEKARAN, B.; JOSEPHSON, J. R.; BENJAMINS, V. R. What are ontologies, and why do we need them? **IEEE Intelligent Systems and their applications**, IEEE, v. 14, n. 1, p. 20–26, 1999.

COMPTON, M. et al. The ssn ontology of the w3c semantic sensor network incubator group. **Web semantics: science, services and agents on the World Wide Web**, Elsevier, v. 17, p. 25–32, 2012.

CONSORTIUM, W. W. W. et al. **About the World Wide Web Consortium (W3C)**. 2001.

CROCKFORD, D. **The application/json media type for javascript object notation (json)**. [S.l.], 2006. Available from Internet: <https://tools.ietf.org/html/rfc4627>.

DEERING, S. E. **Internet protocol, version 6 (IPv6) specification**. [S.l.], 1998. Available from Internet: <https://tools.ietf.org/html/rfc8200>.

DIEP, D.; ALEXAKOS, C.; WAGNER, T. An ontology-based interoperability framework for distributed manufacturing control. In: IEEE. **Emerging Technologies and Factory Automation, 2007. ETFA. IEEE Conference on**. [S.l.], 2007. p. 855–862.

DJURDJANOVIC, D.; LEE, J.; NI, J. Watchdog agent—an infotronics-based prognostics approach for product performance degradation assessment and prediction. **Advanced Engineering Informatics**, Elsevier, v. 17, n. 3-4, p. 109–125, 2003.

EISENHAUER, M.; ROSENGREN, P.; ANTOLIN, P. Hydra: A development platform for integrating wireless devices and sensors into ambient intelligence systems. In: **The Internet of Things**. [S.l.]: Springer, 2010. p. 367–373.

ENDERT, A. et al. Semantic interaction: Coupling cognition and computation through usable interactive analytics. **IEEE Computer Graphics and Applications**, IEEE, v. 35, n. 4, p. 94–99, 2015.

ESPÍNDOLA, D. et al. Integrating intelligent maintenance systems and spare parts supply chains. **IFAC Proceedings Volumes**, Elsevier, v. 45, n. 6, p. 1017–1022, 2012.

ESPÍNDOLA, D. B. **Uma abordagem baseada em modelo para integração e gerenciamento dos dados de sistemas de manutenção inteligente através do uso de técnicas de realidade mista**. Porto Alegre, RS: UFRGS - Universidade Federal do Rio Grande do Sul, 2011. (Tese de Doutorado).

EVANS, D. The internet of things: How the next evolution of the internet is changing everything. **CISCO white paper**, v. 1, n. 2011, p. 1–11, 2011.

FAN, P.-f.; ZHOU, G.-z. Analysis of the business model innovation of the technology of internet of things in postal logistics. In: IEEE. **Industrial Engineering and Engineering Management (IE&EM), 2011 IEEE 18Th International Conference on**. [S.l.], 2011. p. 532–536.

FARAHZADI, A. et al. Middleware technologies for cloud of things-a survey. **Digital Communications and Networks**, Elsevier B.V., n. January, p. 1–13, 2017. ISSN 2352-8648. Available from Internet: <http://dx.doi.org/10.1016/j.dcan.2017.04.005>.

FIELDING, R. T.; TAYLOR, R. N. Principled design of the modern web architecture. **ACM Transactions on Internet Technology (TOIT)**, ACM, v. 2, n. 2, p. 115–150, 2002.

FIORINI, S. R. et al. A suite of ontologies for robotics and automation [industrial activities]. **IEEE Robotics & Automation Magazine**, IEEE, v. 24, n. 1, p. 8–11, 2017.

FIWARE, P. **Core Platform for the Future Internet. Private Public Partnership Project (PPP)**. 2017. Available from Internet: <https://www.fiware.org>.

FIWARE, P. **Welcome to the FIWARE Wiki**. 2017. Available from Internet: <https://forge.fiware.org/plugins/mediawiki/wiki/fiware/index.php/Main_Page>.

FRANCE, R.; RUMPE, B. Model-driven development of complex software: A research roadmap. In: IEEE COMPUTER SOCIETY. **2007 Future of Software Engineering**. [S.l.], 2007. p. 37–54.

GANDHEWAR, N.; SHEIKH, R. Google android: An emerging software platform for mobile devices. **International Journal on Computer Science and Engineering**, v. 1, n. 1, p. 12–17, 2010.

GIUSTO, D. et al. **The internet of things: 20th Tyrrhenian workshop on digital communications**. [S.l.]: Springer Science & Business Media, 2010.

GÓMEZ-PÉREZ, A.; BENJAMINS, R. Overview of knowledge sharing and reuse components: Ontologies and problem-solving methods. In: **Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI 99) Workshop KRR5: Ontologies and Problem-Solving Methods: Lesson Learned and Future Trends**. IJCAI and the Scandinavian AI Societies. CEUR Workshop Proceedings, 1999. v. 18. Ontology Engineering Group - OEG. Available from Internet: <http://oa.upm.es/6468/>.

GRINBERG, M. **Flask web development: developing web applications with python**. [S.l.]: "O'Reilly Media, Inc.", 2014.

GRUBER, T. R. et al. A translation approach to portable ontology specifications. **Knowledge acquisition**, Academic Press, v. 5, n. 2, p. 199–220, 1993.

GUBBI, J. et al. Internet of things (iot): A vision, architectural elements, and future directions. **Future generation computer systems**, Elsevier, v. 29, n. 7, p. 1645–1660, 2013.

GUINARD, D.; TRIFA, V. Towards the web of things: Web mashups for embedded devices. In: **Workshop on Mashups, Enterprise Mashups and Lightweight Composition on the Web (MEM 2009), in proceedings of WWW (International World Wide Web Conferences), Madrid, Spain**. [S.l.: s.n.], 2009. v. 15.

HENNING, K. **Recommendations for implementing the strategic initiative INDUSTRIE 4.0**. [S.l.], 2013. Available from Internet: <http://thuvienso.dastic.vn:8080/dspace/handle/TTKHCNDaNang_123456789/357>.

HOLLER, J. et al. **From Machine-to-machine to the Internet of Things: Introduction to a New Age of Intelligence**. [S.l.]: Academic Press, 2014.

IBM. **Watson Internet of Things**. 2017. Available from Internet: <https://www.ibm.com/internet-of-things>.

JAZDI, N. Cyber physical systems in the context of industry 4.0. In: IEEE. **Automation, Quality and Testing, Robotics, 2014 IEEE International Conference on**. [S.l.], 2014. p. 1–4.

JIRKOVSKỲ, V.; OBITKO, M. Enabling semantics within industry 4.0. In: SPRINGER. **International Conference on Industrial Applications of Holonic and Multi-Agent Systems**. [S.l.], 2017. p. 39–52.

KATASONOV, A. et al. Smart semantic middleware for the internet of things. **Icinco-Icso**, v. 8, p. 169–178, 2008.

KILJANDER, J. et al. Semantic interoperability architecture for pervasive computing and internet of things. **IEEE access**, IEEE, v. 2, p. 856–873, 2014.

KIM, J.; LEE, J.-W. Openiot: An open service framework for the internet of things. In: IEEE. **Internet of Things (WF-IoT), 2014 IEEE World Forum on**. [S.l.], 2014. p. 89–93.

KORTUEM, G. et al. Smart objects as building blocks for the internet of things. **IEEE Internet Computing**, IEEE, v. 14, n. 1, p. 44–51, 2010.

KRCO, S.; POKRIC, B.; CARREZ, F. Designing iot architecture (s): A european perspective. In: IEEE. **Internet of Things (WF-IoT), 2014 IEEE World Forum on**. [S.l.], 2014. p. 79–84.

LACY, L. W. **OWL: Representing information using the web ontology language**. [S.l.]: Trafford Publishing, 2005.

LAMY, J.-B. Owlready: Ontology-oriented programming in python with automatic classification and high level constructs for biomedical ontologies. **Artificial intelligence in medicine**, Elsevier, v. 80, p. 11–28, 2017.

LASI, H. et al. Industry 4.0. **Business & Information Systems Engineering**, Springer, v. 6, n. 4, p. 239–242, 2014.

LEE, J. E-manufacturing—fundamental, tools, and transformation. **Robotics and Computer-Integrated Manufacturing**, Elsevier, v. 19, n. 6, p. 501–507, 2003.

LEE, J. Smart factory systems. **Informatik-Spektrum**, Springer, v. 38, n. 3, p. 230–235, 2015.

LEE, J.; BAGHERI, B.; KAO, H.-A. A cyber-physical systems architecture for industry 4.0-based manufacturing systems. **Manufacturing Letters**, Elsevier, v. 3, p. 18–23, 2015.

LEE, J. et al. Informatics platform for designing and deploying e-manufacturing systems. In: **Collaborative Design and Planning for Digital Manufacturing**. [S.l.]: Springer, 2009. p. 1–35.

LEE, J. et al. Intelligent prognostics tools and e-maintenance. **Computers in industry**, Elsevier, v. 57, n. 6, p. 476–489, 2006.

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.

LEXINNOVA. **Internet of Things (IoT) Patent Landscape Analysis**. 2015. Available from Internet: <http://www.lex-innova.com/resources-reports/?id=33>.

LIU, M. et al. Intelligent assembly system for mechanical products and key technology based on internet of things. **Journal of Intelligent Manufacturing**, Springer, v. 28, n. 2, p. 271–299, 2017.

LÓPEZ-RIQUELME, J. et al. A software architecture based on fiware cloud for precision agriculture. **Agricultural Water Management**, Elsevier, v. 183, p. 123–135, 2017.

MAINETTI, L.; PATRONO, L.; VILEI, A. Evolution of wireless sensor networks towards the internet of things: A survey. In: IEEE. **Software, Telecommunications and Computer Networks (SoftCOM), 2011 19th International Conference on**. [S.l.], 2011. p. 1–6.

MAYER, S. et al. An open semantic framework for the industrial internet of things. **IEEE Intelligent Systems**, IEEE, v. 32, n. 1, p. 96–101, 2017.

MIRANDA, J. et al. From the internet of things to the internet of people. **IEEE Internet Computing**, IEEE, v. 19, n. 2, p. 40–47, 2015.

MUSEN, M. A. Automated support for building and extending expert models. **Machine learning**, Springer, v. 4, n. 3, p. 347–375, 1989.

NEWCOMER, E.; LOMOW, G. **Understanding SOA with Web services**. [S.l.]: Addison-Wesley, 2005.

NIGGEMANN, O. et al. Data-driven monitoring of cyber-physical systems leveraging on big data and the internet-of-things for diagnosis and contro. **CEUR Workshop Proceedings**, v. 1507, p. 185–192, 2015. ISSN 16130073.

NUÑEZ, D. L.; BORSATO, M. An ontology-based model for prognostics and health management of machines. **Journal of Industrial Information Integration**, Elsevier, v. 6, p. 33–46, 2017.

PETRASCH, R.; HENTSCHKE, R. Process modeling for industry 4.0 applications: Towards an industry 4.0 process modeling language and method. In: IEEE. **Computer Science and Software Engineering (JCSSE), 2016 13th International Joint Conference on**. [S.l.], 2016. p. 1–5.

PLOENNIGS, J. et al. Guest editorial semantic technologies in automation systems. **IEEE Transactions on Industrial Informatics**, IEEE, v. 13, n. 6, p. 3334–3337, 2017.

PRENTZAS, J.; HATZILYGEROUDIS, I. Categorizing approaches combining rule-based and case-based reasoning. **Expert Systems**, Wiley Online Library, v. 24, n. 2, p. 97–122, 2007.

PRUD, E.; SEABORNE, A. et al. Sparql query language for rdf. 2006.

QIN, Y. et al. When things matter_ A survey on data-centric internet of things. **Journal of Network and Computer Applications**, Elsevier, v. 64, p. 137–153, 2016. ISSN 1084-8045. Available from Internet: <http://dx.doi.org/10.1016/j.jnca.2015.12.016>.

RICHTER, A.; KOCH, M. Functions of social networking services. **From CSCW to Web 2.0: European Developments in Collaborative Design Selected Papers from COOP08**, ACM Press, 2008.

RIJGERSBERG, H.; ASSEM, M. V.; TOP, J. Ontology of units of measure and related concepts. **Semantic Web**, IOS Press, v. 4, n. 1, p. 3–13, 2013.

RODRIGUES, N. G. **Arquitetura e middleware para integração dos níveis de gerenciamento industriais baseado em sistemas ciberfísicos**. [S.l.], 2016. Available from Internet: <https://argo.furg.br/?BDTD11506>.

SHA, L. et al. Cyber-physical systems: A new frontier. In: **Machine Learning in Cyber Trust**. [S.l.]: Springer, 2009. p. 3–13.

SHENG, Z. et al. A survey on the ietf protocol suite for the internet of things: Standards, challenges, and opportunities. **IEEE Wireless Communications**, IEEE, v. 20, n. 6, p. 91–98, 2013.

SHROUF, F.; ORDIERES, J.; MIRAGLIOTTA, G. Smart factories in industry 4.0: A review of the concept and of energy management approached in production based on the internet of things paradigm. In: IEEE. **Industrial Engineering and Engineering Management (IEEM), 2014 IEEE International Conference on**. [S.l.], 2014. p. 697–701.

SILVA, T. R. d. **Proposta de ontologia para integração de sistemas de manutenção inteligentes e cadeias de suprimento de peças de reposição**. Porto Alegre, RS: UFRGS - Universidade Federal do Rio Grande do Sul, 2015. (Dissertação de Mestrado).

SILVA, T. R. da; PEREIRA, C. E. Building an ontology for intelligent maintenance systems and spare parts supply chain integration. **IFAC Proceedings Volumes**, Elsevier, v. 47, n. 3, p. 7843–7848, 2014.

SMYTH, N. **Android Studio 3.0 Development Essentials-Android 8 Edition**. [S.l.]: Payload Media, Inc., 2017.

SOLDATOS, J. et al. Openiot: Open source internet-of-things in the cloud. In: **Interoperability and open-source solutions for the internet of things**. [S.l.]: Springer, 2015. p. 13–25.

SOTIRIADIS, S.; STRAVOSKOUFOS, K.; PETRAKIS, E. G. Future internet systems design and implementation: cloud and iot services based on iot-a and fiware. In: **Designing, Developing, and Facilitating Smart Cities**. [S.l.]: Springer, 2017. p. 193–207.

SPIESS, P. et al. Soa-based integration of the internet of things in enterprise services. In: IEEE. **Web Services, 2009. ICWS 2009. IEEE International Conference on**. [S.l.], 2009. p. 968–975.

STANFORD. **"Protege: a free, open source ontology editor and knowledge-base framework**. 2013.

STARK, J. Product lifecycle management. In: **Product Lifecycle Management (Volume 1)**. [S.l.]: Springer, 2015. p. 1–29.

STEINMETZ, C. et al. Ontology-driven iot code generation for fiware. In: IEEE. **Industrial Informatics (INDIN), 2017 IEEE 15th International Conference on**. [S.l.], 2017. p. 38–43.

STRANG, D.; ANDERL, R. Assembly process driven component data model in cyber-physical production systems. In: **Proceedings of the World Congress on Engineering and Computer Science**. [S.l.: s.n.], 2014. v. 2.

SUO, H. et al. Security in the internet of things: a review. In: IEEE. **Computer Science and Electronics Engineering (ICCSEE), 2012 international conference on**. [S.l.], 2012. v. 3, p. 648–651.

TAO, F.; CHENG, J.; QI, Q. IIHub: an Industrial Internet-of-Things Hub Towards Smart Manufacturing Based on Cyber-Physical System. **IEEE Transactions on Industrial Informatics**, v. 3203, n. c, 2017. ISSN 15513203.

THRAMBOULIDIS, K.; CHRISTOULAKIS, F. Uml4iot—a uml-based approach to exploit iot in cyber-physical manufacturing systems. **Computers in Industry**, Elsevier, v. 82, p. 259–272, 2016.

TRAPPEY, A. J. et al. A review of essential standards and patent landscapes for the internet of things: A key enabler for industry 4.0. **Advanced Engineering Informatics**, Elsevier, v. 33, p. 208–229, 2017.

TRUNZER, E. et al. A flexible architecture for data mining from heterogeneous data sources in automated production systems. In: IEEE. **Industrial Technology (ICIT), 2017 IEEE International Conference on**. [S.l.], 2017. p. 1106–1111.

UNGUREAN, I.; GAITAN, N. C.; GAITAN, V. G. An IoT architecture for things from industrial environment. **IEEE International Conference on Communications**, 2014. ISSN 15503607.

VESTBURG, H. Ceo to shareholders: 50 billion connections 2020. **Ericsson, April**, v. 13, 2010.

WANG, S. et al. Towards smart factory for industry 4.0: a self-organized multi-agent system with big data based feedback and coordination. **Computer Networks**, Elsevier, v. 101, p. 158–168, 2016.

WANG, X. H. et al. Ontology based context modeling and reasoning using owl. In: IEEE. **Pervasive Computing and Communications Workshops, 2004. Proceedings of the Second IEEE Annual Conference on**. [S.l.], 2004. p. 18–22.

WU, M. et al. Research on the architecture of internet of things. In: IEEE. **Advanced Computer Theory and Engineering (ICACTE), 2010 3rd International Conference on**. [S.l.], 2010. v. 5, p. V5–484.

YANG, Z. et al. Study and application on the architecture and key technologies for iot. In: IEEE. **Multimedia Technology (ICMT), 2011 International Conference on**. [S.l.], 2011. p. 747–751.

YAO, X.; LIN, Y. Emerging manufacturing paradigm shifts for the incoming industrial revolution. **The International Journal of Advanced Manufacturing Technology**, The International Journal of Advanced Manufacturing Technology, p. 1665–1676, 2016. ISSN 0268-3768. Available from Internet: <http://dx.doi.org/10.1007/s00170-015-8076-0>.