



INSTITUTO TECNOLÓGICO VALE


**Programa de Pós-Graduação em Instrumentação, Controle e Automação de Processos de Mineração - PROFICAM**

**Universidade Federal De Ouro Preto - Escola de Minas**

**Associação Instituto Tecnológico Vale**

**Dissertação**

# Inspeção de Transportadores de Correia: Arquitetura Integrada para uma Plataforma de Inspeção com uso de VANTs

**Richardson Salgado do Nascimento**

**Ouro Preto**

**Minas Gerais, Brasil**

**2018**

**Richardson Salgado do Nascimento**

# Inspeção de Transportadores de Correia: Arquitetura Integrada para uma Plataforma de Inspeção com uso de VANTs

Dissertação apresentada ao curso de Mestrado Profissional em Instrumentação, Controle e Automação de Processos de Mineração da Universidade Federal de Ouro Preto e do Instituto Tecnológico Vale, como parte dos requisitos para obtenção do título de Mestre em Engenharia de Controle e Automação.

Orientador: Prof. Dr. Ricardo Augusto Rabelo Oliveira Morato
Coorientador: Prof. Dr. Luis Guilherme Uzeda Garcia

Ouro Preto
2018

N244i Nascimento, Richardson S. do.
Inspeção de transportadores de correia [manuscrito]: Arquitetura integrada para uma plataforma de inspeção com uso de VANTs / Richardson S. do Nascimento. - 2018.
167f.: il.: color; tabs.

Orientador: Prof. Dr. Ricardo A. R. Oliveira Morato.
Coorientador: Prof. Dr. Luis G. Uzeda Garcia.

Dissertação (Mestrado) - Universidade Federal de Ouro Preto. Escola de Minas. Departamento de Engenharia de Controle e Automação e Técnicas Fundamentais. Programa de Pós Graduação em Instrumentação, Controle e Automação de Processos de Mineração.
Área de Concentração: Engenharia de Controle e Automação de Processos Minerais.

1. Transportadores de Correia. 2. Veículos Aéreos. 3. Veículos autônomos. I. Morato, Ricardo A. R. Oliveira. II. Garcia, Luis G. Uzeda. III. Universidade Federal de Ouro Preto. IV. Titulo.

Catalogação: www.sisbin.ufop.br CDU: 681.5:622.2

Mestrado Profissional em Instrumentação, Controle e Automação de Processos de Mineração - PROFICAM

Inspeção de Transportadores de Correia: Arquitetura Integrada para uma Plataforma de Inspeção com Uso de VANTs

Richardson Salgado do Nascimento

Dissertação defendida e aprovada em 04 de maio de 2018 pela banca examinadora constituída pelos professores:

D.Sc. Ricardo Augusto Rabelo Oliveira
Orientador – Universidade Federal de Ouro Preto (UFOP)

Ph.D. Luís Guilherme Uzeda Garcia
Coorientador – Instituto Tecnológico Vale (ITV)

D.Sc. Marcone Jamilson Freitas Souza
Membro interno – Universidade Federal de Ouro Preto (UFOP)

Ph.D. Jorge Miguel Sá Silva
Membro externo – Universidade de Coimbra (UC)

*Ao meu irmão, Lucas, que me fez subir o primeiro degrau, quando eu mal enxergava a escada. Obrigado por acreditar e ver o melhor em mim!*

## AGRADECIMENTOS

Ninguém chega ao fim de uma jornada como essa sozinho. Definitivamente, não foi diferente pra mim. Para realizar este sonho, tive o apoio de inúmeras pessoas, cada uma com sua parcela individual de contribuição que me permitiu chegar ao final. Primeiramente, meu agradecimento especial a meus orientadores, Prof. Dr. Ricardo Rabelo, Prof. Dra. Andrea Bianchi e Prof. Dr. Luis Uzeda, que são exemplos de dedicação enquanto mestres, engajados na difícil missão de educar. Não haveria sequer uma jornada se não fosse pelo apoio e atenção dedicada por vocês ao longo da realização deste trabalho.

Agradeço também ao corpo docente do ITV e da UFOP, que me desafiaram a dar meu melhor, acedendo a fagulha da curiosidade e a inquietante sensação do querer aprender cada vez mais. Não foi fácil cumprir cada uma das cadeiras, mas o aprendizado conseguido com vocês foi a base para suportar essa jornada. Em tempo, estendo os agradecimentos ao time administrativo do ITV, especialmente, Rúbia e Jamilly, que são pessoas incríveis e fazem toda essa máquina girar.

À Vale, por financiar a pesquisa e dar todas as condições necessárias para a realização do mestrado. Particularmente, agradeço ao Luiz Leal, que foi um grande incentivador para iniciar essa caminhada, à Fabiana Neves, que não mediu esforços para prover as condições para que eu conseguisse concluí-la, ao Derval Beltrame, que acumulou funções em diferentes ocasiões para que eu pudesse assistir às aulas e ao Dr. Tobias Frank, cuja trajetória profissional e conhecimento são motivos de inspiração. Tenho muito orgulho de ser da Vale e conviver com pessoas incríveis como vocês, que carregam em cada atitude o valor de “Fazer Acontecer”!

A jornada também traz consigo agradáveis surpresas, que reforçam que a felicidade está no caminho. Dessa forma, o meu mais profundo agradecimento à amiga e Prof. Dra. Kátia Poles pelas incontáveis e detalhadas revisões realizadas neste trabalho e pelas palavras de apoio nas horas em que eram mais necessárias. Ao Prof. Saul Delabrida, meu orientador na graduação e que teve um novo papel na minha vida acadêmica, ensinar-me a escrita científica. Ao amigo Regivaldo Carvalho, com quem compartilhei inúmeras horas de trabalho árduo durante a realização da pesquisa e escrita de artigos. Obrigado por tornar isso mais leve!

Por fim e não menos importante, agradeço a Deus, aos meus pais, Adriana e Pascoal, aos meus irmãos, Jéssika e Lucas, e aos meus amigos. Cada um de vocês sabe a importância que teve nessa etapa de minha vida. Nenhum agradecimento é suficiente para expressar minha gratidão diante do apoio que recebi e da compreensão nos inúmeros momentos de ausência. Saibam que tudo isso é por vocês! Meu mais sincero obrigado!

*”Suba o primeiro degrau com fé. Não é necessário que você veja toda a escada. Apenas dê o primeiro passo.”*

*Marthin Luther King Jr.*

## RESUMO

Os rolos de transportadores de correias são um dos ativos mais críticos da indústria da mineração, cujas falhas trazem grandes impactos para as operações. Para evitá-las, são realizadas inspeções regulares de forma sensitiva (sem uso de instrumentos de medição), o que diminui a qualidade da inspeção e ainda expõe os inspetores a um ambiente com riscos, calor e poeira. As técnicas automatizadas para monitoramento de condições dos rolos possuem limitações que dificultam sua adoção a curto prazo, o que justifica a utilização de uma estrutura de sensoriamento móvel, aqui chamada Plataforma de Inspeção, de implementação mais rápida. Assim, o presente trabalho apresenta uma revisão das principais técnicas de monitoramento de condições de rolos, propõe uma arquitetura para suportar a construção e evolução incremental de Plataformas de Inspeção e apresenta protótipos que validam os conceitos-chave discutidos na arquitetura, particularmente a integração da plataforma com sistemas corporativos. O trabalho empregou Veículos Aéreos Não Tripulados (VANTs) como base para os protótipos e testes em campo, além de construir um ambiente virtual de um porto para aplicações baseadas em VANTs em um simulador.



**Palavras-chaves**: Transportadores de Correia, Rolos, Integração, Veículos Aéreos Não Tripulados.

## ABSTRACT

Conveyor belt rollers are one of the most critical assets in the mining industry and their failures bring major impacts to operations. To avoid them, inspectors regularly perform sensitive inspections (without measurement instruments), but they lack quality and expose labor to a hazardous environment, with heat and dust. Automated techniques for rollers' condition monitoring have limitations that hinder their adoption in the short-term. This supports the use of a mobile sensor structure, here called Inspection Platform, which is fast to implement. Therefore, this work presents a revision of the most important condition monitoring techniques for rollers, proposes an architecture to support the construction and incremental evolution of Inspection Platforms and presents prototypes that validate key concepts discussed in the architecture, mainly its integration with enterprise systems. This work employed Unmanned Aerial Vehicles (UAVs) to support the prototypes and field tests. A virtual environment of a port was built to evaluate UAV-based applications in a simulator.

**Keywords**: Belt Conveyor, Rollers, Integration, Unmanned Aerial Vehicles.

## LISTA DE FIGURAS

# LISTA DE TABELAS

## LISTA DE SIGLAS E ABREVIATURAS

3GPP - *3rd Generation Partnership Project*
ABNT - Associação Brasileira de Normas Técnicas
ANAC - Agência Nacional de Aviação Civil
AE - *Acoustic Emission*
AMQP - *Advanced Message Queuing Protocol*
ANATEL - Agência Nacional de Telecomunicações
API - *Application Programming Interface*
BVLOS - *Beyond Visual Line of Sight*
CAN - *Controller Area Network*
CE - *Conformité Européenne*
CMMS - *Computerized Maintenance Management System*
CoAP - *COnstrained Application Protocol*
CORBA - *Common Object Request Broker Architecture*
C-OTDR - *Coherent-Optical Time Domain Reflectometry*
DCOM - *Distributed Component Object Model*
DOFS - *Distributed Optical Fiber Sensors*
DTLS - *Datagram Transport Layer Security*
DTS - *Distributed Temperature System*
EASA - *European Aviation Safety Agency*
ERP - *Enterprise Resource Planning*
ESB - *Enterprise Service Bus*
ESC - *Electronic Speed Controller*
FAA - *Federal Aviation Administration*
FC - *Flight Controller*
FPS - *Frames per Second*
FPV - *First Person View*
GCS - *Ground Control Station*
GNSS - *Global Navigation Satellite System*
GPS - *Global Positioning System*
HIL - *Hardware-In-The-Loop*
HSPA+ - *Evolved High Speed Packet Access*
HTTP - *Hypertext Transfer Protocol*
HTTPS - *Hypertext Transfer Protocol Secure*
ICEIS - *International Conference on Enterprise Information Systems*
IETF - *Internet Engineering Task Force*
IMU - *Inertial Measurement Unit*
IoT - *Internet of Things*

IPSec - *Internet Protocol Security Protocol*
JMS - *Java Message Service*
JSON - *JavaScript Object Notation*
Li-Po - Polímero de Lítio
MEC - *Mobile Edge Computing*
MQTT - *MQ Telemetry Transport*
MSHA - *Mine Safety and Health Administration*
OFDR - *Optical Frequency Domain Reflectometry*
OLCR - *Optical Low Coherence Reflectometry*
OTDR - *Optical Time Domain Reflectometry*
P - Controlador Proporcional
PI - Controlador Proporcional Integral
PIMS - *Plant Information Management System*
QoS - *Quality of Service*
RAN - *Radio Access Network*
REST - *REpresentational State Transfer*
RFID - *Radio-Frequency IDentification*
ROS - *Robotic Operating System*
RPA - Aeronave Remotamente Pilotada
RTK - *Real Time Kinematic*
RTP - *Real Time Protocol*
SASL - *Simple Authentication and Security Layer*
SDK - *Software Development Kit*
SIL - *Software-In-The-Loop*
SISANT - Sistema de Aeronaves Não Tripuladas
SOA - *Service Oriented Architecture*
SMTP - *Simple Mail Transport Protocol*
SSL - *Secures Socket Layer*
STOMP - *Simple/Streaming Text Orientated Messaging Protocol*
SVM - *Support Vector Machine*
TC - Transportador de Correia
TCP - *Transmission Control Protocol*
TI - Tecnologia da Informação
TLS - *Transport Layer Security*
TMPM - Terminal Marítimo de Ponta da Madeira
UART - *Universal Asynchronous Receiver-Transmitter*
UDP - *User Datagram Protocol*
UPE - Unidade de Processamento Externa
UPM - Unidade de Processamento Móvel

USB - *Universal Serial Bus*
VANT - Veículo Aéreo Não Tripulado
VLOS - *Visual Line Of Sight*
VPN - *Virtual Private Network*
VR - *Virtual Reality*
W3C - *World Wide Web Consortium*
WPD - *Wavelet Packet Decomposition*
WSDL - *Web Service Description Language*
WSN - *Wireless Sensor Network*
XML - *eXtensible Markup Language*
XMPP-IoT - *Extensible Messaging and Presence Protocol - Internet of Things*

# SUMÁRIO

# 1 Introdução

Em processos da cadeia de mineração, que envolvem mina, usina de beneficiamento, pelotização e porto, é extenso o uso de Transportadores de Correia (TC) para a movimentação de grandes quantidades de material, conforme Figura 1. Apenas na Vale, uma das maiores empresas de mineração do mundo, estão presentes mais de 2.000 TCs, totalizando 1.000 km de extensão de correia e aproximadamente 1,6 milhões de rolos, conforme dados coletados com a área de Engenharia de Médio e Longo Prazo de Portos da mineradora Vale.


Figura 1 – Visão de um pátio, onde se concentra a maior extensão dos transportadores de correia. Fonte: autor

Alguns dos componentes do TC não representam desafios significativos de inspeção por i) já existirem tecnologias de monitoramento de condições consolidadas, caso do sistema de acionamento, ou ii) por estarem agrupados em uma pequena região, caso dos raspadores e sistema de esticamento (LODEWIJKS et al., 2016). Porém, outros estão fisicamente espalhados ao longo do TC, em grande quantidade, sem técnicas eficientes para o seu monitoramento. É o caso dos rolos, que suportam a correia e, consequentemente, o material transportado ao longo de sua extensão.

Falhas nos rolos geram perdas operacionais significativas, com prejuízos materiais elevados e impactos na produção, especialmente pelo potencial de incêndios no caso de sobreaquecimento (YANG, 2014). Dada a necessidade de prevenir esse tipo de falha, que pode ter consequências catastróficas, a prática atual é a de realização de inspeções desses componentes, com periodicidade variável de acordo

com as características e uso de cada TC. Porém, essa atividade possui inúmeros desafios, que são listados a seguir (LODEWIJKS, 2004):

- Conforme dados de espaçamento e quantidade de rolos apresentados na Tabela 1, a inspeção de um TC de 1,40 km implica em uma caminhada de 2,80 km e na verificação de aproximadamente 4.760 rolos, o que leva em torno de 3,2 horas[1];
- Condições pouco agradáveis no entorno do TC, como ruído excessivo, calor, poeira e exposição do inspetor a diversos tipos de riscos, como quedas em mesmo nível, projeção de materiais e animais peçonhentos;
- Grande dependência da experiência do inspetor e de sua acuidade visual e auditiva, já que a inspeção é basicamente sensitiva, com observação de ruídos emitidos pelo rolamento dos rolos e avaliação visual da superfície em busca de avarias e travamentos. Mesmo com inspetores experientes, este processo ainda está sujeito a falhas humanas, o que diminui a qualidade da inspeção;
- O monitoramento de condições, como a medição de vibração, captura do áudio dos rolos e o registro de temperatura com uso de termografia não é prático para se aplicar em todas as inspeções, dado o grande volume de componentes, o tempo gasto na execução das medições e a dificuldade de se obter as medidas dos pontos de interesse.

Tabela 1 – Informações gerais sobre os principais tipos de rolos de TC

| | **Rolos** | |
|---|---|---|
| | **Carga** | **Retorno** |
| **Quantidade (em 1,4 km)** | 4200 | 560 |
| **Espaçamento** | 1 m | 2,5 m |
| **Custo por Unidade** | R$ 300 - R$ 400 | R$ 500 - R$ 1000 |
| **Vida Útil** | 3 a 4 anos | 5 a 10 anos |

Fonte: autor, a partir de dados da Engenharia de Médio e Longo Prazo de Portos da mineradora Vale

Diante destes problemas, que estão presentes mesmo nos casos mais simples de inspeção, percebe-se a necessidade de desenvolver mecanismos automatizados que consigam garantir a integridade dos ativos monitorados e reduzir a exposição de trabalhadores a ambientes naturalmente perigosos, como os locais onde os TCs operam. O monitoramento de um rolo por si só não é complexo, com técnicas

[1] Considerando uma velocidade média de caminhada de 5 km/h e 2 segundos de inspeção para cada rolo

bem estabelecidas baseadas em sinais acústico, térmico e vibração. Porém, como monitorar mais de 200.000 rolos presentes apenas no Terminal Marítimo de Ponta da Madeira (TMPM), um dos portos da mineradora?

Procurando responder essa pergunta, no presente trabalho é feita uma discussão das principais técnicas de monitoramento de condições dos rolos, que variam de soluções baseadas no sensoriamento individual de cada rolo, passam por técnicas que baseiam-se na instalação de sensores fixos em cada transportador e, finalmente, chegam às estruturas de sensoriamento móvel, acopladas ou independentes do TC. Com base nas características de cada uma dessas modalidades, é proposta uma "**Arquitetura Integrada**" que suporte uma solução de inspeção de rolos baseada em uma estrutura de sensoriamento móvel e independente do TC, aqui chamada "**Plataforma de Inspeção**".

A viabilidade da utilização de uma Plataforma de Inspeção para o monitoramento desta grande quantidade de ativos baseia-se em uma condição fundamental: a integração com sistemas corporativos já existentes. Tais sistemas suportam diferentes processos de negócio e necessitam receber ou prover dados à Plataforma de Inspeção. Dado o volume em questão, não seria produtivo delegar a um operador o registro manual dos defeitos detectados pela Plataforma no sistema de gestão de manutenção (CMMS, do inglês *Computerized Maintenance Management System*). Neste caso, desconsiderando o retrabalho, a ausência de integração pode resultar em consequências diversas, desde inofensivos erros de digitação, que prejudicam a qualidade da informação, até a completa ausência do registro de um defeito grave, que pode resultar em um incêndio de grandes proporções.

Por conta disso, a arquitetura proposta tem como um dos pilares a integração da Plataforma de Inspeção com sistemas corporativos usando diferentes técnicas de integração. Ainda que o objetivo da arquitetura proposta seja ser flexível e independente do mecanismo de locomoção (portador), neste trabalho, para demonstrar a viabilidade, foram escolhidos os Veículos Aéreos Não Tripulados (VANTs). Assim, foram desenvolvidos protótipos que procuram demonstrar a capacidade de integração nos principais casos de uso, como o envio de defeitos identificados em campo para o sistema de manutenção e a publicação contínua de dados de telemetria e sensores da plataforma. Também foi desenvolvido um modelo simplificado do porto em um ambiente de simulação de VANTs para testes e outras aplicações.

Dessa forma, o presente trabalho representa um esforço na direção de melhorar as práticas atuais de inspeção de rolos na indústria da mineração. Sua principal contribuição é a proposição de uma arquitetura que suporte a construção de Plataformas de Inspeção para rolos e sua integração com sistemas corporativos, com a discussão das principais técnicas e protocolos que suportem esse processo e seus

desafios.

## 1.1 Justificativa

Quando se avalia o problema em questão, é possível perceber que existem diferentes elementos com potencial de gerar falhas graves. O primeiro deles é a natureza dos rolos: eles possuem dois rolamentos que sofrem desgate natural e que podem travar, gerando assim atrito e aquecimento. O segundo é que a correia, feita de borracha ou lona, está em contato direto com os rolos, o que significa que um princípio de incêndio em um rolo encontra um mecanismo de propagação muito favorável e que pode se estender por uma grande região. Por fim, tem-se a dificuldade de monitoramento dos rolos, dada sua quantidade e dispersão, o que torna a inspeção um grande desafio. O resultado da combinação destes elementos é o potencial de ocorrência de eventos catastróficos, como os registrados na Figura 2a, ocorrido no TMPM (São Luís, Maranhão), e na Figura 2b, em uma operação de fertilizantes (Cubatão, São Paulo).

(a) Incêndio em torre de transferência em TMPM. Fonte: IMIRANTE.COM, 2015

(b) Incêndio no TC de uma operação de fertilizantes. Fonte: G1, 2017

Figura 2 – Dois exemplos de incêndios de grandes proporções iniciados em TCs

Felizmente, nem todo incêndio atinge as proporções desses exemplos, porém, eles ocorrem de forma muito frequente nas operações e geram prejuízos. Dados extraídos de sistemas internos da empresa Vale mostram que, entre 2014 e Outubro de 2016, apenas no TMPM (Sistema Norte) e no Porto de Tubarão (Sistema Sudeste), foram registrados mais de R$ 2,7 milhões em perdas materiais por conta de incêndios causados por falhas nos rolos, totalizando 600 horas de parada na operação, vide Figura 3. Além disso, ainda segundo dados extraídos destes sistemas, especificamente no TMPM, foram registradas em 2016 mais de 169 horas

de parada com causa “Princípio de incêndio”, correspondendo a 52% de todas as falhas registradas nos roletes, que são o conjunto dos rolos com o suporte.

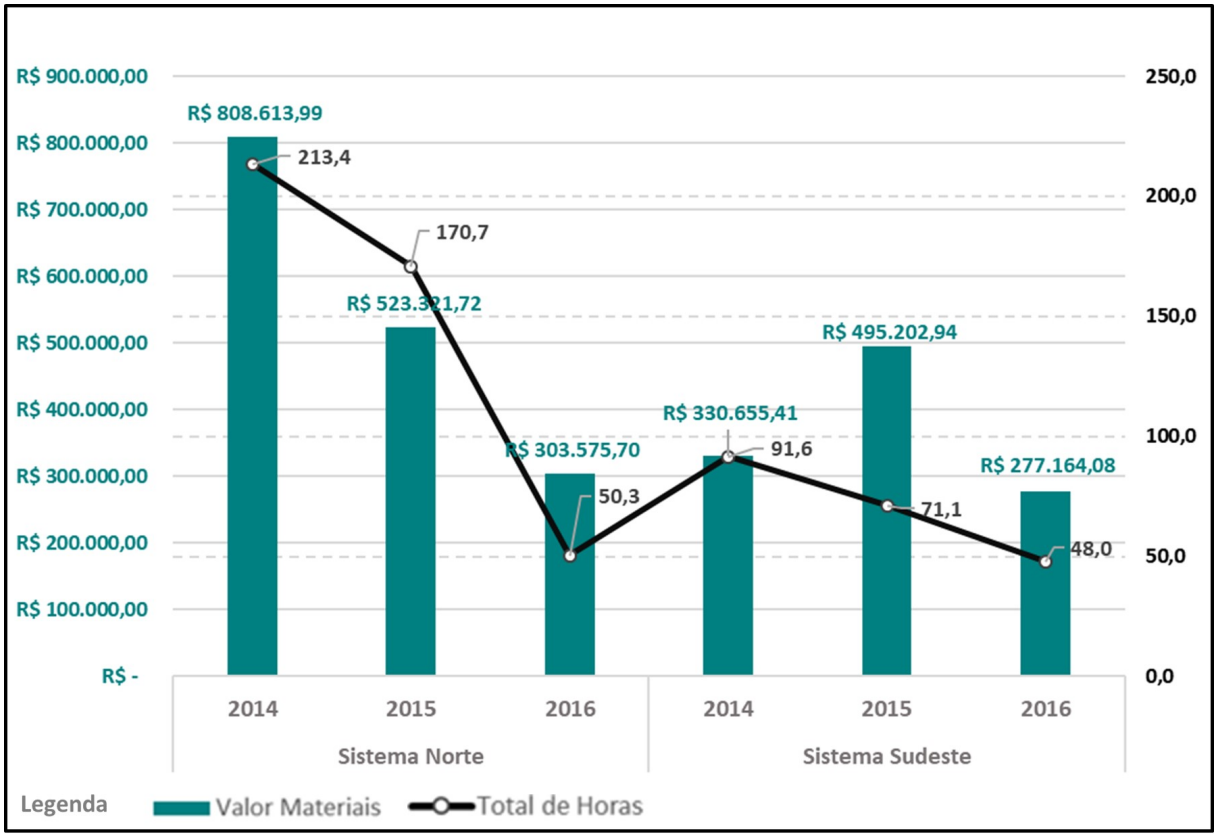



Figura 3 – Consequências financeiras das falhas em rolos em duas operações portuárias da mineradora Vale. Fonte: autor, com dados obtidos de sistema interno da mineradora

Este não é um problema exclusivo destas operações, mas da indústria mineral e de materiais a granel como um todo. Além do aspecto financeiro, existem também os riscos à saúde e segurança dos trabalhadores envolvidos em atividades de inspeção e manutenção de transportadores. Ruff, Coleman e Martini (2011) avaliaram registros de acidentes graves registrados pelo órgão norte-americano responsável pela segurança das operações de mineração (MSHA, do inglês *Mine Safety and Health Administration*). Conforme mostrado na Figura 4, entre 2000 e 2007, foram registrados 562 acidentes, sendo os TCs responsáveis pela maioria deles, correspondente a 14%. Destes eventos, mais da metade ocorreu durante atividades de manutenção ou inspeção no TC.

Desta forma, o presente trabalho procura contribuir na solução deste problema ao propor uma arquitetura que suporte definir Plataformas de Inspeção capazes de melhorar o monitoramento dos rolos e reduzir a exposição dos trabalhadores atualmente envolvidos nesta atividade.

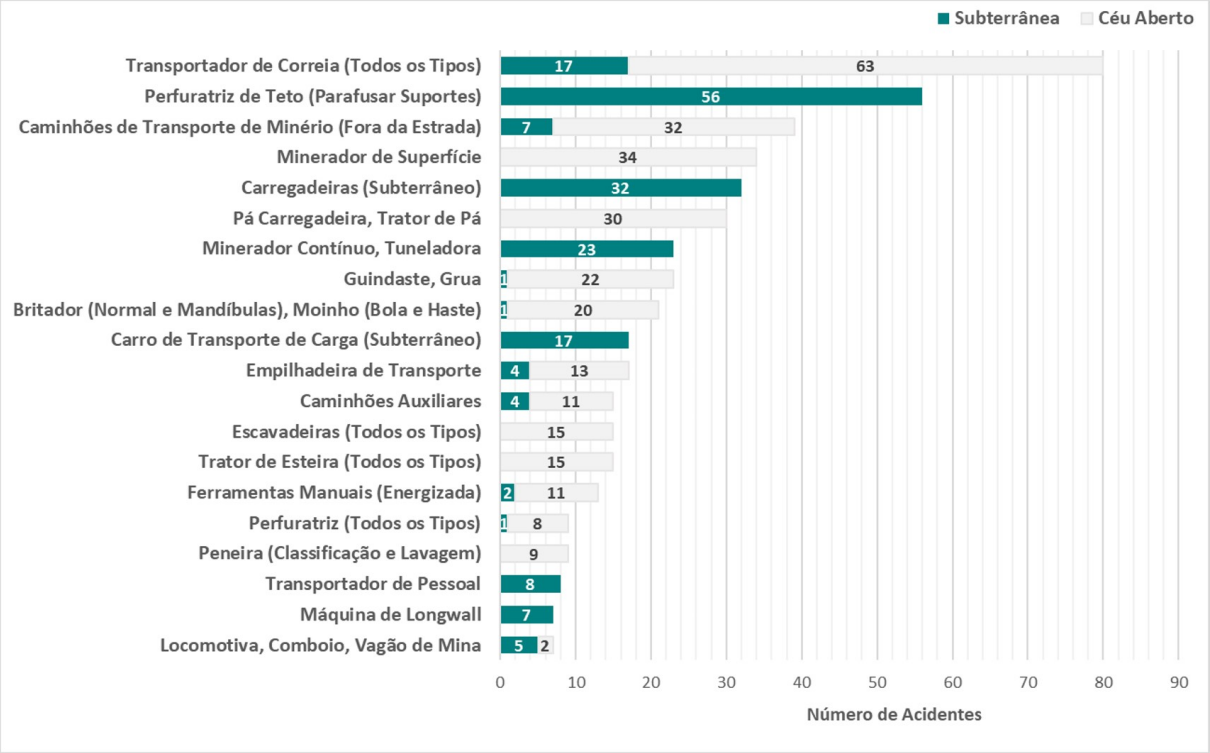



Figura 4 – Transportadores de correia lideram os acidentes na indústria de mineração norte-americana. Fonte: RUFF; COLEMAN; MARTINI, 2011

## 1.2 Objetivos

Este trabalho tem como objetivo geral avaliar técnicas e soluções para suportar a definição de uma arquitetura que suporte a construção e evolução incremental de Plataformas de Inspeção de rolos de TCs, com a sua integração a sistemas existentes. Especificamente, os objetivos do trabalho são:

- Revisar e discutir as principais técnicas disponíveis para o monitoramento de condições de rolos de transportadores de correia;
- Desenvolver uma arquitetura que suporte a construção e evolução incremental de Plataformas de Inspeção adequadas à inspeção de rolos, habilitando sua integração aos sistemas existentes;
- Avaliar protocolos e técnicas de integração compatíveis e adequados aos diferentes casos de uso e interações que a Plataforma de Inspeção possui com sistemas corporativos;
- Desenvolver protótipos que se encaixam nas camadas da arquitetura definida e demonstrem suas capacidades em termos de integração com sistemas existentes;

- Validar em campo um protótipo da Plataforma de Inspeção por meio da realização de uma inspeção simulada em área controlada e limitada.

## 1.3 Publicações

As discussões referentes ao estado da arte no monitoramento de condições de rolos, a arquitetura proposta para a construção da Plataforma de Inspeção e, principalmente, sua integração com as aplicações existentes tiveram como resultado o artigo *An Integrated Inspection System for Belt Conveyor: Advancing in an Enterprise Architecture*, que foi apresentado na *19th International Conference on Enterprise Information Systems* (ICEIS), classificado com Qualis B1 na área de Ciências da Computação na data da publicação.

Também teve como resultado um resumo para apresentação oral com o título *An UAV-based Framework for Automated Thermographic Inspection of Belt Conveyors in the Mining Industry* na *1st International Conference Mines of the Future*, organizada pela RWTH Aachen University, ainda sem Qualis.

## 1.4 Organização do Texto

O restante do texto deste trabalho foi organizado da seguinte forma:

- O capítulo "**2 - Transportadores de Correia e Rolos**" apresenta os principais conceitos dos Transportadores e Rolos, que constituem o problema em questão;
- No capítulo "**3 - Trabalhos Relacionados**" é realizada uma discussão das principais técnicas disponíveis para o monitoramento de condições de rolos, onde apresenta-se o porquê da escolha da abordagem móvel para a Plataforma de Inspeção;
- O capítulo "**4 - Tecnologias**" detalha conceitos fundamentais que suportam as escolhas de tecnologias utilizadas na definição da Arquitetura Integrada, como VANTs, técnicas e protocolos de integração e ferramentas de simulação e desenvolvimento;
- No capítulo "**5 - Desenvolvimento**" apresenta-se a Arquitetura Integrada e os protótipos desenvolvidos para demonstrar algumas das suas capacidades,

com a discussão dos testes e resultados sendo realizada no capítulo “**6 - Testes e Resultados**”;

- Finalmente, o capítulo “**7 - Considerações Finais**” apresenta as considerações finais e procura delinear potenciais trabalhos futuros.

# 2 Transportadores de Correia e Rolos

Este capítulo define e especifica o problema discutido no presente projeto, que é a inspeção de rolos de transportadores de correia. Para isso, o capítulo foi dividido em três seções:

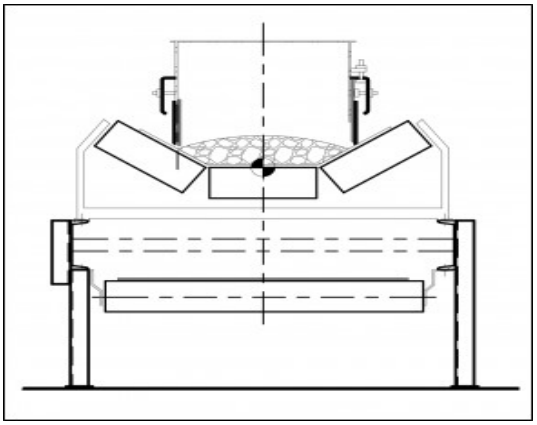
- A seção "**2.1 - Transportadores de Correia**" apresenta os principais tipos de transportadores de correia usados na mineração e faz uma breve descrição de seus principais componentes;
- Em seguida, a seção "**2.2 - Roletes, Rolos e Detecção de Falhas**" aprofunda a explicação dos rolos, descrevendo sua estrutura, seu modo de falha e os principais sinais que podem ser usados para o seu monitoramento;
- Por fim, a seção "**2.3 - Conclusões do Capítulo**" faz um apanhado dos principais pontos discutidos nas seções anteriores e apresenta as conclusões deste capítulo.

## 2.1 Transportadores de Correia

Os Transportadores de Correia (TC) são um dos meios mais utilizados para movimentar grandes quantidades de materiais a granel. Quando comparados com outras alternativas de transporte, como frotas de caminhões ou trens, os TCs apresentam vantagens relativas à confiabilidade, versatilidade, segurança, consumo de energia e custos, que os posicionam como um meio eficiente e seguro para manter fluxos contínuos de materiais entre operações, com extensões que podem variar de centenas de metros a dezenas de quilômetros (CONVEYOR EQUIPMENT MANUFACTURERS ASSOCIATION, 2014).

O tipo de transportador mais comum na indústria da mineração, foco deste estudo, é o que emprega correias e roletes como suporte, vide esquema na Figura 5a. Seus custos de instalação e manutenção são baixos em comparação às alternativas de transporte, conseguem operar com inclinações verticais de até 20º e raios de curva horizontais entre 1 e 2 km, sendo apropriados para o transporte eficiente de materiais por diferentes distâncias e terrenos (MCGUIRE, 2009), conforme mostrado na Figura 5b.

Existem variações básicas nesse tipo de transportador em relação à forma do berço de carga (plano ou côncavo) e ao tipo de correia empregada (lisa, com nervuras ou contenções) que atendem a diferentes tipos de materiais (SWINDERMAN



(a) Visão transversal de um transportador tradicional. Fonte: INGENIUM DESIGN, 2017

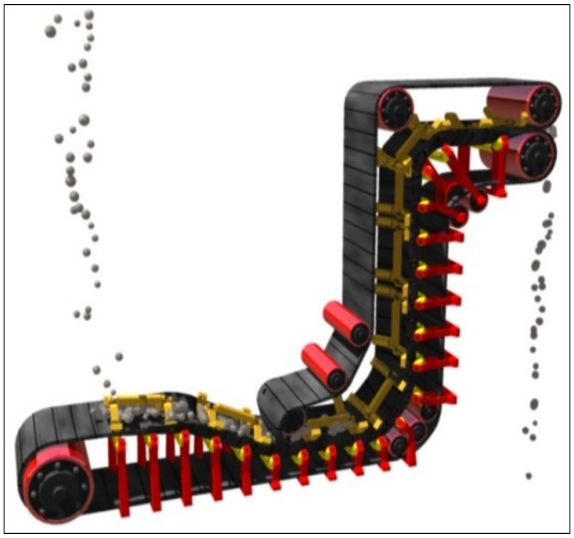
(b) Exemplo de uso de transportador na mineração. Fonte: METSO, 2017

Figura 5 – Representação esquemática de um transportador de correias e aplicação em um cenário real de longa distância na mineração

et al., 2009). Porém, mesmo com tais adaptações, eles não conseguem realizar curvas fechadas nem superar grandes inclinações. Dessa forma, outros tipos de transportadores são usados para atender tais situações, por exemplo:

- **Sanduíche:** São geralmente usados em operações com grandes desníveis verticais. Eles comprimem o material firmemente entre duas correias para conseguir superar diferentes tipos de inclinações verticais, que variam de 60º a 90º, o que não é possível se conseguir com transportadores tradicionais, uma vez que o material tende a deslizar em aclives (CONVEYOR EQUIPMENT MANUFACTURERS ASSOCIATION, 2014). Como pontos de atenção, esse tipo de transportador não é eficiente para materiais muito finos (como cimento) e seus custos são elevados, visto que emprega elementos adicionais, como uma segunda correia (Figura 6a). Resguardadas essas limitações, seu uso é factível em pontos específicos para superar grandes desníveis (SANTOS, 1983), como no exemplo mostrado na Figura 6b.

- **Pocket:** é um tipo especial de transportador com o objetivo de elevar o material. Nesse caso, a correia em si tem suportes de borracha que formam divisões (ou bolsos) com bordas laterais enrugadas, facilitando o encaixe do material (Figura 7a). Com essa configuração, são apropriados para superar inclinações que variam de 20º a 90º e podem ser usados em locais com espaço limitado, como dentro de navios, para efetuar a descarga de seus porões, como mostrado na Figura 7b. Como pontos negativos, a correia utilizada é mais cara, por conta da necessidade de formar os bolsões. Além disso, a granulometria do material transportado é limitada pelo tamanho dos bolsões, especialmente



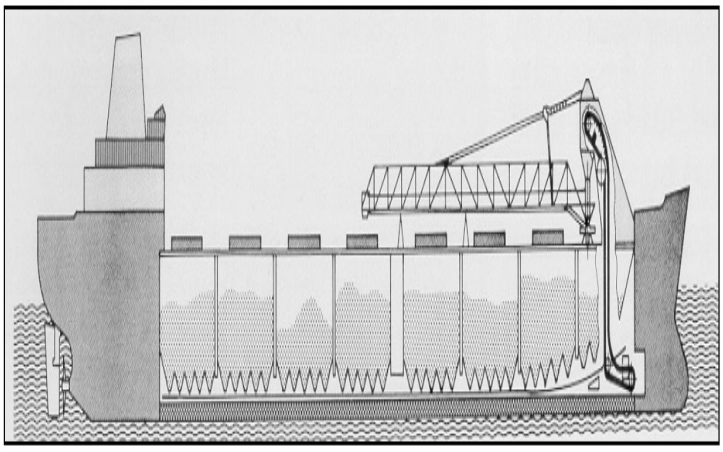
(a) Representação de um transportador do tipo Sanduíche. Fonte: 3D TOWER, 2017

(b) Correia Sanduíche operando em grande inclinação. Fonte: DOS SANTOS INTERNATIONAL, 2017

Figura 6 – Esquema em 3D de um TC do tipo Sanduíche e aplicação em cenário real de inclinação elevada

em ângulos de inclinação mais elevados (CONVEYOR EQUIPMENT MANUFACTURERS ASSOCIATION, 2014).

(a) Correia Pocket com grande inclinação

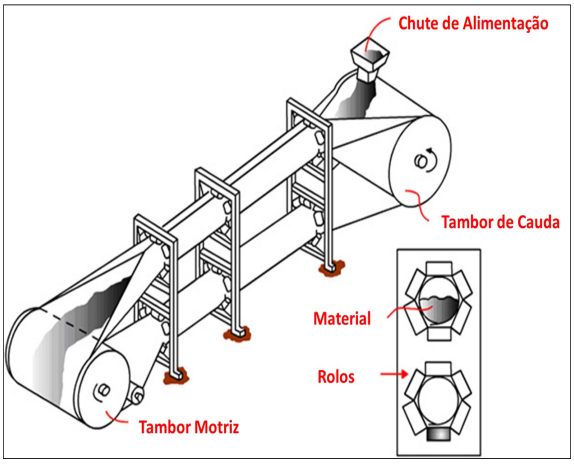


(b) Esquema de navio com sistema de descarga usando transportador do tipo Pocket

Figura 7 – Diferentes aplicações de transportadores do tipo Pocket para elevação de material granulado. Fonte: CONVEYOR EQUIPMENT MANUFACTURERS ASSOCIATION, 2014

- **Tubular:** a parte central da correia é arranjada para formar um tubo, em que o material fica enclausurado, com baixa emissão de particulados (Figura 8a). Com isso, é possível realizar curvas horizontais com diferentes inclinações sem que o material seja projetado para fora do transportador e ainda atingir inclinações verticais de até 30º. Como principais desvantagens, destacam-se

o custo de se construir e manter por conta do maior número de componentes e a menor capacidade de carga específica por largura de correia (MCGUIRE, 2009). Um exemplo de uso desse tipo de transportador é apresentado na Figura 8b.




(a) Representação de um TC do tipo Tubular. Fonte: traduzido de Probelt Global (2017)

(b) Exemplo em campo de um TC do tipo Tubular. Fonte: ZOOMLINE, 2017

Figura 8 – Esquema de um TC do tipo Tubular e exemplo de uso em uma operação real de grande extensão

Independentemente do tipo de transportador, seu objetivo é formar um sistema de transporte, que: i) receba um material oriundo de um TC ou equipamento (viradores de vagão ou recuperadoras, por exemplo); ii) transporte-o ao longo de uma rota específica, por distâncias e trajetos adaptáveis; iii) descarregue-o em uma pilha, navio ou o transfira para outro TC. Resguardadas adaptações específicas discutidas anteriormente, todos os tipos de TC possuem um conjunto comum de componentes principais para realizar a função de sistema de transporte, conforme esquema apresentado na Figura 9.

Não é o objetivo do presente trabalho pormenorizar cada um dos componentes que forma um TC, apenas dar ao leitor uma visão geral que suporte seu entendimento ao longo do texto. Caso seja do interesse, Conveyor Equipment Manufacturers Association (2014) apresenta informações completas a esse respeito. Dessa forma, a lista a seguir resume as explicações disponíveis nesta bibliografia e apresenta uma visão básica dos principais componentes de um TC:

- **Chute de Alimentação** e **Chute de Descarga:** o primeiro é responsável por receber o material a partir de outro TC ou equipamento e posicioná-lo na correia para que possa ser transportado. Já o chute de descarga é usado para transferir o material transportado para um outro TC. Quando a descarga do

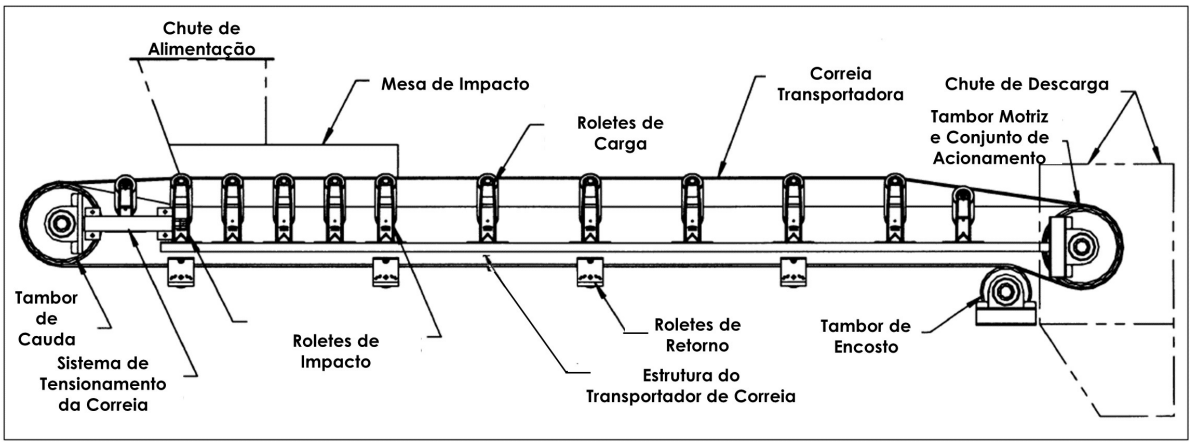



Figura 9 – Principais componentes de um Transportador de Correias. Fonte: adaptado de CONVEYOR EQUIPMENT MANUFACTURERS ASSOCIATION, 2014

material é feita diretamente em pilhas ou mesmo em navios, geralmente ele não é usado. Além disso, em alguns casos, é necessário descarregar total ou parcialmente o material em um ou vários pontos intermediários do TC. Para isso, pode-se adotar desviadores ou *trippers*.

- **Mesa de Impacto:** como o material a ser transportado vem de uma região mais elevada em relação ao TC atual, existe impacto quando o material é despejado pelo Chute de Alimentação. Logo, essa região é preparada para absorver e acomodar o material que será transportado;

- **Roletes de Impacto:** são posicionados abaixo das mesas de transferência de modo a absorver o impacto do material despejado pelo Chute de Alimentação. Nessa região, os roletes são colocados próximos uns aos outros, em geral a cada 30 cm, e empregam rolos com revestimentos especiais para absorção de impacto. Os rolos são melhor explicados na seção 2.2;

- **Correia Transportadora:** pode ser vista como o principal componente de um TC, já que é o elemento de contato com o material e que responde por entre 30 e 40% do custo total do TC. São compostas pela carcaça e por uma cobertura. Esses dois elementos podem ter diferentes tipos de composições para fornecer à correia propriedades específicas às mais diferentes aplicações. Algumas das características resultantes da composição da correia são a resistência à tração, abrasão, alongamento, tipo de material suportado e ambiente de operação, visto que alguns compostos são específicos para operação em condições de temperaturas extremas e com exposição a elementos químicos.

- **Roletes de Carga:** são responsáveis por suportar a correia e, consequentemente, o material transportado. Nos TCs tradicionalmente usados na mineração, os roletes de carga são espaçados a cada 1 m, mas essa distância varia

de acordo com o projeto para condições específicas do TC ou da carga. Cada rolete é formado por um suporte e por três rolos, inclinados entre si em ângulos que normalmente variam de 0º a 45º. A inclinação varia em função do ângulo de acomodação do material, espessura da correia, velocidade de operação e tonelagem de transporte necessária, podendo mudar dentro de um mesmo TC, conforme exemplo na Figura 10;

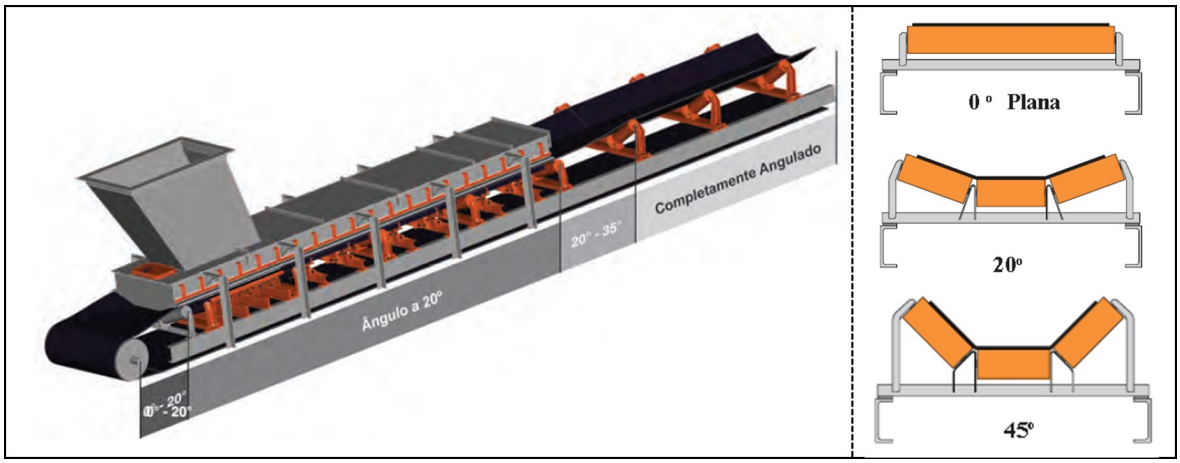



Figura 10 – Variação de ângulos de inclinação dos roletes para diferentes condições.
Fonte: adaptado de SWINDERMAN et al., 2009

- **Roletes de Retorno:** suportam a parte inferior da correia, que está sem carga. Os roletes de retorno ficam geralmente espaçados entre 1,5 e 2,5 m, com variações de acordo com o projeto. Podem empregar 1 ou 2 rolos, podendo também desempenhar a função de alinhamento da correia em alguns projetos;

- **Conjunto de Acionamento:** é geralmente composto por um motor elétrico, acoplamento hidráulico e um sistema de transmissão. É responsável por movimentar a correia ao fornecer tração para o Tambor Motriz, além de regular sua velocidade de operação;

- **Tambores (Motriz, de Encosto e Cauda):** os tambores são geralmente feitos de aço e desempenham diferentes papeis em um TC. O Tambor Motriz é conectado ao conjunto de acionamento e fornece tração à correia. O Tambor de Encosto aumenta o ângulo de contato da correia com o Tambor Motriz, auxiliando na tração. Por fim, o Tambor de Cauda retorna a correia para sua posição inicial e também pode desempenhar funções relativas ao tensionamento da correia;

- **Sistema de Tensionamento da Correia:** também conhecido como sistema de esticamento, é responsável por tensionar a correia durante o acionamento do TC e compensar mudanças na sua extensão, que ocorrem em função das condições de operação e mesmo por conta da temperatura. Assim, esse sistema

procura manter a tensão ideal de operação da correia em todos os momentos. No exemplo de TC mostrado na Figura 9, é apresentado um sistema de esticamento por parafuso, em que o deslocamento do eixo do Tambor de Cauda tensiona a correia. Porém, existem outros sistemas que empregam, por exemplo, a gravidade através de um contrapeso ligado a um tambor.

- **Estrutura do Transportador de Correia:** é o conjunto de todos os elementos que suportam e dão sustentação ao TC, como estruturas metálicas, treliças, longarinas, torres, parafusos, conectores, soldas, etc.

Com base no esquema da Figura 9, percebe-se que alguns dos componentes do TC são únicos e sempre estarão fixos na cabeça (região do Chute de Descarga) e na cauda (Chute de Alimentação) do TC. Tais componentes, como os tambores, conjunto de acionamento e sistema de esticamento, não apresentam desafios relevantes no seu monitoramento, pois são acessíveis para inspeções e não são numerosos. Por outro lado, a quantidade de roletes, e por conseguinte dos rolos, varia em função da extensão do TC, que pode medir quilômetros e, assim, utilizar quantidades vultuosas destes componentes.

Tais condições materializam-se em relevantes desafios para o monitoramento e inspeção destes componentes. Assim, a seção a seguir detalha a estrutura dos rolos e seu modo de falha, além de alguns dos princípios para a detecção de defeitos.

## 2.2 Roletes, Rolos e Detecção de Falhas

Antes de apresentar a estrutura dos rolos em si, é importante realizar uma distinção de nomenclatura entre os **roletes** e os **rolos**, que muitas vezes são confundidos. Segundo a norma NBR 6177 (ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS, 2016), denomina-se rolete o conjunto formado por um ou mais rolos montados em um suporte, cuja função é a de guiar e sustentar a correia transportadora. Por sua vez, ainda segundo a mesma norma, os rolos são elementos cilíndricos capazes de girar em seu próprio eixo com o objetivo de apoiar a correia. A Figura 11 apresenta um TC em campo destacando essas diferenças.

A classificação de um rolete é alterada de acordo com a sua aplicação, com as funções mais comuns de carga, impacto e retorno explicadas na seção anterior. Além delas, conforme ilustrado na Figura 12, o rolete pode ser do tipo auto alinhante (controle do deslocamento lateral da correia), de transição (posicionados próximos aos tambores para mudar a concavidade da correia), em catenária (suspensos com

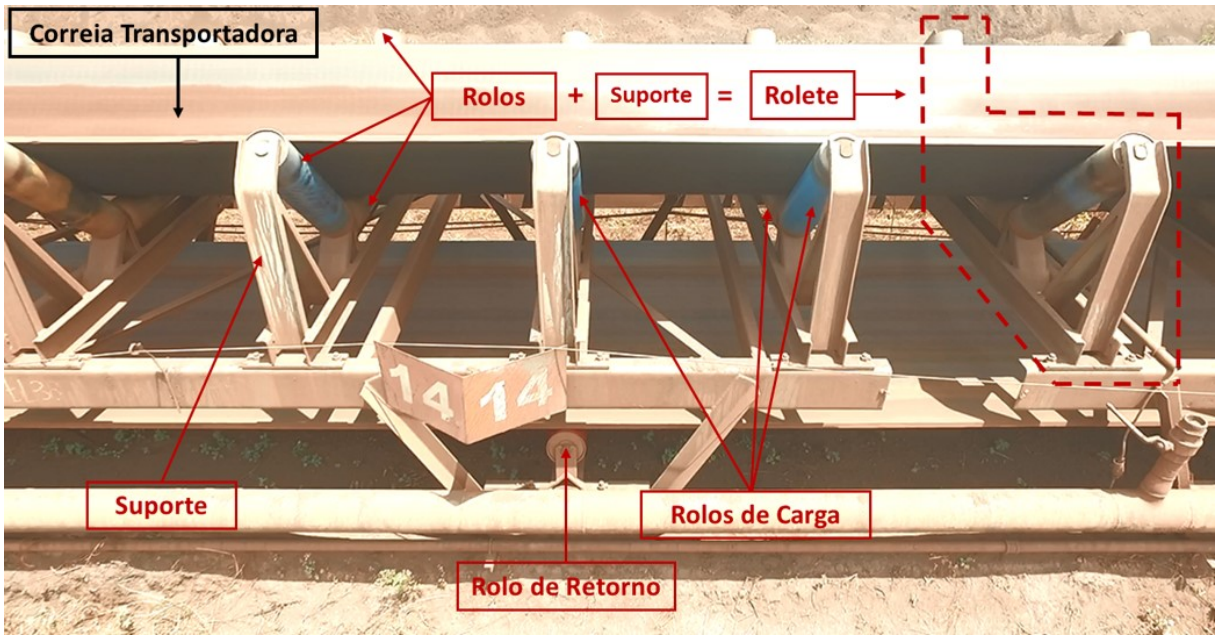



Figura 11 – Nomenclatura de alguns dos elementos do TC. Fonte: autor.

ligações articuladas para auxiliar na centralização da correia) e helicoidais (usado no retorno para desprender o material da correia) (ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS, 2016).

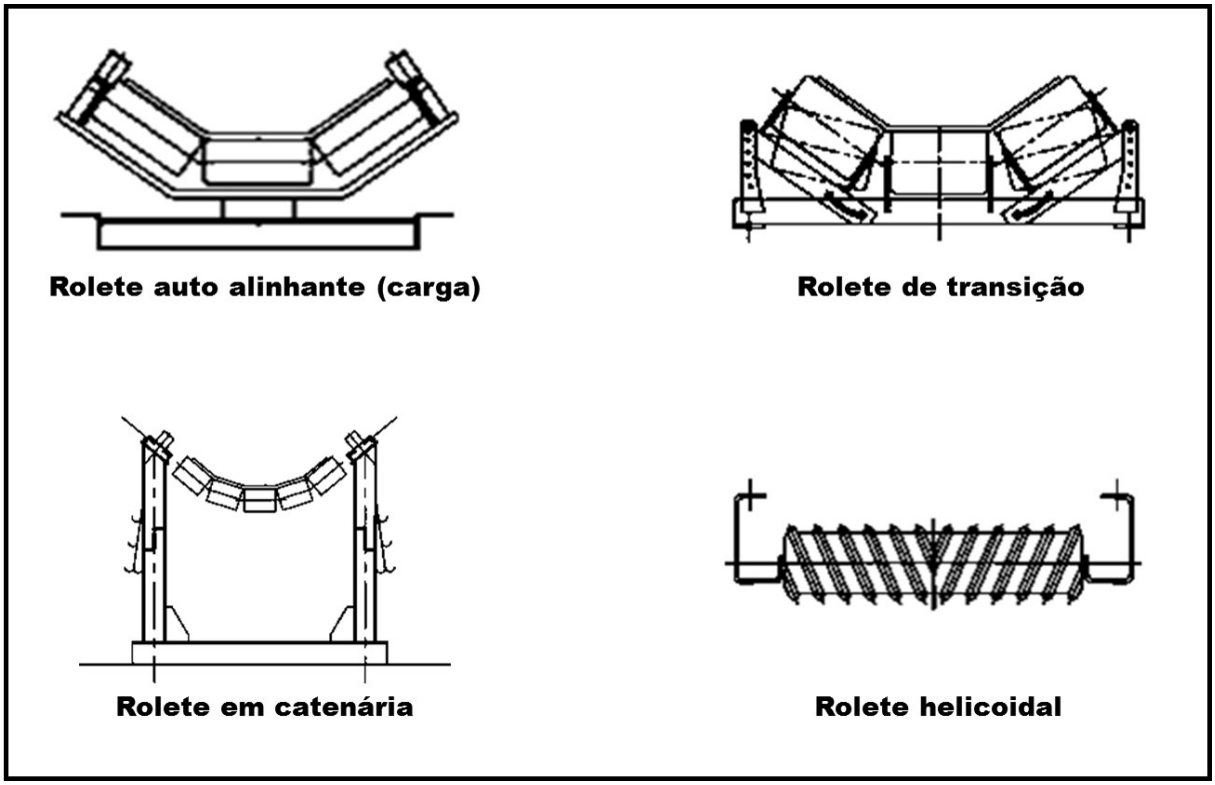
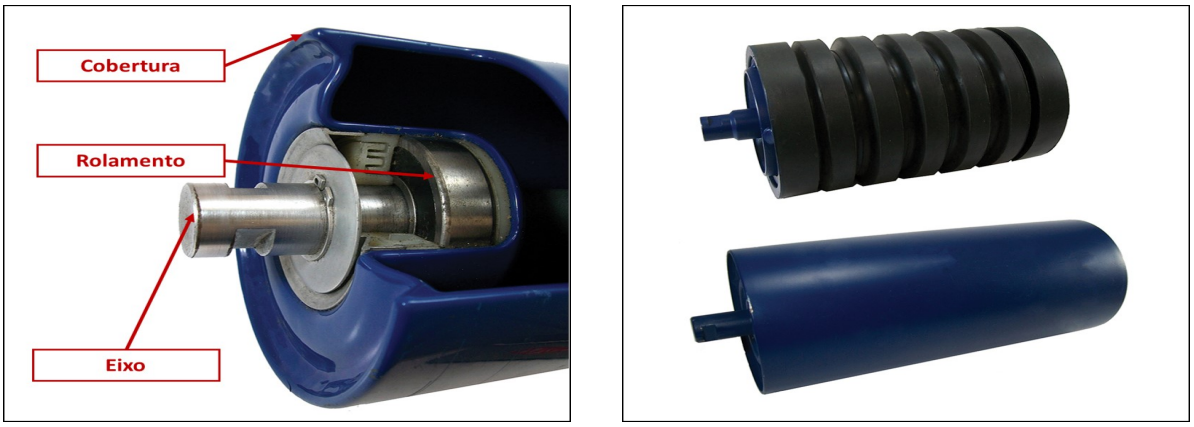



Figura 12 – Tipos especiais de roletes. Fonte: adaptado de ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS, 2016

Independentemente do tipo de rolete, os rolos são compostos por um eixo estacionário que liga dois rolamentos (esquerdo e direito) e um revestimento cilíndrico

geralmente metálico (REICKS, 2008), conforme mostrado na Figura 13a. Cabe destacar que os rolos de impacto normalmente recebem uma cobertura adicional de borracha (Figura 13b) para melhorar sua função de absorção de impacto, mas ainda assim mantém a estrutura básica previamente descrita.




(a) Principais partes que compõem um rolo

(b) Exemplos de um rolo de impacto (superior) e de carga (inferior)

Figura 13 – Partes principais dos rolos e exemplos de alguns tipos de rolos. Fonte: adaptado de HCD CONVEYORS, 2017

Por ser o elemento de contato direto com a correia, o rolo está suscetível ao desgaste natural e a diferentes tipos de falhas, que reduzem seu ciclo de vida. O **revestimento** está sujeito a rasgos e fissuras, causados pela degradação inerente ao uso, pelo atrito com um ponto específico da correia quando um rolamento trava ou pela queda de materiais perfurantes sobre ele, como o exemplo mostrado na Figura 14a. Independente da causa, é possível detectar danos no revestimento apenas pela observação de um inspetor, sem o uso de instrumentos especializados. Por sua vez, os **rolamentos** podem travar, superaquecer ou mesmo quebrar (Figura 14b) em virtude da deterioração ou de imperfeições em sua fabricação. Defeitos em rolamentos, especialmente em sua fase inicial, podem não ser detectáveis sem o uso de instrumentos especializados, o que potencializa o problema (REICKS, 2008).

Como os rolamentos são os componentes dos rolos que apresentam os maiores desafios na sua inspeção, o monitoramento deve primariamente ser capaz de avaliar o estado deles. Atualmente, os principais instrumentos disponíveis para esse fim baseiam-se nos mesmos princípios que permitem determinar a condição de qualquer equipamento com partes rotativas usando três tipos de sinais:

- **Acústico:** a interação entre os elementos internos do rolamento gera Emissões Acústicas (AE, do inglês *Acoustic Emissions*) que possuem uma frequência específica de acordo com a rotação de operação e a geometria do rola-

(a) Dano no revestimento do rolo (b) Rolamento com avarias

Figura 14 – Rolos inutilizado por conta de danos em seu revestimento e rolo com falha avançada no rolamento. Fonte: autor

mento. Falhas nos rolamentos alteram os padrões de AE de diferentes formas, permitindo detectar e caracterizar defeitos em uma fase bastante prematura (HAWMAN; GALINAITIS, 1988), o que define este sinal como preditivo. Em contrapartida, Hemmati, Orfali e Gadala (2016) reportam as dificuldades na obtenção dos sinais e no processamento necessário para extração dos atributos de interesse, especialmente em ambientes ruidosos. Ainda assim, esse tipo de sinal é muito usado, com instrumentos de inspeção específicos para sua obtenção, como o Ultraprobe 10,000 (UE SYSTEMS, 2017).

- **Térmico**: ainda que o rolamento possua lubrificação interna, existe um nível de atrito natural quando ele está girando, dissipado em forma de calor. A temperatura resultante do atrito varia em função de diversos fatores, como a taxa de operação do TC, a temperatura ambiente e mesmo a proximidade a outras fontes de calor. Falhas ou desgaste no rolamento ocasionam aumento da fricção entre suas partes, provocando uma elevação da temperatura em diferentes níveis de acordo com a gravidade da falha. Assim, a elevação da temperatura de um rolamento acima de algum limiar, geralmente a partir de 5º C em comparação aos rolamentos adjacentes, indica defeitos em andamento no rolo, sendo que quanto maior a temperatura medida, mais avançado o estágio de defeito do rolo (YANG, 2014), o que define esse sinal como o mais reativo dentre os disponíveis. Câmeras térmicas, como a Flir E60 (FLIR, 2017), são amplamente utilizadas para obtenção da temperatura de rolamentos durante inspeções.

- **Vibração**: de acordo com características específicas em sua construção (número de elementos rotativos internos, ângulo de contato e inclinação) e a rotação resultante da taxa de operação do TC, cada rolamento tem uma frequência

natural de vibração. Alterações nesta frequência são indicativos de defeitos, que também podem ser classificados de acordo com as assinaturas detectadas para determinar o elemento interno defeituoso e ainda o tipo de defeito (GIRDHAR; SCHEFFER, 2004). Esse sinal ainda pode ser considerado preditivo, visto que detecta defeitos em fases iniciais, porém, AlShammari e Addali (2015) ponderam que o sinal não é efetivo quando a rotação é menor que 100 rpm, visto que as frequências geradas abaixo desse limiar não são facilmente detectáveis. Comercialmente, há uma gama de produtos para medir a vibração de rolos, como o PCE-VT 2700S (PCE INSTRUMENTS, 2017).

Apesar de ser possível empregar esses três tipos de sinais para avaliar um rolamento e, desse modo, monitorar as condições de um rolo, cada um deles possui características específicas que precisam ser avaliadas ao se construir a solução de monitoramento. Tanto o sinal acústico quanto a vibração conseguem detectar defeitos em fases iniciais, mas o sinal acústico, mesmo que em baixas velocidades de rotação, consegue fazê-lo de forma mais prematura, quando as frequências emitidas sequer são audíveis (TANDON; CHOUDHURY, 1999). Além disso, segundo Hecke, He e Qu (2014), o sinal acústico tem melhores condições de classificar a falha quando comparado à vibração. Por fim, ambos os sinais trazem consigo desafios na captura e processamento, visto que a operação ocorre em ambientes ruidosos e com fontes de vibração diversas.

Ainda que o sinal térmico sofra interferências relativas às condições ambientais e seja o mais reativo, visto que a elevação de temperatura ocorre quando o defeito já está afetando de forma significativa o rolamento (YANG, 2014), é o que possui captura mais simples, ou seja, com menos processamento de sinais, o que o qualifica para ser empregado em sistemas relativamente simples.

Assim, a escolha do sinal a ser usado no sistema de monitoramento de condições depende da avaliação destas características e das capacidades do sistema a ser criado, como capacidade de aquisição do sinal em questão, a quantidade de recursos computacionais disponíveis para o processamento de sinais, dentre outros aspectos. O Capítulo 3 apresenta algumas das plataformas de monitoramento de rolos que estão atualmente disponíveis, considerando os sinais adotados por cada uma delas e apresentando aspectos positivos e negativos de cada uma delas.

## 2.3 Conclusões do Capítulo

Este capítulo apresentou a fundamentação teórica sobre os transportadores de correias e rolos, que consistem o problema abordado no trabalho. Foram discutidos

os tipos mais comuns de TCs encontrados nas operações, ressaltando suas características mais marcantes e o objetivo de negócio que cada tipo de transportador procura suprir. Com base nas informações apresentadas, fica clara a importância deste equipamento para operações que lidam com materiais a granel, particularmente empresas de mineração.

Também foram apresentados os principais componentes que compõem um TC, com uma breve discussão sobre cada um deles. Foi dada atenção especial aos rolos, particularmente os de carga e de retorno, que representam o maior desafio de monitoramento por serem numerosos e geograficamente dispersos. Na sequência, foram discutidos os modos de falha destes componentes, onde pontuou-se que a maioria das falhas ocorre nos dois rolamentos que cada rolo possui, ainda que também ocorram quebras e perfurações na cobertura. Por fim, foram discutidos os três principais sinais disponíveis para monitoramento de condições dos rolos: acústico, térmico e vibração.

Com base no exposto, conclui-se que qualquer tipo de solução desenvolvida para o monitoramento de rolos de transportadores de correias deve ser capaz de avaliar a condição de seus rolamentos por meio dos sinais disponíveis. Por conta disso, o capítulo a seguir apresenta e discute as principais técnicas atualmente disponíveis para o monitoramento de condições de rolos, definindo qual o tipo de abordagem seguido neste trabalho.

# 3 Trabalhos Relacionados

Neste capítulo são apresentadas as principais técnicas de monitoramento de condições de rolos discutidas na literatura. Ele está organizado da seguinte forma:

- A seção "**3.1 - Soluções para o Monitoramento de Condições de Rolos**" apresenta as principais soluções, comerciais ou laboratoriais, disponíveis para realizar o monitoramento dos rolos;
- Por sua vez, a seção "**3.2 - Discussão da Abordagem para a Plataforma de Inspeção**" apresenta um resumo comparativo entre as técnicas disponíveis e discute a abordagem adotada neste trabalho;
- Por fim, a seção "**3.3 - Conclusões do Capítulo**" revisa os principais pontos abordados na seção e apresenta as conclusões relativas às soluções de monitoramento.

## 3.1 Soluções para o Monitoramento de Condições de Rolos

Dada a relevância dos Transportadores de Correia para a indústria mineral, os rolos, que são um dos componentes mais críticos e suscetíveis a falhas, são amplamente discutidos na literatura, com diferentes abordagens propostas para realizar o monitoramento de sua condição.

Entretanto, ainda há variação em relação ao nível de desenvolvimento e viabilidade de cada uma das propostas. Já é possível encontrar no mercado soluções maduras, cuja viabilidade de adoção depende unicamente de fatores comerciais e operacionais, enquanto outras ainda estão em desenvolvimento. Porém, mesmo as soluções maduras ainda trazem consigo restrições que precisam ser observadas.

Desta forma, a Figura 15 apresenta um resumo das principais soluções para o monitoramento de condições de rolos e as próximas subseções discutem os três principais grupos de soluções.

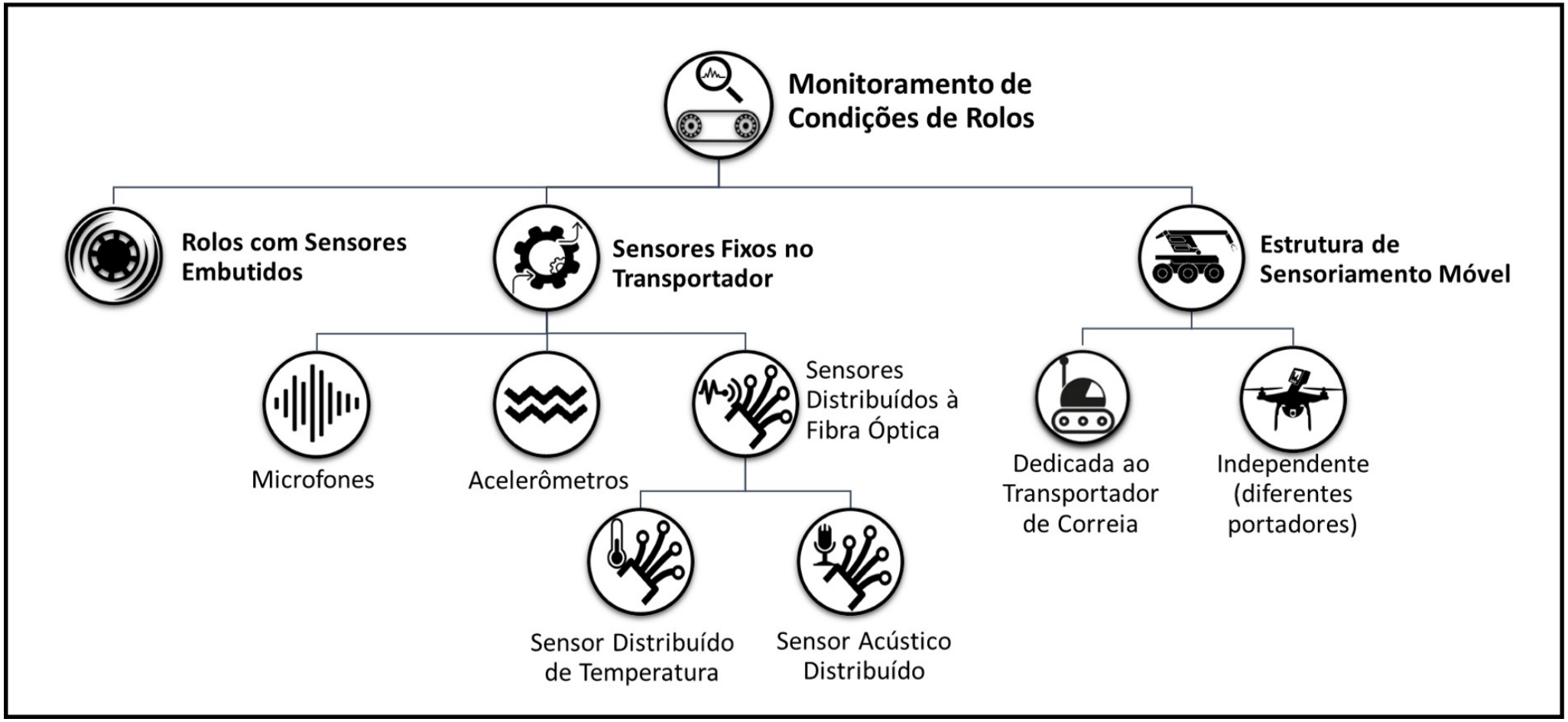



Figura 15 – Principais técnicas e soluções para o monitoramento de condições de rolos. Fonte: autor

### 3.1.1 Rolos com Sensores Embutidos

Dentre os tipos de soluções disponíveis, o primeiro grupo apresentado na Figura 15 é o que trata de embutir o sensoriamento diretamente no rolo. As propostas de Lodewijks et al. (2016) e Norris e Moutzouris (2014) incorporam a um rolo tradicional diferentes tipos de sensores (acústico, térmico e de vibração) e *hardware* de comunicação, dotando-o de capacidades para capturar e transmitir sua condição para um controlador central. Alguns modelos mais sofisticados contam ainda com mecanismos de captura de energia a partir do movimento do rolo. Com essa abordagem, o rolo, que é tradicionalmente passivo, torna-se um dispositivo inteligente, que pode ser visto como parte de uma rede de sensores sem fio (WSN, do inglês *Wireless Sensor Network*). Por conta disso, também é chamado de "*Smart Idler*" por um dos fabricantes que comercializa o produto, a Vayeron Pty (2016) (Figura 16).

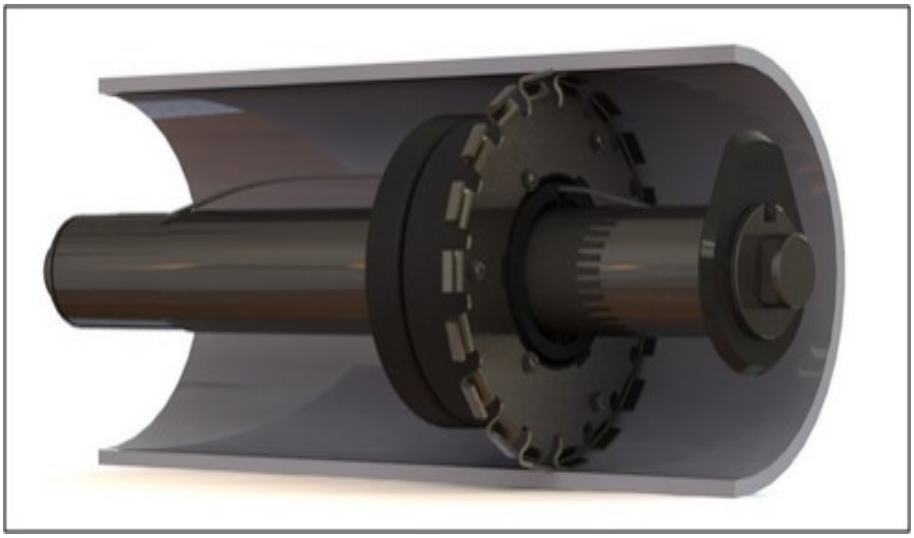


Figura 16 – Esquema da parte interna de um rolo inteligente. Fonte: VAYERON PTY, 2016

Com capacidade de medir, identificar e reportar condições anormais em seus componentes internos e também no revestimento, essa solução consegue endereçar os requisitos de monitoramento individual dos rolos e antecipar falhas, dado o uso de sinais preditivos, como o acústico (VAYERON PTY, 2016). Porém, para que se consiga monitorar completamente um TC, todos os rolos já instalados devem ser substituídos pelo modelo inteligente, pois basta que um único rolo sem a tecnologia falhe para que o TC como um todo seja colocado em risco. Logo, sua adoção tende a ser mais vantajosa em novas operações, que já sejam comissionadas com o uso exclusivo deste tipo de rolo, ou ainda em TCs com grande histórico de problemas, como a própria Vayeron Pty (2016) recomenda. Para operações existentes, embora ainda possível, é menos razoável justificar a troca em massa por modelos inteligentes dado o grande volume de componentes e, consequentemente, dos

custos envolvidos, embora não tenha sido possível obter informações confiáveis a esse respeito para uma melhor discussão.

### 3.1.2 Sensores Fixos no Transportador

O segundo grupo mostrado na Figura 15 traz soluções que propõem o uso de sensores instalados na estrutura do TC como uma forma de reduzir a quantidade de sensores envolvidos, monitorando múltiplos rolos por vez. Li et al. (2013) propõem a instalação de acelerômetros ao longo do TC para capturar a vibração (Figura 17), *Wavelet Packet Decomposition* (WPD) (COIFMAN; WICKERHAUSER, 1992) para isolar e identificar as frequências de interesse e *Support Vector Machine* (SVM) (CORTES; VAPNIK, 1995) para classificar a falha. De forma similar, Tan et al. (2015) usam redes neurais e classificadores para reconhecer diferentes tipos de falha com base na vibração. O sinal acústico também pode ser usado. Um exemplo é a proposta de Jiang e Cao (2015), com a aplicação de *Wavelet Transform* para detectar falhas em frequências ultrassônicas nos rolos.

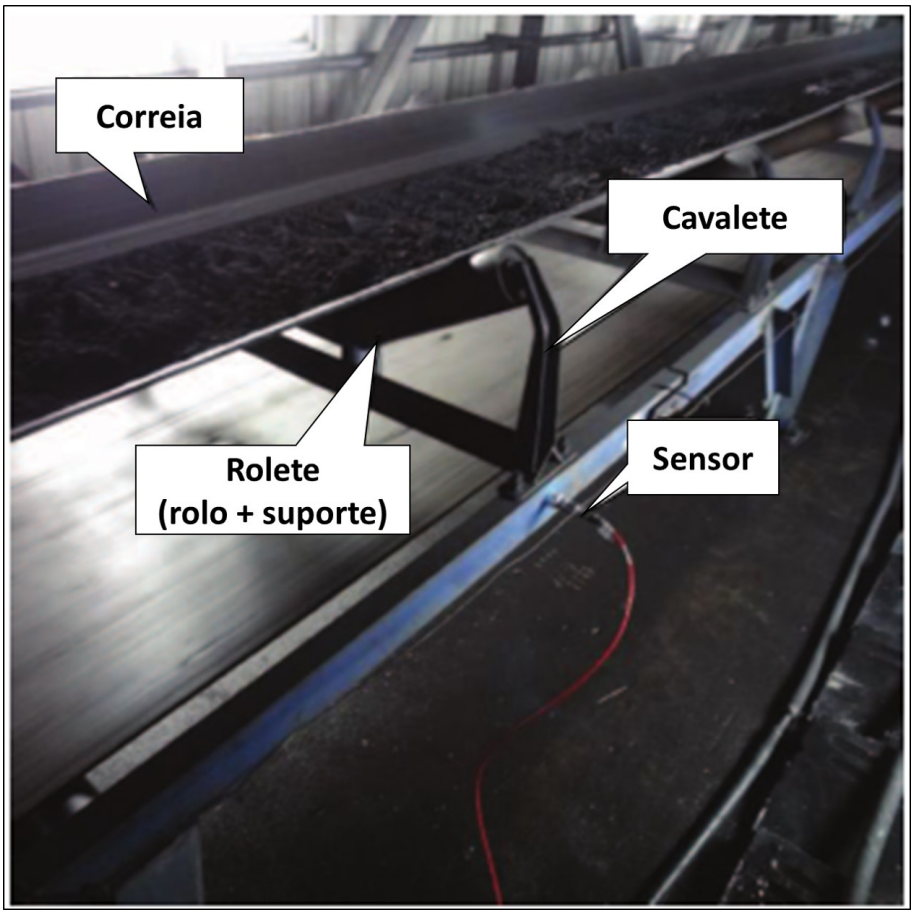



Figura 17 – Solução de monitoramento com uso de acelerômetros na estrutura do TC. Fonte: traduzido de LI et al., 2013

Independentemente do sinal adotado, acústico ou vibração, o processamento traz grandes desafios para isolar interferências e ainda identificar o rolo que está ori-

ginando o sinal de defeito. Além disso, a instalação de microfones e acelerômetros ao longo do TC pode não ser prática e gerar necessidade de manutenção desses dispositivos, o que potencializa o problema de inspeção já existente por conta da adição de novos elementos. No caso da solução de Li et al. (2013), foi instalado um acelerômetro a cada 6 metros, o que ainda é um número significativo quando se considera que existem TCs que se estendem por quilômetros.

Como uma forma de não aumentar o número de elementos sensores proporcionalmente à extensão do TC, um outro tipo de solução possível, que também se enquadra no caso de sensoriamento instalado na estrutura do TC, é o uso de Sensores Distribuídos à Fibra Óptica (DOFS, do inglês *Distributed Optical Fiber Sensors*). Esse tipo de solução emprega o cabo de fibra ótica tanto como elemento de sensoriamento quanto meio de transmissão. De forma resumida, as tecnologias baseiam-se na medição de diferentes tipos de espectros de espalhamento, conhecidos como *Raman*, *Brillouin* e *Rayleigh*, que naturalmente ocorrem à medida que um feixe de luz se propaga em um cabo de fibra óptica (ROGERS, 1988).

Elementos externos ao cabo, como vibração, ondas sonoras e mesmo variações de temperatura interferem de diferentes formas nos espectros citados, permitindo, assim, inferir o valor proporcional à grandeza da perturbação e o ponto de origem com diferentes resoluções (ou grau de precisão). Para isso, empregam-se diferentes princípios para se medir o espectro de interesse, como a OTDR, do inglês *Optical Time Domain Reflectometry*, a OFDR (*Optical Frequency Domain Reflectometry*), a OLCR (*Optical Low Coherence Reflectometry*), que são amplamente discutidas no trabalho de Ribeiro (2011), e a C-OTDR, (*Coherent-Optical Time Domain Reflectometry*), proposta por King et al. (1987).

Atualmente, o meio mais comum de monitoramento de rolos usando fibras ópticas é conhecido como Sensor Distribuído de Temperatura (ou DTS, do inglês *Distributed Temperature System*). Usando o OTDR, Hu et al. (2011) testaram um sistema DTS de 10 km de extensão em uma mina subterrânea de carvão e conseguiram medir variações de temperatura com incerteza de 2 ºC e resolução espacial de 3 metros. Porém, os autores relataram a influência de fatores ambientais que afetaram as medições, concluindo a necessidade de uma calibração meticulosa e específica para cada TC como principal ponto negativo do DTS.

Com essas dificuldades em mente, Yang (2014) avaliou diferentes materiais, conectores e arranjos de cabo para isolar os elementos influenciadores na medição da temperatura. O autor concluiu que existe uma troca entre a complexidade da instalação, a acurácia na temperatura medida e o tempo de resposta do sistema, sugerindo a instalação do cabo de fibra óptica com contato direto com cada rolamento, conforme esquema na Figura 18. O OTDR, usado nos sistemas DTS, ainda

vem recebendo melhorias, como no trabalho de Rosolem et al. (2017) e existem diferentes produtos no mercado (AP SENSING, 2017; LIOS TECHNOLOGY, 2017) baseados na tecnologia para medições de temperatura com variada precisão e resolução espacial.

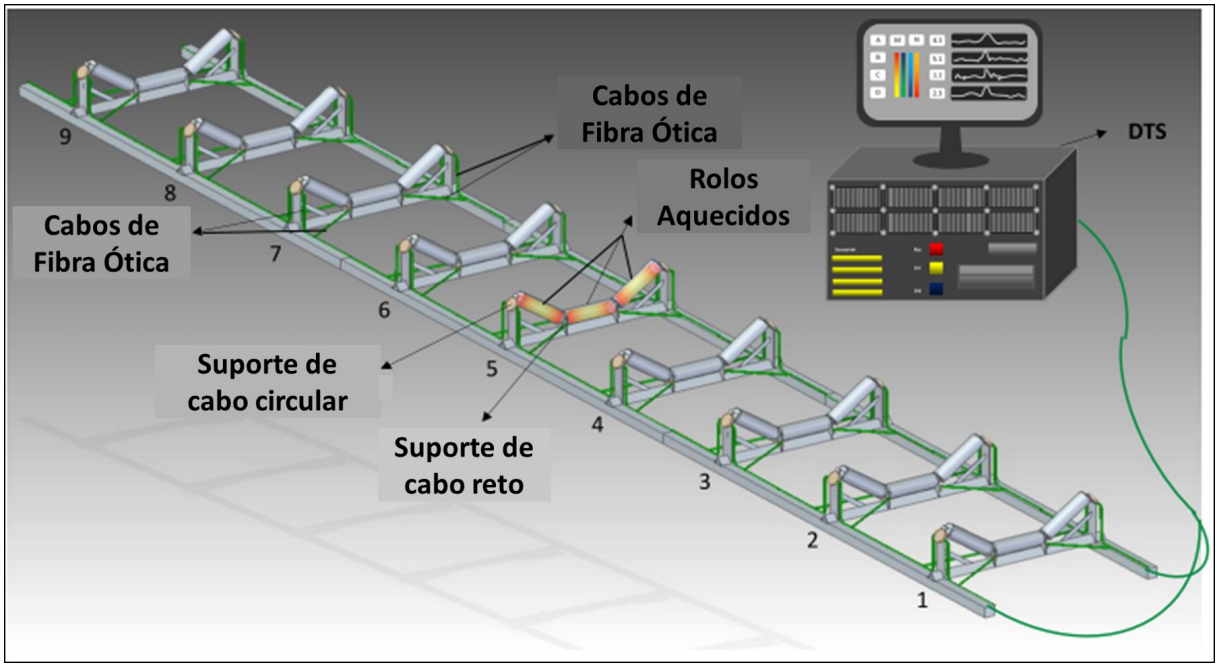



Figura 18 – Sistema DTS para monitoramento da temperatura de rolos por fibra óptica. Fonte: traduzido de YANG, 2014

Outra abordagem de monitoramento de rolos usando fibra óptica é conhecida como Sensor Acústico Distribuído (DAS, do inglês *Distributed Acoustic Sensor*). Se comparado ao DTS, seu uso ainda é relativamente recente (MARTINS et al., 2017), com pesquisas ainda em andamento, como a de Hicke et al. (2017), que investiga a sua viabilidade para o monitoramento de rolos e outras aplicações industriais, e testes em campo, como os executados por Wilson (2017). Neste trabalho, o autor destaca algumas das dificuldades no uso do DAS, como a sensibilidade do cabo de fibra a ruídos e o quanto isso é potencializado pelo fato dos TCs serem naturalmente ruidosos. Com isso, foi necessário aperfeiçoar o processamento de sinais, atingindo-se recentemente um estágio em que é possível detectar e acompanhar com meses de antecedência a evolução de falhas em elementos internos do rolamento. Apesar da tecnologia ainda não estar disponível comercialmente, essa característica é um diferencial em relação a outros métodos, pois permite que a equipe de manutenção monitore o comportamento e se planeje para efetuar a troca apenas quando de fato for necessário, maximizando o uso dos rolos sem colocar em risco a operação.

### 3.1.3 Estrutura de Sensoriamento Móvel

O último grupo de soluções para o monitoramento de condições de rolos mostrado na Figura 15 é o que propõe soluções móveis. Neste caso, a estrutura de sensoriamento pode estar montada e, portanto, dedicada a um único TC ou ser uma estrutura de sensoriamento independente, que pode ser transportada por diferentes tipos de portadores e, assim, ser utilizada para inspecionar mais de um TC, ainda que um por vez.

Seguindo a primeira abordagem, Yang, Zhang e Ma (2016) apresentam um robô dotado de câmera termográfica que se move por dentro da estrutura do TC (Figura 19) e consegue capturar imagens de diferentes componentes. Algoritmos de processamento de imagens são aplicados para encontrar as regiões de interesse, classificar os componentes encontrados em rolos, motores ou tambores e extrair a informação da temperatura do mesmo. Após essa etapa, os dados coletados são comparados a um banco de dados, que possui valores de referência de temperatura para cada tipo de componente, o que permite classificar a temperatura em diferentes níveis (normal, alerta ou defeito).

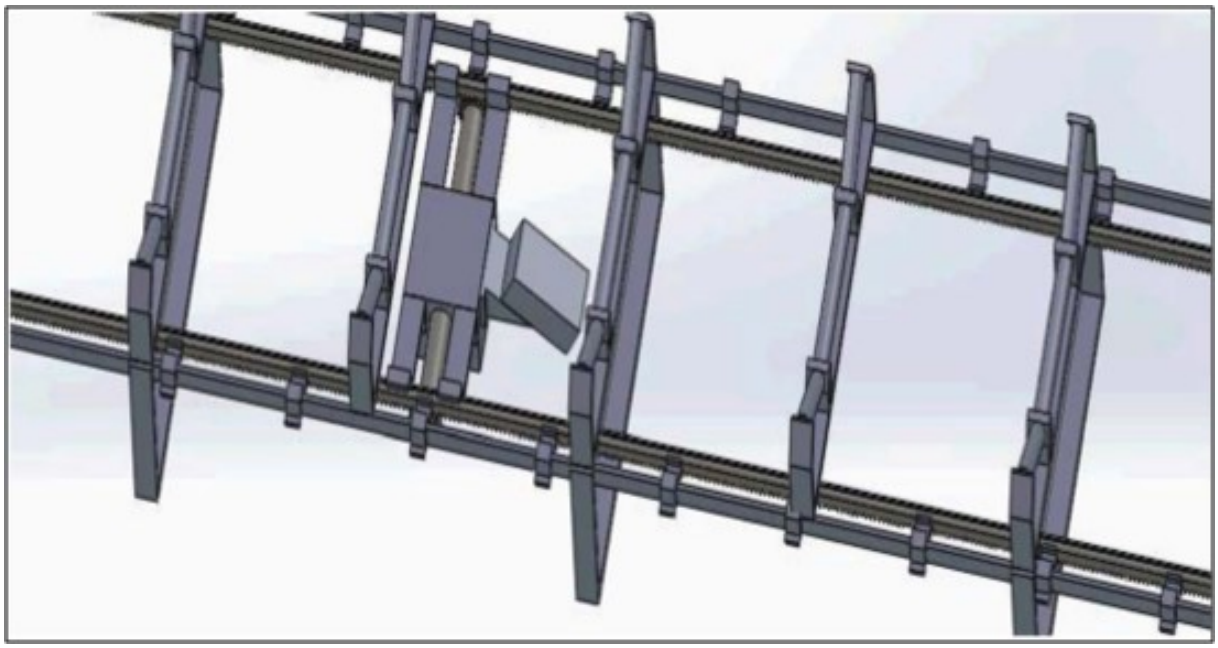


Figura 19 – Robô móvel deslocando-se pela estrutura interna do TC. Fonte: YANG; ZHANG; MA, 2016

Embora a abordagem seja promissora e permita um monitoramento praticamente constante dos rolos, dependendo da velocidade e frequência com que o robô irá se deslocar no TC, a proposta de Yang, Zhang e Ma (2016) esbarra nas dificuldades inerentes à adaptação do TC para que o robô possa se deslocar. Em TCs existentes, nem sempre é possível ter a parte interna livre devido à instalação de cabos e estruturas metálicas auxiliares. Além disso, em algumas situações, as

partes internas do TC podem ser bloqueadas pelo acúmulo de material que cai da correia, o que pode restringir ou inviabilizar a solução.

Isto posto, uma forma de minimizar essas limitações, é adotar uma solução de sensoriamento que seja externa ao TC. O último tipo de solução mostrado na Figura 15 representa soluções independentes, que não estão acopladas a um TC e, portanto, podem ser usadas para inspecionar diferentes TCs. É essa a linha adotada pela ABB Technology AG (2014), que descreve uma estrutura de sensoriamento independente que pode ser transportada por um robô terrestre, por um robô que se desloca em um trilho paralelo ao transportador ou por um VANT, conforme Figura 20. A estrutura de sensoriamento é composta por uma câmera térmica e microfones, o que a habilita capturar a temperatura atual e predizer falhas usando o sinal acústico.

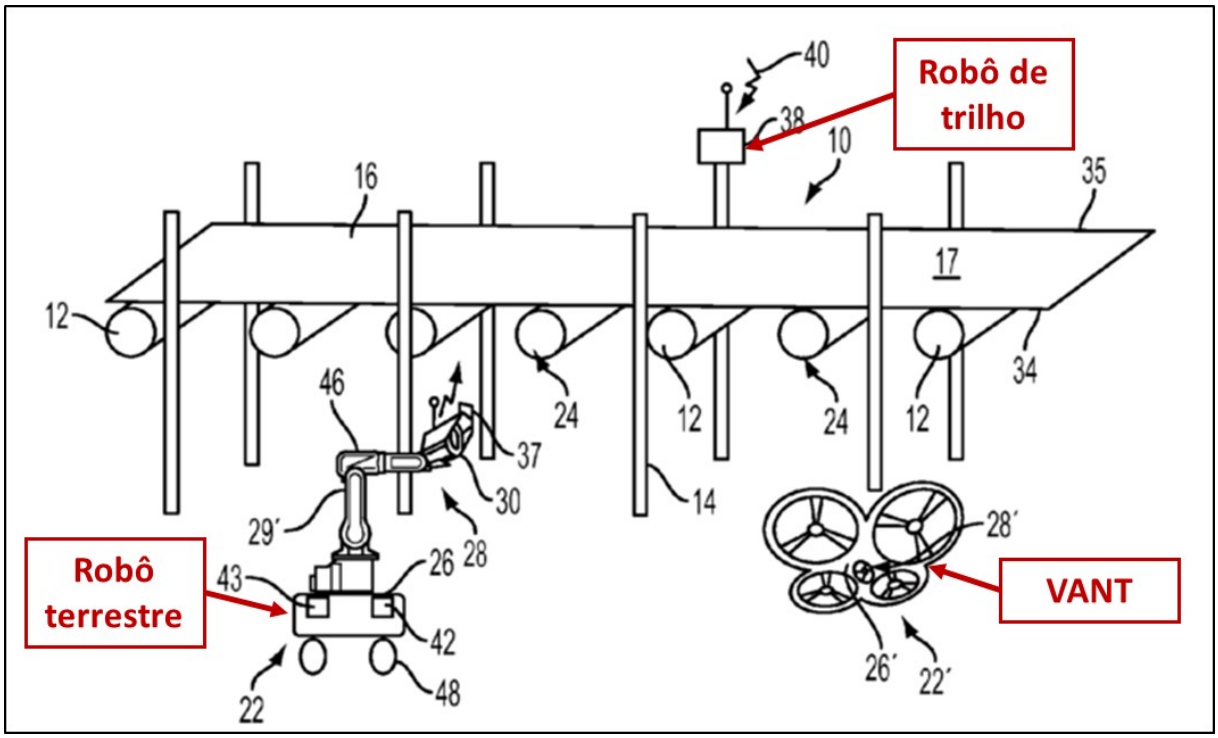



Figura 20 – Diferentes tipos de portadores propostos na solução de monitoramento da ABB. Fonte: adaptado de ABB Technology AG, 2014

De forma similar, Yong et al. (2014) descrevem um sistema composto por um VANT autônomo que transporta uma estrutura de sensores, formada por uma câmera térmica, um sensor de RFID e uma câmera de alta-resolução, além de uma estação terrestre para processamento de dados. A inovação dos autores foi o uso de fitas reflexivas coladas na estrutura do TC, no chão e em outros equipamentos para orientar a navegação do VANT durante a inspeção. Dessa forma, a câmera é usada para reconhecer as fitas e informar ao VANT qual ação deve ser realizada, por exemplo, subir ou continuar o voo de forma horizontal, paralelamente ao TC. Todos os dados capturados pelo VANT são transmitidos para a estação terrestre, que

é responsável por analisa-los e avaliar se existe algum defeito na posição. Um ponto negativo específico dessa abordagem é o fato dos autores terem usados etiquetas de identificação por radiofrequência (RFID, do inglês *Radio Frequency Identification*) para identificar a posição do rolo sendo inspecionado. Em um cenário real de uma grande operação, que pode ter mais de 200.000 rolos, a instalação desse elemento adicional em cada rolo pode ser onerosa.

Os métodos de inspeção que se baseiam em Estruturas de Sensoriamento Móvel trazem consigo as desvantagens dos portadores escolhidos. Por exemplo, o uso de VANTs tem como limitação a autonomia de bateria e o peso máximo transportável (*payload*); veículos ou robôs terrestres podem ter dificuldades em acessar os locais onde os TCs operam, com a presença de escadas e espaços fechados; robôs que se movem em trilhos interna ou externamente ao TC requerem manutenção, como lubrificação, e adaptações nas estruturas existentes; inspetores carregando a estrutura de sensores são geralmente lentos e ficam expostos a ambientes insalubres. Além disso, essas estruturas não conseguem prover monitoramento *online*, mas ciclos de inspeção que também irão variar de acordo com o portador.

Por outro lado, essas estruturas são as que podem ser colocadas em operação de forma mais rápida, sem um investimento inicial significativo. No caso da Estrutura de Sensoriamento Móvel e Independente, é possível ainda que uma mesma estrutura inspecione múltiplos TCs, o que aumenta a flexibilidade e também contribui na redução do custo com sensoriamento. Além disso, cada portador pode ter características mais adequadas para cada tipo de situação, por exemplo, o VANT para inspeção de TCs em pátios abertos e o robô terrestre para inspeção de TCs em áreas fechadas. Assim, esse tipo de solução também é positiva em termos de adaptabilidade.

## 3.2 Discussão da Abordagem para a Plataforma de Inspeção

Conforme foi visto ao longo da seção anterior, cada um dos grupos de soluções apresentado possui aspectos positivos e negativos, que devem ser levados em conta na definição do tipo de solução para o monitoramento de rolos. A Tabela 2 apresenta uma visão consolidada e resumida destes pontos.

Tabela 2 – Principais características dos grupos de soluções de monitoramento de condições de rolos

| Grupo de Solução de Monitoramento | Exemplos de Soluções Neste Grupo | Aspectos Positivos | Aspectos Negativos |
|---|---|---|---|
| **Rolos com Sensores Embutidos** | - Soluções comercialmente chamadas de *smart-idlers* (ou rolos inteligentes). | - Monitoramento online e individual dos rolos;<br>- Múltiplos sensores para monitoramento preditivo;<br>- Identificação individual dos rolos. | - Custos de instalação;<br>- Requer substituição de todos os rolos já instalados para monitorar completamente um TC. |
| **Sensores Fixos no Transportador** | - Sensores tradicionais: microfones e acelerômetros;<br>- Soluções de monitoramento baseados em fibra ótica (Temperatura ou Acústico). | - Monitoramento online;<br>- Algumas soluções conseguem monitorar individualmente os rolos;<br>- Soluções baseadas em fibra reduzem o número de sensores, o que é interessante para TCs longos. | - Novos sensores podem aumentar a necessidade de manutenção;<br>- Sensores baseados em fibra ainda possuem calibração complexa e sofrem com interferências. |
| **Estrutura de Sensoriamento Móvel** | - Sensores montados em *cable/rail-drones* (internos ou externos à estrutura do TC);<br>- Estrutura de sensoriamento independente e instalável em portadores móveis, como VANTs, robôs terrestres, etc. | - Menores custos iniciais;<br>- Adaptabilidade: mesma estrutura de sensoriamento pode ser levada por diferentes portadores;<br>- Flexibilidade: uma estrutura pode monitorar múltiplos TCs. | - Dificuldade de adaptação do TC em soluções dedicadas;<br>- Monitoramento apenas durante o ciclo de inspeção (variável);<br>- Cada portador tem suas próprias limitações. |

Fonte: autor

Assim, com base nas características apresentadas na Tabela 2, percebe-se que o grupo de **Rolos com Sensores Embutidos** e o de **Sensores Fixos no Transportador**, particularmente as soluções baseadas em fibra ótica, são os melhores tecnicamente, por prover monitoramento constante e, em algumas abordagens, preditivo, permitindo que a área de manutenção se planeje melhor para fazer a substituição. Por conta disso, a longo prazo, endereçadas as questões de custos e calibração, essas são as soluções que devem dominar o mercado.

Por outro lado, as classificadas como **Estruturas de Sensoriamento Móvel**, especialmente as Independentes, possuem um custo inicial menor, além de vantagens em termos de adaptabilidade e flexibilidade, o que as caracterizam como uma boa opção para uma fase de transição, até que todos os rolos tenham sensores embutidos ou monitoramento por fibra. Por conta disso, essa foi a abordagem adotada neste trabalho, visto que essas soluções são capazes de melhorar a qualidade da inspeção e ainda reduzir a exposição dos trabalhadores a riscos, o que endereça os principais problemas presentes na prática atual, puramente sensitiva.

Como as **Estruturas de Sensoriamento Móvel** não monitoram de forma constante os rolos, é importante que as temperaturas medidas e eventuais defeitos encontrados sejam imediatamente reportados para a equipe de manutenção. Por conta disso, este trabalho se diferencia das demais abordagens ao propor uma Plataforma de Inspeção que seja integrada aos sistemas corporativos, reduzindo o atraso entre a detecção de um defeito em campo e o mesmo ser visualizado pelas equipes responsáveis pela correção. Além disso, optou-se para este trabalho o uso de VANTs como portadores, mas a arquitetura aqui proposta poderia ser adotada por outros tipos de portadores que se adequem a condições específicas de cada TC, o que é outro diferencial da abordagem adotada neste trabalho.

## 3.3 Conclusões do Capítulo

O presente capítulo apresentou uma revisão das principais técnicas e soluções disponíveis para o monitoramento de condições de rolos de transportadores de correia. Dada a complexidade do problema, é notável que as soluções existentes não estão totalmente consolidadas, considerando que algumas delas estão em estágio laboratorial ou em escala industrial limitada. Esta constatação reforça a necessidade de se propor novas soluções ou técnicas, uma vez que o problema está de fato em aberto.

Todas as soluções avaliadas com base na literatura utilizam pelo menos um dos sinais disponíveis (acústico, térmico e vibração) para monitoramento do TC, con-

forme discutido em "**2.2 - Roletes, Rolos e Detecção de Falhas**". Dessa forma, o que as diferencia é o tipo de abordagem em relação ao posicionamento dos sensores para obtenção destes sinais: nos rolos, no transportador ou em um portador externo, dedicado ou não ao TC. Assim, as soluções avaliadas foram agrupadas e explicadas seguindo essas possibilidades: "**Rolos com Sensores Embutidos**", "**Sensores Fixos no Transportador**" e "**Estrutura de Sensoriamento Móvel**".

O primeiro grupo é o que tem maior potencial a longo prazo, pois consegue monitorar individualmente cada um dos rolos, ainda que tenha uma curva de adoção íngreme, já que cada um dos rolos existentes precisa ser substituído. O segundo grupo procura atingir o monitoramento em tempo real utilizando um número limitado de sensores, que não cresça proporcionalmente ao número de rolos. Neste caso, destacam-se as soluções baseadas em fibra ótica, especialmente a baseada no sinal acústico, que consegue antecipar falhas. Resolvidas as questões de calibração ainda em aberto, tem grande potencial. Por fim, o último grupo traz uma abordagem diferente, onde o sensoriamento deixa de estar no TC ou nos rolos e passa a ser carregado por um portador, o que aumenta a flexibilidade e tende a reduzir os custos iniciais, ainda que não se obtenha monitoramento em tempo real de todos os rolos.

Isto posto, a seção "**3.2 - Discussão da Abordagem para a Plataforma de Inspeção**" apresentou um comparativo desses grupos de solução, com seus principais aspectos positivos e negativos. Nessa comparação, conclui-se que a abordagem proposta para o presente trabalho encaixa-se no grupo de "**Estrutura de Sensoriamento Móvel**", diferenciando-se dos demais trabalhos ao propor uma Plataforma de Inspeção flexível, que permita adotar diferentes tipos de portadores e sensores, adequados às variadas características encontradas nas operações. Outro diferencial é a integração com sistemas existentes como uma forma de melhorar o processo de inspeção de rolos e reduzir o tempo entre a detecção das falhas em campo e a ciência das equipes de manutenção. Dessa forma, o capítulo a seguir descreve as tecnologias utilizadas para propor uma Arquitetura Integrada que suporte tais requisitos.

# 4 Tecnologias

Este capítulo discute os conceitos e tecnologias empregados para definição da Arquitetura Integrada e na criação dos protótipos para sua validação. Ele foi dividido divido em cinco seções:

- A seção "**4.1 - Veículos Aéreos Não Tripulados (VANTs)**" apresenta os VANTs, que foram os portadores da estrutura de sensoriamento utilizados nos protótipos da Plataforma de Inspeção. Optou-se neste trabalho a dedicar uma seção inteira a eles dada a novidade do seu uso para esse tipo de aplicação, ainda que outros portadores possam ser igualmente utilizados;
- Na sequência, a seção "**4.2 - Paradigmas de Computação: *Edge*, *Fog* e *Cloud***" apresenta brevemente conceitos que foram usados para definir pontos de processamento dentro da Arquitetura Integrada proposta;
- Já a seção "**4.3 - Técnicas e Protocolos de Integração**" discute conceitos envolvidos na integração da Plataforma de Inspeção com sistemas existentes e os fatores que levaram à escolha dos protocolos de comunicação adotados nos protótipos para suportar os diferentes tipos de interação;
- Por sua vez, a seção "**4.4 - Ferramental para Protótipos e Simulação**" detalha alguns dos conceitos das ferramentas usadas na construção dos protótipos, como o uso de Simulação e os *kits* de desenvolvimento;
- Por fim, a seção "**4.5 - Conclusões do Capítulo**" revisa os pontos centrais abordados ao longo do capítulo e apresenta as conclusões.

## 4.1 Veículos Aéreos Não Tripulados (VANTs)

Os VANTs ou UAVs, da sigla em inglês *Unmanned Aerial Vehicles*, vem promovendo uma verdadeira revolução em diferentes áreas em que podem ser usados, como agricultura de precisão (GONZALEZ-DUGO et al., 2013), inspeção de linhas de transmissão (LUQUE-VEGA et al., 2014), monitoramento de painéis solares (MESAS-CARRASCOSA et al., 2017), e na indústria de cinema e fotografia (FLEUREAU et al., 2016). Apesar dos desafios relativos à regulamentação e privacidade, que ainda são um entrave para uma adoção ainda mais abrangente, os VANTs vem se integrando à sociedade de maneira aparentemente irreversível.

Dentre as inúmeras áreas que podem se beneficiar dos VANTs, é crescente a sua utilização na indústria da mineração, por exemplo, para realizar mapeamento 3D de grandes áreas, inspeção de regiões perigosas ou inacessíveis e ainda na avaliação de presença humana antes do desmonte usando explosivos em uma frente de lavra (CREAGH, 2017). Este trabalho propõe uma nova aplicação no contexto da mineração, que é o uso de VANTs como um dos possíveis portadores de uma estrutura de sensoriamento para a inspeção e monitoramento de transportadores de correias e seus rolos, formando uma Plataforma de Inspeção. Sua flexibilidade, agilidade e outras características, melhor detalhadas ao longo do texto, os caracterizam como uma ótima ferramenta para essa atividade.

É importante destacar que a definição de VANT/UAV usada neste trabalho é a do departamento de Defesa Norte-Americano, que é geralmente a mais aceita e referenciada por outros autores. Dessa forma, segundo o Department of Defense (2001), VANT é um:

> "Veículo aéreo motorizado que não transporta um operador humano, usa forças aerodinâmicas para a sustentação aérea, pode voar de maneira autônoma ou ser pilotado por controle remoto, pode ser descartável ou recuperável e pode transportar uma carga útil letal ou não-letal. Veículos balísticos ou semi-balísticos, mísseis de cruzeiro e projéteis de artilharia não são considerados veículos aéreos não tripulados."

Tendo como referência esta definição, a presente seção aborda alguns temas relativos a VANTs, utilizados como portadores nos protótipos construídos para validação da Arquitetura Integrada discutida neste trabalho. Para tanto, o texto foi organizado da seguinte forma:

- A subseção "**4.1.1 - Um Breve Histórico dos VANTs**" aborda a evolução destas aeronaves, seu uso inicialmente para fins militares e a recente popularização;
- Na sequência, a subseção "**4.1.2 - Sistema VANT**" apresenta as principais partes desse tipo de veículo e características que influenciam o tipo de aplicação;
- Dando continuidade, a subseção "**4.1.3 - Principais Componentes de um Quadricóptero**" apresenta os elementos fundamentais desse tipo de aeronave, um dos tipos mais apropriados para a inspeção de transportadores;
- Na sequência, a subseção "**4.1.4 - Dinâmica de Voo de um Quadricóptero**" discute como as forças atuantes na aeronave resultam em diferentes tipos de movimentos;

- Por sua vez, a subseção "**4.1.5 - Controle do Voo de um Quadricóptero**" discorre sobre a importância de sistemas de controle para o voo de um quadricóptero e explica alguns dos mecanismos que o viabilizam;
- Por fim, a subseção "**4.1.6 - Regulamentação: O grande desafio**" discute a relevância desse tema, abordando brevemente o histórico da legislação de VANTs em alguns países e aspectos da recente regulamentação deste assunto no Brasil.

### 4.1.1 Um Breve Histórico dos VANTs

Apesar da recente popularização do uso civil dos VANTs, Valavanis e Vachtsevanos (2015) descrevem que a concepção de uma máquina voadora não-tripulada data de aproximadamente 2500 anos atrás, na Grécia antiga, com o primeiro exemplar em 425 a.C. sendo creditado a Archytas. Segundo o autor, existem também relatos em documentos na China antiga, por volta de 400 a.C., descrevendo conceitos similares. Leonardo Da Vinci, por volta de 1483 d.C., também desenhou uma máquina que seria capaz de autopropelir e flutuar, além de um pássaro mecânico articulado, em 1508 d.C. Todos esses conceitos ainda são bastante rudimentares e não se enquadram na definição atual de VANTs usada neste trabalho. Apesar disso, podem ser vistos como precursores das ideias de VANTs modernos e reflexo do fascínio que essas máquinas despertam.

O potencial de uso militar desses dispositivos contribuiu para o seu desenvolvimento, com os primeiros relatos datados de 1916, durante a Primeira Guerra Mundial. Dois fatores principais contribuíram para a concepção do que é considerado o primeiro VANT: a invenção por Elmer Sperry de um estabilizador automático giroscópico e a demonstração, em 1898, de Nicola Tesla do controle remoto usando ondas de rádio, cujo conceito foi adaptado para a aviação por Peter Hewitt (TARANTOLA, 2013). A junção dessas tecnologias resultou na criação do avião automático de Hewitt-Sperry (Figura 21), com o objetivo de carregar explosivos e lançá-los em um alvo (KEANE; CARR, 2013). Devido a diferentes problemas técnicos no lançamento, tanto por catapultas quanto por outros métodos, e à baixa acurácia, perdeu-se, momentaneamente, o interesse no uso dos VANTs como veículos de ataque (VALAVANIS; VACHTSEVANOS, 2015). Além disso, com o fim da Primeira Grande Guerra, a necessidade de desenvolvimento de novos armamentos deixou de ser prioridade para os militares, o que colocou de lado, por algum tempo, o esforço no aprimoramento desses veículos (KEANE; CARR, 2013).

Apesar disso, o desenvolvimento dos VANTs não foi completamente abandonado. Eles eram vistos com grande potencial para uso como alvos no treinamento

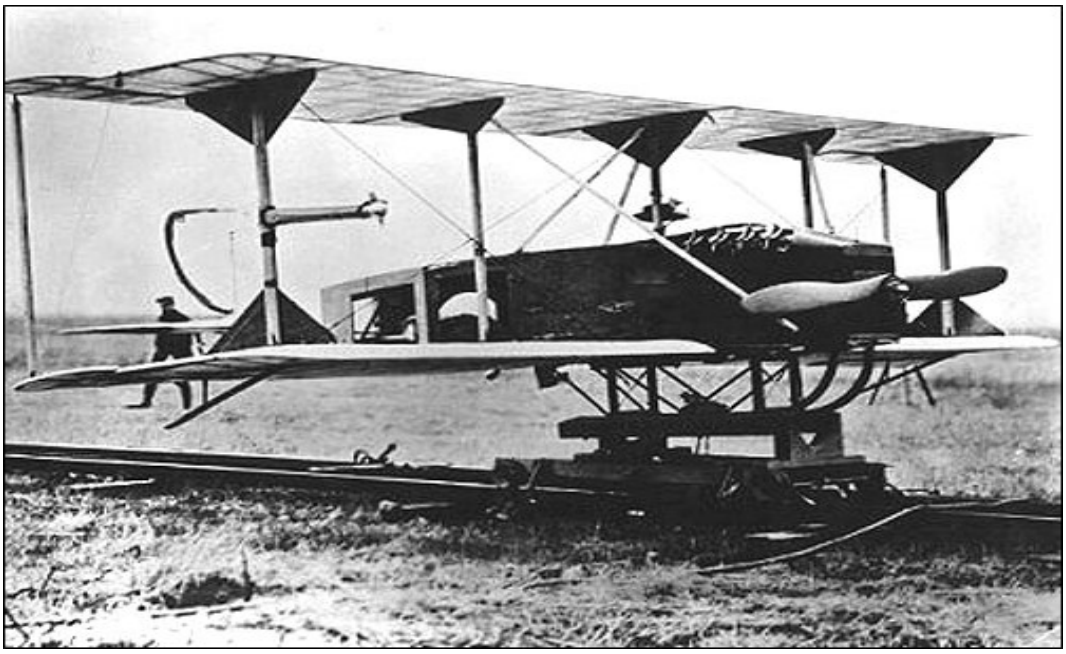
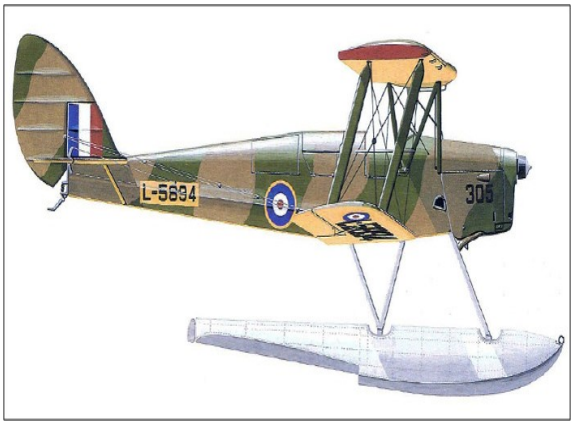


Figura 21 – Avião automático de Hewitt-Sperry, considerado o primeiro VANT nos moldes conhecidos atualmente. Fonte: LEU, 2015

de pilotos de caça, de artilharia anti-aérea em navios e outras atividades que demandavam alvos móveis para prática. O exemplo de maior destaque desse tipo de uso é o projeto britânico que resultou no Havilland DH-82B (Figura 22a), também conhecido como *Queen Bee*. Segundo Keane e Carr (2013), o seu sucesso à época, com mais de 400 unidades produzidas entre 1933 e 1943, e o fato de seu nome remeter a abelha-rainha, contribuiu para que os VANTs também sejam conhecidos como "*drones*", ou seja, zangões. Do lado americano, um entusiasta no desenvolvimento de aviões rádio-controlados, Reginald Denny, foi contratado pelo exército americano em 1941 e forneceu mais de 3800 VANTs para treinamento de artilharia anti-aérea ao longo da Segunda Guerra Mundial. A Figura 22b mostra um dos seus primeiros protótipos, o RP-1, que posteriormente foi adaptado e gerou diferentes variações (KEANE; CARR, 2013).

Ao longo da Segunda Grande Guerra, as forças armadas dos Estados Unidos desenvolveram diferentes iniciativas visando usar VANTs como armas de ataque. O "Projeto Aphrodite" é o exemplo de maior destaque, cujo objetivo era converter bombardeiros B-17, que já estavam deixando de ser viáveis operacionalmente, em aviões operados remotamente por rádio, carregando grandes quantidades de explosivos para atacar de forma suicida fortificações ligadas ao Eixo. Ao longo de 1944, a Força Aérea americana promoveu diferentes missões desse projeto e os resultados foram inconsistentes, com vários eventos de perda de controle do VANT, tornando-o uma bomba voadora sem rumo, e erro dos alvos por grandes distâncias. Tais problemas, em combinação com o fracasso de um projeto similar da marinha, o Anvil, levaram ao cancelamento da iniciativa e novamente à perda de interesse no




(a) VANT britânico *Queen Bee*. Fonte: VINTAGE WINGS OF CANADA, 2013

(b) Protótipo RP-1 de Reginald Denny. Fonte: NAUGHTON, 2006

Figura 22 – Exemplos de VANTs desenvolvidos para serem usados como alvos em treinamentos militares.

desenvolvimento de VANTs como armas para missões de ataque (KEANE; CARR, 2013).

Os VANTs voltam a ganhar destaque no período conhecido como Guerra Fria, que teve início imediatamente após o fim da Segunda Guerra Mundial. Nesse período, era comum a realização de missões aéreas de reconhecimento em território inimigo, muitas vezes com resistência. Apenas a força aérea americana perdeu 163 combatentes entre 1950 e 1969 em missões deste tipo (KEANE; CARR, 2013). Adicionalmente, a perda de dois aviões-espiões do tipo U-2, um sobre a União Soviética e outro sobre Cuba, foi o estopim para retomar de forma massiva o financiamento de projetos de desenvolvimento de VANTs para missões de reconhecimento (VALAVANIS; VACHTSEVANOS, 2015).

Nessa linha, o *SD-1*, também conhecido como *MQM-57 Falconer* (Figura 23a), foi o primeiro modelo desenvolvido pelos Estados Unidos em meados da década de 1950, com mais de 1500 unidades produzidas até 1970. Paralelamente, a *Ryan Aeronautical Company* desenvolveu o *Ryan Model 147 Firefly* (Figura 23b), posteriormente renomeado para *Lightning Bug*. Ele recebeu diferentes adaptações, com variações especializadas em reconhecimento e em missões de ataque, sendo usado em mais de 1000 missões na Guerra do Vietnã. A União Soviética também desenvolveu alguns modelos, sendo o mais relevante deles o *TU-143* (Figura 23c), que incorporou paraquedas para reduzir os custos operacionais do modelo anterior, o *DBR-1*, que tornou-se inviável por não ser recuperável. Ainda durante a Guerra Fria, Israel também teve seu papel de pioneirismo no desenvolvimento de VANTs, sendo o *Scout* (Figura 23d) seu modelo mais importante (VALAVANIS; VACHTSEVANOS, 2015; KEANE; CARR, 2013).

(a) VANT americano MQM-57 Falconer

(b) VANT americano AQM-34Q, uma das variações do Ryan Model 147

(c) TU-143, VANT desenvolvido pela União Soviética

(d) Scout, o VANT desenvolvido pela Força Aérea de Israel

Figura 23 – Exemplos de VANTs desenvolvidos durante o período da Guerra Fria. Fonte: VALAVANIS; VACHTSEVANOS, 2015

Os avanços obtidos durante a Guerra Fria e na Guerra do Golfo foram os pilares para o desenvolvimento dos VANTs modernos. O VANT *Mastiff*, desenvolvido pela indústria israelense em 1973, serviu como base para a construção do projeto americano do *General Atomics MQ-1 Predator* (VALAVANIS; VACHTSEVANOS, 2015), cuja produção iniciou-se em 1995 e somente agora, no final de 2017, iniciou seu processo de descontinuidade. O sucesso com esse modelo levou ao lançamento, em 2001, de sua evolução, o *General Atomics MQ-9 Reaper* (Figura 24a). Juntos, estes modelos alcançaram, em 2016, a marca de 4 milhões de horas de voo, 90% delas em missões de reconhecimento e ataque em zonas de guerra no Oriente Médio (TOMKINS, 2016), consolidando os VANTs como uma parte praticamente indispensável das forças armadas de um país.

É interessante perceber que, apesar da evolução dos VANTs estar intimamente ligada a seu potencial de uso militar, as aplicações civis beneficiam-se muito de tais avanços. Um dos exemplos mais claros é o VANT *Ikhana* (Figura 24b), desenvolvido

em 2006 pela NASA para ser usado em missões científicas diversas a partir de uma adaptação direta do VANT militar *Predator* (CONNER, 2015). Ainda assim, esse é um VANT caro e pouco acessível mesmo para grandes centros de pesquisa.

(a) VANT militar MQ-9 Reaper. Fonte: TOMKINS, 2016

(b) Ikhana, adaptação pela NASA do Predator para uso científico. Fonte: CONNER, 2015

Figura 24 – Exemplos de VANTs modernos para uso em combate e civil.

Não é clara a linha que delimita a transição entre os primeiros VANTs civis, como o Ikhana, e VANTs popularmente usados hoje, como os da fabricante DJI (Figura 25a) e da Parrot (Figura 25b). Até pouco tempo atrás, o uso civil de VANTs era comum apenas entre empresas especializadas, que vendiam serviços específicos, e aeromodelistas, que desenvolviam praticamente toda a estrutura do veículo, rádio-controle e precisavam ainda ter grandes conhecimentos para a pilotagem. Dessa forma, pode-se apontar como elementos que contribuíram para a popularização o fato de eles terem se tornado mais baratos, portáteis e de fácil operação mesmo para leigos (THE ECONOMIST, 2015). De acordo com Gao (2015), as principais evoluções técnicas que propiciaram tais características são:

- Desenvolvimento das baterias de polímero de Lítio (Li-Po), que conseguem armazenar grandes quantidades de carga em tamanhos compactos e ainda entregá-la de forma rápida;
- Redução do preço e miniaturização de componentes eletromecânicos, como os motores com ímãs permanentes e sem escovas (do inglês *brushless*), que conseguem fornecer as grandes quantidades de torque necessárias para VANTs, particularmente para os que pairam em posições fixas;
- Popularização dos *smartphones*, o que levou à produção em massa e consequente redução de custo de sensores, como acelerômetros, giroscópios e receptores de sistemas de posicionamento global (GNSS, do inglês *Global*

*Navigation Satellite System*). Tais sensores são importantes para a estabilização do VANT, facilitando sua operação, e também são os pilares para voos autônomos.

Olhando adiante, é fato que os VANTs possuem um potencial gigantesco ainda inexplorado. Kushleyev et al. (2013) apostam na miniaturização e no conceito de enxame (Figura 25c), em que diversos micro-VANTs voam cooperativamente para habilitar diferentes aplicações, como shows aéreos (Figura 25d), mapeamento de ambientes internos e geração de imagens de uma região por diferentes ângulos e perspectivas, reduzindo a oclusão. Segundo os autores, a redução do tamanho diminui os custos de fabricação, permite o acesso a espaços restritos, aumenta a agilidade e ainda reduz as consequências no caso de acidentes. A colaboração de múltiplos micro-VANTs permitiria sobrepor as limitações individuais de tamanho e aumentar a capacidade total de carga (ou *payload*). Outros autores, como Pope e Cutkosky (2016), apontam como tendência a bioinspiração, em que animais da natureza servem como referência para a construção de VANTs, buscando, assim, imitar suas principais características para obter vantagens ou resolver problemas ainda em aberto.

Ainda que o futuro dos VANTs esteja aberto e repleto de possibilidades, é provável que os sistemas modernos continuem adotando uma estrutura similar à já adotada hoje em relação aos seus componentes. A próxima subseção apresenta as principais partes que formam um sistema VANT genérico e como elas se encaixam para habilitar as diferentes aplicações possíveis.

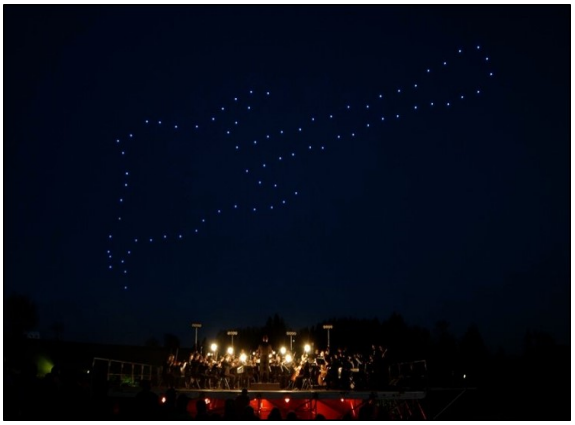
### 4.1.2 Sistema VANT

Apesar do termo VANT geralmente referir-se ao veículo aéreo em si, é importante lembrar que o fato de ele não ser tripulado, leva à necessidade de sistemas adicionais externos e internos à aeronave para realizar sua pilotagem e controle de missão. Tais elementos podem ser tão simples quanto um rádio-controle e um receptor, que estabelece a comunicação com a aeronave, quanto um arcabouço de controle terrestre complexo, usando comunicações via satélite. Dessa forma, esta subseção irá avaliar os VANTs sob uma perspectiva um pouco mais abrangente, olhando o conjunto de partes ou componentes que formam o que é chamado de **Sistema VANT**. Essa estrutura genérica, que vale para qualquer tipo de VANT aderente à definição adotada neste trabalho, é apresentada de forma esquemática na Figura 26.

(a) Matrice 600, da fabricante DJI, para uso profissional. Fonte: DJI, 2017b

(b) Modelo híbrido Swing, um dos VANTs da Parrot. Fonte: PARROT, 2017

(c) Enxame de micro-VANTs de 75 g. Fonte: KUSHLEYEV et al., 2013



(d) Uso de um enxame de micro-VANTs no Rock in Rio 2017. Fonte: MELLO, 2017

Figura 25 – Exemplo de alguns VANTs de uso civil e demonstração do conceito de enxame de micro-VANTs.

#### 4.1.2.1 Veículo Aéreo

O veículo aéreo pode ser visto como o ponto central de um Sistema VANT. Segundo Fahlstrom e Gleason (2012), ele pode ser subdivido em: estrutura, unidade de propulsão, controles de voo, sistema de alimentação e comunicação. Não é escopo deste trabalho detalhar todos esses subsistemas. Portanto, caso seja do interesse do leitor aprofundar em algum deles, recomenda-se a bibliografia de Fahlstrom e Gleason (2012), onde é possível obter informações completas e detalhadas de cada um deles. Por sua vez, a estrutura será melhor explicada, visto que é um dos parâmetros mais relevantes por determinar as características e aplicações possíveis de um VANT.

A estrutura do VANT influencia completamente a dinâmica de voo, já que ela determina os diferentes princípios aerodinâmicos usados para sustentação da ae-

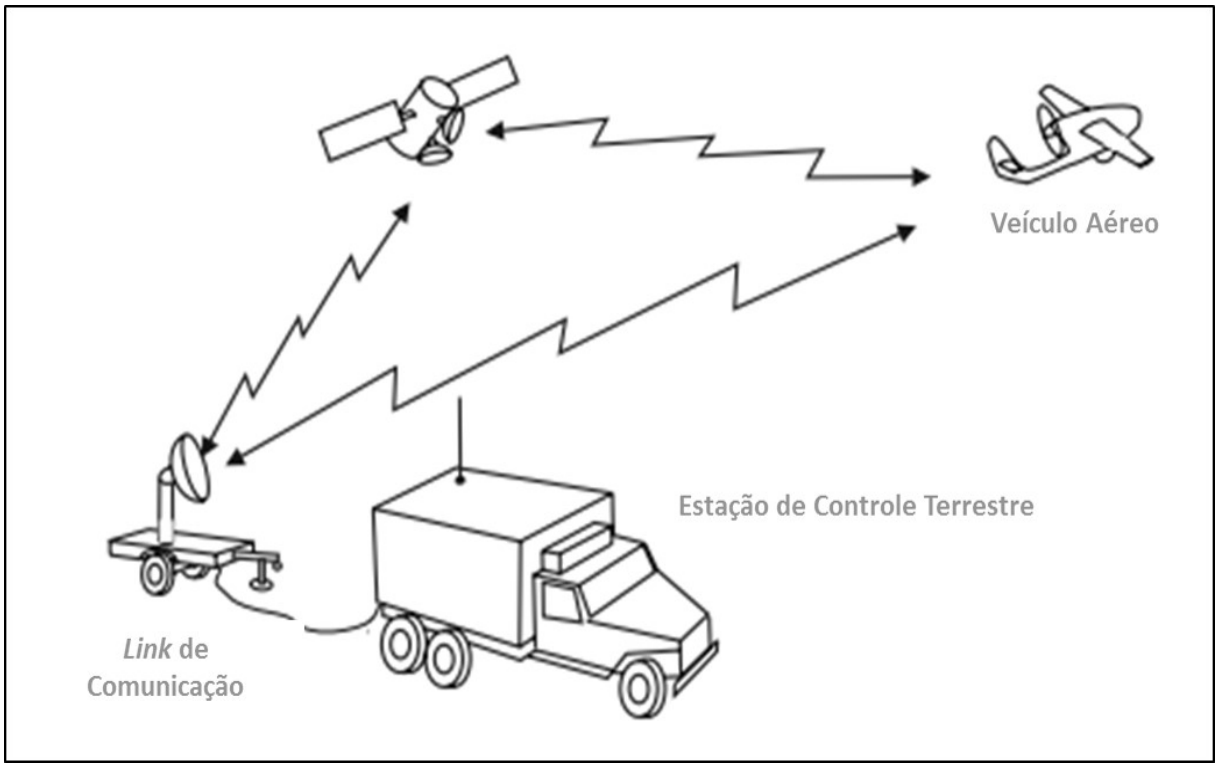



Figura 26 – Esquema simplificado dos principais componentes de um Sistema VANT. Fonte: adaptado de FAHLSTROM; GLEASON, 2012

ronave. Para entender melhor este ponto, uma analogia simples é pensar em quão diferente é o voo de um helicóptero em relação a um avião. Ou seja, cada um adota diferentes princípios de sustentação para atingir o objetivo de se manter no ar. Assim, conforme Fahlstrom e Gleason (2012), os tipos de estrutura mais comuns são:

- **Asa Fixa (*fixed-wing*)**: consiste de uma asa fixa rígida com um determinado coeficiente aerodinâmico. O voo se torna possível por conta da sustentação que é naturalmente gerada pelas asas à medida que um determinado impulso empurra o veículo adiante. Geralmente, utiliza-se uma hélice acoplada a um motor, elétrico ou a combustão, como o sistema de propulsão. Por conta da necessidade de sustentação, esse tipo de estrutura requer um sistema de lançamento para atingir as velocidades necessárias para iniciar e manter o voo, podendo utilizar uma pista, catapultas ou outras formas, o que reduz sua flexibilidade. Por outro lado, essa estrutura tende a ser mais eficiente, pois consegue usar a sustentação natural do ar para aumentar a sua autonomia e cobrir maiores distâncias, o que permite ainda carregar cargas mais pesadas. Um exemplo disso é o VANT mostrado na Figura 24a, planejado para incursões militares de longas distâncias carregando diferentes tipos de cargas (DOBROKHODOV, 2014).
- **Asa Rotativa (*rotary wing*)**: esse tipo de estrutura usa um ou mais rotores, isto é, um motor acoplado a duas ou três lâminas, para gerar a sustentação ne-

cessária para o voo. Por conta disso, são menos eficientes que os VANTs de asa fixa, mas conseguem pairar sobre pontos fixos, voar em áreas fechadas e realizar decolagem e aterrissagem verticais, o que os torna extremamente flexíveis. O número de rotores pode variar, mas a regra geral é que quanto mais rotores, maior a complexidade, custo e carga que o VANT consegue carregar. A dinâmica de voo é totalmente influenciada pelo número de rotores, já que são as variações de torque em cada rotor que permitem que o VANT voe em todas as direções. Atualmente, os VANTs com quatro rotores (quadricópteros ou *quadrotors*), conforme exemplos na Figura 25a e Figura 25c, são os mais comuns (POWERS; MELLINGER; KUMAR, 2015).

A partir dessas características, a Tabela 3 apresenta uma comparação dos principais tipos de estrutura, com considerações gerais de cada um e aplicações que são adequadas para VANTs de Asa Fixa e Asa Rotativa. Existem ainda modelos híbridos, que tentam combinar os dois sistemas acima para sobrepor as limitações individuais. Um exemplo simples desse tipo pode ser visto na Figura 25b, que mostra um VANT para recreação que adota o sistema de Asa Rotativa para decolagem e aterrissagem, e, ao atingir velocidades maiores, utiliza uma Asa Fixa para conseguir maior autonomia e realizar manobras.

Com base nas considerações apresentadas na Tabela 3, é importante notar que, ainda que possua pontos negativos, a estrutura de Asa Rotativa é mais flexível. Modelos de Asa Fixa não conseguiriam voar de forma estável e segura em baixas altitudes, nem acessar as diferentes áreas onde esses equipamentos operam, muitas vezes em espaços restritos ou com obstáculos ao redor. Por fim, o fato de VANTs de Asa Rotativa conseguirem planar em uma posição fixa permite uma análise mais cuidadosa de pontos específicos, característica que pode ser fundamental durante a inspeção.

Por conta disso, quando se avalia a camada **Portador** da Arquitetura Integrada proposta na seção “5.2 - Arquitetura Integrada”, percebe-se que, para Sistemas VANTs, apenas os de estrutura de Asa Rotativa possuem atributos que habilitam seu uso para o problema em questão: a inspeção de rolos de transportadores de correias. Assim, a validação da arquitetura proposta restringiu-se ao uso deste tipo de portador.

#### 4.1.2.2 Estação de Controle Terrestre

A segunda parte que compõem um Sistema VANT, conforme mostrado na Figura 26, é a Estação de Controle Terrestre (GCS, do inglês *Ground Control Station*). Segundo Fahlstrom e Gleason (2012), esse elemento pode variar em termos de

Tabela 3 – Principais características e aplicações das estruturas de Asa Fixa e Rotativa

| Estrutura | Considerações | Principais Aplicações |
|---|---|---|
| Asa Fixa | - Usam o próprio ar como sustentação, sendo mais eficientes e apropriados para levar cargas mais pesadas por distâncias e velocidades maiores;<br>- Requerem estrutura de lançamento e aterrisagem;<br>- Precisam manter um fluxo de ar constante sob a asa, o que impede seu uso para aplicações que demandem manter uma posição fixa;<br>- Podem empregar sistemas de propulsão elétricos ou a combustão;<br>- Geralmente mais caros e mais difíceis de operar. | - Mapeamento de grandes áreas abertas;<br>- Aplicações militares de longo alcance e altitudes;<br>- Missões em condições climáticas adversas. |
| Asa Rotativa | - Precisam criar sua própria sustentação através dos rotores, o que diminui sua eficiência e os torna apropriados para cargas e distâncias menores;<br>- Conseguem decolar e aterrisar de forma vertical, mesmo com pouco espaço livre;<br>- Sistema de estabilização, que permite que o VANT plane em uma posição fixa;<br>- Modelos multi-rotor possuem redundância no caso de falha, o que aumenta a segurança do voo;<br>- Por conta da necessidade de alimentar precisamente cada rotor para manter a estabilidade, apenas sistemas de propulsão elétricos são factíveis atualmente;<br>- Diversas faixas de preço, mas geralmente acessíveis e fáceis de operar. | Missões em áreas urbanas ou espaços restritos;<br>- Inspeção de estruturas diversas;<br>- Fotografia, filmagem e demais aplicações que exijam posicionamento preciso e estável. |

Fonte: adaptado de DOBROKHODOV, 2014 e POWERS; MELLINGER; KUMAR, 2015

complexidade e em relação às funções realizadas, mas, em geral, a GCS é responsável por:

- Apresentar, próximo de tempo real, o vídeo das câmeras embarcadas no veículo;
- Processar e apresentar os dados de sensores e telemetria de voo;
- Permitir o envio de comandos para o VANT, o que inclui a pilotagem em si e o

controle de sua carga (câmera, míssil, etc).

Naturalmente, a complexidade da GCS varia muito em função do VANT e do tipo de aplicação. Por exemplo, um VANT militar, como o MQ-9 Reaper, possui em sua GCS diferentes funções, que incluem o comando da aeronave, o controle da carga sendo transportada, estrutura para comunicação entre as bases e monitores para acompanhamento dos sensores e da missão, conforme Figura 27a. Todo voo dessa aeronave é realizado por um piloto e por um operador de sensores, assim, o primeiro conta com instrumentos para pilotagem, como *joysticks* e visão em primeira pessoa (FPV, do inglês, *First Person View*), enquanto o segundo monitora, por meio de visores, o funcionamento dos diferentes sensores disponíveis na aeronave. Nesse caso, a GCS é tão avançada que ela pode estar até mesmo distribuída entre diferentes locais físicos, por exemplo, a decolagem do VANT pode ser feita por uma equipe no Iraque e a continuação da missão por outro piloto em uma base aérea americana (SCHEVE, 2008).

(a) GCS do MQ-9 Reaper, com diferentes funções e dois operadores.Fonte: SUBBARAMAN, 2013

(b) Controle físico e aplicativo específico formam o GCS do Mavic Pro. Fonte: DJI, 2017c

Figura 27 – Exemplos de Estações de Controle Terrestre com diferentes níveis de complexidade

Por outro lado, VANTs mais simples, que atendem a aplicações menos complexas, podem ter uma GCS igualmente reduzida. Neste caso, as funções geralmente estão limitadas a receber o vídeo transmitido pela aeronave, realizar sua pilotagem, controlar a câmera e acompanhar os dados de telemetria de voo, como velocidade e posição. Para isso, alguns modelos contam com apenas um aplicativo móvel para *smartphones* ou *tablets* que permite realizar todas essas opções. Em outros casos, como o DJI Mavic Pro, a GCS é geralmente a combinação de um controle físico (*joystick*) e um aplicativo do fabricante rodando em um *smartphone*, que provê as demais funções, conforme apresentado na Figura 27b. Apenas o controle físico

pode ser usado, mas sem a visualização do vídeo em FPV do VANT e dados limitados sobre a telemetria de voo. Da mesma forma, é possível operar o VANT apenas com o aplicativo móvel, mas com menor precisão no voo (DJI, 2017c).

Dessa forma, não existe um modelo definido de estação de controle terrestre que atenda a qualquer tipo de VANT, mas sim uma combinação de múltiplas funções, atendidas de diferentes formas, que permitam que o VANT complete o seu objetivo e missões. Em um dos protótipos do presente trabalho, foi adotada como GCS o controle físico do VANT e uma aplicação móvel desenvolvida sob medida para prover as funções desejadas na inspeção dos rolos, por exemplo, criar missões seguindo coordenadas específicas (ou *waypoints*), além de outros recursos melhor discutidos na subseção "**5.4.1 - Protótipo 1 - Aplicativo Móvel e Sistema de Controle da Inspeção**".

#### 4.1.2.3 *Link* de Comunicação

É imprescindível que o Veículo Aéreo e a Estação de Controle Terrestre (GCS) consigam se comunicar para que seja possível o controle e a recepção de dados dos sensores do veículo. Geralmente, a comunicação é realizada de forma bidirecional, o que demanda equipamentos de recepção e transmissão tanto no veículo quanto na GCS (FAHLSTROM; GLEASON, 2012). Naturalmente, a seleção das tecnologias de comunicação apropriadas em Sistemas VANTs não é trivial, dado que inúmeros fatores têm influência nessa decisão, especialmente os itens abaixo resumidos por Brown, McHenry e Jaroonvanichkul (2014):

- **Porte da Aeronave:** pode variar de poucos centímetros a muitos metros, o que influencia no tamanho e peso dos equipamentos de comunicação que a mesma é capaz de carregar;
- **Alcance e Tipo de Operação**: alguns VANTs são pensados para voar apenas a curtas distâncias, com linha de visada (VLOS, do inglês *Visual Line Of Sight*), enquanto outros são concebidos para longas distâncias e operação sem linha de visada (BVLOS, do inglês *Beyond Visual Line of Sight*). Assim, a tecnologia de comunicação deve suportar o tipo de operação;
- **Robustez:** VANTs militares e civis de missão crítica demandam canais de comunicação seguros, à prova de invasores, que consigam suportar interferências e tentativas de bloqueio (*jamming*), logo, devem ter equipamentos capazes de lidar com tais situações;
- **Tipo de Carga e Aplicação:** alguns sensores e equipamentos transportados pelo VANT podem ter requisitos específicos de comunicação, como baixa la-

tência e necessidade de transmitir grandes quantidades de dados. Um exemplo são os VANTs usados para filmagens profissionais equipados com câmeras 4K, que demandam grande largura de banda e que não podem ter atrasos significativos entre transmissão e recepção;

- **Regulamentação:** comunicações sem fio devem usar faixas de frequência específicas, licenciadas ou de livre utilização em uma região ou país. Além disso, os equipamentos de rádio para transmissão e recepção também devem ser homologados por agências, como a Agência Nacional de Telecomunicações (ANATEL) no Brasil. Dessa forma, nem toda frequência ou equipamento que permita otimizar as características acima são passíveis de utilização em um Sistema VANT.

Com base em todos esses aspectos, os *links* de comunicação estabelecidos variam muito entre os VANTs, com complexidade e robustez do sistema geralmente proporcional à sua aplicação. Por exemplo, o MQ-9 Reaper (Figura 24a) utiliza uma combinação de sistemas de comunicação para suportar as diferentes etapas de voo, conforme ilustrado na Figura 28. Durante a decolagem, aterrissagem e operação VLOS, a comunicação entre o VANT e a GCS ocorre diretamente por antenas com linha direta de visada, enquanto *links* de comunicação via satélite passam a operar quando o VANT se afasta e a operação passa a ser BVLOS (CUADRA; WHITLOCK, 2014).

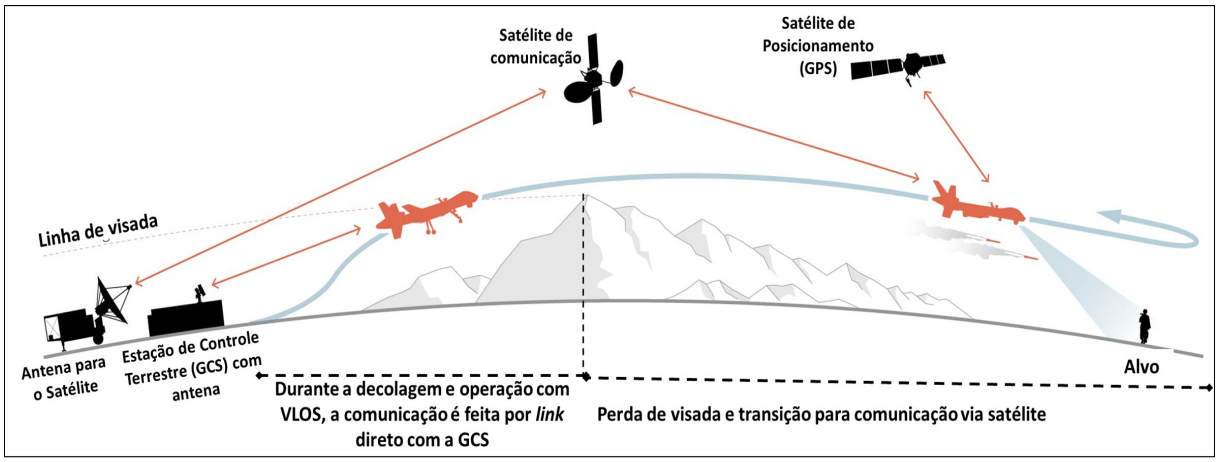



Figura 28 – Esquema dos *links* de comunicação usados durante a operação de um MQ-9 Reaper. Fonte: adaptado de CUADRA; WHITLOCK, 2014

Por sua vez, VANTs civis tendem a adotar *links* de comunicação mais simples, dado que a maioria dos dispositivos opera apenas em distâncias relativamente curtas, dentro da linha de visada do operador e da GCS. Segundo Elston et al. (2014), o padrão IEEE 802.11 é o mais comum de ser encontrado, pois consegue estabelecer conexões ponto-a-ponto (GCS x VANT), possui boa largura de banda para transmissão de vídeo, possui equipamentos de transmissão e recepção pequenos

e baratos, tem alcance satisfatório em áreas abertas e adota frequências de uso livre na maior parte dos países.

Em contrapartida, justamente por serem livres, as frequências usadas pelo padrão, particularmente a de 2,4 GHZ, é extremamente poluída e sofre interferência de diversos aparelhos, o que limita o alcance em áreas urbanas. Por conta disso, os VANTs adotados no presente trabalho, como o DJI Mavic Pro e o DJI Inspire 1, adotam tecnologias proprietárias para aumentar a robustez na comunicação, procurando reduzir os problemas relativos ao uso de frequências abertas. O primeiro usa o *OcuSync*, mais recente e que adota algoritmos de compressão e alteração dinâmica de frequências para atingir o alcance máximo de 7 km na transmissão de vídeos em 1080p (720p quando há interferências) com latência de 130 ms (DJI, 2017c). Já o segundo adota o *LightBridge*, que é amplamente utilizado pela gama de produtos da fabricante e tem o alcance máximo de 5 km, com latência na ordem de 220 ms na transmissão de vídeo em 720p (DJI, 2017a).

Assim, mesmo baseados em frequências de livre utilização, ao adotar tais padrões, esses VANTs conseguem oferecer comunicação estável e transmissão de vídeo de alta qualidade com baixa latência, sem interferir com os equipamentos críticos das áreas industriais onde irão voar, que usam frequências específicas e regulamentadas. Esse conjunto de características é positivo para a aplicação de inspeção de rolos proposta neste trabalho e reforça a escolha de tais equipamentos como portadores da estrutura de sensoriamento posteriormente detalhada.

Dessa forma, dado que a aplicação em questão é nova e desafiadora, pode ser necessário, futuramente, desenvolver um VANT customizado para ela. Isto posto, as próximas subseções procuram aprofundar a explicação sobre os componentes principais, dinâmica de voo e o algoritmo de controle de voo de um quadricóptero, que é um VANT de Asa Rotativa com quatro rotores.

### 4.1.3 Principais Componentes de um Quadricóptero

Para ilustrar os componentes, dinâmica de voo e algoritmo de controle, foram escolhidos os quadricópteros por serem populares e versáteis. De qualquer forma, é importante ressaltar que os princípios aqui discutidos valem para outras quantidades de rotores, como os hexacópteros (6 rotores) e octacópteros (8 rotores).

Ainda que os componentes de um quadricóptero possam ter grande variação em função de sua aplicação e grau de sofisticação, aqui são discutidos os elementos fundamentais, ou seja, o mínimo necessário para a concepção de um VANT desse tipo. Como praticamente qualquer tipo de operação demanda a visão FPV, os componentes que adicionam tal capacidade, ainda que opcionais, também foram

descritos. Dessa forma, a Figura 29 apresenta os componentes e a lista a seguir detalha cada um deles, seguindo as explicações de Liang (2018) e Wang (2015):

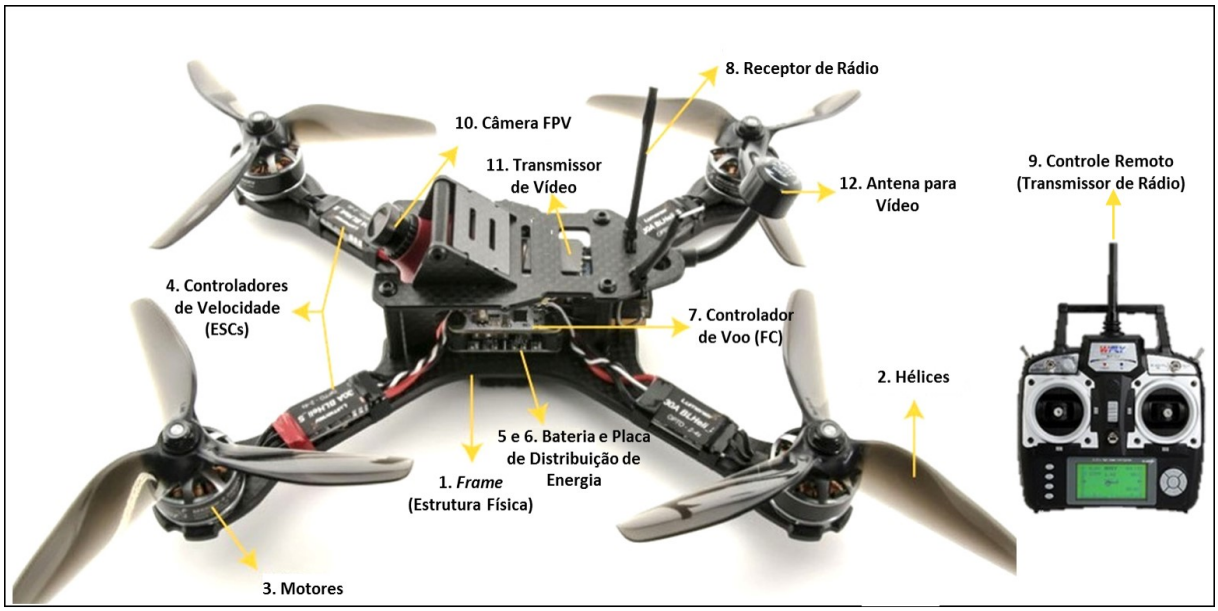



Figura 29 – Principais componentes de um quadricóptero. Fonte: adaptado de Liang (2018)

1. **Estrutura física (*frame*):** também conhecido como corpo, é o elemento que dá forma ao VANT e que carrega os demais componentes. No caso de quadricópteros, os tipos de estrutura mais comuns são os em forma de “X”, cruz (“+”) ou “H”. Geralmente, o tipo de aplicação é que determina o *frame* utilizado, por exemplo, a estrutura em “H” favorece a agilidade, apropriada para manobras rápidas em VANTs de competição, enquanto a estrutura em “X” auxilia na estabilidade e suavidade de voo, facilitando sobrevoar algum ponto em missões de inspeção.

2. **Hélices (*propellers*):** são os elementos que, quando giradas por um motor, interagem com o ar para gerar sustentação e viabilizar o voo. Dessa forma, tem grande efeito na velocidade, carga suportada e manobrabilidade do quadricóptero. Para tanto, o comprimento e inclinação são os principais influenciadores: enquanto hélices longas aumentam a capacidade de carga e geram elevação desde baixas rotações, hélices curtas conseguem ser mais eficientes para aumentar a velocidade e capacidade de manobra, mas demandam mais energia dos motores para sua rotação. Assim, deve-se combinar adequadamente as hélices com os motores para formar rotores eficientes para a aplicação pretendida.

3. **Motores:** são os responsáveis por girar as hélices. Na maior parte dos casos, são utilizados motores sem escova (ou *brushless*), mais eficientes, confiáveis

e silenciosos que os motores tradicionais. É importante que os motores sejam leves e que consigam minimizar o consumo de energia, por ser o componente que mais demanda bateria.

4. **Controladores de velocidade (*Electronic Speed Controller* - ESC):** como será melhor detalhado na subseção a seguir, é a variação de velocidade de rotação dos rotores que permite os diferentes tipos de movimento que um quadricóptero pode realizar. Dessa forma, este componente é responsável por controlar eletronicamente cada um dos motores, modulando os sinais oriundos do controlador de voo na corrente de alimentação do motor correspondente à taxa desejada de rotação do motor.

5. **Bateria:** é o elemento que fornece energia para os motores e demais elementos do VANT, como sensores e câmeras. O tipo mais comum são as de LiPo, dada sua capacidade de descarregar rapidamente as altas quantidades de energia que os motores demandam.

6. **Placa de Distribuição de Energia:** além da bateria em si, esta placa é necessária para distribuir a energia para todos os componentes, de acordo com suas necessidades específicas.

7. **Controlador de Voo (*Flight Controller* - FC:)** pode ser visto como o cérebro do VANT, responsável por todo o controle do voo. Para isso, o FC possui diversos tipos de sensores embarcados (acelerômetros, giroscópios, barômetros, magnetômetro, etc) que permitem saber a condição de voo ou orientação atual. Com base no estado lido dos sensores e do tipo de comando recebido do piloto, o FC calcula qual deve ser a rotação de cada um dos motores para atingir o estado desejado e os envia por meio de sinais para cada um dos ESC. A subseção "**4.1.5 - Controle do Voo de um Quadricóptero**" detalha como este controle é feito.

8. **Receptor de Rádio:** por definição, o VANT é um veículo não tripulado, portanto, precisa ser controlado remotamente de alguma forma. A forma mais comum é através de radiofrequência. Assim, o quadricóptero deve ter um componente que recebe os comandos enviados pelo controle remoto e os entrega ao FC. É importante destacar que o receptor é parte de uma dupla "transmissor/receptor", portanto, deve utilizar frequências e protocolos compatíveis com o transmissor.

9. **Controle Remoto (Transmissor de Rádio):** parte da GCS, responsável por enviar os comandos desejados ao quadricóptero por meio de radiofrequência. É com ele que é realizada a pilotagem, determinando o estado desejado da aeronave em relação a velocidade, rotação e direção de voo.

10. **Câmera FPV:** juntamente com o transmissor e antena de vídeo, é um dos componentes opcionais. Ela fornece ao piloto a visão frontal do quadricóptero, o que auxilia a pilotagem e, em alguns casos, permite operações além da linha de visada (BLOS). Por conta disso, as câmeras FPV não focam em qualidade, mas sim em baixa latência e ampla gama dinâmica, para imagens mesmo contra luz direta.

11. **Transmissor de Vídeo:** este componente é necessário para receber os sinais da câmera FPV e transmiti-los, por radiofrequência, aos dispositivos de exibição das imagens. Geralmente, é utilizado um dos canais na faixa de frequência de 5,8 GHz.

12. **Antena para Vídeo:** fisicamente conectada ao transmissor de vídeo, a antena determina o alcance e potência / qualidade do sinal transmitido para o receptor de vídeo, que apresenta as imagens ao piloto.

Com o uso de tais componentes é possível montar e voar quadricópteros ou suas variações com mais motores. Modelos comerciais mais modernos utilizam componentes adicionais para tornar a operação mais segura e facilitar a pilotagem, melhorando a experiência do usuário. Alguns exemplos são os sensores de ultrassom e infravermelho para detecção de obstáculos, sensores de posicionamento de alta precisão, sistema de redundância para baterias, etc. Além disso, o tipo de aplicação do VANT também pode demandar componentes adicionais, como gimbal para conexão de câmeras de alta resolução e mecanismos para controle da carga (*payload*).

A subseção a seguir apresenta, simplificadamente, a física por trás do voo de um quadricóptero e como alguns de seus componentes principais influenciam nela.

### 4.1.4 Dinâmica de Voo de um Quadricóptero

A dinâmica de voo de um quadricóptero pode ser totalmente definida com base nas leis de *Newton* e *Euler*. À medida que os rotores do VANT giram, as hélices empurram o ar para baixo. Conforme Terceira Lei de Newton, essa força provoca uma reação do ar, que empurra as hélices para cima, gerando sustentação de maneira proporcional à velocidade de rotação do rotor (empuxo). Porém, se tal movimento fosse executado por apenas um rotor, provocaria um giro da estrutura do VANT no sentido oposto ao de rotação, como um helicóptero que gira descontroladamente ao perder seu rotor traseiro. Para anular tal efeito, os pares de rotores do VANT giram de forma contrária: dois no sentido horário e dois no sentido anti-horário. As

forças e demais elementos que influenciam na dinâmica de voo são exemplificados na Figura 30.

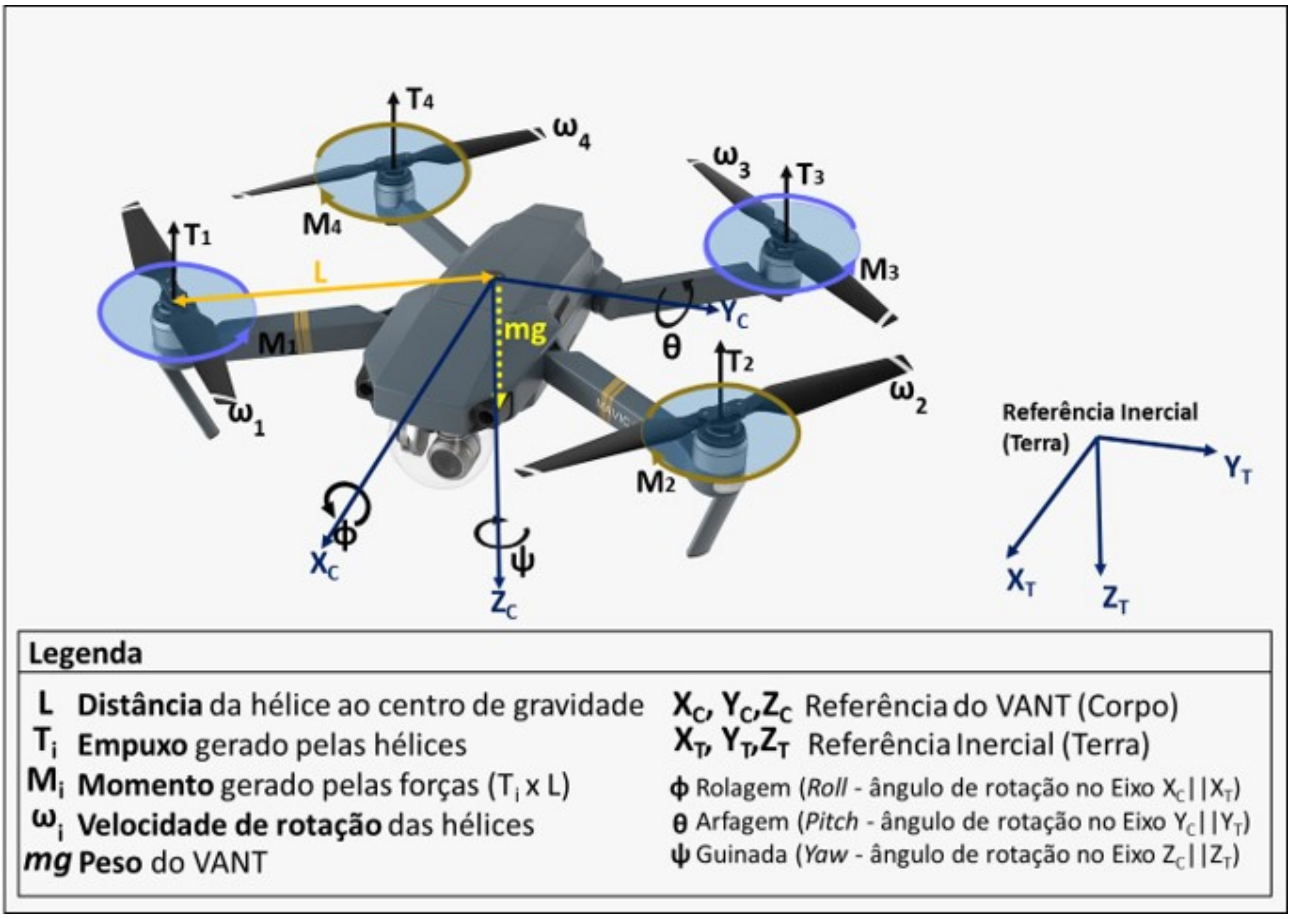



Figura 30 – Esquema de forças e elementos que influenciam no voo de um quadricóptero. Fonte: autor

Com base nessa figura, é possível perceber que existem dois sistemas de referência diferentes da atitude (ou orientação) do VANT: Inercial, que é a referência da Terra em si, e a do próprio corpo do VANT. Conforme Yali e Yuanxi (2012) explicam, a transformação entre estes dois sistemas de referência permite caracterizar a atitude do VANT em termos de ângulos de *Euler*: $\phi$, que é a **Rolagem** (ou *Roll*) ao longo do eixo X, $\theta$ que representa a **Arfagem** (ou *Pitch*) no eixo Y e, finalmente, $\psi$ que é a **Guinada** (ou *Yaw*) no eixo Z.

Dessa forma, a resultante das forças atuando sobre o VANT, onde há influência da rotação dos seus rotores, pode provocar deslocamento longitudinal ao longo dos eixos de referência e alteração nos valores dos ângulos de *Euler*. Os diferentes tipos de movimentos obtidos a partir disso são apresentados de forma didática nas subseções a seguir. Para informações detalhas e o modelo matemático completo, recomenda-se a bibliografia de Yali e Yuanxi (2012) e de Bresciani (2008).

#### 4.1.4.1 *Hover*

É o equilíbrio completo do VANT, fazendo com que ele plane sobre uma posição fixa. Nesse estado, representado na Figura 31 e resumido por Thu e Gavrilov (2017), há equilíbrio de:

- Forças: $\sum_{i=1}^{4} T_i = -mg$
- Direções: $T_{1,2,3,4} || g$
- Momentos: $\sum_{i=1}^{4} M_i = 0$
- Velocidade de rotação das hélices: $(\omega_1 + \omega_3) - (\omega_2 + \omega_4) = 0$

Tais condições implicam que os ângulos de *Euler* são nulos, ou seja, $\phi = \theta = \psi = 0$.

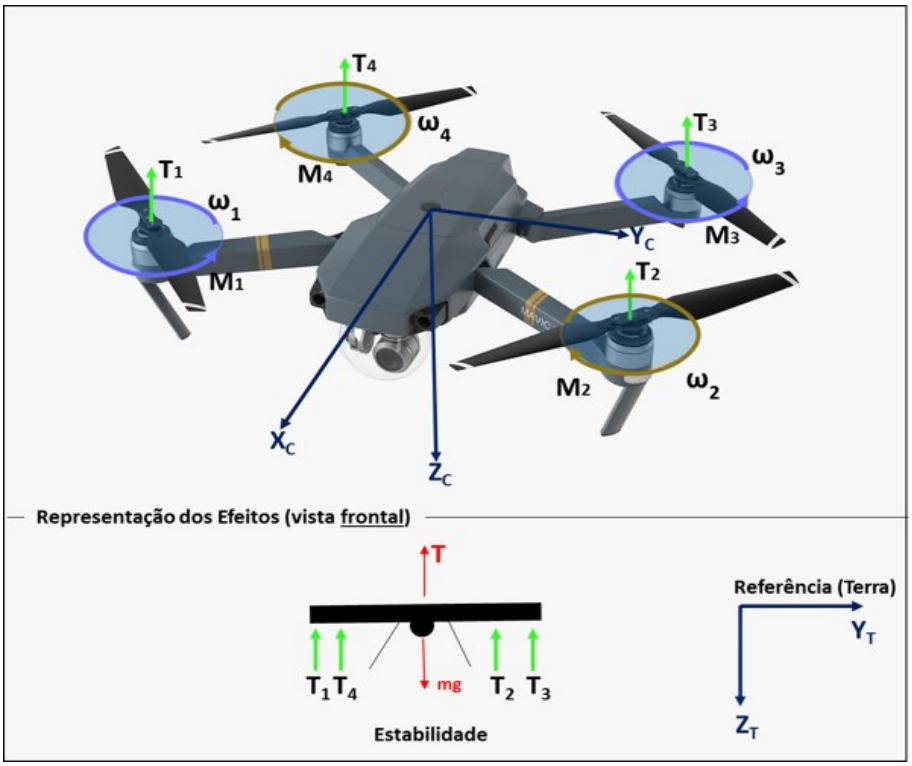



Figura 31 – Estabilidade do quadricóptero quando ele plana (*hover*). Fonte: autor

#### 4.1.4.2 *Deslocamento Vertical (Throttle)*

No movimento anterior, o empuxo das hélices é proporcional ao peso do VANT. Para que haja deslocamento vertical (eixo Z), todas as demais condições devem ser mantidas e a velocidade de rotação das hélices deve desequilibrar tal relação, gerando empuxo maior ou menor que o peso. Ou seja (THU; GAVRILOV, 2017):

- Desequilíbrio de forças: $\sum_{i=1}^{4} T_i \neq -mg$
- Direções: $T_{1,2,3,4} || g$
- Momentos: $\sum_{i=1}^{4} M_i = 0$
- Velocidade de rotação das hélices: $(\omega_1 + \omega_3) - (\omega_2 + \omega_4) = 0$

Neste contexto, o quadricóptero irá subir se o empuxo gerado for superior ao peso do VANT ($\sum_{i=1}^{4} T_i > -mg$) ou irá descer caso contrário ($\sum_{i=1}^{4} T_i < -mg$), vide esquema na Figura 32.

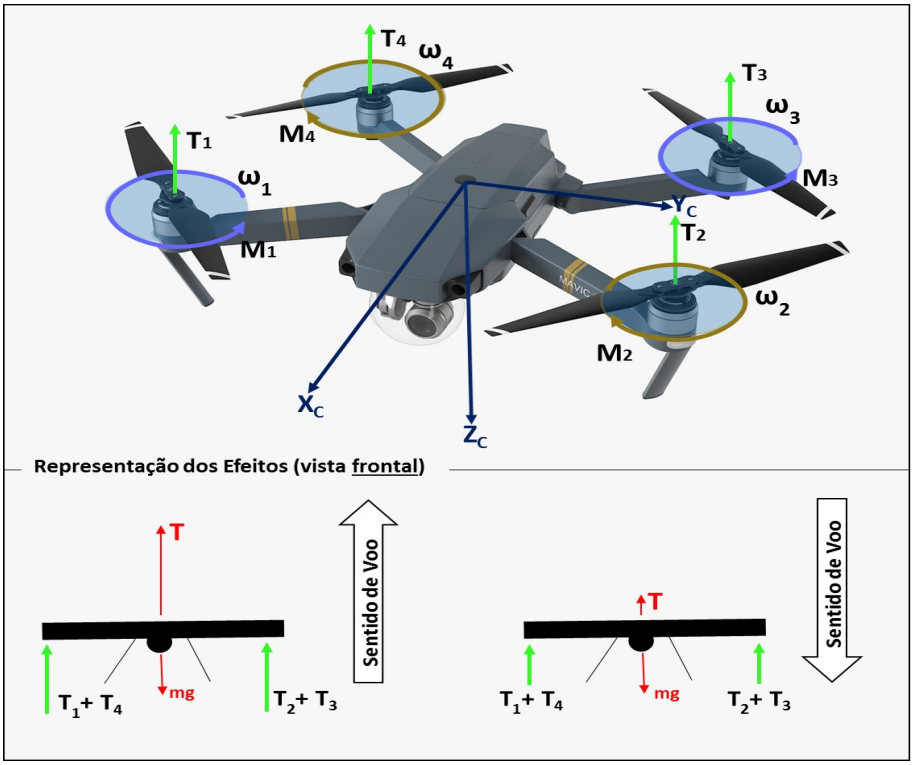



Figura 32 – Representação do movimento *throttle* de um quadricóptero. Fonte: autor

#### 4.1.4.3 *Roll*

O movimento de rolagem é obtido quando altera-se, de forma proporcional, a velocidade de rotação das hélices laterais do VANT (YALI; YUANXI, 2012). Isso faz com que ele incline-se para o lado oposto ao dos motores que tiveram a velocidade aumentada. Assim, o empuxo total é decomposto em duas forças: a de elevação (*lift* ou $T_L$), que tenta fazer com que o VANT suba, e a de arrasto (*drag* ou $T_D$), que provoca um deslocamento ao longo eixo Y, vide esquema da Figura 33.

Como visto no esquema, a alteração de velocidade dos motores laterais por uma constante $K_R$ provoca os seguintes efeitos (THU; GAVRILOV, 2017):

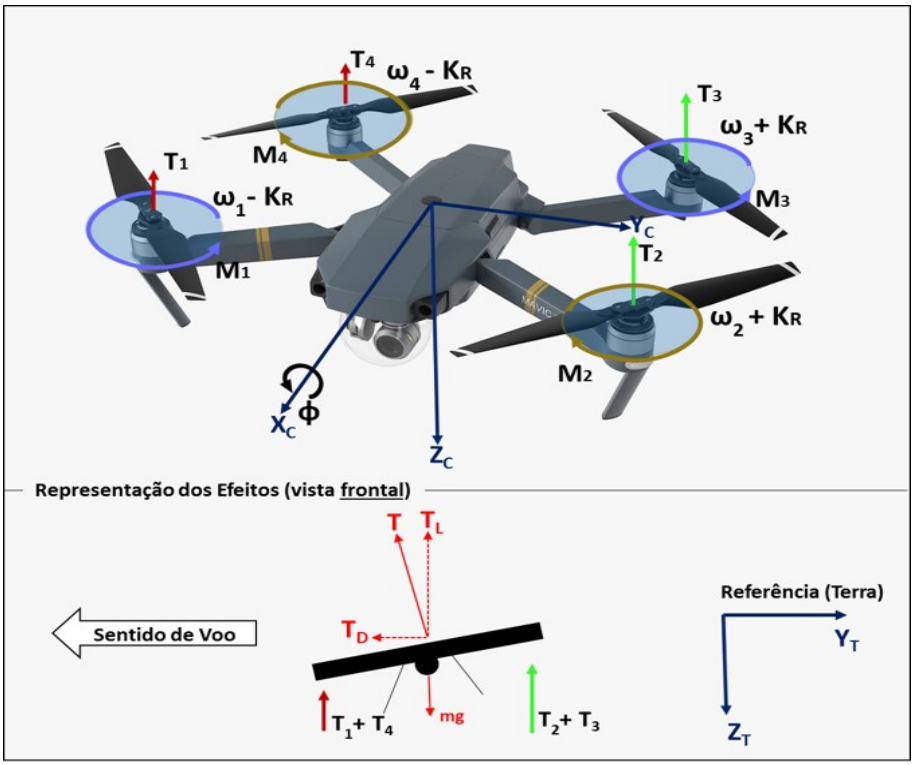



Figura 33 – Representação do movimento *roll* de um quadricóptero. Fonte: autor

- Desequilíbrio de forças: $\sum_{i=1}^{4} T_i \neq -mg$. Agora, essa força é decomposta em duas:
  - Força de Elevação (*Lift*): $T_L = T \cos \phi$
  - Força de Arrasto (*Drag*): $T_D = T \sin \phi$
- Desequilíbrio de direções: $T_{1,2,3,4}$ não é paralelo a $g$
- Desequilíbrio de Momentos: $\sum_{i=1}^{4} M_i \neq 0$
- Velocidade de rotação das hélices desbalanceada: $(\omega_1 + \omega_3) - (\omega_2 + \omega_4) \neq 0$

#### 4.1.4.4 *Pitch*

Similar ao *roll*, este movimento é obtido quando se altera proporcionalmente a velocidade de rotação das hélices na parte traseira ou frontal do VANT por uma constante $K_Y$, fazendo com que ele incline-se pra frente ou pra trás e desloque ao longo do eixo X (YALI; YUANXI, 2012).

O efeito das forças nesse caso é idêntico ao do movimento de rolagem, também ocorrendo arrasto. A Figura 34 apresenta o efeito das forças no movimento de *pitch* com uma representação lateral do VANT.

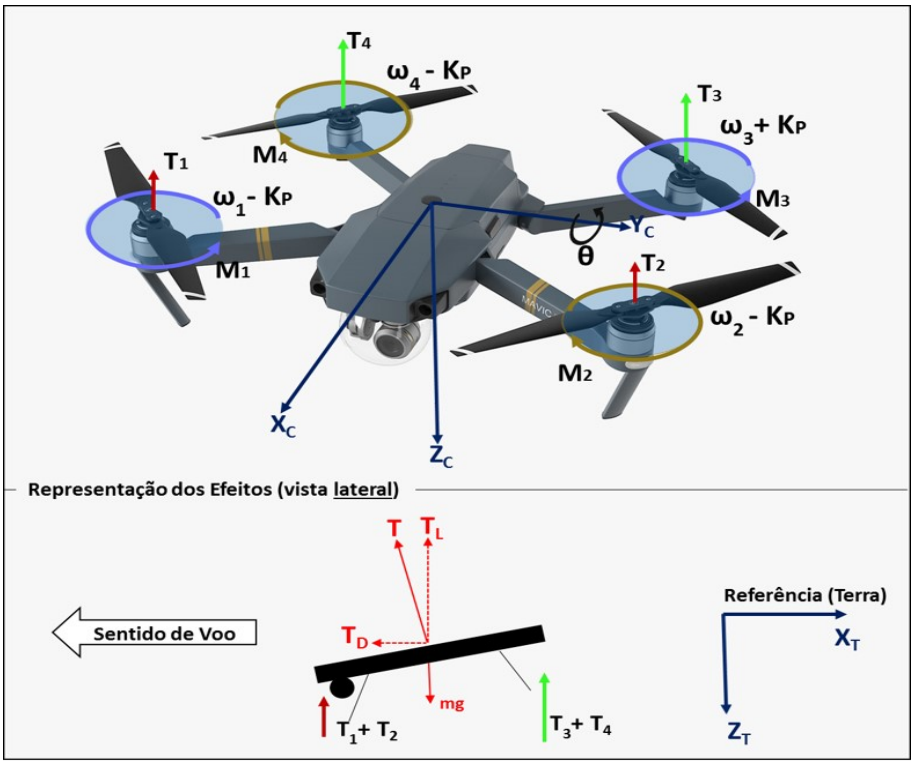



Figura 34 – Representação do movimento *pitch* de um quadricóptero. Fonte: autor

#### 4.1.4.5 *Yaw*

Quando a velocidade das quatro hélices é alterada de forma combinada entre os pares de hélices que giram no mesmo sentido, o quadricóptero irá realizar o movimento *Yaw*, girando em torno do eixo Z. Para isso, um par de motores irá girar mais rápido, enquanto o segundo par irá girar mais devagar, gerando um desequilíbrio. Com exceção da rotação, todas as demais forças estão equilibradas (THU; GAVRILOV, 2017):

- Equilíbrio de forças: $\sum_{i=1}^{4} T_i = -mg$
- Equilíbrio de Direções: $T_{1,2,3,4} || g$
- Equilíbrio de Momentos: $\sum_{i=1}^{4} M_i = 0$
- Velocidade de rotação das hélices desbalanceada: $(\omega_1 + \omega_3) - (\omega_2 + \omega_4) \neq 0$

Assumindo que o valor de aumento e redução de velocidade das hélices é uma constante $k_y$, obtém-se a taxa de rotação no eixo Z: $\dot{\psi} = K_y((\omega_1 + \omega_3) - (\omega_2 + \omega_4))$. Essas informações são representadas na Figura 35.

Em resumo, é possível notar que todos os tipos de movimento de um quadricóptero dependem diretamente da velocidade de rotação de seus motores, o que determina ainda sua velocidade de deslocamento e atitude (orientação). Porém,

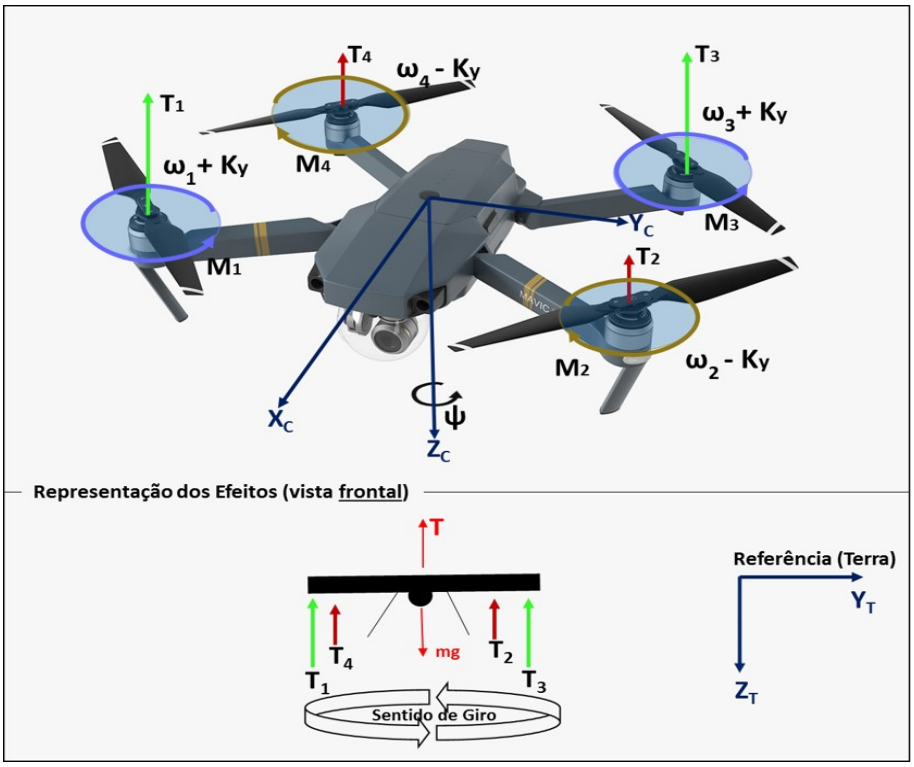



Figura 35 – Representação do movimento *yaw* de um quadricóptero. Fonte: autor

conforme as dinâmicas aqui apresentadas, os movimentos precisam ser coordenados entre todos os motores. Assim, a próxima subseção apresenta os desafios nessa tarefa e apresenta um sistema de controle de exemplo que conseguiu tornar os quadricópteros acessíveis.

### 4.1.5 Controle do Voo de um Quadricóptero

Conforme Bristeau et al. (2011) destacam, os quadricópteros são instáveis em malha aberta, tornando-os praticamente incontroláveis sem mecanismos apropriados. Assim, torná-los fáceis e seguros para que sejam acessíveis a pessoas comuns significa empreender um grande esforço na criação de sistemas de controle que consigam endereçar alguns desafios, por exemplo (BRISTEAU et al., 2011):

- O piloto deve prover apenas comandos em alto nível, que devem ser interpretados e executados pelo controlador de voo (FC) para se obter os diversos tipos de movimento possíveis;
- O VANT está sujeito a uma diversidade de condições e intempéries que afetam o voo, por exemplo, velocidade do vento, falhas na comunicação da aeronave com a estação de controle (GCS) ou perda de sinal do sistema de navegação GNSS;
- Pode ser necessário reduzir a velocidade ou interromper abruptamente a mo-

vimentação do veículo em situações de emergência ou na iminência de acidentes.

Tais fatores são críticos e, de certa forma, o tratamento de tais situações foi determinante para a popularização dos quadricópteros. Sem sistemas de controle adequados para prover tais características, é provável que as aeronaves teriam sua adoção restrita a aplicações muito específicas ou a entusiastas. Dessa forma, para exemplificar alguns dos mecanismos de controle presentes nesse tipo de aeronave, foi utilizada como referência a arquitetura do Parrot AR.Drone, lançado em 2010 e o primeiro quadricóptero comercial que utilizava um *smartphone* como controle remoto (OLIVER, 2017), o que reforça sua acessibilidade e aptidão para o público em geral.

Conforme discutido na subseção anterior, um dos grandes desafios para a movimentação do quadricóptero é a determinação correta de sua atitude (ou ângulos de *Euler*) e de sua velocidade de deslocamento ao longo dos eixos X, Y e Z. Assim, o sistema de controle precisa saber a atitude e velocidade do VANT a todo momento para calcular a diferença entre o estado atual e o estado solicitado pelo piloto (*setpoint*).

Dessa forma, a unidade de medição inercial (IMU, do inglês *Inertial Measurement Unit*), formada por acelerômetros e giroscópios, é a principal responsável por fornecer a atitude e velocidade. Porém, a IMU sozinha possui erros de desalinhamento com a estrutura do VANT, bias e ruídos que, mesmo com calibração meticulosa na linha de montagem, ainda atrapalham a determinação da atitude e velocidade. Por conta disso, uma das inovações desta arquitetura é a utilização de um sistema de visão para servir como referência, composto por uma câmera e algoritmos de processamento. As imagens capturadas sequencialmente do chão a 60 Hz são processadas para estimar a velocidade de deslocamento. Com isso, é possível avaliar a medição da IMU tendo uma referência, o que permite corrigir as leituras e obter a atitude de forma mais precisa (BRISTEAU et al., 2011).

Assim, a Figura 36 apresenta a arquitetura de controle do AR.Drone. Os elementos em cinza são os sensores, que fornecem a leitura bruta dos sinais de entrada, e atuadores, neste caso, apenas os quatro motores, já que é a variação de velocidade de rotação que resulta nos movimentos. Em laranja estão destacados os blocos que calculam e ajustam os *setpoints* nas diferentes etapas até estabelecer a velocidade de rotação necessária para realizar o movimento comandado pelo piloto ou por algum modo de navegação automatizada. Por fim, em azul, estão os algoritmos auxiliares, como o de visão, que suportam a correção de erros dos sensores da IMU e, assim, ajudam a obter um voo mais preciso e seguro.

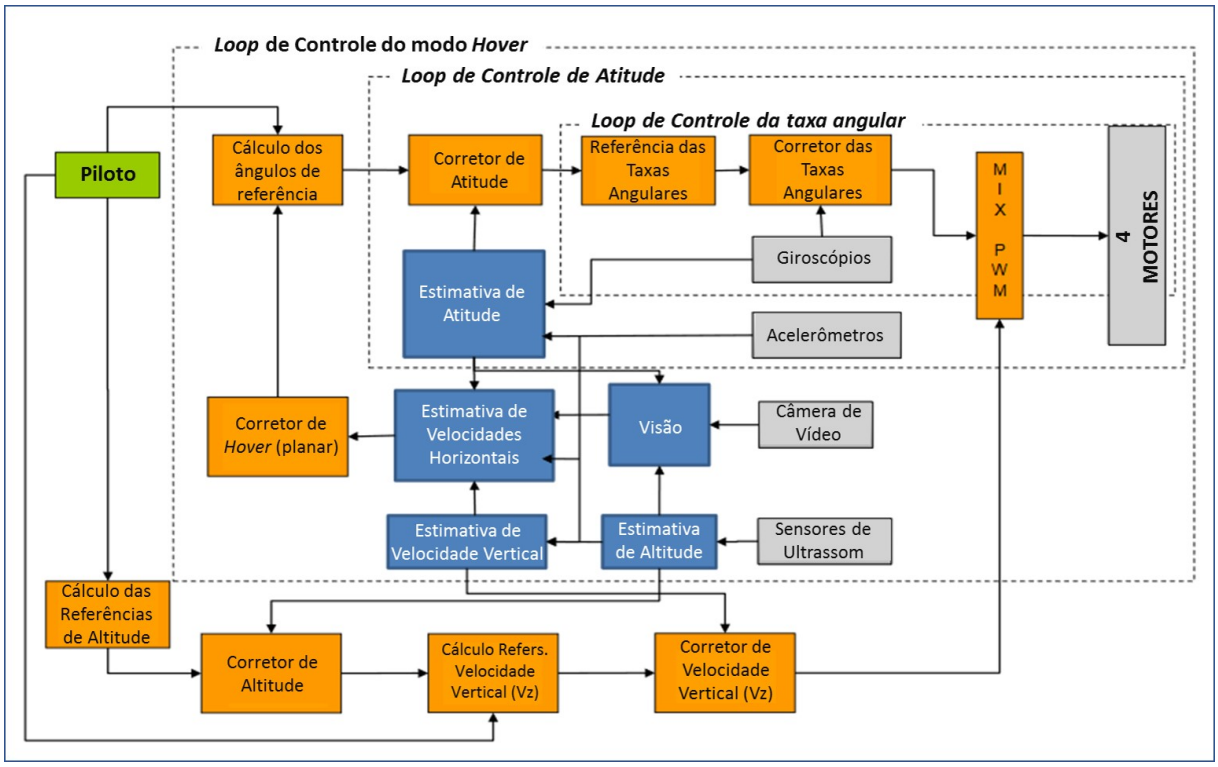



Figura 36 – Arquitetura de controle do Parrot AR.Drone, usada como referência. Fonte: traduzido de Bristeau et al. (2011)

Os elementos descritos anteriormente estão agrupados no controlador de voo (FC), que possui diferentes *loops* de controle. O mais abrangente deles é o "***Loop* de Controle do modo *Hover***" (planar) que utiliza um Controlador Proporcional Integral (PI) na estimativa de velocidade para alcançar o *setpoint* de velocidade e atitude zero, ou seja, a completa estabilidade. O segundo deles é o "***Loop* de Controle de Atitude**", que também utiliza um controlador PI para definir a taxa angular necessária para se obter o movimento solicitado pelo piloto à partir da atitude atual. Finalmente, o mais simples deles é o "***Loop* de Controle da Taxa Angular**", que utiliza Controladores Proporcionais (P) para controlar a taxa de rotação dos motores a partir da referência dada pelo *loop* superior (BRISTEAU et al., 2011).

Com esta hierarquia de controle foi possível estabilizar o voo e reproduzir o complexo conjunto de movimentos de um quadricóptero de forma acessível ao piloto, já que não há necessidade dele estar no controle o tempo todo. Por exemplo, o "*Loop* de Controle do modo *Hover*" se encarrega do controle do voo na ausência de comandos do usuário, mantendo a posição e atitude estáveis (zero), mesmo na presença de vento com baixa intensidade. Outro exemplo é o controle da trajetória de voo até a imobilidade. Se a aeronave está realizando um movimento a comando do piloto e ele o interrompe, o sistema de controle se encarrega de traçar uma trajetória para reduzir a atitude e deslocamento a zero, em um intervalo de tempo proporcional à velocidade de voo no momento da interrupção do piloto (BRISTEAU et al., 2011).

É evidente que os quadricópteros mais modernos de hoje contam com FCs ainda mais avançados e com sensores adicionais. Por exemplo, sistemas de navegação GNSS de alta precisão permitem determinar a velocidade instantânea da aeronave, tornando obsoleto o sistema auxiliar de visão do AR.Drone. Tais sistemas também podem ser usados para gravar a posição de decolagem e retornar automaticamente para ele em caso de falhas ou perda de comunicação. Adicionalmente, bússolas incorporadas ao sistema de controle auxiliam na orientação, desde que não exista interferência magnética.

Todos estes avanços são combinados em sistemas cada vez mais complexos, permitindo voos com grande grau de autonomia ou totalmente autônomos. Assim, ações como decolagem, pouso e sobrevoo podem ser programadas em *softwares* auxiliares, que geram missões e as transformam em instruções para que o FC execute. Assim, um último aspecto que merece atenção é a regulamentação, assunto discutido na próxima subseção. Ela é extremamente relevante, visto que o uso de VANTs segue orientações específicas em diferentes países, sendo que alguns deles restringem voos autônomos.

### 4.1.6 Regulamentação: O grande desafio

Independentemente das direções que os VANTs sigam no futuro, é fundamental que a regulamentação seja bem estabelecida, incorporando-os, de forma segura e definitiva, à sociedade. O grande desafio da regulamentação, segundo Rambaldi (2016), é que ela deve trazer segurança e privacidade à população, sem gerar barreiras intransponíveis para empresas e indústrias criarem aplicações úteis ou recreativas baseadas nesses veículos.

Com a recente evolução tecnológica, que tornou diversos modelos de VANTs acessíveis à população em diferentes países, governos vem sendo pressionados em relação ao controle e regulamentação destes veículos, já que sua utilização vem se espalhando com ou sem normatização. Nessa linha, vários países já se mobilizaram e publicaram orientações para definir as fronteiras no uso civil de VANTs, cada um com diferentes níveis de controle e permissividade.

Desde 2008, cada país membro da União Europeia é responsável por definir suas próprias regras para VANTs com peso inferior a 150 kg, conforme regulamentação “EC 216/2008” (EUROPEAN PARLIAMENT, 2008). Como isso gera fragmentações e barreiras para a expansão do mercado, a União Europeia, por meio de sua agência de segurança de aviação (EASA, do inglês *European Aviation Safety Agency*), tem como meta, até 2018, estabelecer uma nova regulamentação única e completa para os VANTs. Isso inclui tanto o aspecto de operação, com critérios

claros que delimitam regras de voo, quanto questões técnicas relativas à fabricação e comercialização desses veículos, o que inclui a certificação de VANTs com o selo "CE" (*Conformité Européenne*), que atesta a conformidade do produto com a legislação europeia em diferentes parâmetros, como saúde, segurança e meio ambiente (EUROPEAN AVIATION SAFETY AGENCY, 2017).

Nos Estados Unidos, o órgão de administração da aviação (FAA, do inglês *Federal Aviation Administration*) incluiu, em junho de 2016, uma nova seção no documento de regulamentação do sistema de aviação americano, a "*Part 107 - Small Unmanned Aircraft Systems*" (FEDERAL AVIATION ADMINISTRATION, 2016). Essa inclusão tem como objetivo classificar VANTs civis, definir critérios de certificação de pilotos e estabelecer limitações operacionais para o voo desses veículos.

No Brasil, a Agência Nacional de Aviação Civil (ANAC) publicou em Maio de 2017 a "RBAC-E 94 - Requisitos Gerais para Aeronaves Não Tripuladas de Uso Civil" (AGÊNCIA NACIONAL DE AVIAÇÃO CIVIL, 2017b), que estabelece a regulamentação para o voo de VANTs no território brasileiro. Ela procurou incorporar as principais ideias e avanços obtidos em legislações precursoras, como a americana, a europeia e a australiana, que está vigente desde 2002 e com a última atualização em agosto de 2017 (CIVIL AVIATION SAFETY AUTHORITY, 2017).

A legislação brasileira estabelece a nomenclatura de "**Aeromodelo**" para todo VANT usado para recreação e "**Aeronave Remotamente Pilotada**" (RPA, do termo original em inglês *Remotely Piloted Aircraft*) quando o uso é comercial, experimental ou corporativo. Como o próprio nome denota, a aeronave deve ser pilotada remotamente, visto que voos autônomos[1] são vetados. Os RPAs foram divididos em três classes, com diferentes requerimentos regulatórios:

- **Classe 1**: para veículos acima de 150 kg de peso total (aeronave e carga). Essa classe está sujeita a um processo de certificação individual, similar a aeronaves tripuladas, sendo necessário que essa certificação conste no Registro Aeronáutico Brasileiro;

- **Classe 2**: para veículos entre 25 e 150 kg de peso total. Os fabricantes devem realizar o processo de certificação, que é por modelo e não individual como na Classe 1. Também devem constar no Registro Aeronáutico Brasileiro para poderem voar;

- **Classe 3**: para veículos com peso total inferior a 25 kg. A ANAC subdivide esta

[1] O texto da legislação (RBAC-E 94) define autônomo apenas como "sem intervenção de piloto remoto", sem deixar claro se missões por *waypoint*, onde o piloto pode assumir o controle a qualquer momento, se enquadram nesse critério

classe em: a) veículos que voam sem linha de visada ou acima de 120 m; e b) veículos que voam abaixo desse teto e com linha de visada. No primeiro caso, o fabricante deve certificar o modelo, que deve ainda constar no Registro Aeronáutico Brasileiro. Já no segundo caso, basta um cadastro do VANT e do operador no Sistema de Aeronaves Não Tripuladas (SISANT), não havendo necessidade de aprovação ou certificação do projeto pela ANAC. Os VANTs usados neste projeto, que serão melhor detalhados no Capítulo 5, encaixam-se nesta classe.

Ainda segundo Agência Nacional de Aviação Civil (2017b), veículos com peso total inferior a 250 g, independentemente do tipo de uso, estão isentos de registro, mas devem obedecer à distância mínima de 30 metros de terceiros, que vale para voos com veículos de qualquer classe. Os pilotos devem possuir idade mínima de 18 anos e, dependendo da classe de aeronave que irão operar, possuir documentação especial, como certificado médico, licença de voo e habilitação emitidas pela ANAC. A Tabela 4 resume os principais pontos estabelecidos pela norma vigente no Brasil.

Sem discutir se a legislação brasileira é moderna ou permissiva o bastante, é fato que ela traz segurança jurídica e permite o desenvolvimento de novos modelos de negócios usando essas aeronaves, o que, por si só, é positivo. Por ser tão recente, é esperado que melhorias venham a ser incorporadas com o tempo e com a evolução da tecnologia. Portanto, é fundamental que os órgãos regulatórios procurem sempre trabalhar para que os VANTs e demais inovações possam ser incorporados à sociedade da forma mais segura possível, sem minar o desenvolvimento de novas aplicações que contribuam para a melhoria de bem estar, qualidade de vida e outros importantes aspectos sociais.

## 4.2 Paradigmas de Computação: *Edge*, *Fog* e *Cloud*

Quando se avalia as abordagens para processamento de dados nos últimos dez anos, percebe-se que vem ocorrendo uma massiva migração da infra-estrutura antes existente nas empresas (ou *on-premises*) para serviços e plataformas em nuvem, a chamada *Cloud Computing*. De forma simplificada, a nuvem pode ser vista como um grande provedor de recursos para processamento, armazenamento, hospedagem de aplicações, transferência de dados, etc., em que o consumidor paga apenas pelo uso que faz de cada serviço oferecido por grandes provedores como Amazon, Google e Microsoft (ARMBRUST et al., 2010).

Tabela 4 – Resumo dos principais pontos introduzidos na legislação brasileira sobre VANTs

| | **RPAS Classe 1** | **RPAS Classe 2** | **RPAS Classe 3** | **Aeromodelos** |
|---|---|---|---|---|
| **Registro da Aeronave?** | Sim | Sim | BVLOS: Sim<br>VLOS: Sim | Sim |
| **Aprovação do Projeto?** | Sim | Sim | Apenas BVLOS ou acima de 400 pés | Não |
| **Limite de Idade para Operação?** | Sim | Sim | Sim | Não |
| **Certificado Médico?** | Sim | Sim | Não | Não |
| **Licença e Habilitação?** | Sim | Sim | Apenas para operações acima de 400 pés | Apenas para operações acima de 400 pés |
| **Local de Operação** | A distância do voo não poderá ser inferior a 30 metros horizontais de pessoas não envolvidas ou anuentes com a operação, exceto se houver barreira mecânica e suficientemente robusta para isolar e protegê-las. Tal restrição não é válida para operações de órgãos de segurança pública, polícia, fiscalização tributária e aduaneira, de combate a vetores de transmissão de doenças, de defesa civil e/ou do corpo de bombeiros (ou operadores contratados por tais órgãos). | | | |

Fonte: consolidado a partir de Agência Nacional de Aviação Civil (2017a)

Esse movimento é explicado por diferentes fatores, onde destacam-se a redução dos custos, infinitamente menores na nuvem, a escalabilidade dos serviços, que permite que as empresas ou aplicações cresçam ou diminuam de forma rápida e sob demanda, e a confiabilidade, uma vez que o provedor se encarrega de *backups* e redundâncias, estabelecendo níveis de serviço (ARMBRUST et al., 2010). Comparativamente, se uma empresa com infra-estrutura *on-premises* necessita de um novo servidor de banco de dados, ela deve comprar o *hardware*, adquirir licenças de *software* e alocar um profissional para fazer a instalação. Na computação em nuvem é possível, com alguns cliques em uma página *web*, aprovisionar uma máquina virtual já com o sistema operacional e serviço de banco de dados instalados e prontos para uso.

Porém, a realidade de IoT, envolvendo dispositivos restritos em termos de processamento e largura de banda, muitas vezes em aplicações de missão crítica, traz consigo outros requisitos, onde nem sempre é possível enviar os dados para a nuvem e aguardar seu processamento para tomar uma ação, já que a latência

poderia comprometer o resultado. Com isso, surgiram os conceitos de *Fog* e *Edge Computing*, que, de forma bastante resumida, procuram trazer recursos computacionais para mais próximo dos dispositivos que geram os dados, levando ou obtendo dados da nuvem apenas quando necessário (SHI et al., 2016).

A computação *Edge* é um conceito, resumido por Shi et al. (2016) como um conjunto de tecnologias que permite que o processamento de dados seja feito na borda (*edge*) da rede, ou seja, no caminho entre o dispositivo gerador do dado e a nuvem. Isso se justifica porque o poder de processamento evoluiu e se tornou barato, enquanto a transmissão de dados é afetada pela distância em relação ao local de processamento. Mesmo no caso da fibra ótica, há um aumento de 1 ms a cada 100 km de distância (SONG; KIM; MUKHERJEE, 2010). Dessa forma, em alguns casos, é preferível processar parte dos dados localmente a transmiti-los na totalidade para a nuvem. Isso reduz a latência, o tráfego na rede e permite que algumas aplicações não sejam interrompidas na ausência de conexão com os serviços em nuvem, algo essencial em alguns casos, como carros e cidades inteligentes (PANG; TAN, 2004). Porém, nem sempre é possível ou conveniente embarcar todo poder de processamento nos dispositivos em si ou nos *gateways* de comunicação com a nuvem.

Por conta disso, a computação *Edge* é reforçada pela proposta da chamada *Fog Computing* que, essencialmente, procura prover serviços de nuvem dentro de uma área geográfica limitada, suportando os diferentes dispositivos conectados a ela. Ainda que não conte com todo poder da nuvem, a *Fog* é um passo intermediário entre processamento e latência, permitindo que aplicações de missão crítica sejam executados e que diversos dispositivos interajam entre si, numa escala bem mais abrangente que a obtida na computação *Edge* (BONOMI et al., 2012; VAQUERO; RODERO-MERINO, 2014).

Um exemplo de implementação do conceito de *Fog* é a *Mobile Edge Computing* (MEC) que, conforme Liang (2017) explica, procura prover um ambiente de nuvem para processamento de dados dentro da área de cobertura de redes móveis (RAN, do inglês *Radio Access Network*). Ou seja, as estações-base, que permitem a conectividade dos dispositivos móveis, também proveem processamento e serviços dentro da área coberta por ela. Esse conceito é colocado como uma das tecnologias essenciais para o 5G e vem sendo padronizado pelo *3rd Generation Partnership Project* (3GPP), conforme trabalho de Hu et al. (2015).

A Figura 37 demonstra como esses três conceitos se relacionam, ressaltando abrangência de cada um desses paradigmas, puramente local no caso do *Edge* e em escala global na *Cloud*. Além disso, o poder de processamento é inversamente proporcional à latência na transmissão de dados. Enquanto a *Cloud* é a que oferece

maior poder computacional, praticamente ilimitado, é a que possui maior latência para receber e devolver os dados para os dispositivos, na base da pirâmide. Por fim, apesar da figura representar alguns dispositivos já como parte da camada *Edge*, vale lembrar que alguns deles não possuem processamento embarcado, podendo utilizar dispositivos de rede inteligentes para prover tais funcionalidades, também conhecidos como *Edge Gateways*.

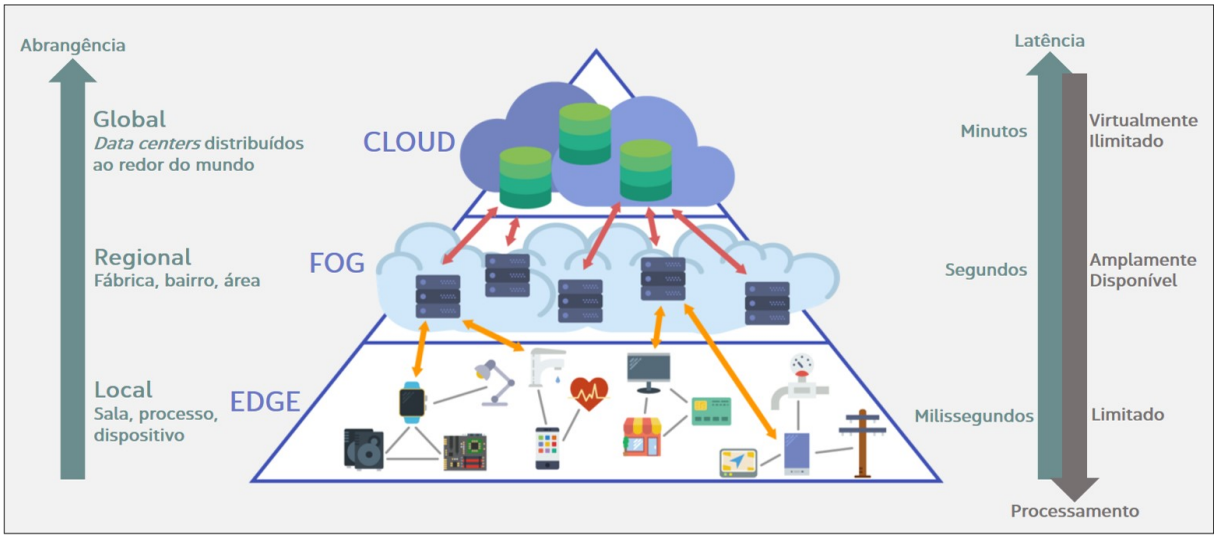



Figura 37 – Representação dos paradigmas de computação *Edge*, *Fog* e *Cloud* com sua respectiva abrangência. Fonte: adaptado de SPOTNITZ, 2017

É parte da definição de arquitetura de cada sistema ou plataforma a avaliação dos paradigmas empregados, ou seja, se os requisitos de latência comportam comunicação direta com a nuvem para processamento ou se existem demandas de tempo real que trazem a necessidade de processamento mais próximo dos dispositivos. No presente trabalho, esses conceitos foram parcialmente empregados para suportar a definição de camadas da arquitetura integrada proposta, particularmente as camadas de "**Extração de Dados de Sensores**" e "**Insights e Conhecimento**", que são melhor descritas na seção "**5.2 - Arquitetura Integrada**".

## 4.3 Técnicas e Protocolos de Integração

Quando se avalia a Plataforma de Inspeção proposta neste trabalho, é importante destacar que a mesma precisa se encaixar em um arcabouço de sistemas e soluções já existentes, que possibilitam a gestão integrada dos processos de inspeção e manutenção dos ativos, neste caso, dos rolos dos transportadores de correia. Por conta disso, ela deve prover informações a esses sistemas e receber dados que auxiliem a inspeção, o que só pode ser conseguido por meio da integração entre tais agentes.

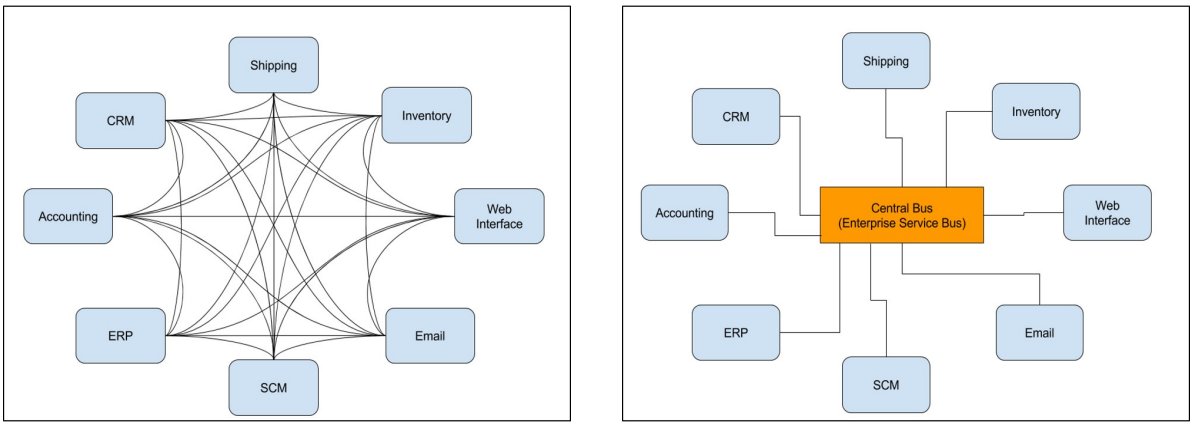
Por conta disso, esta seção apresenta os principais conceitos, ferramentas, técnicas e protocolos envolvidos quando há a necessidade de integração de aplicações heterogêneas, focando nos principais casos de uso esperados para integração da Plataforma de Inspeção com Sistemas Corporativos. Os conceitos e protocolos abordados foram utilizados para suportar a definição da camada de **Integração** da Arquitetura Integrada, apresentada na subseção "**5.2.4 - Integração**". Para atingir tais objetivos, a presente seção foi organizada da seguinte forma:

- Primeiramente, a subseção "**4.3.1 - Arquitetura Orientada a Serviços**" discute princípios para construção de serviços que visam a interoperabilidade e baixo acoplamento na comunicação entre as aplicações;
- Em seguida, a subseção "**4.3.2 - Barramento de Integração**" apresenta as principais funções deste elemento e como ele pode facilitar a comunicação entre aplicações;
- Por sua vez, a subseção "**4.3.3 - Casos de Uso x Padrões de Mensagens**" classifica os principais tipos de interação de uma Plataforma de Inspeção com sistemas corporativos em padrões de troca de mensagem existentes na literatura;
- A subseção "**4.3.4 - Comunicação no Padrão *Request-Response***" explora os principais protocolos e técnicas usadas nessa forma de comunicação e discute sua complementariedade;
- Por fim, a subseção "**4.3.5 - Comunicação no Padrão *Publish-Subscribe***" apresenta alguns dos protocolos de comunicação mais populares nessa modalidade e conclui a seção com uma comparação entre eles e a definição de qual será utilizado na Plataforma de Inspeção.

### 4.3.1 Arquitetura Orientada a Serviços

Tradicionalmente, quando há a necessidade de comunicação entre dois sistemas, adota-se uma integração ponto-a-ponto, em que comunicam-se diretamente, sem nenhum tipo de agente intermediário. Infelizmente, esse tipo de abordagem costuma trazer mais problemas e desafios do que soluções de fato, visto que cada parte envolvida pode ser escrita em uma linguagem de programação distinta, utilizarem protocolos incompatíveis e ter poder computacional desproporcional, desbalanceando a comunicação. Além disso, em ambientes empresariais complexos, com múltiplas aplicações que necessitam se comunicar, esse tipo de abordagem torna-se um problema para ser gerenciada, aumentando a complexidade de projetos de migração, decomissionamento e implantação de novas soluções. Conforme

exemplo na Figura 38a, esse é o motivo pelo qual essa prática de integrações ponto-a-ponto também é conhecida como “Integração Espaguete” (INDRASIRI, 2016).




(a) Abordagem de integração ponto-a-ponto ou “espaguete”, de difícil administração

(b) Adição do ESB como elemento central de comunicação entre aplicações

Figura 38 – Abordagem de integração ponto-a-ponto em comparação ao uso de um ESB. Fonte: FERNANDO, 2016

Uma das formas propostas para reduzir esses e outros problemas na integração entre sistemas, é o conceito de Arquitetura Orientada a Serviços (SOA, do inglês *Service-Oriented Architecture*), proposto por Thomas Erl. Ele estabelece uma série de princípios que tem como objetivo final promover a interoperabilidade ao abstrair detalhes de implementação de cada parte envolvida em uma integração e definir “Serviços” que se tornam a forma de comunicação entre as pontas. O SOA estabelece oito princípios, apresentados com os nomes originais em inglês para preservar a essência de cada um e traduzidos livremente pelo autor para melhorar o entendimento (ERL, 2008):

1. ***Standardized Service Contract* (Contrato de Serviço Padronizado)**: estabelece os “termos” e “condições” com que um serviço é prestado, ou seja, quais são as funcionalidades e estruturas de dados que regulam a comunicação entre as partes. É o elemento básico para que as partes envolvidas consigam se entender;

2. ***Service Loose Coupling* (Desacoplamento de Serviço):** procura diminuir a dependência entre as partes, promovendo desacoplamento. Isso permite que um serviço evolua sua lógica, implementação interna e capacidades sem que seus consumidores atuais sejam afetados por isso;

3. ***Service Abstraction* (Abstração de Serviço):** suporta o princípio anterior, de desacoplamento, ao estabelecer meta-dados e outros mecanismos que tor-

nam detalhes internos da implementação de cada serviço transparentes para seus consumidores;

4. ***Service Reusability* (Reutilização de Serviço):** cada serviço pode ser usado por múltiplos consumidores, cada um com seu caso de uso específico, sem a necessidade de adaptações e reescrita. Para tanto, o serviço deve suportar completamente uma função de negócio, ainda que esta seja apenas uma pequena parte de um processo de negócio maior;

5. ***Service Autonomy* (Autonomia do Serviço):** o serviço deve ser desenvolvido de forma a conseguir lidar, por si só, com diferentes questões encontradas em um ambiente produtivo, como tipos de rede, ambientes, etc. Tais detalhes não devem ser de conhecimento de seus consumidores e fazem parte de sua lógica interna;

6. ***Service Statelessness* (Independência de Estado do Serviço):** cada requisição a um serviço deve ser isolada e tratada de forma independente das requisições anteriores, ou seja, os dados usados só são conhecidos e válidos pela duração da chamada específica;

7. ***Service Discoverability* (Descoberta do Serviço):** os contratos e meta-dados do serviço devem ser redigidos de modo a facilitar sua descoberta por consumidores externos que demandam a funcionalidade que o mesmo provê, promovendo o reuso;

8. ***Service Composability* (Composição de Serviços):** os serviços devem ser criados da forma mais atômica possível, ou seja, para suportar um objetivo específico. Assim, cada serviço resolve um pequeno problema, enquanto a composição deles resolve problemas (ou requisitos de negócio) cada vez maiores.

Por si só, esses princípios já auxiliam na organização e reduzem parte dos problemas enfrentados em integrações ponto-a-ponto, mas não resolvem todas as situações. Por exemplo, imagine que uma empresa utiliza o Sistema “A” para gestão de manutenção, que expõe um serviço para “Criação de Notas de Manutenção” para seus consumidores. Se a empresa fizer melhorias no Sistema “A”, considerando os princípios de SOA, seus consumidores não seriam afetados, o que, por si só, é positivo. Porém, e se o Sistema “A” for substituído pelo novo Sistema “B” para gestão de manutenção? Todos os consumidores teriam que saber que existe um novo provedor e direcionar a comunicação para o serviço de criação de notas da nova aplicação, o que é um dos maiores problemas na abordagem de “Integração Espaguete”, exemplificada na Figura 38a.

Dessa forma, a implementação completa dos princípios de SOA geralmente é acompanhada de um elemento central, que procura concentrar os serviços numa camada única, vide exemplo na Figura 38b, aumentando o nível de abstração. Neste caso, os sistemas envolvidos em uma integração comunicam-se com os serviços expostos no Barramento de Integração (ESB, do inglês *Enterprise Service Bus*), e não diretamente com o serviço da outra aplicação. Dessa maneira, durante a troca do Sistema "A" para o Sistema "B", exemplificada anteriormente, o serviço de "Criação de Notas de Manutenção" no ESB seria reapontado para o novo Sistema "B", sem que os consumidores ao menos soubessem que outra aplicação passou a ser responsável pela gestão das notas de manutenção. Segundo Indrasiri (2016) é esse tipo de centralização que traz ganhos à empresa ao reduzir a complexidade em projetos de Tecnologia da Informação (TI).

Assim, em grandes corporações, como as da indústria de mineração, para as quais o presente trabalho é relevante, é praticamente impossível realizar integrações sem o uso de um Barramento de Integração (ou ESB), visto que, acima de tudo, ele é um grande facilitador. Dessa forma, a subseção a seguir apresenta formalmente o conceito de ESB e as principais funções que ele deve desempenhar para suportar os conceitos de orientação a serviços discutidos.

### 4.3.2 Barramento de Integração

Conforme Indrasiri (2016), o ESB não é uma ferramenta em si, mas um padrão de arquitetura, que também pode ser visto como uma melhor prática, que tem como objetivo prover um barramento de comunicação comum que permite que aplicações heterôgeneas e geograficamente distribuídas consigam comunicar-se de forma transparente e centralizada.

Para que o ESB consiga suportar os oito princípios da arquitetura orientada a serviços, ele geralmente deve possuir um conjunto mínimo de funcionalidades, que foram resumidos abaixo seguindo descrições de diferentes autores:

- **Orquestração**: o ESB precisa combinar múltiplos serviços para suportar processos de negócio complexos, ou seja, ele orquestra a invocação de serviços "atômicos", especializados em atividades específicas, que se complementam e suportam o processo em questão, escondendo da aplicação solicitante essa complexidade (DUNPHY et al., 2009);
- **Conversão de Protocolos:** um ESB deve suportar diferentes protocolos de comunicação por meio de adaptadores, o que o habilita a comunicar-se com diferentes serviços e aplicações usando o(s) protocolo(s) específico(s) de cada

um. Dessa forma, as pontas envolvidas na integração não precisam saber o protocolo de comunicação da outra, pois o ESB se encarrega de fazer a conversão (INDRASIRI, 2016);

- **Transformação de Mensagem:** assim como os protocolos, cada aplicação suporta estruturas de dados e formatos de arquivos distintos. Portanto, um ESB deve conseguir realizar a tradução de dados entre sistemas, alterando o conteúdo das mensagens para que elas se tornem compatíveis com o contrato do serviço sendo consumido (DUNPHY et al., 2009);

- **Mediação:** com base no conteúdo da mensagem, o ESB deve ser capaz de fazer o roteamento e entrega para os serviços correspondentes. Da mesma forma, ele também precisa filtrar mensagens que não devem ser entregues ou guardá-las por um tempo para realizar a entrega mediante um evento específico (INDRASIRI, 2016);

- **Controle de Fluxo:** cada sistema provedor de um serviço possui suas próprias limitações em termos de capacidade de processamento. Dessa forma, o ESB deve assegurar que a entrega de mensagens seja feita de acordo com a capacidade das partes envolvidas na integração, sem sobrecarregar uma delas ou atrasar o processo (CHAPPEL, 2004);

- **Adaptação de Modos de Comunicação:** cada serviço pode ter diferentes formas de comunicação, como síncrona e assíncrona. Como analogia, a comunicação síncrona pode ser vista como o envio de uma carta com aviso de entrega e rastreamento, em que o remetente consegue acompanhar e confirmar a entrega. O segundo caso é um envio tradicional, em que uma vez despachada, não é possível saber se e quando foi recebida. Assim, cabe ao ESB fazer a adaptação das formas de comunicação entre os diferentes serviços e aplicações que operem com cada uma dessas formas (ENDREI et al., 2004).

De acordo com Keen et al. (2004), existem ainda funções adicionais que podem ser vistas como fundações de um ESB, portanto, não foram detalhadas na lista anterior. Alguns exemplos são os diretórios de descoberta de serviços, mecanismos de autenticação e segurança, consoles de gestão e monitoramento dos serviços e conectores (*gateways*), que abstraem para os consumidores detalhes de conectividade entre redes, ou seja, se um serviço está rodando em nuvem ou localmente.

É importante destacar que cada solução de mercado pode ter diferentes níveis de abrangência para tais funcionalidades. Por exemplo, em relação à função de

“Conversão de Protocolos”, algum fabricante pode suportar mais ou menos protocolos e padrões de indústria que outros. Isso significa que as ferramentas de mercado, como TIBCO Business Works (TIBCO SOFTWARE, 2017), Apache ServiceMix (APACHE SOFTWARE FOUNDATION, 2017) e Mulesoft Anypoint Platform (MULESOFT, 2017), para citar apenas alguns exemplos, diferenciam-se na completude com que suportam cada requisito, mas para serem consideradas um ESB, devem suportar minimamente o conjunto de funcionalidades discutido nesta subseção.

A Figura 39 apresenta, de forma esquemática, como o ESB habilitou a integração da Plataforma de Inspeção com as aplicações existentes. Nesta figura é possível observar os diferentes adaptadores disponíveis, alguns dos serviços que são providos pelos sistemas corporativos e exemplos dos tópicos usados para troca de mensagens. É importante destacar que diversas das funções do ESB facilitaram os fluxos de integração desenvolvidos no projeto, por exemplo, realizando a conversão entre protocolos e abstraindo detalhes de conexão de rede. Mais detalhes de como o ESB suportou esse processo e de como ele se encaixa na arquitetura proposta são discutidos no Capítulo “**5 - Desenvolvimento**”.

### 4.3.3 Casos de Uso x Padrões de Mensagens

O fato de já existir um ESB implementado e disponível para uso na empresa onde foi realizado este projeto não exclui a necessidade de avaliar os casos de uso e requisitos de integração que a Plataforma de Inspeção possui em relação aos sistemas corporativos existentes. Dessa forma, é a partir da avaliação desses requisitos que são determinados os protocolos e tecnologias mais aderentes a serem usadas para conexão com o ESB e aplicações provedoras ou consumidoras de informações.

Apesar de diversos casos de uso serem possíveis, os principais foram reunidos em três tipos de interação, que agrupam características semelhantes e, em última instância, irão direcionar o tipo de tecnologia de integração a ser adotado para o dado em questão. Essas interações são exemplificadas na Figura 39 e melhor explicadas nos itens abaixo:

- **Ação Durante a Inspeção:** representa as ações que o inspetor ou a Plataforma de Inspeção, de forma automática, pode realizar ao longo da inspeção, enviando dados para sistemas corporativos. Nesse caso, as interações são disparadas pela Plataforma de Inspeção, que é provedora da informação, em mensagens orientadas a eventos (“*event-driven*”), ou seja, enviadas quando um evento específico acontece. Um exemplo é a abertura de uma Ordem no

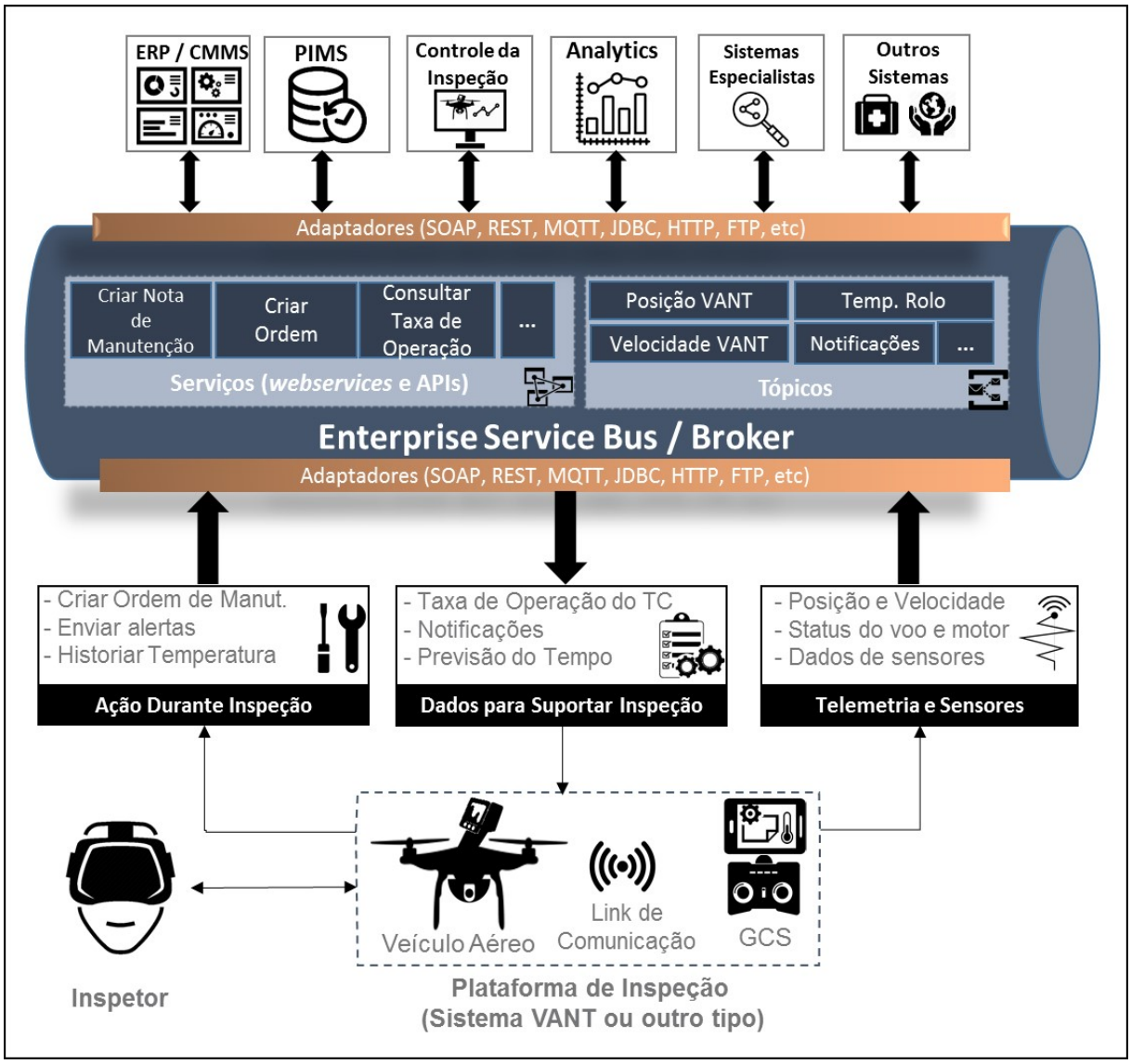



Figura 39 – Visão geral de como a plataforma de inspeção se encaixa aos sistemas existentes e alguns casos de uso de integração. Fonte: autor

Sistema de Manutenção (CMMS, do inglês *Computerized Maintenance Management System*) quando um defeito é detectado pela plataforma. É importante que tais eventos tenham garantia de entrega, visto que se a ordem não é gerada, o defeito não será tratado;

- **Dados para Suportar a Inspeção:** dizem respeito a informações que podem suportar o processo de inspeção e que são do domínio de outros sistemas. Eles podem disparar mensagens mediante eventos específicos para alertar o inspetor para alguma situação ou configurar a Plataforma de Inspeção remotamente. Por exemplo, o sistema de “Controle de Inspeção” pode enviar configurações atualizadas da câmera térmica para lidar com variações de umidade e temperatura ambiente. Também é importante garantir a entrega das mensagens, visto que falhas no processo podem resultar em uma inspeção com

dados incorretos ou incompletos;

- **Telemetria e Sensores:** são os dados gerados automaticamente pela Plataforma de Inspeção e seus diversos sensores. Os dados irão variar de acordo com o portador utilizado, mas como características comuns, nascem de forma praticamente contínua, sem interação do usuário, e em pequenas quantidades por mensagem. Exemplificando, uma Plataforma de Inspeção baseada em um Sistema VANT como portador pode publicar os dados de posição (Latitude, Longitude e Altitude) do veículo aéreo de forma contínua para que o sistema de "Controle de Inspeção" e qualquer outro sistema interessado consiga acompanhar, remotamente, o deslocamento da aeronave. Nesse caso, se uma mensagem for perdida, não há grande prejuízo para o processo como um todo e diferentes aplicações podem receber os dados publicados.

Ao avaliar as características destas interações diante dos vários **Padrões de Mensagem** organizados por Erl (2008), percebe-se que elas se encaixam em dois deles, conhecidos como ***Request/Response*** e ***Publish/Subscribe***. Enquanto os grupos de interações chamados **"Ação Durante a Inspeção"** e **"Dados para Suportar a Inspeção"** possuem propriedades que se encaixam no primeiro paradigma, o grupo referido como **"Telemetria e Sensores"** possui características de interação mais próximas ao segundo. Esses Padrões de Mensagem são brevemente explicados abaixo:

- ***Request/Response:*** um Consumidor (Cliente) faz uma solicitação (***Request***) a um Provedor de Serviço (Servidor) e o mesmo devolve uma resposta (***Response***). Nesse caso, é o cliente quem coordena o processo, fazendo as solicitações (*Requests*) à medida que ele necessita consumir um serviço para obter ou enviar um dado ao provedor do mesmo. Apesar de a comunicação entre esses agentes poder ser direta, conforme já explicado na subseção "**4.3.2 - Barramento de Integração**", o uso de um ESB traz diversas vantagens e seu uso foi adotado na representação desse paradigma na Figura 40 (RODRÍGUEZ-DOMÍNGUEZ et al., 2012). Uma analogia para apoiar no entendimento desse padrão é um serviço de entrega de pizza. Um cliente faz o pedido a uma pizzaria (provedor do serviço), que a prepara e realiza a entrega mediante o que foi solicitado pelo cliente. A pizza só é preparada e enviada se houver um pedido de um cliente;

- ***Publish/Subscribe:*** conforme explicado por Hohpe e Woolf (2012), neste paradigma as aplicações comunicam-se por meio de **Tópicos** ou **Filas**. Consumidores que estejam interessados em receber atualizações de um determinado

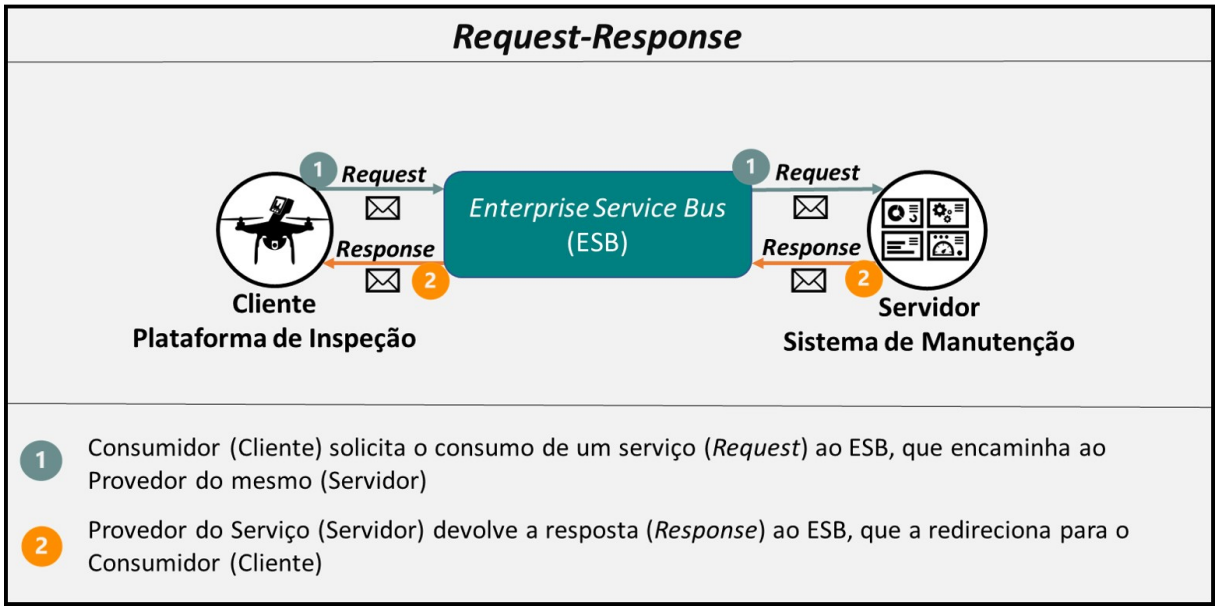



Figura 40 – Padrão de Mensagem *Request-Response* para interação entre aplicações. Fonte: autor

dado fazem uma assinatura no respectivo tópico ou fila, tornando-se um assinante (***Subscriber***). Uma ou várias aplicações podem ser provedoras de algum tipo de dado, sendo conhecidas como ***Publishers***). Assim, cada nova mensagem publicada em um tópico ou fila por um ***Publisher*** é entregue a todos os ***Subscribers*** que se inscreveram para receber tal informação. Geralmente adota-se um ***Broker***, elemento central que desacopla os elementos envolvidos na comunicação, conforme ilustrado na Figura 41. Por exemplo, quando um jornal (*Publisher*) finaliza a edição mais recente, todos os assinantes (*Subscribers*) a receberão por meio de um entregador (*Broker*). Utopicamente, o jornal é produzido mesmo que não existam assinantes e a entrega é feita de forma independente a cada um deles.

Por si só, esses estilos de interação já filtram quais tipos de protocolos são ou não apropriados para atender os casos de uso da Plataforma de Inspeção. Ou seja, usar um protocolo que suporte apenas *Publish-Subscribe* para uma situação em que claramente é necessária uma resposta não seria uma abordagem eficiente. Porém, além desse aspecto, existem ainda questões técnicas que influenciam a escolha, como consumo de bateria, complexidade, largura de banda, bibliotecas e ferramentas, etc. Dessa forma, as subseções a seguir discutem alguns dos protocolos mais adequados para atender a esses dois estilos de comunicação, ainda que alguns deles possam ser usados com ambos paradigmas. Assim, o texto foi dividido para abordar as comunicações via Serviços, que são majoritariamente no estilo *Request-Response* e via Mensageria, basicamente *Publish-Subscribe*.

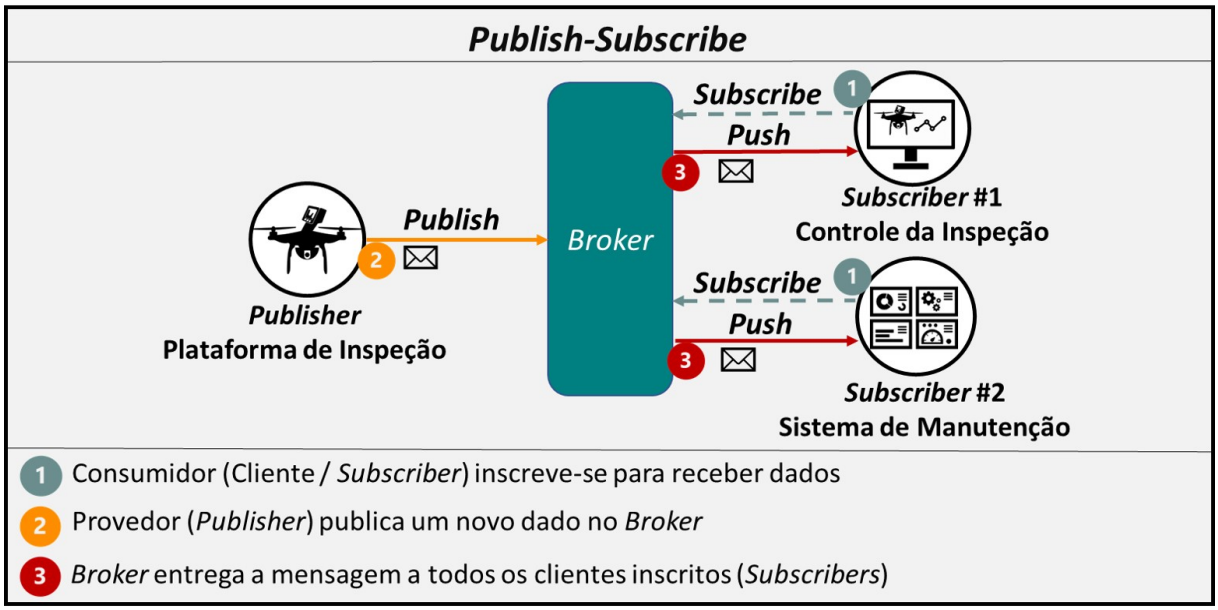



Figura 41 – Padrão de Mensagem *Publish-Subscribe* para interação entre aplicações. Fonte: autor

### 4.3.4 Comunicação no Padrão *Request-Response*

Para habilitar a comunicação entre aplicações usando os paradigmas da arquitetura SOA, abordada na subseção "**4.3.1 - Arquitetura Orientada a Serviços**", geralmente utilizam-se os ***web services***. Existem várias definições para esse termo, mas, de forma geral, ele é usado para designar serviços expostos por aplicações permitindo que elas se comuniquem de forma desacoplada em ambientes distribuídos. Nesse contexto, também é comum que os *web services*, especialmente os baseados na arquitetura *REpresentational State Transfer* (REST), sejam chamados de APIs (do inglês *Application Programming Interface*).

Diferentes tecnologias e protocolos podem ser usados para viabilizar esse tipo de comunicação distribuída. A seguir, é feita uma breve explicação das duas principais vertentes atuais no desenvolvimento de *web services*, que são as baseadas em SOAP (sem acrônimo desde 2007) e REST. Ambas foram usadas neste trabalho como uma forma de realizar interações *Request-Response* entre a Plataforma de Inspeção e sistemas existentes.

#### 4.3.4.1 *Web Services* SOAP

Os *web services* SOAP podem ser vistos como uma combinação de diferentes especificações e tecnologias que emergiram no início dos anos 2000 e que foram combinadas para habilitar a interoperabilidade entre aplicações. Curbera et al. (2002) os descrevem com base em três áreas principais: **Comunicação**, **Descrição de Serviços** e **Descoberta de Serviços**. Além delas, utiliza-se ***eXtensible***

***Markup Language*** (XML) como um idioma em comum que une essas diferentes áreas, conforme esquema da Figura 42.

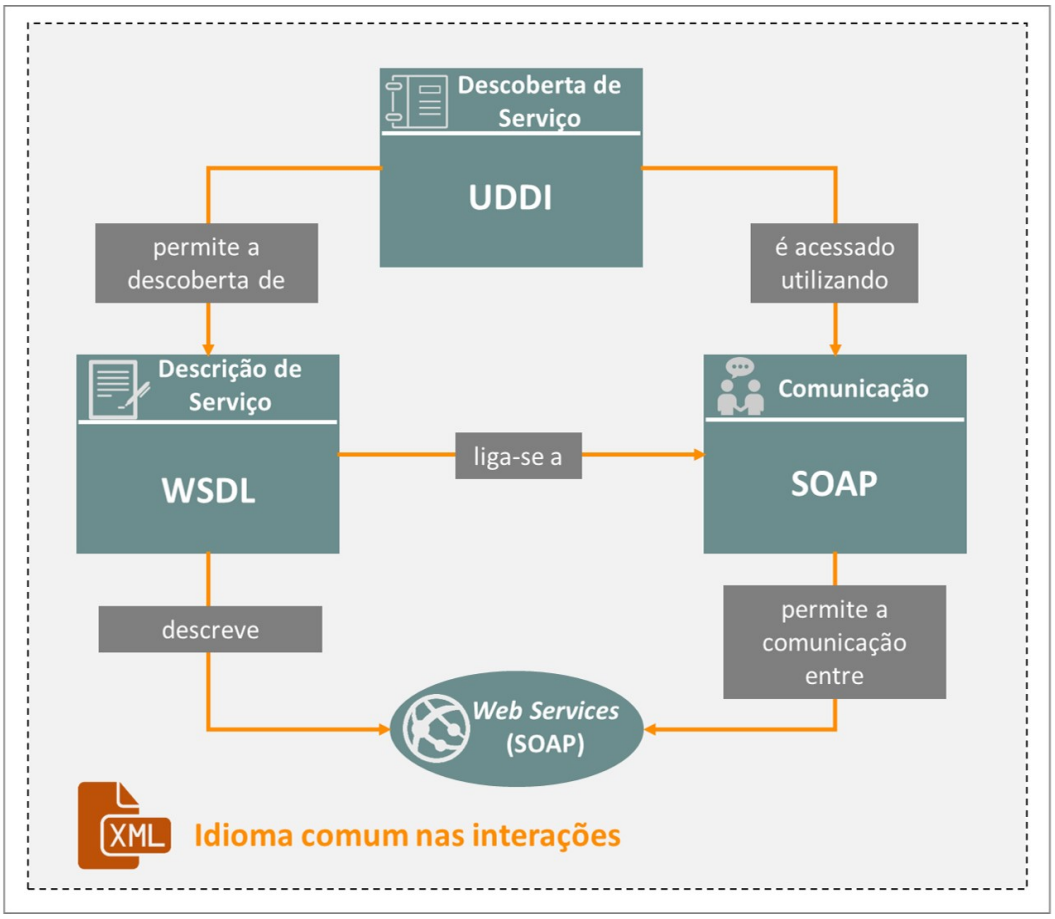



Figura 42 – Principais tecnologias e especificações que permitem a construção de *Web Services* SOAP. Fonte: autor

A primeira delas, que diz respeito à **Comunicação**, é baseada em um padrão específico, o **SOAP**. Ele foi desenvolvido em 1998 por funcionários da Microsoft como uma alternativa aos protocolos para comunicação entre aplicações que existiam à época, como o CORBA (*Common Object Request Broker Architecture*) e DCOM (*Distributed Component Object Model*), já que apresentavam problemas diversos para uso via internet. Ele foi criado buscando ser agnóstico a linguagens, suportar diferentes mecanismos de transporte, como o amplamente usado HTTP (*Hypertext Transfer Protocol*) e ser extensível, podendo acomodar melhorias em diferentes aspectos mantendo a compatibilidade. A especificação “SOAP 1.2” foi aceita pela World Wide Web Consortium (W3C) em 2003, sendo assim um padrão internacional ainda amplamente usado (SKONNARD, 2003).

Fundamentalmente, o SOAP permite a troca de informações estruturadas entre aplicações heterogêneas e distribuídas, sempre no formato XML. Este, por sua vez, é dividido em quatro seções: ***Envelope***, elemento responsável por indicar que o XML em questão é uma mensagem SOAP e agrupar as demais partes; ***Header***, que

é um cabeçalho opcional, podendo conter, por exemplo, informações de autenticação e dados para auxiliar no roteamento da mensagem; ***Body***, elemento obrigatório contendo o corpo da mensagem a ser transportada e ***Fault***, que é uma estrutura opcional para receber erros ocorridos no processamento (SKONNARD, 2003).

A segunda área, **Descrição de Serviços**, é suportada pelo ***Web Services Description Language*** (WSDL). Ele é considerado como o "contrato" que rege a forma como um serviço é provido por uma aplicação, suportando o primeiro princípio do SOA (Contrato de Serviço Padronizado). Para isso, o WSDL permite ao consumidor saber como usar o serviço, ou seja, como fazer a requisição (*Request*), determinando o protocolo de comunicação a ser usado, quais operações são suportadas, os tipos de dados da mensagem e o tipo de retorno (*Response*) que o serviço possui. Essencialmente, é um documento XML que descreve todas essas informações (GONCALVES, 2013).

Finalmente, a terceira área é a **Descoberta de Serviços**. Para isso, existe o ***Universal Description, Discovery and Integration*** (UDDI), que funciona como um catálogo dos serviços disponíveis, sendo a analogia mais próxima com o antigo catálogo de "Páginas Amarelas", que provia uma forma para que consumidores descobrissem e soubessem como contatar provedores de serviços. O UDDI é exatamente isso, um servidor central, geralmente parte do ESB, que possui um registro central dos serviços. Cada um deles é descrito no formato XML, contendo informações de categorização, dados de contato do provedor e o WSDL do mesmo. O UDDI foi desenvolvido para suportar a "Descoberta do Serviço", um dos oito princípios de SOA. Do mesmo modo que as "Páginas Amarelas", o UDDI também caiu em desuso, sendo pouco usado atualmente (GONCALVES, 2013).

Os *web services* baseados em SOAP no contexto de interações *Request-Response* ainda são amplamente utilizados, pois possuem diversas vantagens, tais como: a independência de plataforma e linguagem, o uso de protocolos de transporte que são amigáveis para uso em ambientes com *firewall*, tratamento de erros embutido no protocolo e amplo suporte de organizações, o que se reflete num grande número de bibliotecas para sua utilização em ESBs e ferramentas de programação. Em contrapartida, por serem extremamente baseados em XML, os *web services* SOAP tendem a ser mais lentos, pois a criação e interpretação do XML é mais pesada por conta do *overhead* que essa linguagem possui (DOGLIO, 2015; MUELLER, 2013). Na prática, isso se reflete em problemas de performance em cenários com grande volume de requisições, tornando o SOAP menos favorável para uso em aplicações móveis ou em dispositivos com limitações de processamento.

Dessa forma, a alternativa mais comum para esses cenários e que vem ganhando cada vez mais espaço é o uso de um *web service* / API REST, que tende a

ser mais leve, simples e rápida, conforme discutido na subseção a seguir.

#### 4.3.4.2 APIs REST

É importante destacar que o REST não é um protocolo, tecnologia ou mesmo um padrão. A ideia que motivou a sua criação, proposta em 2000 por Roy Fielding em sua tese de doutorado, é a definição de um estilo arquitetural para a interação entre aplicações em ambientes distribuídos. Na prática, o que Fielding (2000) propôs são **restrições** (ou *constraints*) que, quando seguidas, trazem escalabilidade, simplicidade, performance e reuso. Tais restrições não serão detalhadas neste trabalho, pois fogem do objetivo de explicar o funcionamento geral. Se for interesse do leitor, recomenda-se a leitura do Capítulo 5 da tese de Fielding (2000). Como curiosidade, conforme explica Pinkham (2016), o termo **RESTful** API (totalmente REST em tradução livre) só deveria ser aplicado quando todas as restrições propostas por Roy são seguidas na construção da mesma.

Conforme Doglio (2015) explica, quando se avalia o acrônimo REST, ou seja, "Transferência de Representação de Estado" em tradução livre, é possível inferir o que o REST essencialmente é: a **transferência** do estado de algum **recurso** usando uma **representação**. Essas três partes são explicadas nos próximos parágrafos e representadas na Figura 43, que ilustra o consumo de uma API REST para a consulta dos valores de uma *tag* disponível no Sistema Historiador, mais conhecido como *Plant Information Management System* (PIMS).

A **Transferência** diz respeito à interação entre as partes, ou seja, à comunicação em si. Na prática, interações no estilo REST são baseadas nos protocolos HTTP/HTTPS, o que leva ao uso dos "verbos" definidos por eles para realizar a requisição (*Request*) e os códigos de retorno para indicar o status da resposta (*Response*). A Tabela 5 apresenta, de forma não exaustiva, os principais verbos e exemplos de códigos de retorno possíveis nestas situações. A lista completa desses códigos de retorno está disponível em Internet Engineering Task Force (2017).

Os **Recursos** (e seu estado) podem ser virtualmente qualquer coisa: uma página WEB, os dados de um cliente em uma aplicação, uma imagem, etc. Dessa forma, é importante que seja possível identificá-lo de forma unívoca, usando para isso a *Unique Resource Identifier* (URI). Ela contém todo o caminho que permite que um recurso seja localizado, ou seja, não se trata apenas do "código" do recurso, mas de seu endereço completo. Para ilustrar, no caso da Figura 43, a ideia é recuperar os valores armazenados em uma *tag* (Recurso) no PIMS (Servidor). Visto que a *tag* buscada é a "TR1305", localizada no PIMS do Porto, foi usada a URI "/pims/porto/tags/tr1305/". Não há uma regra absoluta na definição da URI, mas, de

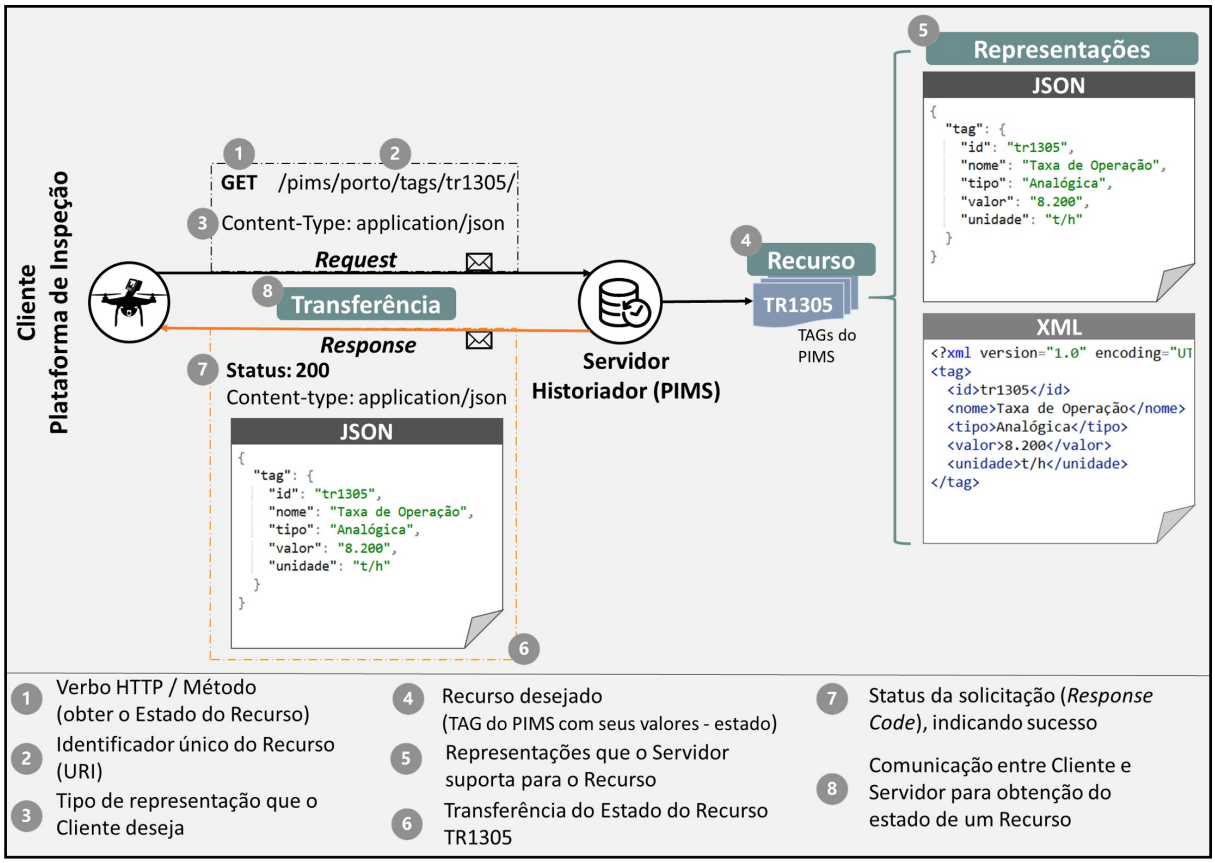



Figura 43 – Esquema de interação entre duas aplicações usando o estilo arquitetural REST. Fonte: autor

Tabela 5 – Exemplos de verbos (*Request*) e códigos de status de retorno (*Response*)

| Verbo HTTP | Ação Geralmente Associada | Status de Retorno (Exemplos) |
|---|---|---|
| GET | Acessar ou obter um Recurso (somente leitura) | 200 (OK)<br>404 (Não localizado) |
| POST | Criar um novo Recurso | 201 (Criado)<br>409 (Conflito - Recurso já existente) |
| PUT | Atualizar ou substituir Recurso existente | 200 (OK)<br>204 (Sem conteúdo - sucesso, mas sem mensagem de retorno) |
| DELETE | Apagar recurso existente | 200 (OK)<br>204 (Sem conteúdo - sucesso, mas sem mensagem de retorno)<br>401 (Não autorizado)<br>405 (Método não permitido) |

Fonte: adaptado de DOGLIO, 2015 e NEWMARCH, 2017

modo geral, ela deve ser de simples entendimento para facilitar o uso pelo Cliente, facilitando a descoberta (DOGLIO, 2015).

Por sua vez, a **Representação** é a abstração utilizada para retratar o recurso em questão, por exemplo, um arquivo em XML ou em *JavaScript Object Notation*

(JSON), uma página HTML, etc. Neste caso, um mesmo recurso pode ter uma ou múltiplas representações. Essa situação também é ilustrada na Figura 43, onde a *tag* TR1305 é representada tanto em JSON quanto em XML, com o estado do recurso, ou seja, seus dados, sendo equivalentes. É importante notar que o Consumidor especifica na chamada qual o formato que ele prefere por meio do campo *Content-Type*, o que também é conhecido como negociação de conteúdo (*Content Negotiation*) (NEWMARCH, 2017).

Com base nessas características, percebe-se que APIs baseadas em REST têm uma filosofia mais simples, resultando em leveza, performance e eficiência. Mumbaikar e Padiya (2013) comparou *web services* SOAP e APIs REST para um mesmo caso de uso, mostrando que mensagens em REST eram entre 9 e 10 vezes menores e o tempo de processamento em SOAP entre 5 e 6 vezes maior. Isso torna APIs REST apropriadas para interações em aplicações WEB ou em dispositivos móveis, geralmente mais restritos. Em alguns casos, apenas com uma URL, é possível enviar uma requisição ao Servidor com todos os parâmetros necessários, sem precisar criar e enviar um XML ou outro tipo de arquivo no corpo da mensagem, algo impensável com SOAP, mais rígido e formal (NEWMARCH, 2017).

Em contrapartida, o REST parte do princípio de comunicação Cliente-Servidor, o que o torna menos apropriado para ambientes distribuídos, mesmo que ainda seja possível o uso com um ESB (DOGLIO, 2015). Além disso, o fato do REST não guardar o estado, ou seja, cada interação ser única e não possuir contexto, dificulta casos de uso em que há a necessidade de combinar vários serviços para suportar um processo de negócio mais complexo (PINKHAM, 2016).

Assim, é possível perceber que *web services* SOAP e APIs REST não são necessariamente comparáveis, muito menos concorrentes: SOAP é um protocolo padronizado, enquanto REST é um estilo arquitetural que ganha cada vez mais espaço (MUELLER, 2013). Porém, o uso combinado de ambos é interessante, pois conseguem se complementar. Ambos são apropriados para interações no estilo *Request-Response*, mas, geralmente, utiliza-se o SOAP para suportar serviços corporativos robustos (financeiro, manutenção, etc) em um ESB. Já o REST pode ser usado para expor tais serviços para clientes externos, por exemplo, para uma aplicação móvel para consultar ou criar uma ordem de manutenção, sem necessariamente ter que conhecer todo o contrato do serviço existente (MUEHLEN; NICKERSON; SWENSON, 2005).

Um exemplo prático dessa complementariedade é apresentado na subseção "**5.4.2 - Protótipo 2 - Envio de defeitos da UPE para o Sistema de Manutenção**", onde tanto *web Services* SOAP quanto APIs REST foram usados para suportar as interações *Request-Response* da Plataforma de Inspeção com sistemas corporati-

vos. Já na próxima seção é discutido o uso dos protocolos para suportar as interações *Publish-Subscribe*, que possuem outro grupo de características e, portanto, demandam outros tipos de tecnologias.

### 4.3.5 Comunicação no Padrão *Publish-Subscribe*

O conceito de Internet das Coisas (IoT, do inglês *Internet of Things*) já é uma realidade há alguns anos e, essencialmente, permite que diferentes tipos de dispositivos conectem-se a redes de comunicação, como a internet, para habilitar novas aplicações e expandir possibilidades. Além disso, alguns dispositivos podem comunicar-se entre si, diretamente ou por meio de algum intermediário, o que é conhecido como M2M (*Machine-to-Machine*) (WU et al., 2011).

Tais dispositivos possuem características únicas, como escassez de processamento e memória, pouca largura de banda para envio e recebimento de dados e autonomia limitada, já que, geralmente, são alimentados por bateria (THANGAVEL et al., 2014). Com a popularização destes dispositivos e diante de tais restrições, foi necessário o desenvolvimento de novos tipos de protocolos de comunicação, pensados desde sua concepção para lidar com essa nova realidade. Na sua maioria, esses protocolos trabalham com o paradigma *Publish-Subscribe*, adequado ao caso de uso geralmente associado a estes dispositivos, que é a disponibilização de dados para múltiplos consumidores.

Ao avaliar a Plataforma de Inspeção, proposta neste trabalho, os dados classificados na subseção anterior como de **"Telemetria e Sensores"**, que são enviados de forma contínua e em pequenas quantidades, podem ser vistos como um caso de IoT e se enquadram no estilo de interação *Publish-Subscribe*. Ainda que existam inúmeros protocolos que poderiam ser avaliados por suportarem o estilo acima, como o *Simple/Streaming Text Orientated Messaging Protocol* (STOMP) (STOMP, 2012), a extensão IoT do *Extensible Messaging and Presence Protocol - Internet of Things* (XMPP-IoT) (LINDBORG, 2017), *Java Message Service* (JMS) (ORACLE, 2013), para citar alguns, a avaliação foi restrita aos três protocolos mais utilizados e referenciados na literatura. Dessa forma, as próximas subseções apresentam alguns dos protocolos criados ou adaptados para IoT e M2M, delineando seu histórico e características mais relevantes. Finalmente, é feita uma discussão comparativa para justificar o protocolo escolhido a ser usado na Plataforma de Inspeção.

#### 4.3.5.1 MQTT

O protocolo MQTT (*MQ Telemetry Transport*), foi criado em 1999 por Andy Stanford-Clark e Arlen Nipper com o objetivo inicial de prover o menor consumo de bateria

e consumo de banda em dispositivos de monitoramento de tubulações de óleo que usavam conexões via satélite. Atualmente, ele é padronizado pela "*OASIS Standard*" e encontra-se na versão 3.1.1 (BANKS; GUPTA, 2014).

O MQTT funciona a partir do padrão de integração conhecido como *publish-subscribe*. Conforme explicado por Hohpe e Woolf (2012), neste padrão as aplicações comunicam-se por meio de **Tópicos**, local onde o gerador da informação publica uma mensagem (***Publisher***) e consumidores interessados na mesma assinam, tornando-se inscritos (***Subscribers***). Cada nova mensagem publicada em um **Tópico** por um ***Publisher*** é entregue a todos ***Subscribers*** por um ***Broker***, o que garante uma comunicação totalmente desacoplada entre as partes. Um exemplo de como ele funciona é apresentado na Figura 44.

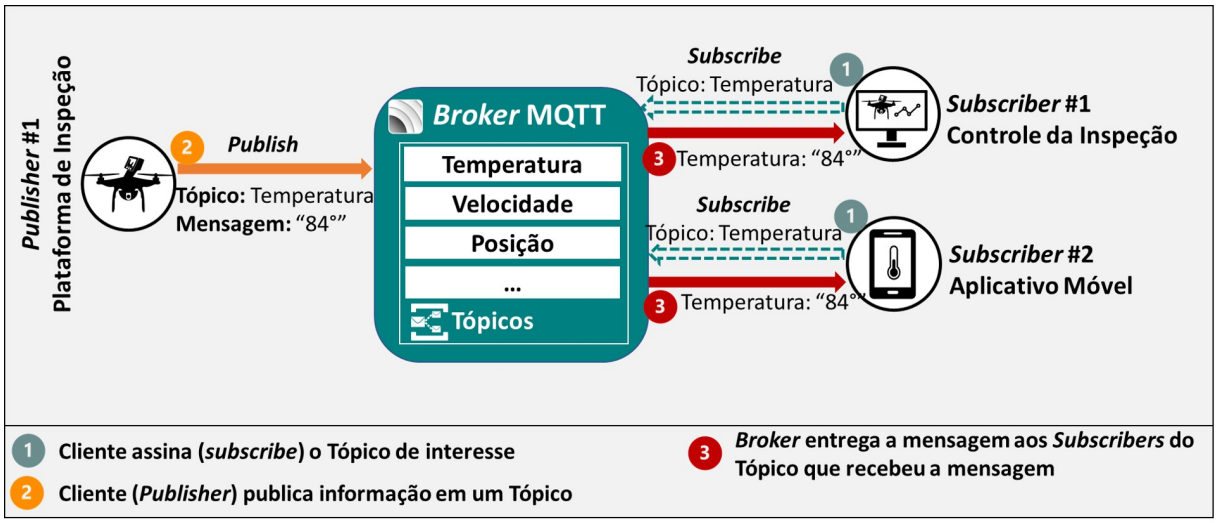



Figura 44 – Exemplo dos agentes envolvidos e funcionamento do padrão *Publish-Subscribe* com MQTT. Fonte: autor

Neste caso, os agentes envolvidos são os **Clientes** (*publisher* ou *subscriber*) e o ***Broker***. Um cliente pode ser qualquer tipo de dispositivo, desde um microcontrolador até um servidor, rodando a biblioteca do MQTT e que esteja conectado a um *Broker*. Este, por sua vez, é o coração de qualquer implementação do padrão *publish-subscribe*, podendo se apresentar como uma das funcionalidades presentes em um ESB, ou ser um barramento de mensagens totalmente separado. Independente da forma, ele é responsável por receber as mensagens, decidir quem deve recebê-las por meio do controle das subscrições, e realizar a entrega. Por fim, ele também é responsável pela autorização e autenticação dos clientes.

O MQTT trabalha de forma orientada à conexão, ou seja, ele exige que os Clientes e o *Broker* estabeleçam uma conexão usando o protocolo TCP (*Transmission Control Protocol*) para se comunicarem. Além disso, opcionalmente, esse canal pode ser estabelecido de forma segura usando os protocolos TLS (*Transport Layer Security*) e SSL (*Secures Socket Layer*), porém, a autenticação é simples,

apenas com usuário e senha. Ainda sobre conexões, o protocolo suporta um modo "durável", em que as mensagens, que podem ter tamanho máximo de 256 MB, são guardadas para posterior entrega a clientes, mesmo que eles, eventualmente, desconectem-se do *Broker*, o que diminui as chances de perdas de mensagens (NAIK, 2017).

Outra funcionalidade, que é uma das mais interessantes do MQTT, são os diferentes níveis de qualidade de serviço (QoS, do inglês *Quality of Service*). Por meio deles, é possível definir diferentes tipos de confiabilidade na entrega das mensagens, o que é extremamente útil no contexto de IoT e M2M, visto que os dispositivos nem sempre contam com *links* de comunicação estáveis e confiáveis. Assim, diferentes níveis podem ser adotados para balancear velocidade e confiabilidade (LAMPKIN et al., 2012):

- **QoS 0:** a mensagem é entregue **no máximo uma vez**, ou seja, o Cliente receptor da mensagem não avisa ao *Broker* que ela foi recebida. Por conta disso, é o nível mais rápido e indicado quando velocidade é mais importante que confiabilidade;
- **QoS 1:** a mensagem é entregue **pelo menos uma vez**, mas pode ser recebida mais de uma vez. Com isso, ele consegue ser geralmente rápido e confiável, mas exige que o Cliente tenha mecanismos de controle de duplicidade para os casos em que a mensagem é entregue mais de uma vez;
- **QoS 2:** a mensagem é entregue **exatamente uma vez**, o que significa que este é o nível com maior confiabilidade. Por outro lado, os mecanismos para garantir a entrega de forma unívoca também o fazem ser o mais lento.

Por fim, é fato que o MQTT é um dos protocolos mais populares para os casos de uso de IoT. Algumas das explicações para isso são o fato de ele ser aberto, livre de *royalties*, de fácil utilização (poucos métodos permitem realizar todas as funcionalidades descritas) e contar com bibliotecas para as linguagens de programação mais populares, além de amplo suporte de grandes organizações, como Facebook, Amazon, IBM e Cisco (LAMPKIN et al., 2012).

#### 4.3.5.2 AMQP

O *Advanced Messaging Queuing Protocol* (AMQP) foi desenvolvido em 2003 no banco JPMorgan Chase com o objetivo de melhorar a interoperabilidade, segurança e confiabilidade na troca de dados entre aplicações usadas na instituição. Até então, a comunicação via mensagens era baseada em ferramentas e *brokers* proprietários,

cada qual com suas próprias implementações de protocolos. Essa abordagem aumentava a dependência de fornecedores específicos e limitava a comunicação com parceiros (VINOSKI, 2006). Em 2008, um grupo de trabalho formado por diversos bancos e empresas da área de tecnologia definiu a primeira versão de uma especificação aberta e livre para o AMQP, conhecida como “AMQP 1.0”. Posteriormente, em 2012, ele foi aceito como um padrão pela OASIS Standard e, desde 2014, o AMQP 1.0 também é padronizado na forma da ISO/IEC 19464:2014.

Para atingir os requisitos de interoperabilidade, o AMQP trabalha com diferentes elementos que desacoplam a comunicação entre Clientes (*Publisher* e *Subscriber*). A estrutura interna do *Broker* é composta por dois elementos principais. O primeiro deles é conhecido como ***Exchange***, que recebe as mensagens enviadas pelos clientes do tipo *Publisher* e as redistribui internamente para o segundo elemento, chamado de Fila (***Queue***). Estas são responsáveis por armazenar e realizar a entrega das mensagens para os clientes do tipo *Subscriber*. O cabeçalho de cada mensagem é publicado com uma chave de roteamento, que é usada pelas tabelas de ***Binding*** para rotear as mensagens de um *Exchange* para uma ou várias filas (*Queues*) (CARMO, 2012). A Figura 45 apresenta um esquema geral de uma comunicação usando o AMQP, demonstrando como esses elementos interagem.

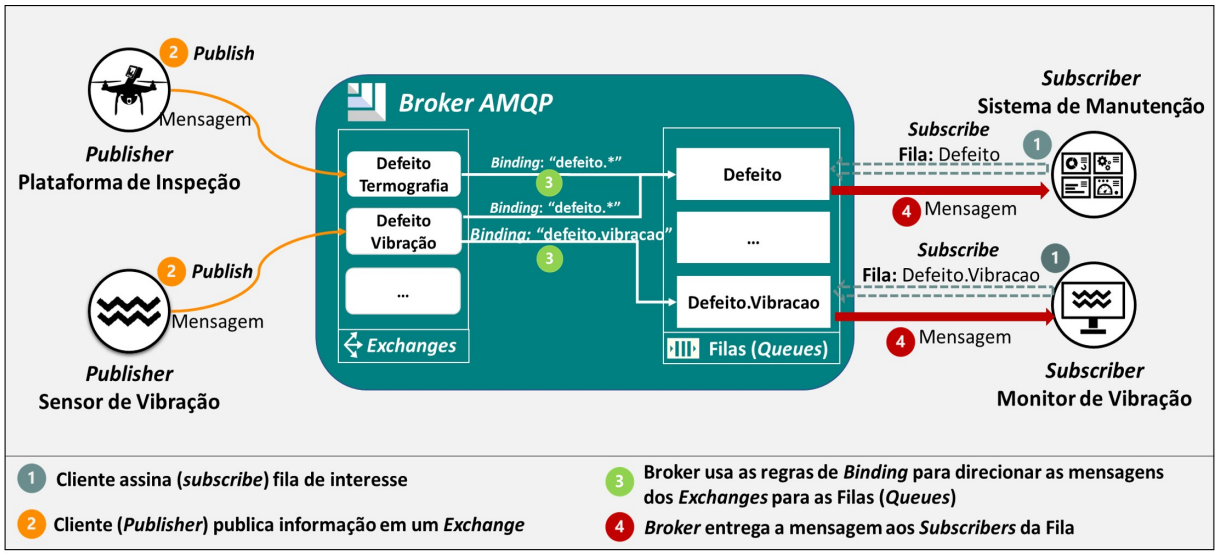



Figura 45 – Exemplo de um processo de integração com os principais elementos envolvidos no protocolo AMQP. Fonte: autor

Apesar de atualmente ser amplamente usado no contexto de IoT e M2M, o AMQP foi criado para suportar troca de mensagens em aplicações bancárias, o que explica algumas de suas características relacionadas à robustez. A primeira delas refere-se à forma de uso da fila, que pode ser compartilhada entre múltiplos consumidores (pública) ou exclusiva de apenas um deles (privada). A segunda é o fato das filas poderem ser criadas de forma persistente, ou seja, são automaticamente recriadas se o *Broker* for reiniciado. A terceira e última são os tipos de

entrega suportados. Apesar de oferecer um modo sem garantias, focado em velocidade (*Unsettle Format*), suporta também a entrega garantida (*Settle Format*), o que assegura o recebimento, mas delega ao consumidor tratar eventuais duplicidades (NAIK, 2017).

Em relação ao transporte, o AMQP utiliza o protocolo TCP, ou seja, é orientado a conexão. Esse canal pode ser protegido usando SSL/TLS ou *Internet Protocol Security Protocol* (IPSec) e a autenticação dos Clientes com o *Broker* pode ser feita via *Simple Authentication and Security Layer* (SASL) (NAIK, 2017). Além de sua robustez, outra vantagem do protocolo é que o tamanho das mensagens pode ser negociado, com limite de 4 GB, aumentando sua flexibilidade (LUZURIAGA et al., 2015).

#### 4.3.5.3 CoAP

O *Constrained Application Protocol* (CoAP) foi proposto em 2010 por um grupo de trabalho do *Internet Engineering Task Force* (IETF) com objetivo de habilitar a comunicação via HTTP para dispositivos restritos em termos de processamento, bateria e rede para transmissão de dados. Ele é o mais recente dentre os protocolos M2M avaliados e, por conta disso, incorpora diversas das qualidades e características dos protocolos anteriores, baseando-se no estilo arquitetural REST e buscando promover uma ponte entre o mundo de IoT e a *web* tradicional (BORMANN; CASTELLANI; SHELBY, 2012).

Para que isso seja possível, o CoAP tem princípios similares ao HTTP, mas os aplica de uma forma mais leve e adequada a dispositivos restritos. Dessa forma, ele também utiliza verbos (*POST*, *GET*, *PUT*, etc.) para solicitar ações específicas e uma variação dos códigos de retorno para representar as respostas (2.00 - *OK*, 4.04 para *Not Found*, etc.), conforme já abordado na subseção "**4.3.4.2 - APIs REST**". Alguns dispositivos, conhecidos como *proxies*, tem a função de garantir a interoperabilidade entre HTTP e o CoAP, ao traduzir as requisições de um formato para o outro. Todo esse processo é melhor representado na Figura 46, onde é possível ver as formas de comunicação entre Internet e ambientes restritos, além da pilha dos protocolos HTTP e CoAP (BORMANN; CASTELLANI; SHELBY, 2012).

Além do aspecto de interoperabilidade, a Figura 46 ainda apresenta outra informação relevante sobre o modo de funcionamento do CoAP, que é o uso do *User Datagram Protocol* (UDP) como protocolo de transporte. Dessa forma, ele não é orientado à conexão e, assim, implementa mecanismos de confiabilidade, em que as mensagens podem ser enviadas como "confirmáveis" ou "não-confirmáveis". No primeiro caso, enquanto o dispositivo que enviou a mensagem não receber a confir-

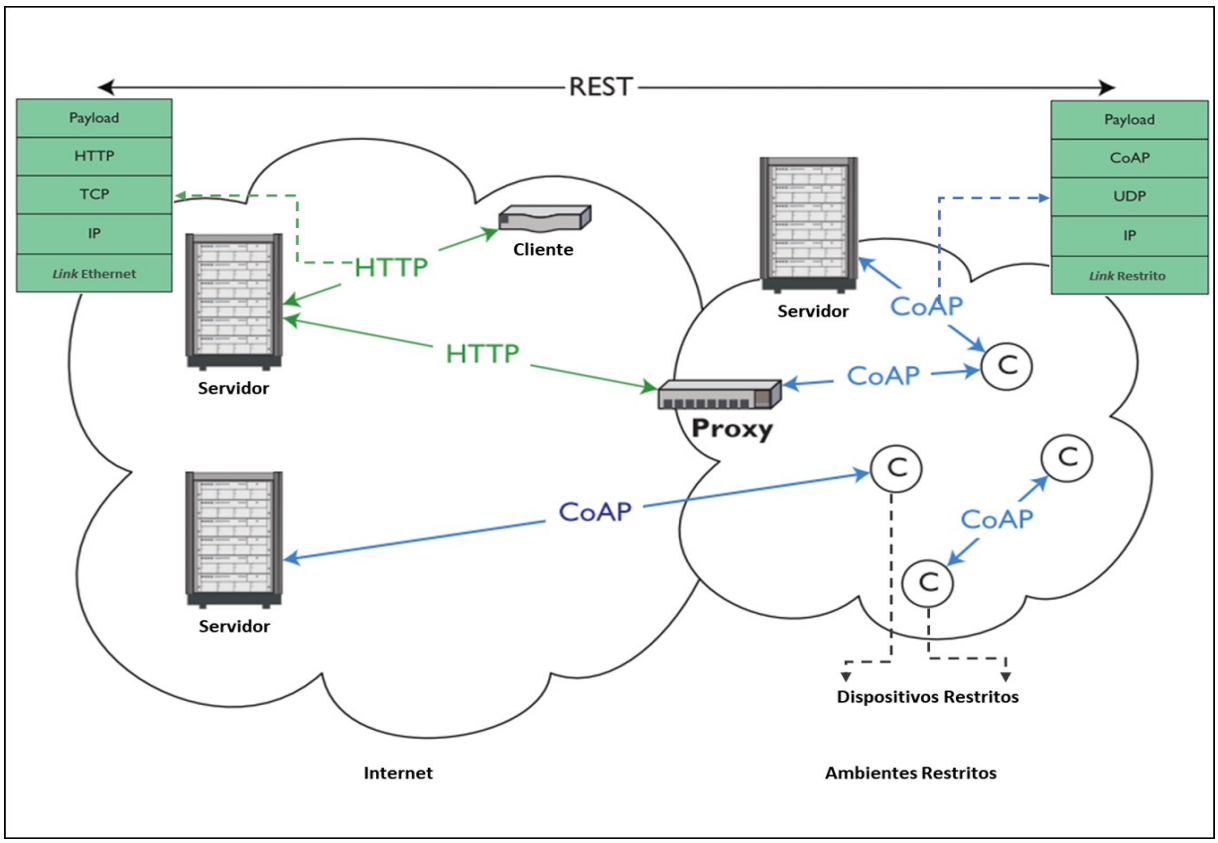



Figura 46 – Interoperabilidade entre Internet e Ambientes Restritos com o protocolo CoAP. Fonte: adaptado de BORMANN; CASTELLANI; SHELBY, 2012

mação de recebimento (*acknowledgment*) pelo destinatário, ele faz retransmissões periódicas da mensagem. Já no segundo caso, a mensagem é enviada apenas uma vez e sem nenhum tipo de confirmação, o que é mais rápido, mas sem confiabilidade (NAIK, 2017; BORMANN; CASTELLANI; SHELBY, 2012).

Tais características, que são próprias do estilo de arquitetura REST, baseado em **URIs** e **Recursos**, são inerentes ao padrão de mensagens *Request-Response*, com as comunicações iniciadas sempre de um cliente para um servidor. Porém, o CoAP possui artifícios para suportar o estilo *Publish-Subscribe*, utilizando para isso a **Observação de Recursos** (ou "*Observe*"). Nesse caso, o cliente faz uma requisição (*GET*) à URI de um recurso, indicando que deseja fazer a observação dele. A partir deste momento, o servidor envia para o cliente qualquer atualização ocorrida no recurso observado, sem ação do cliente, vide esquema na Figura 47. De forma comparativa, as **URIs** funcionam como os **Tópicos** do MQTT ou as **Filas** do AMQP. Como uma vantagem em relação a esses protocolos, o CoAP oferece negociação de conteúdo, permitindo que o cliente e o servidor negociem qual a representação do recurso que é mais adequada (XML, JSON, etc) (THANGAVEL et al., 2014).

Por fim, é importante ainda destacar duas características relevantes do CoAP: a descoberta de recursos, que é inspirada e similar à encontrada no HTTP, e o uso

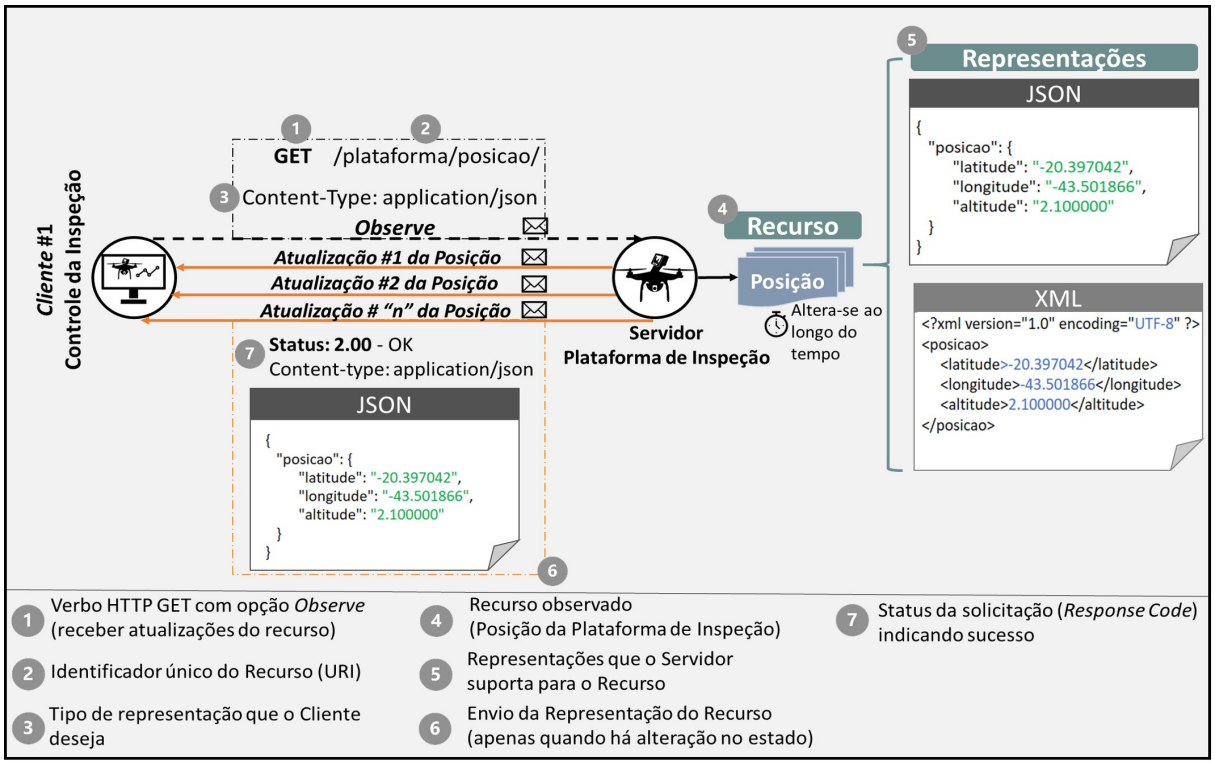



Figura 47 – Funcionamento do CoAP emulando o *Publish-Subscribe* com negociação de conteúdo. Fonte: autor

de DTLS (do inglês *Datagram Transport Layer Security*), que adiciona uma camada de segurança ao canal por onde as mensagens de até 64 KB por pacote são transportadas. Por ser relativamente recente, o CoAP ainda vem recebendo recursos adicionais, o que o posiciona como uma alternativa viável para habilitar a comunicação de dispositivos IoT com APIs e serviços já existentes usando a arquitetura REST (NAIK, 2017).

#### 4.3.5.4 Discussão Sobre Protocolos de Mensageria

Com base nas características apresentadas até então, percebe-se que os três protocolos são adequados para suportar as interações *Publish-Subscribe* da Plataforma de Inspeção com sistemas corporativos. No entanto, cada um deles foi desenvolvido e evoluiu com um propósito específico, o que diz muito de suas características fundamentais.

O MQTT foi criado para envio de baixas quantidades de dados em redes não confiáveis, o que se materializa no seu recurso de QoS com três níveis, incluindo um que garante a entrega da mensagem exatamente uma vez. O AMQP, por sua vez, foi desenvolvido para troca de dados entre instituições bancárias e, assim, tem em seu cerne questões de segurança muito bem desenvolvidas. Já o CoAP nasceu com a interoperabilidade entre internet e dispositivos restritos em mente, sendo o mais simples para habilitar comunicações no estilo REST de ponta-a-ponta. Essas são as

Tabela 6 – Resumo das principais características dos protocolos avaliados

| **Característica** | **MQTT** | **AMQP** | **CoAP** |
|---|---|---|---|
| Ano Criação | 1999 | 2003 | 2010 |
| Arquitetura de Comunicação | Cliente/*Broker* | Cliente/*Broker*<br>Cliente/Servidor | Cliente/*Broker*<br>Cliente/Servidor |
| Padrões de Mensagem | *Publish/Subscribe* | *Publish/Subscribe*<br>*Request/Response* | *Request/Response*<br>*Publish/Subscribe*<br>(opção *observe*) |
| Tamanho da Mensagem | Médio<br>(até 256 MB) | Negociável<br>(até 4 GB) | Pequeno<br>(até 64 KB) |
| Tamanho do Cabeçalho | 2 bytes | 8 bytes | 4 bytes |
| Orientado a Conexão? | Sim<br>(TCP) | Sim<br>(TCP) | Não<br>(UDP) |
| Segurança | TLS/SSL | TLS/SSL<br>IPSec<br>SASL | DTLS |
| Confiabilidade | Três níveis:<br>0: No máximo uma vez<br>1: Pelo menos uma vez<br>2: Exatamente uma vez | Dois níveis:<br>*Settle*: No máximo uma vez<br>*Unsettle*: Pelo menos uma vez | Dois níveis:<br>Não-Confirmável:<br>No máximo uma vez<br>Confirmável:<br>Pelo menos uma vez |
| Padronização | OASIS | OASIS<br>ISO/IEC | IETF |
| Principais Apoiadores | Facebook, IBM, Cisco, Red Hat, Amazon, dentre outros | Microsoft, JPMorgan, Bank of America, Goldman Sachs, dentre outros | Comunidade, Cisco, Contiki, Erika, IoTivity, dentre outros |

Fonte: adaptado de NAIK, 2017, LUZURIAGA et al., 2015 e CARO et al., 2013

características mais relevantes de cada um desses protocolos, mas ainda existem inúmeros aspectos que precisam ser avaliados, conforme resumo apresentado na Tabela 6.

Os itens resumidos na Tabela 6 apresentam informações relevantes que dão indícios sobre o funcionamento e a aplicabilidade de cada um dos protocolos. Porém, apenas com base neles, não é possível entender o comportamento em situações reais. Por conta disso, alguns autores realizaram trabalhos comparando alguns desses protocolos em diferentes condições, buscando avaliar aspectos como robustez, consumo de banda, bateria, etc. Ainda que não tenham sido encontrados resultados experimentais comparando diretamente os três protocolos, existem trabalhos que comparam diretamente dois deles, como AMQP x MQTT (LUZURIAGA et al.,

2015) e CoAP x MQTT (CARO et al., 2013), o que permite aprofundar a análise, ainda que separadamente.

No trabalho de Luzuriaga et al. (2015), que comparou o MQTT e o AMQP para uso em redes veiculares, os autores concluem que o MQTT é mais apropriado para cenários de uso em que a rede não é confiável, já que este tem menor atraso na entrega dos pacotes (*jitter*), ainda que com pouca diferença em relação ao AMQP. Além disso, ele consegue garantir a ordem de entrega das mensagens aos consumidores mesmo quando há oscilações ou picos no envio. Este comportamento, especialmente em condições adversas de rede, é extremamente desejável, pois reduz o esforço de desenvolvimento de aplicações que utilizam o MQTT.

Em uma comparação entre os protocolos CoAP e MQTT para sensoriamento de multidões usando *smartphones* (*crowdsensing*), Caro et al. (2013) demonstram que mensagens com ou sem garantia de entrega geram menos tráfego com o protocolo CoAP. A maior parte da diferença é explicada pelos protocolos de transporte utilizados, que é mais complexo no caso do MQTT, que utiliza TCP, e simples no CoAP, que adota UDP. Outro aspecto avaliado, o tempo de tráfego de mensagens confirmáveis (CoAP) ou com QoS 1 (MQTT), demonstrou que o CoAP consegue ser até 20% mais eficiente que o MQTT por conta do tamanho de pacote menor. Por outro lado, mesmo com o recurso de mensagens confirmáveis do CoAP, não foi possível garantir a entrega de todos os pacotes quando enviados em pequenos intervalos. Neste cenário, o MQTT conseguiu atingir taxas entre 86% e 100% de entrega, enquanto o CoAP variou entre 65% e 96% no melhor cenário.

Tais resultados destacam dois aspectos importantes: a leveza do CoAP e a confiabilidade do MQTT. O CoAP não precisa realizar conexões para transmitir dados e depende puramente da pilha de protocolos do HTTP. Dessa forma, depende de poucos recursos do cliente, consumindo menos bateria que o MQTT para transmitir no mesmo nível teórico de confiabilidade (CARO et al., 2013). Porém, na prática, o maior nível de confiabilidade do CoAP não consegue garantir taxas de entrega similares à do MQTT em condições de rede equivalentes, que é o segundo aspecto relevante desta análise. Além disso, o MQTT possui um nível adicional de QoS, que garante a entrega de mensagens sem duplicidade, o que, novamente, simplifica o desenvolvimento de aplicações (LAMPKIN et al., 2012).

Em resumo, os três protocolos avaliados são apropriados para IoT e M2M e vem ganhando cada vez mais espaço por serem capazes de suportar diferentes casos de uso (PERTEL et al., 2017). Especificamente para suportar as interações *Publish-Subscribe* da Plataforma de Inspeção com sistemas corporativos, o protocolo MQTT foi selecionado por apresentar um conjunto de características mais favorável:

- É leve e consegue garantir a entrega de mensagens em redes não-confiáveis sem gerar excesso de tráfego;
- Possui recursos que facilitam o desenvolvimento de aplicações, como a garantia de ordem de mensagens e a entrega sem duplicidades;
- Está há mais tempo no mercado, sendo suportado por diversos *brokers*;
- Possui bibliotecas disponíveis em várias linguagens de programação, que o habilita a funcionar tanto em dispositivos embarcados extremamente simples quanto em máquinas robustas;
- Conforme gráfico na Figura 48, é o que apresentou maior interesse de pesquisas no Google nos últimos 3 anos. Apesar de não ser conclusivo, pode significar maior adoção dentre os protocolos avaliados.

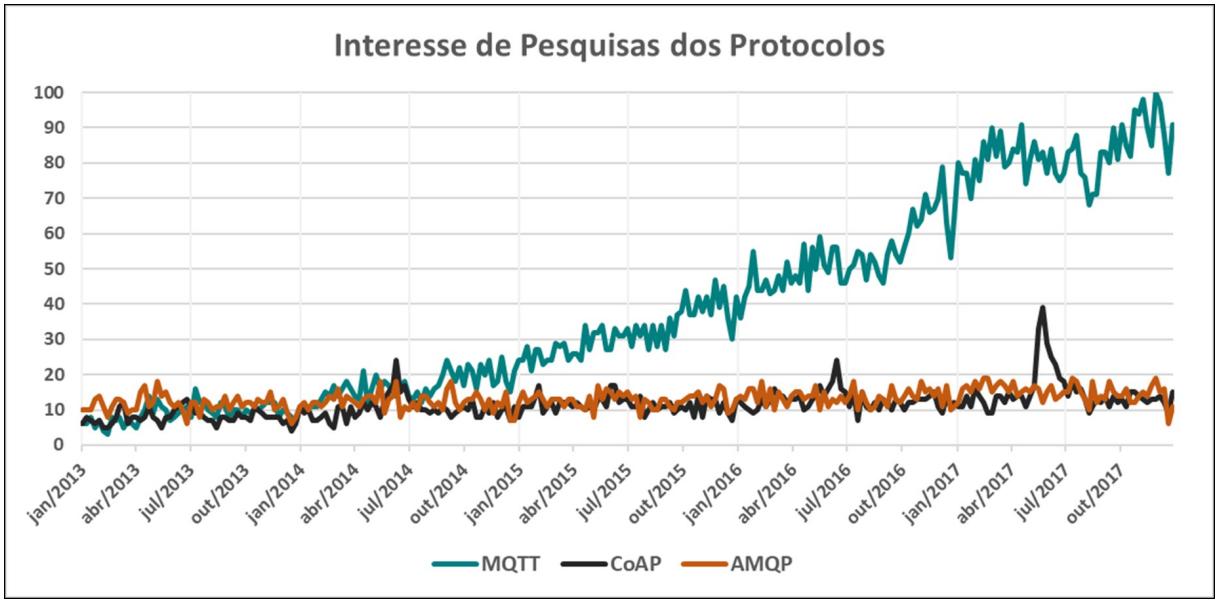



Figura 48 – Interesse de pesquisa no Google dos protocolos avaliados entre janeiro de 2013 e dezembro de 2017. Fonte: adaptado a partir de dados de Google Trends, 2018

## 4.4 Ferramental para Protótipos e Simulação

A presente seção apresenta a base teórica das principais ferramentas utilizadas para a construção de protótipos para validação da Arquitetura Integrada e para a simulação de portadores do tipo VANT. Para tanto, ela foi organizada da seguinte forma:

- A subseção “**4.4.1 - Desenvolvimento para VANTs**” aborda o desenvolvimento de aplicações, tanto regulares quanto embarcadas, para VANTs da fabricante DJI, utilizada no presente projeto;
- Já a subseção “**4.4.2 - Simulação**” discute a importância do uso da simulação no contexto de VANTs e apresenta dois simuladores utilizados no presente projeto para diferentes propósitos.

### 4.4.1 Desenvolvimento para VANTs

Os VANTs são um dos portadores possíveis para a Plataforma de Inspeção, conforme melhor discutido na subseção “**5.2.1 - Portador**”. Dessa forma, é necessário o desenvolvimento de aplicações e ferramentas que permitam o uso deste portador para a inspeção de rolos de transportadores de correia. Enquanto é possível realizar a construção totalmente do início, sem nenhum tipo de acelerador, os kits de desenvolvimento, ou SDK (do inglês *Software Development Kit*), permitem que os fabricantes exponham funcionalidades do seu produto em uma camada acessível aos desenvolvedores, de forma segura e controlada, de modo que eles criem de forma mais rápida aplicações baseadas no produto, expandindo seu uso.

No presente trabalho foram adotados como portadores da Plataforma de Inspeção VANTs comerciais da fabricante DJI. Ela disponibiliza dois tipos de SDKs para o desenvolvimento de aplicações para seus produtos, cada uma com objetivos específicos:

- ***DJI Mobile SDK***: é compatível com toda a linha de produtos da fabricante e habilita a criação de aplicações que rodam em dispositivos com sistema operacional Android ou iOS. Diferentes tipos de aplicativos podem ser desenvolvidos, com interação com o VANT em si, alguns subsistemas e acessórios, como os diferentes modelos de câmeras que alguns VANTs suportam. Alguns sensores internos são inacessíveis e a taxa de obtenção de dados de telemetria do VANT é de 10 Hz (DJI, 2017d);
- ***DJI Onboard SDK***: é restrita à linha de VANTs “Matrice”, focada no uso profissional. Ela é apropriada para sistemas autônomos ou que operem além da linha de visada, sem controle remoto ou *link* de comunicação. Além disso, habilita a interação em baixo nível com alguns sensores, inclusive de terceiros, com capacidade de obtenção de dados de telemetria a 200 Hz, o que permite controle avançado e preciso do voo (DJI, 2017e).

A Figura 49 demonstra como as SDKs interagem com os Produtos da DJI. É possível perceber que a ***DJI Mobile SDK*** é disponibilizada para ser usada em conjunto

com as SDKs do Android ou iOS, o que permite desenvolver aplicações móveis nessas plataformas capazes de se comunicar com os produtos da DJI. Com exceção de alguns modelos, como o DJI Spark e o Phantom 3 *Standard*, que conectam-se por WiFi, o dispositivo móvel deve ser fisicamente conectado ao controle remoto via *Universal Serial Bus* (USB), que transmite os comandos da aplicação para o VANT por meio do *link* sem fio estabelecido (DJI, 2017d). Por sua vez, a ***DJI Onboard SDK*** é disponibilizada por meio de pacotes para sistemas operacionais ou plataformas de sistemas embarcados, como algumas distribuições Linux e o *Robotic Operating System* (ROS)[2]. A aplicação é executada em um dispositivo embarcado que é transportado pelo VANT e se conecta fisicamente ao controlador de voo por meio de interface *Universal Asynchronous Receiver-Transmitter* (UART).

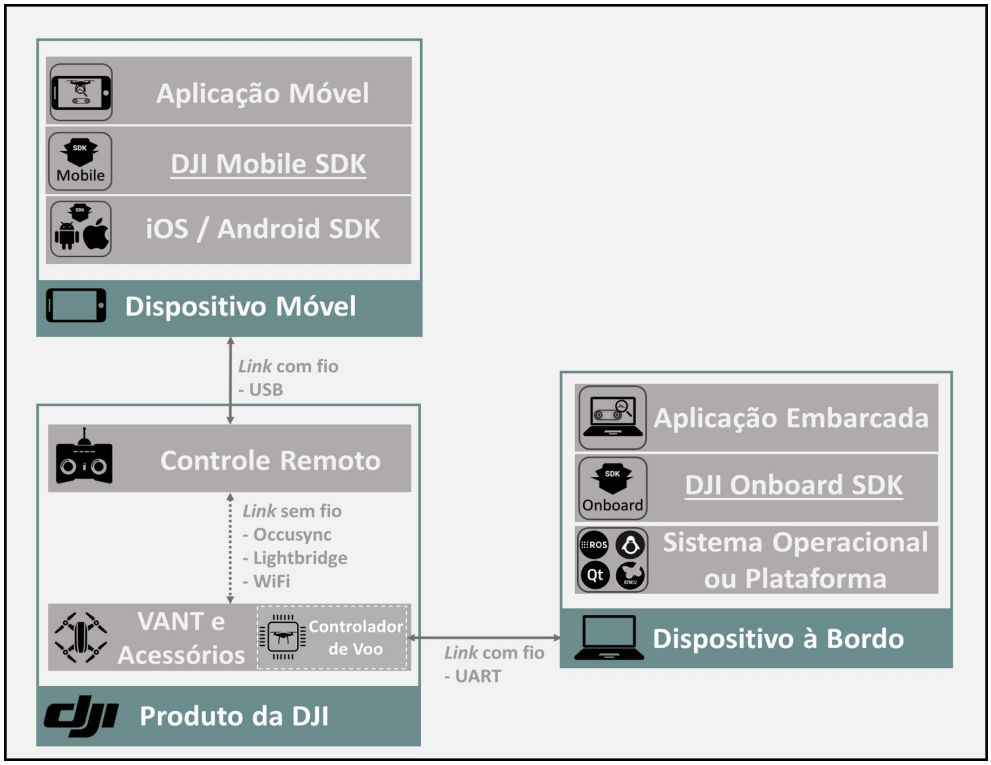



Figura 49 – Formas de interação das SDKs da DJI com seus produtos. Fonte: autor

Ainda que ambas possam ser usadas de forma combinada, cada uma das SDKs possui objetivos e características específicas, que precisam ser avaliadas frente ao tipo de aplicação desejado. Assim, a Tabela 7 apresenta um resumo das principais características das SDK disponíveis, bem como o tipo de operação mais apropriado para cada uma delas.

Não foi necessária a utilização da *DJI Onboard SDK* no presente trabalho. Isso se deve ao fato do tipo de operação nessa etapa não ser autônomo (também não permitido pela legislação brasileira) e não ter sido necessário realizar intervenções no comportamento de voo. Mesmo a taxa de 10 Hz para obtenção de dados de

2 Para mais informações, veja http://www.ros.org/about-ros/

Tabela 7 – Principais características das SDKs disponibilizadas pela DJI

| Categoria | *DJI Onboard SDK* | *DJI Mobile SDK* |
|---|---|---|
| Plataformas | Linux, STM32, ROS | Android, iOS |
| Linguagens | C++ | Java, Swift, Objective C |
| Produtos DJI Suportados | - Matrice: M100, M600, M210<br>- Controladores: A3, N3 | - Séries: Phantom, Inspire e Matrice<br>- Mavic Pro, Spark<br>- Controladores: A3, N3 |
| Conexão com VANT | Com Fio | - Disp. Móvel x Controle: Com Fio<br>- Controle x Aeronave: Sem Fio (Lightbrige, Occusync, WiFi) |
| Taxa de Comandos e Telemetria | 200 Hz | 10 Hz |
| Tipos de Missões | - Waypoint (seguir rota)<br>- Hotpoint (sobrevoar ponto) | - Waypoint<br>- Hotpoint<br>- Follow Me (seguir controle)<br>- Active Track (seguir objeto - reconhecimento de imagem)<br>- TapFly (voar para o ponto clicado no vídeo FPV) |
| Controles Relacionados à Câmera | - Capturar foto<br>- Iniciar/parar vídeo | - Capturar foto<br>- Iniciar/parar vídeo<br>- Configurar Exposição<br>- Reprodução de mídia |
| Tipo de Operação | - Autônomo<br>- Além do alcance do controle remoto<br>- Voos complexos, sem possibilidade de planejar via missões tradicionais | - No alcance do controle remoto<br>- Sem interação com sensores de terceiros |

Fonte: adaptado de DJI, 2017e

sensores para telemetria de voo, oferecida pela *DJI Mobile SDK*, já é suficiente para a a aplicação de inspeção de rolos. Dessa forma, a subseção a seguir apresenta, resumidamente, a estrutura da *DJI Mobile SDK*.

#### 4.4.1.1 Estrutura da *DJI Mobile SDK*

A DJI possui uma extensa linha de produtos, que vai desde estabilizadores de imagens a VANTs para uso recreativo e profissional. Além disso, cada um desses dispositivos pode receber acessórios ou evoluções ao longo do tempo, como novos tipos de câmeras e sensores. Dessa forma, a *DJI Mobile SDK* foi definida como uma estrutura abstrata, que permite que a linha de produtos evolua sem que as aplicações desenvolvidas tornem-se incompatíveis entre os produtos existentes e os novos que venham a ser lançados (DJI, 2017f).

Para que isso seja possível, a SDK é estruturada conforme mostrado na parte superior esquerda da Figura 50. É possível perceber que existe uma hierarquia ge-

nérica, que pode ser expandida ao longo do tempo. Por exemplo, o Produto pode ser tanto uma aeronave (*DJIAircraft*, exemplificada na parte superior direita da imagem) quanto um estabilizador (*DJIHandheld*). Caso a fabricante lance uma nova categoria de produtos, ela poderia ser incorporada. Cada **Produto** possui ainda seus **Componentes** (*link* de comunicação, câmera, gimbal, bateria, etc.) que, por sua vez, podem ainda ter **Subcomponentes** (por exemplo, *Lightbridge* dentro do *link* de comunicação). Os componentes e subcomponentes podem ter métodos diversos para acessar seu estado e configurações, e, de forma similar aos produtos, novos componentes e subcomponentes podem ser adicionados futuramente.

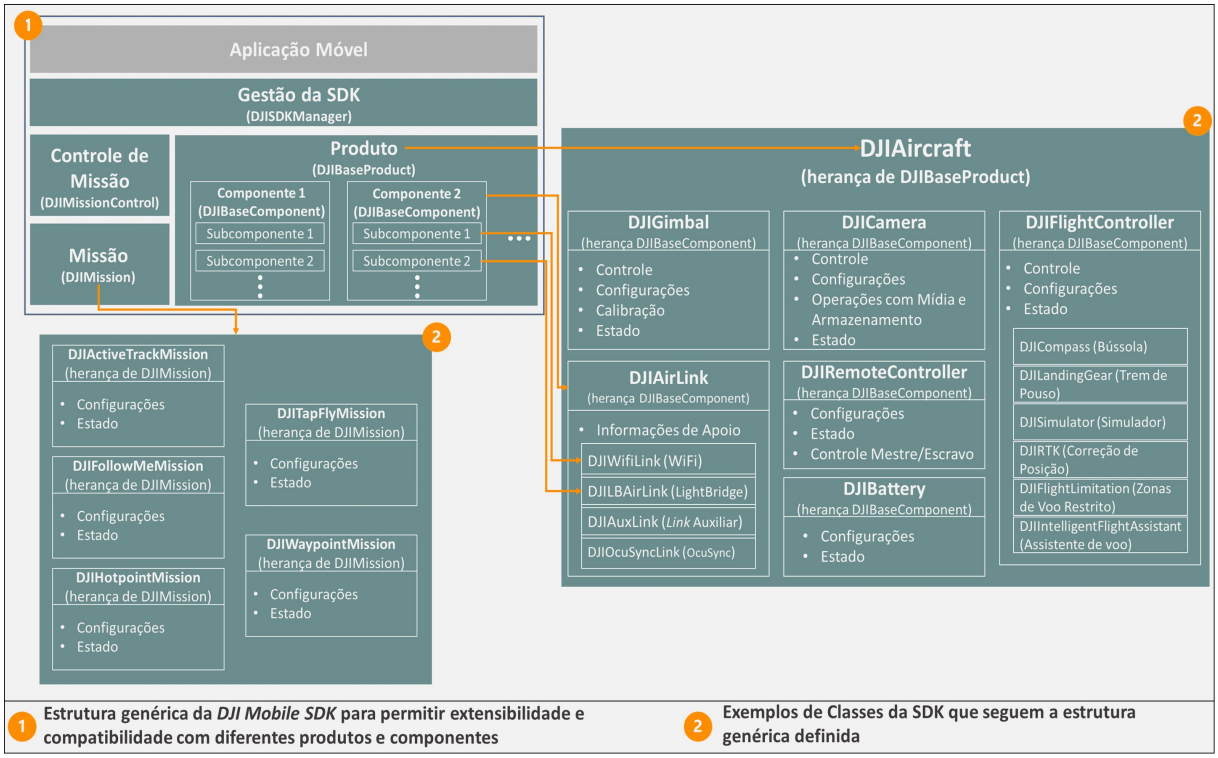



Figura 50 – Estrutura genérica da *DJI Mobile SDK* para permitir extensibilidade. Fonte: adaptado de DJI, 2017f

A conexão com o produto e registro da aplicação junto à DJI é feita pela classe de **Gestão da SDK** (*DJISDKManager*), com a qual a **Aplicação Móvel** interage para ter acesso às demais classes. Outra parte importante habilitada pela SDK é a gestão de missões, que são voos seguindo diferentes diretrizes, por exemplo, seguir um objeto, sobrevoar um local, seguir rota, etc. A classe de **Missão** (*DJIMission*) permite planejar, em tempo de execução da aplicação, diferentes tipos de missões, listados na Tabela 7 e exemplificados na parte inferior esquerda da Figura 50. Por fim, a classe de **Controle de Missão** (*DJIMissionControl*) permite controlar a execução de missões (carregar para o VANT, iniciar, interromper, etc.), podendo ainda coordenar múltiplas missões de forma sequencial (DJI, 2017f).

Todas essas funcionalidades da *DJI Mobile SDK*, particularmente a obtenção de

dados de sensores, o planejamento e execução de missões de *waypoints* foram utilizadas na construção da aplicação móvel para suportar a validação da arquitetura, que é melhor descrita na subseção "**5.4.1.1 - Aplicativo Móvel - Missões por *Waypoint***".

### 4.4.2 Simulação

Particularmente nos casos em que a Plataforma de Inspeção adota um VANT como portador, a simulação é extremamente importante, visto que situações não previstas nos algoritmos podem ser perigosas e afetar o comportamento de voo, colocando em risco o VANT, o patrimônio de terceiros e mesmo pessoas nas proximidades. Assim, a simulação descrita no presente trabalho teve como foco a camada de portador da Arquitetura Integrada, melhor descrita na seção "**5.2.1 - Portador**".

Existem duas modalidades principais de simulação: a primeira delas, conhecida como ***hardware-in-the-loop*** (HIL), combina partes reais (ou físicas) do sistema em questão com componentes ou ambientes simulados. Por exemplo, o uso do controle físico de um VANT para simular seu voo dentro de um ambiente simulado é caracterizado como HIL. A segunda modalidade é conhecida como ***software-in-the-loop*** (SIL), em que a totalidade do sistema é simulada, sem partes físicas. As simulações HIL são melhores, pois conseguem ser mais detalhadas e realistas (SILVA, 2008), sendo assim preferíveis na maior parte dos casos. Ainda assim, o uso de SIL é adequado em diversas situações, particularmente quando deseja-se antecipar ou validar aspectos específicos antes da construção física de qualquer parte do sistema.

Desta forma, o presente projeto utilizou simulações HIL como um estágio anterior aos testes em campo dos protótipos que envolviam voos de portadores do tipo VANT, seja em áreas operacionais ou mesmo em ambientes controlados. Para isso, foram adotadas duas abordagens principais: o desenvolvimento de um ambiente virtual do porto, que remete à área operacional onde os VANTs irão voar, e o uso do próprio simulador disponibilizado pela DJI, a fabricante dos VANTs usados como portadores neste trabalho. As subseções a seguir têm como objetivo apresentar as ferramentas e tecnologias utilizadas para suportar as duas abordagens descritas.

#### 4.4.2.1 Simulador Microsoft AirSim

Existem diversos simuladores disponíveis no mercado capazes ou mesmo focados na simulação de VANTs, como o Gazebo, Hector, RotorS e jMavSim. Porém, conforme Shah et al. (2017) explicam, ainda que consigam simular bem a dinâmica

de voo de diferentes tipos de VANTs e seus sensores, eles pecam na criação e renderização de ambientes, diminuindo o realismo ou prejudicando o teste de algoritmos que se baseiam no processamento de vídeo ou imagens, o que é amplamente utilizado para treinamento de algoritmos de inteligência artificial (*machine learning*).

Por conta disso, pesquisadores ligados à Microsoft desenvolveram o AirSim (SHAH et al., 2017), criado para permitir a simulação da dinâmica de diferentes tipos de veículos nas modalidades SIL e HIL. Inicialmente, os autores focaram o desenvolvimento em VANTs de asa rotativa, permitindo o uso de controladores de voo [3] reais, o que aumenta muito o realismo da simulação. O AirSim funciona como um *plugin* executado sobre a Unreal Engine (EPIC GAMES, 2017), que é um motor gráfico e físico extremamente popular, usado no desenvolvimento de jogos e aplicações diversas, capaz de renderizar gráficos de última geração e física realista.

É essa combinação um dos maiores diferenciais do AirSim em relação às outras plataformas: ele permite a simulação da dinâmica de um veículo, incluindo VANTs, em ambientes visualmente e fisicamente realistas, o que o torna apropriado para obtenção de dados para o treinamento de algoritmos de inteligência artificial e visão computacional. A versão disponível em Janeiro de 2018 disponibiliza dois tipos de veículos: VANT com estrutura de asa rotativa e carro. A Figura 51 apresenta uma visão geral da arquitetura do AirSim, demonstrando a forma de interação entre os elementos internos e externos para simulações HIL. Já a lista a seguir apresenta um breve descritivo dos principais elementos que compõem o simulador (à direita da imagem) (SHAH et al., 2017):

- **Modelagem do Veículo:** o simulador parte de uma estrutura de corpo rígida onde diferentes partes podem receber atuadores que geram força e torque. O modelo do veículo também inclui parâmetros como massa, inércia, arrasto angular e linear, coeficientes de fricção, etc., que interagem com o Motor de Física. Com essa estrutura, diferentes tipos de veículos podem ser modelados, o que reforça a extensibilidade do AirSim;

- **Modelagem do Ambiente:** existem diferentes fenômenos naturais que atuam sobre o veículo e que precisam ser considerados para aumentar o realismo. Dessa forma, os autores geraram modelos que permitem simular a gravidade, o campo magnético da Terra, a densidade e a pressão do ar (de acordo com altitude);

- **Motor de Física:** os autores adotaram 6 fatores para registrar a cinemática de cada parte do corpo que compõe um veículo: posição, orientação, velocidade

[3] Elemento central que calcula a propulsão em cada motor de acordo com a entrada no controle físico ou aplicação via SDK, como PX4 e ROSFlight

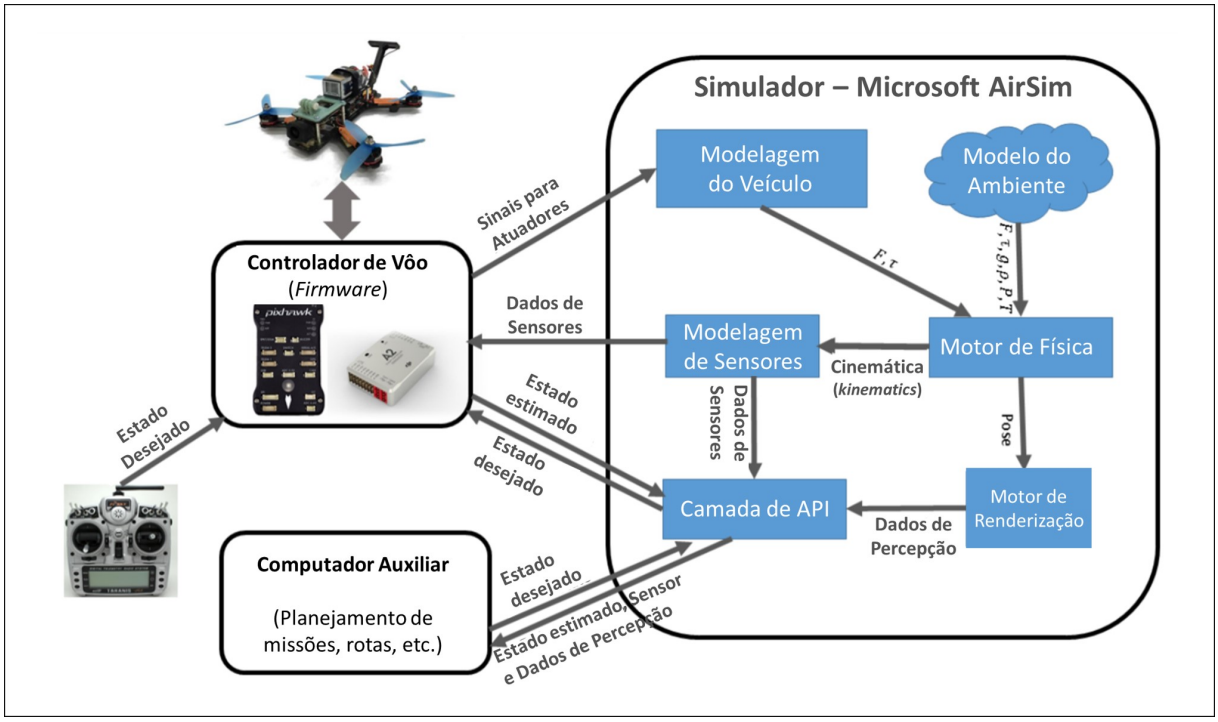



Figura 51 – Arquitetura do simulador Microsoft AirSim e interação com elementos externos. Fonte: traduzido de SHAH et al., 2017

e aceleração linear, velocidade e aceleração angular. Assim, este componente calcula o próximo estado cinemático e a pose, ao avaliar tais fatores a partir das forças e torque atuando sobre eles, com uma frequência de atualização de até 1000 Hz, o que permite simulações em tempo real. Existe ainda a modelagem física da própria Unreal Engine, que fornece informações sobre colisões para o motor de física, como posição de impacto e profundidade;

- **Modelagem de Sensores:** o AirSim implementa alguns dos sensores mais comuns em veículos reais, como barômetro, acelerômetro, giroscópio, magnetômetro e GPS. Como um diferencial, as implementações são interface abstratas, o que permite a customização do modelo de cada um desses sensores e ainda a implementação de novos sensores;

- **Motor de Renderização:** é o papel desempenhado pela Unreal Engine. Segundo os autores, ela foi escolhida por sua capacidade técnica avançada para renderização gráfica, por possuir uma loja própria que permite a aquisição de componentes gráficos prontos, incluindo ambientes, texturas e outros, e por estar disponível para os principais sistemas operacionais (Linux, Windows e OSX);

- **Camada de API:** permite a interação com o veículo por meio de comandos em diversas linguagens de programação, como C++, C#, Python, Java, etc. Na prática, é possível controlar o veículo, obter vídeos, imagens, dados de

sensores, etc., permitindo a criação de aplicações complexas que interagem com o simulador.

Além destes elementos, a Figura 51 também apresenta elementos externos ao simulador, que interagem com ele para simulações HIL. De forma resumida, comandos são enviados usando um **Controle** (ou *joystick*) físico, determinando o estado desejado para o veículo (acelerar, subir, descer, etc.). Eles são recebidos pelo **Controlador de Vôo** (ou pelo *firmware* do mesmo), que efetua os cálculos necessários para determinar a propulsão em cada motor e os encaminha para um **Veículo Real** ou para o **Simulador**. Para que ele faça os cálculos de forma precisa, ele recebe os dados de sensores, enviados pelo Simulador ou pelo Veículo Real. Por fim, opcionalmente, é possível utilizar um **Computador Auxiliar** para programar missões e rotas, que também interagem com o simulador por meio das APIs.

É exatamente com esta última funcionalidade que o AirSim apresenta a última de suas características mais interessantes: os algoritmos desenvolvidos para interação com o veículo podem funcionar tanto no ambiente de simulação como também ser embarcados em um veículo real, sem ajustes. Isso é possível porque a **Camada de API** do simulador, também conhecida como *AirLib*, é implementada com arquitetura Cliente-Servidor. Assim, o componente **Servidor** pode ser executado de forma independente no **Computador Auxiliar**, embarcado no veículo, comunicando-se com o **Controlador de Vôo** por meio de seu protocolo específico, como o MavLink no caso do controlador Pixhawk PX4 (MEIER; HONEGGER; POLLEFEYS, 2017). A aplicação **Cliente** então envia os comandos na linguagem que preferir, como Python ou C++, sem necessariamente saber se a execução está sendo feita no simulador ou em um veículo real (LOVETT, 2017).

A Figura 52 apresenta um VANT customizado que utiliza o *AirLib*, conforme arquitetura apresentada no último parágrafo. Neste caso, conforme explicado por Lovett (2017), foi empregado o **Controlador de Vôo** PX4, que se conecta via USB ao **Computador Auxiliar**, no caso, uma placa *Gigabyte Brix BXi7-5500* executando o Ubuntu, uma das distribuições Linux. Assim, este computador abriga o componente Servidor e também a aplicação Cliente, o que permite que o VANT realize missões de forma completamente autônoma. Essa arquitetura permitiria ainda que o Cliente fosse executado em outro computador e transferisse os comandos para o Servidor via WiFi.

Com base nestas informações, percebe-se que o AirSim apresenta características interessantes relativas a funcionalidades e extensibilidade, mesmo sendo um projeto relativamente recente, iniciado no final de 2016. O fato de ser executado usando a Unreal Engine abre caminho para a criação dos mais diversos tipos de


Figura 52 – Exemplo de um VANT real que utiliza a Camada de APIs *AirLib*. Fonte: LOVETT, 2017

ambientes, com física e modelagem tão realistas quanto a capacidade de modelagem à disposição. O fato de sua camada de APIs, a *AirLib*, poder ser embarcada em veículos reais e comunicar-se com alguns controladores de vôo, como o PX4, facilita o desenvolvimento de aplicações customizadas, diminuindo a barreira entre simulação e mundo real. Por fim, o fato de ser disponibilizado como código aberto e ter vários contribuidores ativos (MICROSOFT, 2018), indica que o AirSim continuará a ser aprimorado e a receber novas funcionalidades, tanto para VANTs quanto para outros tipos de aplicações, como carros autônomos.

Este conjunto de atributos levou à utilização do AirSim para simular um ambiente que remete ao pátio de materiais de um porto, que é uma das áreas possíveis para utilização da Plataforma de Inspeção para rolos de transportadores de correias. Esse ambiente pode ser utilizado tanto para teste de aplicações customizadas quanto para o treinamento de operadores da Plataforma, antes de efetivamente irem a campo com o equipamento real. O fato de utilizarem o controle físico no ambiente simulado, por si só, já tende a aumentar a familiaridade com o equipamento. A seção “**5.5 - Simulação**” apresenta mais detalhes sobre o ambiente desenvolvido e algumas aplicações customizadas para controle e obtenção de dados do portador no simulador.

Visto que nesta etapa do projeto foram utilizados para validação da Arquitetura Integrada para a Plataforma de Inspeção apenas VANTs comerciais, da fabricante DJI, o AirSim não foi empregado para testar as aplicações desenvolvidas com a SDK, descrita em “**4.4.1 - Desenvolvimento para VANTs**”. Assim, a próxima subseção apresenta o simulador utilizado nos testes de aplicações para os VANTs da

fabricante.

#### 4.4.2.2 Simulador da DJI

A DJI disponibiliza uma aplicação *Desktop* para testes de aplicações customizadas desenvolvidas usando sua SDK. O ***DJI Simulator*** (DJI, 2017g) permite simulações HIL, onde o controlador real do VANT é usado para calcular as saídas de acordo com as entradas enviadas pelo controle físico (*joystick*) ou aplicação customizada desenvolvida com a SDK. Além disso, é possível definir informações no ambiente virtual, como latitude e longitude do ponto inicial, e velocidade do vento em diferentes direções.

A aplicação é disponibilizada como parte do *DJI Assistant 2*, disponível apenas para Windows e OSX. Além disso, é necessário que um VANT da fabricante esteja fisicamente conectado ao computador que está executando o simulador para a visualização do ambiente. Um aspecto interessante e que aumenta a flexibilidade é que o simulador também é disponibilizado no controlador de voo do VANT e é acessível por meio da SDK, com diferentes métodos disponíveis para automatizar a execução de testes. Isso habilita a realização de *continuous integation*, ou seja, o desenvolvimento e liberação de novas versões pode ser automatizado passando por testes diretamente no simulador, sem a conexão a um computador, ainda que, neste caso, não seja possível acompanhar visualmente a simulação (DJI, 2017g).

Assim como outros simuladores disponíveis no mercado, o *DJI Simulator* é limitado na recriação de ambientes. Conforme mostrado na Figura 53, ele não permite customização visual, pois utiliza um ambiente padrão, onde a física está limitada à velocidade e direção do vento. Esse tipo de limitação, por exemplo, inviabiliza seu uso para coleta de dados para treinamento de algoritmos de inteligência artificial, como é possível com o *Microsoft AirSim*.

Em relação à simulação dos sensores, é possível obter dados da atitude do VANT (arfagem, guinada e rolagem), acelerômetros, giroscópio, a posição (coordenadas calculadas à partir do ponto inicial) e a velocidade de voo nos 3 eixos (X, Y e Z). Porém, não há suporte para nenhum tipo de simulação dos sensores de obstáculo que alguns VANTs da fabricante possuem e a simulação da câmera limita-se a repetir no ambiente simulado os movimentos do *gimbal*, sem a efetiva captura de imagens ou dados (DJI, 2017g). A Figura 54 apresenta as configurações que podem ser feitas, além dos diferentes dados de sensores que são atualizados em tempo real quando o simulador está conectado ao VANT.

Apesar de permitir uma boa simulação dinâmica do voo do VANT e dar segurança para testes das aplicações antes de ir a campo, este simulador é limitado em

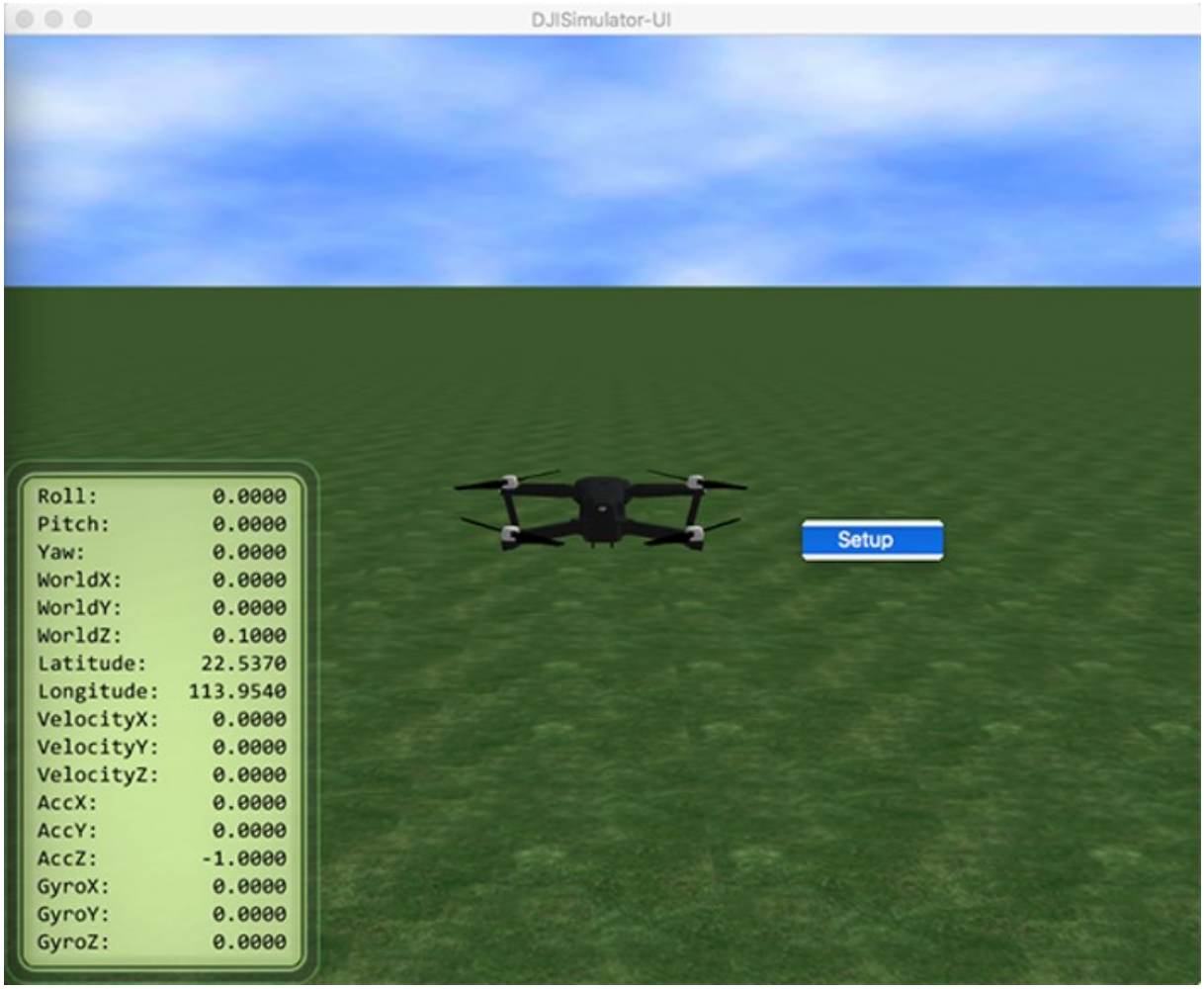



Figura 53 – Ambiente único disponível no *DJI Simulator*. Fonte: DJI, 2017g

diferentes aspectos e, obviamente, só funciona com VANTs da fabricante DJI. Por conta disso, o seu uso no projeto foi restrito aos testes de alguns dos protótipos desenvolvidos baseados em VANTs da DJI, propósito que o mesmo atendeu plenamente, já que os defeitos mais graves na aplicação foram corrigidos com base nos testes do simulador, antes de ir a campo. Por exemplo, o protótipo discutido na seção "**5.4.1 - Protótipo 1 - Aplicativo Móvel e Sistema de Controle da Inspeção**" foi extensivamente testado no simulador antes dos testes em ambientes controlados e na área operacional.

## 4.5 Conclusões do Capítulo

O presente capítulo apresentou as tecnologias, conceitos e ferramentas utilizadas para suportar a definição da Arquitetura Integrada e o desenvolvimento de protótipos e simulação para a sua validação. Inicialmente, a seção "**4.1 - Veículos Aéreos Não Tripulados (VANTs)**", apresentou estes veículos, que são um dos por-

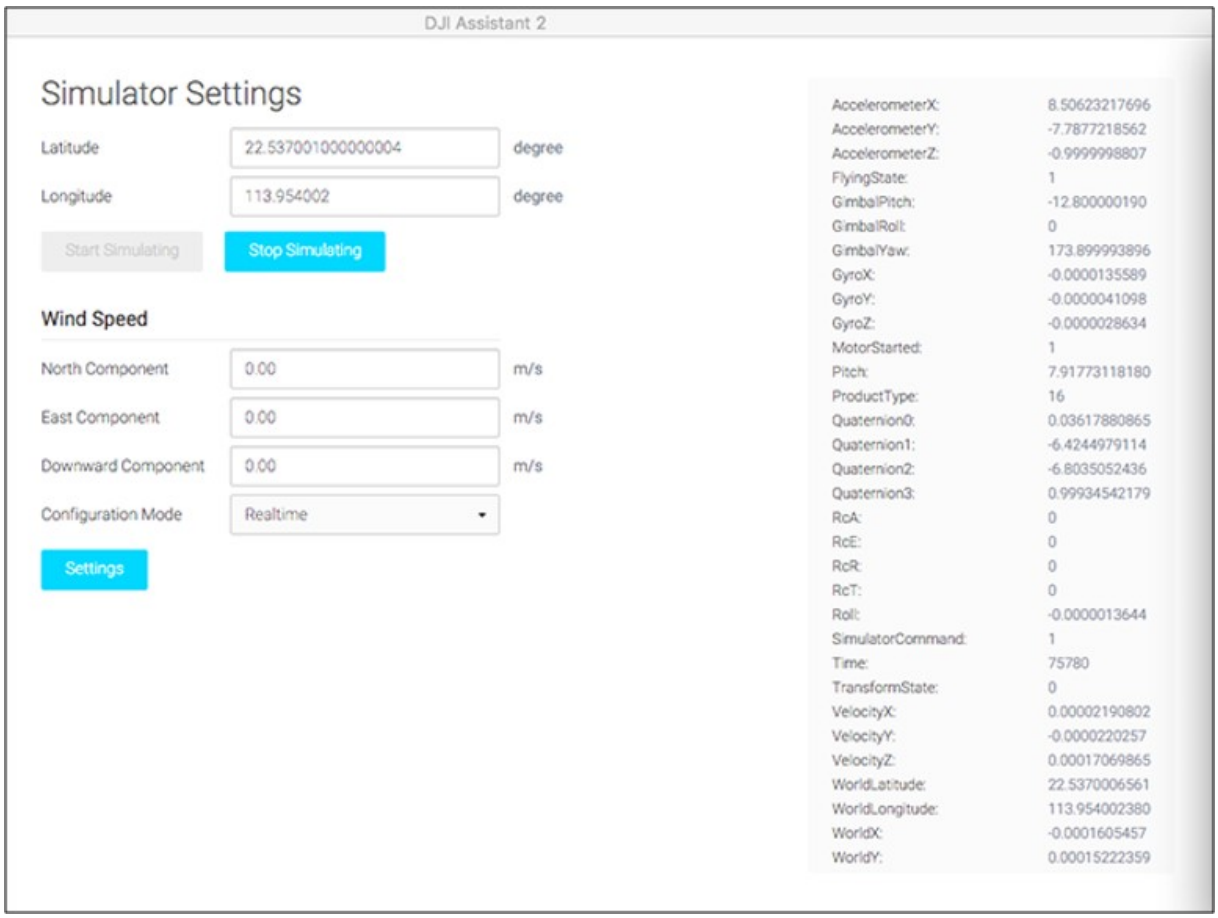



Figura 54 – Configurações e dados de sensores disponíveis no *DJI Simulator*. Fonte: DJI, 2017g

tadores possíveis para transportar a estrutura de sensoriamento da Plataforma de Inspeção. Nela, foi exposto o histórico da evolução da tecnologia, sua forte aplicação para fins militares e sua recente popularização, que habilitou o uso doméstico e em aplicações industriais, como a proposta neste trabalho. Também foram discutidos aspectos relativos à estrutura de Sistemas VANTs, concluindo-se que aeronaves com estrutura de Asa Móvel são mais apropriadas para a aplicação em questão. Na parte final, foram abordados assuntos relativos à regulamentação, com destaque para a legislação brasileira, que, mesmo recente, dá segurança jurídica ao uso comercial destas aeronaves, ainda que não permita voos autônomos.

Em seguida, a seção "**4.3 - Técnicas e Protocolos de Integração**" abordou temas que sustentam o aspecto central da arquitetura proposta: a integração da Plataforma de Inspeção com Sistemas Corporativos. Inicialmente, foi apresentado o conceito de SOA e seu propósito de suportar a comunicação entre aplicações distribuídas usando serviços, permitindo reuso e evolução independente dos componentes. Na sequência, como uma forma prática de implementar essa arquitetura, foi discutido o papel do Barramento de Integração e como ele pode facilitar a comunicação entre aplicações e a administração dos serviços em ambientes complexos.

Com esses conceitos desenvolvidos, foram demonstrados os principais tipos de interação da Plataforma de Inspeção com sistemas corporativos, relacionando cada caso de uso com Padrões de Mensagem bem estabelecidos na literatura, basicamente, *Request/Response* e *Publish/Subscribe*. O primeiro é utilizado em interações síncronas, baseadas em eventos, onde espera-se algum tipo de resposta para cada requisição, mesmo que apenas uma confirmação de recebimento. Para o segundo tipo, a comunicação é feita de forma totalmente independente, já que uma aplicação pode estar publicando dados para os quais não existam consumidores.

Em relação ao padrão *Request/Response*, foram discutidas formas de integração usando o protocolo SOAP, muito comum em aplicações corporativas dada sua robustez, e o estilo de arquitetura REST, que procura simplificar a comunicação com uma abordagem leve e apropriada também a dispositivos móveis. Neste contexto, demonstrou-se que SOAP e REST são complementares, não cabendo comparações diretas. Já para o padrão *Publish/Subscribe*, foram explicados os protocolos AMQP, CoAP e MQTT, que são apropriados para as restrições encontradas em ambientes de IoT e suportam comunicação entre dispositivos (M2M). Para este cenário, ainda que possuam diferenças por conta do propósito que motivou a criação de cada um, concluiu-se que os três protocolos poderiam ser usados neste trabalho, mas optou-se pelo MQTT dada sua simplicidade, robustez e recursos que facilitam o desenvolvimento de aplicações.

Dando prosseguimento, a seção "**4.2 - Paradigmas de Computação: *Edge*, *Fog* e *Cloud***" demonstrou diferentes abordagens para o processamento de dados em ambientes distribuídos, discutindo como tais conceitos inspiraram a definição deste aspecto na Arquitetura Integrada. Foram explicados os motivos que vem levando empresas a migrarem suas aplicações para a nuvem (*cloud*), dentre eles, menores custos e escalabilidade. Também foi discutido como essa abordagem foi desafiada pelos requisitos de algumas aplicações de IoT, como tempos de resposta próximos de tempo real, o que trouxe a necessidade de camadas de processamento mais próximas aos dispositivos, na borda da rede (ou *Edge*). Finalmente, foi explicado o papel do *Fog*, que oferece maior poder de processamento que o *Edge* e posiciona-se em uma camada superior, com abrangência regional, sendo o *Mobile Edge Computing* um exemplo de sua implementação.

Por fim, a seção "**4.4 - Ferramental para Protótipos e Simulação**" apresentou as principais ferramentas e tecnologias utilizadas para suportar a parte prática deste trabalho, tanto na construção de protótipos quanto na parte de simulação. Dessa forma, foram apresentados os *kits* de desenvolvimento (ou SDKs) para realizar a construção de aplicações móveis ou embarcadas capazes de comunicar-se com VANTs da fabricante DJI, usados como portadores nos protótipos apresentados na

seção “**5.4 - Protótipos Desenvolvidos**”. Em seguida, foram desenvolvidos conceitos relativos à simulação de VANTs, visto que foi esse o tipo de portador escolhido para os protótipos. Além do simulador da fabricante DJI, utilizado para testes das aplicações desenvolvidas com a SDK antes de ir a campo, também foi apresentado o Microsoft AirSim, que utiliza a Unreal Engine como motor gráfico e de física, além de ser extensível e aberta por meio de APIs que também podem ser embarcadas em VANTs reais. Essa combinação justificou o uso do AirSim no trabalho para recriar um ambiente de porto, que é uma das áreas mais prováveis de uso da Plataforma de Inspeção com um portador do tipo VANT, o que é melhor detalhado na seção “**5.5 - Simulação**”.

# 5 Desenvolvimento

Este capítulo apresenta as principais etapas de desenvolvimento do trabalho, como a definição da arquitetura integrada e o detalhamento dos protótipos que foram criados para validação da mesma. Desta forma, o capítulo está organizado da seguinte forma:

- Inicialmente, a seção "**5.1 - Metodologia**" discute a forma de desenvolvimento deste trabalho;
- Já a seção "**5.2 - Arquitetura Integrada**" descreve a arquitetura proposta e detalha cada uma de suas camadas;
- Na sequência, a seção "**5.3 - Evolução da Plataforma de Inspeção**" discute uma forma de construção da plataforma que habilita sua evolução modular e incremental;
- Dando prosseguimento, a seção "**5.4 - Protótipos Desenvolvidos**" discute os protótipos usados para validação da arquitetura proposta;
- Por fim, a seção "**5.5 - Simulação**" discute a importância da simulação no contexto do portador VANT, além detalhar o ambiente virtual construído e algumas aplicações.

## 5.1 Metodologia

A solução escolhida para o problema de monitoramento de rolos, usando uma estrutura de sensoriamento móvel e independente do TC (Plataforma de Inspeção) possui desafios, onde se destacam a integração das diferentes partes que a compõem e a comunicação com sistemas externos. Com isso em mente, adotar um método de desenvolvimento tradicional, baseado em atividades sequenciais, pode não ser o mais eficiente, já que os diferentes componentes tem sua evolução própria e não há dependência entre alguns deles. Assim, foi proposta para a construção da plataforma uma abordagem cíclica e incremental, em que diferentes partes são evoluídas individualmente e posteriormente integradas, melhorando-a progressivamente. A Figura 55 apresenta a metodologia adotada.

Para suportar essa forma de evolução, a arquitetura propõe diferentes camadas que buscam separar o portador, o sensoriamento e a extração dos sinais para permitir a evolução de cada uma dessas camadas de forma independente. Em termos

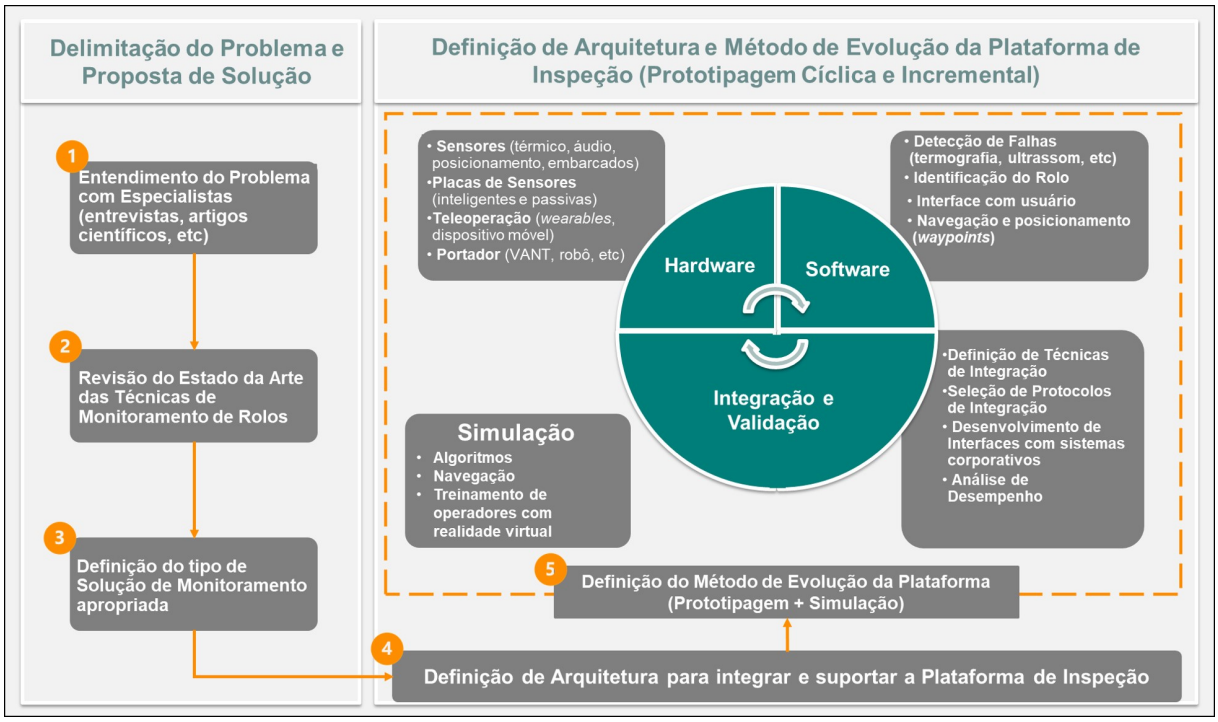



Figura 55 – Metodologia proposta com as principais partes envolvidas. Fonte: autor

práticos, a seção "**5.3 - Evolução da Plataforma de Inspeção**" apresenta uma das possíveis formas de se implementar uma solução modular desse tipo, ainda que diversas outras sejam factíveis. Especificamente para a construção dos protótipos apresentados neste trabalho, foram utilizados equipamentos comerciais.

Isto posto, não é parte do escopo do trabalho a especificação e desenvolvimento de todos os componentes da plataforma em questão, apenas a construção de protótipos que validem os principais aspectos da arquitetura proposta, em especial a integração com sistemas corporativos. Cabe também destacar que o presente trabalho é parte de uma iniciativa maior, com outros pesquisadores atuando em diferentes frentes, como processamento de sinais e detecção de falhas, realidade virtual e aumentada. Assim, a lista abaixo apresenta as principais atividades realizadas para se atingir os objetivos propostos para o trabalho:

- Caracterização do problema, relevância e consequências para a indústria da mineração, por meio de entrevistas com especialistas e revisão de bibliografia especializada;
- Levantamento do estado da arte das soluções disponíveis para o problema de monitoramento de condições de rolos;
- Avaliação das soluções e definição da abordagem diante de restrições encontradas nas operações;

- Proposição de uma Arquitetura Integrada que permita a construção e evolução de uma Plataforma de Inspeção, além de sua integração a sistemas corporativos;
- Revisão de aspectos centrais desta arquitetura, inclusive técnicas e protocolos de integração adequados aos casos de uso de integração da plataforma;
- Uso de prototipagem para validação dos principais aspectos da arquitetura e de simulação para alguns casos de uso do portador do tipo VANT;
- Testes em campo de um protótipo, preferencialmente em uma área operacional com características próximas às que serão encontradas durante as missões de inspeção.

Desta forma, a seção a seguir descreve a arquitetura proposta, com a discussão de cada uma de suas camadas e como elas se combinam para melhorar as práticas atuais de inspeção de rolos.

## 5.2 Arquitetura Integrada

Conforme seção "**3.2 - Discussão da Abordagem para a Plataforma de Inspeção**", a forma escolhida para o monitoramento de rolos envolve o uso de uma Estrutura de Sensoriamento Móvel e independente do TC, aqui chamada **Plataforma de Inspeção**. Para que seja viável, é importante que informações coletadas e processadas por ela sejam disponibilizadas aos sistemas corporativos, da mesma forma que estes devem prover informações que suportem a inspeção. Essa integração de mão dupla, em que ambos se complementam, é o que caracteriza o termo **Arquitetura Integrada**, onde diferentes camadas, tanto internas quanto externas à Plataforma de Inspeção, são usadas em conjunto visando a melhoria da prática atual na inspeção de rolos de transportadores de correia.

Na Arquitetura Integrada, demonstrada na Figura 56, cada uma das **Camadas** possui um ou vários **Componentes** que, quando aplicados individualmente ou de forma combinada, suportam o desempenho de **Funções** que contribuem para a solução do problema em questão. Essa contribuição se reflete no fluxo de dados mostrado na mesma figura. À medida que eles fluem das camadas inferiores para as superiores da arquitetura, sua quantidade diminui, mas sua relevância aumenta. Por exemplo, vários segundos do sinal de áudio bruto capturado por um microfone na camada de **Captura de Dados** não tem o mesmo valor que o seu processamento e registro de defeito na camada de **Sistemas Corporativos**.

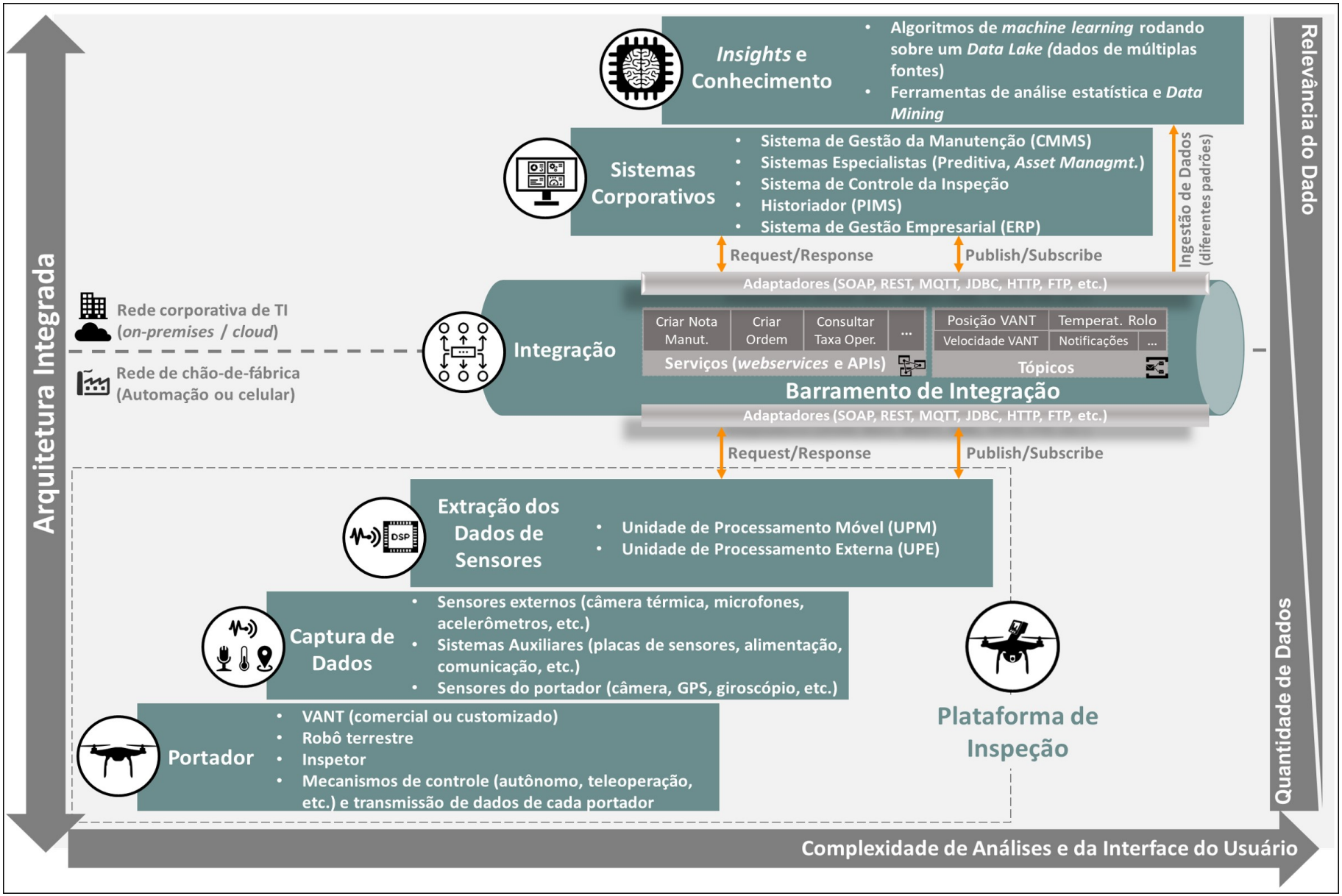



Figura 56 – Arquitetura Integrada proposta para o monitoramento de rolos. Fonte: autor

Outro aspecto que merece destaque na arquitetura proposta refere-se ao tipo de interface de usuário e complexidade de análises. À medida que se avança nas camadas mostradas na Figura 56, as interações tendem a ser mais ricas e complexas. Por exemplo, enquanto a camada de **Captura de Dados** consegue obter e apresentar ao usuário o sinal de áudio e de uma câmera térmica de forma individual, é a camada de **Extração dos Dados de Sensores** que pode combiná-los para determinar se existe ou não um defeito. Por sua vez, a camada de ***Insights* e Conhecimento**, no topo da arquitetura, pode cruzar os dados de defeitos reportados com a taxa de operação do TC, tipo de produto transportado, temperatura ambiente, etc., para realizar análises ainda mais complexas e impossíveis em níveis inferiores.

Desta forma, conforme demonstrado na Figura 56, a Plataforma de Inspeção corresponde apenas às três camadas inferiores. Por si só, a plataforma é sim capaz de realizar as inspeções e obter a condição dos rolos. Porém, perde-se em eficiência e confiabilidade ao não usar a camada de **Integração** para ligá-la às duas camadas superiores. A primeira delas, **Sistemas Corporativos**, pode prover informações relevantes para a plataforma e, claro, precisa receber os defeitos e outras informações para aumentar a produtividade dos processos. Já a segunda, conforme explicado, combina dados da plataforma e dos sistemas corporativos para realizar análises complexas. Assim, fica evidente que todas as camadas possuem importância própria na solução do problema e, portanto, são melhor detalhadas nas próximas subseções.

### 5.2.1 Portador

De forma abrangente, quaisquer mecanismos que consigam carregar uma estrutura de sensores habilitando-os a obter sinais dos rolos de TCs podem ser considerados portadores na arquitetura em questão. Isso inclui VANTs (Figura 57a), robôs terrestres (Figura 57b), inspetores humanos e mesmo veículos. Dessa forma, um dos diferenciais propostos no presente trabalho é que Plataforma de Inspeção deve ser independente do portador, funcionando com diferentes tipos.

Esse atributo é fundamental, já que cada tipo de operação na indústria da mineração tem suas próprias características e desafios, que dificilmente são supridas por um único tipo de portador. Por exemplo, enquanto os VANTs podem ser extremamente rápidos e eficientes em áreas abertas, como os pátios, eles teriam dificuldades com TCs operando em espaços restritos e fechados, como chutes e casas de transferência, áreas que robôs terrestres conseguem acessar com maior eficiência. Assim, cada portador traz consigo suas próprias qualidades e limitações, que precisam ser avaliadas de acordo com as particularidades do local onde irão operar.

(a) VANT DJI Inspire 1 utilizado como portador. Fonte: autor

(b) Robô terrestre adaptado para realização de inspeção de rolos de TCs. Fonte: FRANCA; FERNANDES, 2017

Figura 57 – Exemplos de portadores possíveis para a Plataforma de Inspeção

Esta camada inclui também os mecanismos de controle do portador em questão, exceto quando se trata de um inspetor carregando os sensores. Vários mecanismos de controle podem ser adotados, como teleoperação, orientação visual, navegação autônoma ou por pontos específicos, também conhecidos como *waypoints*. Novamente, o tipo de controle irá variar de acordo com o portador e a operação.

A primeira fase do presente projeto teve como objetivo a inspeção de rolos localizados em pátios, já que estes concentram a maior extensão de TCs em operações da indústria de mineração. Por conta disso, foram priorizados os testes usando VANTs como portador. Para o controle, foram utilizadas dois tipos de estação de controle (GCS): i) uma composta pelo controle remoto e um dispositivo móvel rodando uma aplicação padrão do fabricante e; ii) usando um controle remoto e uma aplicação totalmente customizada em Android, que permite criar missões de inspeção via *waypoints*, melhor discutida na subseção "**5.4.1 - Protótipo 1 - Aplicativo Móvel e Sistema de Controle da Inspeção**".

### 5.2.2 Captura de Dados

O monitoramento dos rolos só é possível se houver sensores capazes de capturar seu estado usando um ou múltiplos sinais apropriados, como o acústico, térmico e vibração, discutidos na seção "**2.2 - Roletes, Rolos e Detecção de Falhas**". Cada um desses sinais possui suas peculiaridades, portanto, exigem sensores específicos e procedimentos apropriados para que as medidas obtidas sejam acuradas e

forneçam a condição real do rolo. De forma genérica, esses sensores serão chamados de estrutura de sensoriamento.

Assim, essa camada tem como objetivo a obtenção dos sinais que permitem avaliar a condição do rolo e de sinais auxiliares, que suportem a identificação dele ou auxiliem na operação da Plataforma de Inspeção (controle e navegação). Para tanto, essa camada é composta por de três tipos de elementos principais:

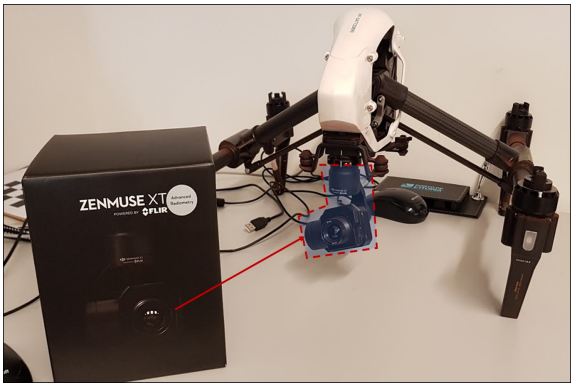
- **Sensores Externos:** são os diferentes componentes que possuem a capacidade de obtenção de algum tipo de sinal, por exemplo, câmeras térmicas (Figura 58a), acelerômetros, arranjos de microfones, módulos de posicionamento GNSS, como o *Global Positioning System* (GPS), leitores de RFID, câmeras, etc. Eles podem ser sensores tradicionais ou *smart-sensors* (MEIJER, 2008), com capacidade de processamento e transmissão embutida. Virtualmente, qualquer tipo de sensor que não esteja embarcado ou ligado ao portador se enquadra nesta categoria;

- **Placas de Sensores e Auxiliares:** muitos dos sensores tradicionais mencionados acima precisam ser conectados e processados para que sejam úteis. Para tanto, pode-se empregar placas de sensores (*Arduino Mega*, *Intel Edison*, etc.) e sistemas auxiliares de alimentação e comunicação, como USB e *Controller Area Network* (CAN) para citar dois exemplos;

- **Sensores do Portador:** alguns portadores possuem sensores, embarcados ou removíveis, que também podem ser utilizados para obter a condição do rolo ou para suportar seu controle ou navegação. Por exemplo, câmeras, receptores GPS/GNSS, giroscópios, ultrassom para detecção de obstáculos, etc. (Figura 58b). De forma geral, qualquer sensor do portador cujos dados sejam acessíveis via API ou outro mecanismo enquadram-se nessa categoria.

Com exceção da última categoria, em que presume-se que os sensores embutidos ou conectáveis ao portador são naturalmente compatíveis, é importante que os **Sensores Externos** e **Sistemas Auxiliares** sejam adequados ao portador que os está transportando. Por exemplo, os VANTs tem limitações de carga, logo, estruturas de sensoriamento que extrapolem essa capacidade não são apropriadas para compor uma Plataforma de Inspeção usando um VANT como portador. Dessa forma, ainda que as camadas da arquitetura tenham funções específicas e visem a interoperabilidade, é o portador que define o que pode ou não ser carregado.

Nos protótipos desenvolvidos foram utilizados VANTs comerciais com seus respectivos sensores. Resumidamente, em relação à camada de *Captura de Dados*, foram utilizadas câmeras térmica e tradicional para avaliação das condições do




(a) Câmera térmica Flir Zenmuse XT, específica para alguns VANTs da DJI

(b) Câmeras embutidas no portador (principal e do sistema de visão)

Figura 58 – Exemplos de sensores que podem compor a camada de Captura de Dados. Fonte: autor

rolo e receptores de GPS, tanto para controle e navegação do VANT quanto para a identificação do rolo por meio de sua posição aproximada. Assim, não houve a utilização de sensores externos e, consequentemente, dos sistemas auxiliares, ainda que seja possível, dada a modularidade da arquitetura proposta.

### 5.2.3 Extração dos Dados de Sensores

Após a obtenção do sinais de interesse, é preciso processá-los para avaliar se correspondem a situações anormais. Esse é o papel realizado por essa camada, que utiliza os dados brutos obtidos dos sensores da camada anterior e aplica diferentes técnicas e algoritmos para avaliação da condição dos rolos e sua identificação. Existe mais de uma forma para realizar o processamento dos dados capturados, como de forma embarcada ou externamente. Na abordagem escolhida para a Plataforma de Inspeção, que utiliza um portador móvel, tem-se um desafio adicional: a conectividade.

Isso ocorre porque o portador da estrutura de sensoriamento e o *hardware* para processamento dos dados podem estar fisicamente separados, o que leva a alguns contratempos. O primeiro e mais importante deles é a falta de cobertura por redes confiáveis e de alta velocidade nas áreas em que os portadores irão operar. Aliado a isso, o volume de dados gerados para capturar os sinais de interesse pode ser significativo. De forma ilustrativa, um minuto de vídeo em resolução 4K (3840 x 2160) capturado a 30 quadros por segundo tem, aproximadamente, 434 MB. Adicionalmente, é importante lembrar que nem todo sinal capturado é de interesse para o processamento de defeitos, o que leva à última parte do problema: o tráfego desnecessário de dados.

Por outro lado, nem sempre é possível ou desejável adicionar poder de proces-

samento ao portador (embarcado ou ao *hardware* que ele transporta) para que todas as análises sejam feitas localmente. Isso se deve ao fato de o portador ser móvel e, assim, depender de alimentação por baterias na maioria das situações. O aumento no processamento significa maior consumo de bateria e, em alguns casos, mais peso. Em alguns tipos de portadores, como os VANTs, essa combinação significa reduzir a distância coberta a cada voo, diminuindo a produtividade.

Para acomodar tais características e restrições, essa camada da arquitetura foi dividida em dois elementos, que agrupam diferentes responsabilidades:

- **Unidade de Processamento Móvel (UPM):** é responsável por realizar o processamento de defeitos e identificação dos rolos de forma embarcada na Plataforma de Inspeção, ou seja, não depende de processamento externo. Assim, ela é composta por diferentes algoritmos de tratamento de sinais e *machine learning* capazes de serem executadas por *hardware* embarcado. Neste caso, o termo embarcado refere-se tanto a elementos de processamento do portador, como a GCS no caso de um VANT, quanto a *hardware* dedicado, fisicamente conectado ou apenas transportado por ele. Nesse contexto, a função desempenhada pela UPM assemelha-se ao conceito de computação *Edge* apresentado na seção "**4.2 - Paradigmas de Computação: *Edge*, *Fog* e *Cloud***";
- **Unidade de Processamento Externa (UPE):** nem sempre é possível processar de forma *online* os sinais capturados, seja por falta de conectividade entre a estrutura de sensoriamento e a UPM ou pela complexidade, visto que algoritmos de processamento de sinais para vibração e áudio podem demandar poder computacional que não está disponível na UPM. Portanto, a UPE permite que sinais capturados pela plataforma sejam armazenados em uma memória secundária e possam ser processados posteriormente de forma *offline*. Assim, a UPE pode ter partes que não são do *hardware* da Plataforma de Inspeção, porém, numa visão lógica, elas a compõem.

Tanto a UPM quanto a UPE podem utilizar diferentes elementos para interação com o usuário, de modo que ele visualize os resultados do processamento realizado. Além disso, elas também contêm as APIs e bibliotecas que permitem o envio e recebimento de mensagens de integração por meio de protocolos adequados, sempre passando pela camada de **Integração**, que é discutida na sequência.

Nos protótipos desenvolvidos, a UPM foi desempenhada por um aplicativo móvel em Android sendo executado como parte da estação de controle (GCS) do VANT, detalhado na subseção "**5.4.1 - Protótipo 1 - Aplicativo Móvel e Sistema de Controle da Inspeção**"). Por sua vez, a UPE foi avaliada por meio de algoritmos executados no MATLAB para processamento *offline* de defeitos a partir de imagens

térmicas capturadas pelo VANT e envio automático do defeito para o Sistema de Manutenção, posteriormente abordado na subseção "**5.4.2 - Protótipo 2 - Envio de defeitos da UPE para o Sistema de Manutenção**". É importante destacar que os algoritmos de detecção de defeitos da UPM e UPE não foram desenvolvidos pelo autor deste trabalho, apenas foram incorporados para mostrar a viabilidade da arquitetura proposta.

### 5.2.4 Integração

Esta camada tem como objetivo viabilizar a integração entre a Plataforma de Inspeção e diferentes sistemas corporativos relacionados à inspeção de rolos. Conforme explicado em detalhes na seção "**4.3 - Técnicas e Protocolos de Integração**", diferentes tipos de técnicas, ferramentas e protocolos são utilizados para que isso seja possível. O **Barramento de Integração**, nomenclatura genérica usada para o *Enterprise Service Bus* (ESB) e *Broker*, é o elemento central dessa camada, pois é ele quem viabiliza a interoperabilidade ao desempenhar diferentes papeis.

O primeiro papel desempenhado é o de ser o ponto de conectividade entre os sistemas corporativos e a Plataforma de Inspeção, visto que os primeiros são hospedados em uma rede de TI, enquanto a plataforma pode estar conectada via rede de chão-de-fábrica ou mesmo por redes celulares. Assim, o Barramento de Integração funciona como uma ponte segura para esses dois mundos, permitindo a comunicação por meio de serviços e tópicos disponíveis no barramento.

O segundo papel está relacionado a conceitos de SOA, como o abstração e reaproveitamento. Os serviços e tópicos expostos no barramento podem ser consumidos pela Plataforma de Inspeção sem que ela precise de detalhes de quais sistemas são os seus provedores. Isso também é válido no sentido oposto. Quando os sistemas corporativos recebem as informações, não precisam saber se quem as enviou foi uma plataforma baseada em VANT com sensores térmicos ou um robô terrestre com sensores acústicos.

O terceiro papel do barramento é o de adaptação de protocolos e tecnologias nas interações "*Request-Response*". A maior parte dos sistemas corporativos expõe seus *web services* usando o protocolo SOAP, mais robusto e apropriado para interação entre eles, enquanto as Plataformas de Inspeção tendem a ser baseadas em *hardware* com menor poder computacional, que as leva a utilizar REST, que é mais leve e apropriado nesses casos. Assim, cabe ao barramento fazer a tradução SOAP x REST para viabilizar a comunicação. Em tempo, as interações *Publish-Subscribe* também podem beneficiar-se desse recurso, por exemplo, permitindo que um sistema publique seus dados via MQTT e outro realize seu consumo via AMQP.

Vale lembrar que várias tecnologias e soluções podem ser usadas de forma combinada como barramento de integração, portanto, não se trata de apenas um *software* ou plataforma única. De forma mais ampla, como uma regra geral, quão mais robusta e provida de funcionalidades é a camada de **Integração**, mais simples se torna para que as aplicações interajam entre si. Desse modo, ela traz pra si a complexidade de modo a simplificar a comunicação entre aplicações heterogêneas, como a Plataforma de Inspeção e os sistemas corporativos.

### 5.2.5 Sistemas Corporativos

Em grandes organizações, como as da indústria de mineração, é comum que diferentes tipos de sistemas sejam utilizados na gestão dos seus processos. Quando se avalia especificamente o Processo de Manutenção, que é o responsável por garantir a integridade dos ativos, incluindo os TCs e seus rolos, vários sistemas suportam suas diferentes fases. Desta forma, essa camada é composta pelos sistemas que possuem relação com o processo de inspeção de rolos e, por conta disso, precisam prover ou receber informações da Plataforma de Inspeção.

Ainda que a maior parte dos sistemas nessa camada seja relacionada ao processo de manutenção em si, é importante observar que a Plataforma de Inspeção também pode demandar informações de outras fontes, por exemplo, de uma aplicação que monitora as condições climáticas e suporte a decisão de iniciar ou não uma missão de inspeção. Com isso em mente e de forma não exaustiva, alguns dos agentes que podem fazer parte dessa camada são:

- **Sistema de Manutenção (CMMS):** responsável pela gestão dos processos de inspeção, planejamento, programação e execução de manutenção (corretiva ou programada). Por conta disso, realiza a gestão da carteira de ordens e notas de manutenção, necessitando, portanto, receber os defeitos identificados pela Plataforma de Inspeção. Ele também pode prover informações diversas à plataforma, como a última data de inspeção em um TC, data de instalação de um rolo, etc;
- **Sistemas Especialistas:** responsáveis por atividades diversas, dentre elas, análises de causa raiz e avaliação preditiva de condição dos rolos. Assim, pode receber dados de temperatura, vibração e acústico dos rolos para compor laudos preditivos;
- **Sistema de Controle da Inspeção:** permite acompanhar de forma remota as atividades da Plataforma de Inspeção, como a posição do portador, dados de sensores e informações relevantes para seu monitoramento. Além disso, ele

pode enviar mensagens ou parâmetros de configuração importantes para a plataforma. Por exemplo, a termografia é afetada pela umidade relativa do ar, logo, esse sistema pode enviar automaticamente valores atualizados para a plataforma ser reconfigurada;

- **Historiador (PIMS):** é responsável por armazenar grandes quantidade de informações dos processos onde está inserido. Por conta disso, o PIMS é o sistema apropriado para receber e armazenar valores de temperatura de cada um dos rolos, permitindo acompanhar seu histórico. A Plataforma de Inspeção também pode recorrer ao PIMS para buscar dados, por exemplo, a taxa de operação de um TC, visto que esta influencia no valor de temperatura dos rolos.

Os protótipos desenvolvidos procuraram demonstrar a interação da plataforma de inspeção com o **Sistema de Manutenção**, já em uso no porto usado como piloto, e com o **Sistema de Controle da Inspeção**, que foi desenvolvido também como um protótipo. Mais detalhes são discutidos na subseção "**5.4.1 - Protótipo 1 - Aplicativo Móvel e Sistema de Controle da Inspeção**".

### 5.2.6 *Insights* e Conhecimento

Além de melhorar o processo de monitoramento em si, a incorporação da Plataforma de Inspeção aos processos de manutenção tem como consequência positiva a geração de dados de condição dos rolos até então inexistentes. A disponibilização desses dados em um repositório apropriado, de forma combinada com informações oriundas de outros sistemas, habilita a realização de análises avançadas, que podem retroalimentar a inspeção de rolos e, portanto, gerar *insights* e conhecimento sobre o processo.

Desse modo, o primeiro objetivo dessa camada é prover uma forma de armazenamento apropriada para receber grandes quantidades de dados gerados pela Plataforma de Inspeção e por demais sistemas. Geralmente, utiliza-se um *Data Lake* definido por Stein e Morrison (2014) como um repositório de baixo custo, apropriado para armazenamento na ordem de petabytes, capaz de preservar os dados em sua forma original (sem agregações, transformações, modelagem, etc.) e que, por conta disso, consegue receber dados estruturados (bancos de dados, arquivos XML, etc.) e não estruturados (arquivos diversos, vídeos, etc.) das mais diversas origens.

Por si só, o uso das técnicas de integração discutidas para a ingestão dos dados em um *Data Lake* tem efeitos práticos nulos. Por conta disso, o segundo objetivo

dessa camada é prover ferramentas e poder computacional para que os Cientistas de Dados[1] consigam explorar a relação dos dados armazenados no *Data Lake* e, assim, extrair valor deles. Isso pode ser feito por meio de ferramentas diversas, como *softwares* de análise estatística, algoritmos de *machine learning* e desenvolvimentos customizados em linguagens de programação como Python e R, populares para este tipo de uso.

Tais análises podem ter resultados diversos. Por exemplo, a avaliação de quais fornecedores possuem rolos com menor vida útil, cruzando dados de compras, obtidos do Sistema de Gestão Empresarial (ERP, do inglês *Enterprise Resource Planning*), com os defeitos registrados pela Plataforma de Inspeção. Outros resultados podem ser melhores algoritmos para análise de defeito dos rolos, que poderiam ser incorporados aos componentes da camada de **Extração de Dados de Sensores** da plataforma. Assim, fecha-se o ciclo da Arquitetura Integrada, em que camadas de diferentes níveis colaboram entre si para que o processo de inspeção de rolos seja aprimorado.

Ainda que seja de grande importância, esta camada não foi avaliada nos protótipos por necessitar de uma grande quantidade de dados para análises, algo inviável no horizonte de realização deste trabalho. Entretanto, a empresa onde os protótipos foram testados já possui um *Data Lake* e mecanismos estabelecidos para sua alimentação, o que simplifica a ingestão de dados gerados pela plataforma. Por fim, não é escopo deste trabalho a ingestão de dados ou sinais brutos capturados pela plataforma (arquivos de vídeo, áudio, etc). Ainda que o *Data Lake* seja apropriado a recebê-los, é necessário discutir e enriquecer a arquitetura com protocolos de *streaming* ou transmissão de arquivos aptos a este objetivo.

## 5.3 Evolução da Plataforma de Inspeção

Para suportar a modularidade e evolução incremental da Plataforma de Inspeção, deve-se permitir agregar ou substituir componentes sem alterar de forma significativa as demais camadas da solução. Se isso não é feito, o acoplamento entre as camadas acaba tornando complexa ou mesmo impedindo a incorporação de melhorias, como novos algoritmos, melhores sensores ou outros tipos de portadores.

Essa abordagem não é muito diferente da discutida para a integração entre sistemas, usando orientação a serviços (SOA) e interações do tipo *Publish-Subscribe*.

[1] Profissional que une profundo conhecimento do negócio, matemática, estatística e de ferramentas de programação para extrair conhecimento e informações relevantes de grandes quantidades de dados

Porém, além de ser usada entre a Plataforma de Inspeção e os Sistemas Corporativos, ela também é aplicável a elementos internos da plataforma. Assim, a Figura 59 demonstra este conceito, que relaciona as camadas da Plataforma de Inspeção com uma implementação possível baseada no *framework* de robótica ROS.

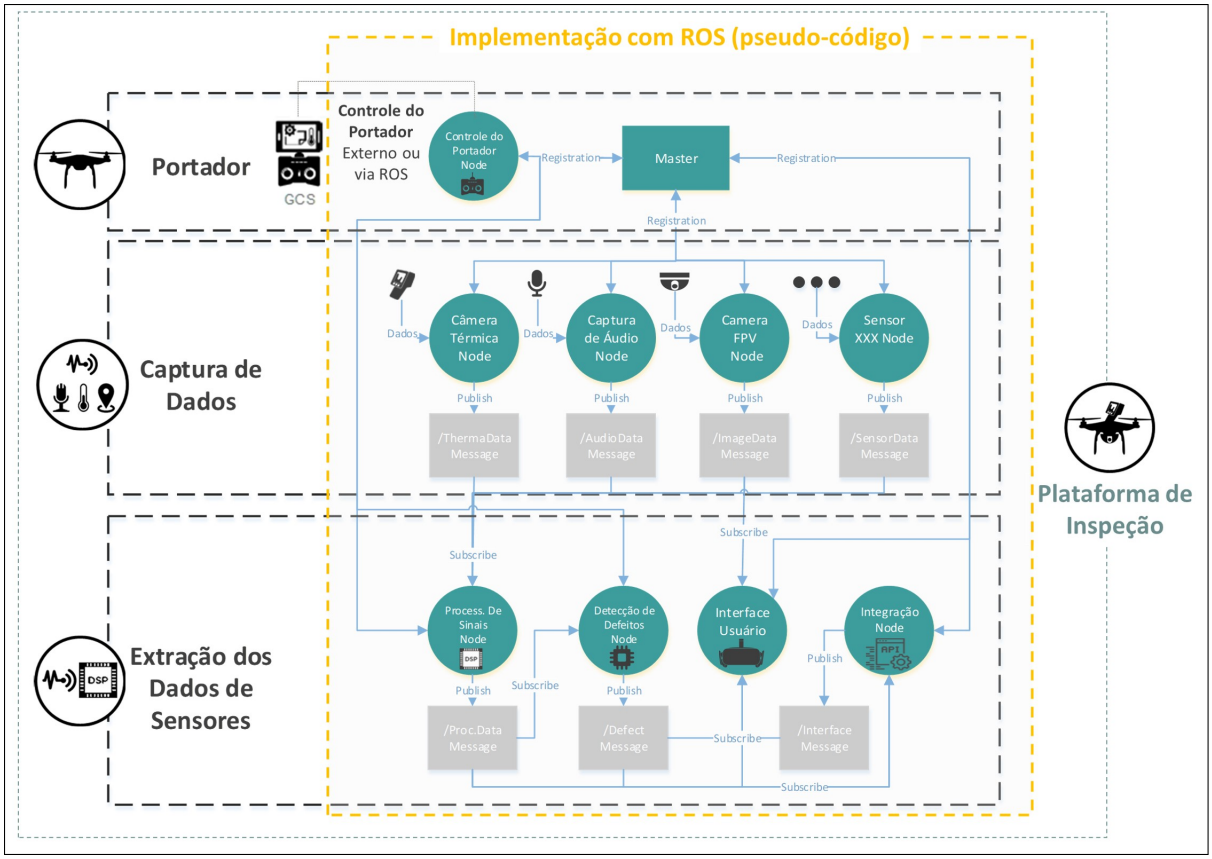



Figura 59 – Relação das camadas da Plataforma de Inspeção com uma implementação conceitual. Fonte: autor

Os elementos classificados como “nós” (*nodes*) são os responsáveis por realizar o processamento ou a comunicação em baixo nível com elementos de *hardware*. Seguindo esse conceito, a substituição de um desses elementos, como a câmera térmica da camada de Captura de Dados, implica na alteração apenas do nó específico dela. Desse modo, desde que mantida a mensagem que esse nó gera, a substituição da câmera por um modelo mais moderno ou de outro fabricante poderia ser transparente para o algoritmo de processamento de imagens. Essa lógica também vale para nós referentes ao processamento de sinais, na camada de Extração dos Dados de Sensores. Por exemplo, caso seja desenvolvido um algoritmo mais robusto de detecção de falhas por áudio, apenas esse nó é alterado, sem que o restante da solução seja impactado.

A troca de mensagens entre os nós ocorre primariamente pelo padrão de mensagem *Publish-Subscribe*. Um nó publica suas mensagens em um Tópico e um ou vários nós interessados na informação inscrevem-se para receber atualizações do

tema. Como já discutido na subseção "**4.3.5 - Comunicação no Padrão *Publish-Subscribe***", a maior vantagem dessa abordagem é o desacoplamento, exatamente o que está sendo buscado para este caso de uso.

Vale destacar que não é necessário que toda solução seja desenvolvida usando o *framework* em questão, seja o ROS ou outro qualquer. Por exemplo, o controle do Portador poderia ser feito, opcionalmente, por mecanismos externos, como a GCS da fabricante, como é o caso do VANT DJI Matrice 600 (DJI, 2017b). Isso traz flexibilidade e permite o uso de diferentes portadores na Plataforma de Inspeção. Assumindo um cenário em que as camadas de Captura de Dados e Extração dos Dados de Sensores sejam autocontidas e desenvolvidas no ROS, essa parte poderia simplesmente ser movida de um portador a outro, sem que a solução de obtenção e processamento de sinais tenha que ser reescrita.

Assim, esta seção apresentou a viabilidade de evolução incremental e modular da plataforma, com um exemplo simplificado de implementação em ROS, um dos *frameworks* disponíveis. O presente trabalho não realizou e testou uma implementação completa deste tipo, visto que optou-se pela utilização de portadores e sensores comerciais para os protótipos, detalhados na sequência.

## 5.4 Protótipos Desenvolvidos

A metodologia proposta para o presente trabalho propõe a prototipação como uma forma de validação de alguns aspectos da **Arquitetura Integrada**, proposta no presente trabalho e descrita na seção "**5.2 - Arquitetura Integrada**". Por conta disso, as subseções a seguir apresentam os protótipos que foram desenvolvidos e testados. O texto foi organizado da seguinte forma:

- A subseção "**5.4.1 - Protótipo 1 - Aplicativo Móvel e Sistema de Controle da Inspeção**" descreve o protótipo que procura validar interações do tipo *Publish-Subscribe* da **Plataforma de Inspeção** com **Sistemas Corporativos**, além da automação das inspeções por meio de rotas predefinidas;
- Na sequência, a subseção "**5.4.2 - Protótipo 2 - Envio de defeitos da UPE para o Sistema de Manutenção**" apresenta um caso de uso do tipo *Request-Response*, ao descrever o envio de notas de manutenção da **Plataforma de Inspeção** para o **Sistema de Manutenção**, um dos **Sistemas Corporativos**.

### 5.4.1 Protótipo 1 - Aplicativo Móvel e Sistema de Controle da Inspeção

Como uma forma de validar o fluxo de dados de **Sensores e Telemetria** entre a **Plataforma de Inspeção** e os **Sistemas Corporativos**, passando pela camada de **Integração**, foi desenvolvido um primeiro protótipo que permite a troca de mensagens no padrão *Publish-Subscribe*. Este protótipo foi desenvolvido com o objetivo de suportar os seguintes requisitos funcionais:

1. Habilitar o envio de dados de sensores e telemetrias da Plataforma de Inspeção para o Sistema de Controle da Inspeção;
2. Fornecer para usuários da sala de controle uma aplicação para monitoramento remoto da Plataforma de Inspeção;
3. Fornecer ao inspetor um método automatizado para controle de um portador do tipo VANT por meio da criação de rotas de inspeção.

Para tanto, o protótipo foi desenvolvido em três partes principais, que se integram e trabalham de forma conjunta, conforme é apresentado nas três subseções a seguir. Os testes e resultados deste protótipo são discutidos posteriormente, na seção "**6.1 - Protótipo 1**".

#### 5.4.1.1 Aplicativo Móvel - Missões por *Waypoint*

As inspeções dos rolos são atividades recorrentes, executadas seguindo rotas predefinidas e com frequências específicas. Por conta disso, com a adoção da Plataforma de Inspeção nessa atividade, percebe-se que a definição de rotas de modo que o portador consiga seguir os pontos estabelecidos pode acelerar o processo e reduzir o esforço de controle do inspetor, que poderá dedicar-se à atividade de inspeção em si.

Tendo como base o VANT como portador, dado o foco inicial de inspeção de TCs em grandes áreas abertas, a primeira parte do protótipo desenvolvido consiste de uma aplicação móvel em Android, que utilizou a *Mobile SDK* da DJI, apresentada na subseção "**4.4.1.1 - Estrutura da *DJI Mobile SDK***", para comunicar-se com VANTs da fabricante. Ela permite ao inspetor definir diversos pontos (*waypoints*) em um mapa que formam uma rota de inspeção, que, durante a execução, é seguida pelo VANT de forma automática. Conforme mostrado na Figura 60, a aplicação desenvolvida possui uma interface customizada, que apresenta dados de telemetria e controle de voo do VANT, a visão da câmera FPV, um mapa da região onde o

VANT se encontra e ícones diversos que permitem ao inspetor criar a rota de inspeção, carregá-la no VANT, iniciar e interromper uma missão. O ícone do VANT representa o seu status (verde ligado, vermelho desligado), a sua posição atual e sua orientação, ou seja, ele repete no aplicativo móvel, em tempo real, os movimentos realizados pelo VANT.

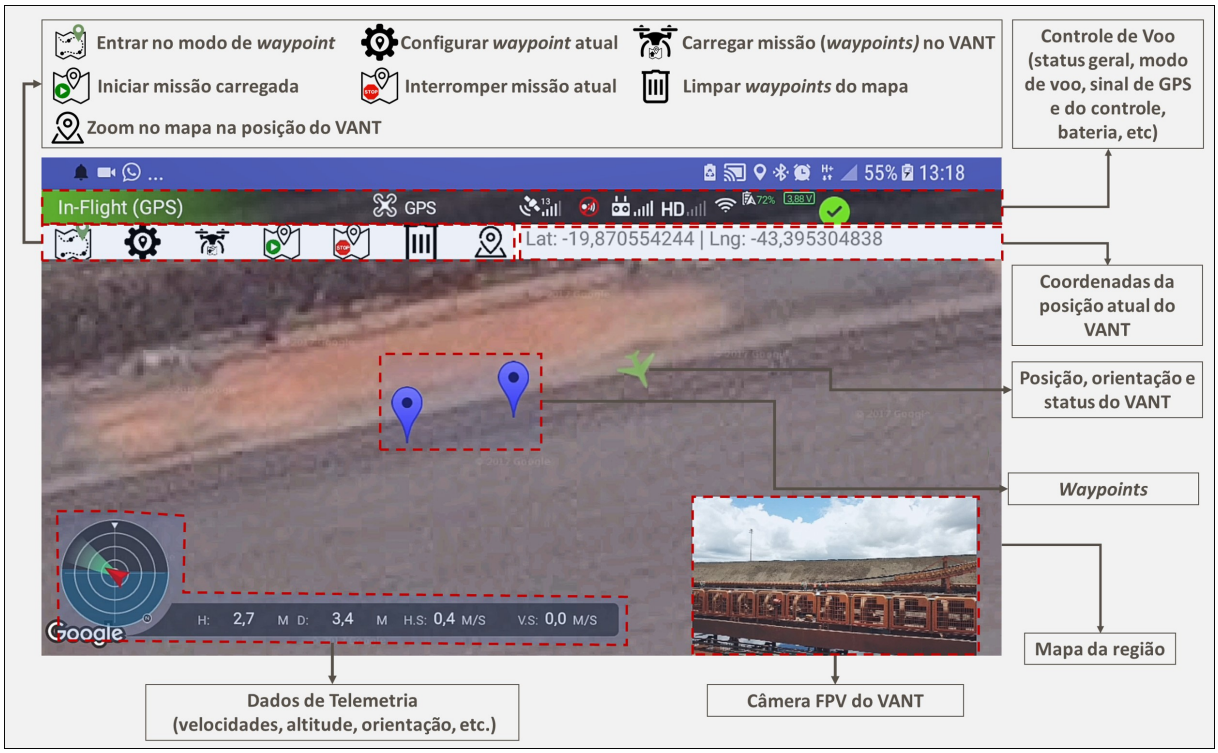



Figura 60 – Visão geral da aplicação Android para controle do portador. Fonte: autor

Para cada um dos *waypoints* definidos, o inspetor pode configurar a altitude, velocidade, a direção de voo do VANT e qual a ação após chegar ao *waypoint* (pousar, retornar ao início, etc.). As configurações possíveis são mostradas na Figura 61 e, com base nelas, é possível adequar a missão em questão a diferentes tipos de situações encontradas em campo.

Além do controle do Portador e interface de usuário, que são duas das funções do protótipo em questão, ele também atua nas camadas de "**Captura de Dados**" e "**Extração de Dados dos Sensores**". No primeiro caso, são capturados os dados de telemetria de voo (status dos motores, velocidade de voo, etc.) e dados de sensores internos (câmera FPV, altitude, latitude, longitude, arfagem, desvio e rolagem). Já a extração desses dados é feita por um processo em segundo plano na aplicação Android que recupera, a cada um segundo, os dados dos sensores por meio de métodos disponíveis na SDK da DJI. Uma vez recuperados e formatados, papel da **Unidade de Processamento Móvel (UPM)**, os dados são publicados via MQTT em tópicos disponíveis em um *broker*. Isso habilita que aplicações diversas acessem e


Figura 61 – Configuração para cada um dos *waypoints*. Fonte: autor

usem os dados que estão sendo publicados, acompanhando as missões do VANT, conforme é discutido nas duas seções subsequentes.

#### 5.4.1.2 Integração - Implementação MQTT

Conforme descrito na seção anterior, os dados de telemetria e sensores podem ser do interesse de diferentes aplicações envolvidas no processo de inspeção. A forma mais eficiente de publicação dos dados, conforme discutido em detalhes na subseção "**4.3.5 - Comunicação no Padrão *Publish-Subscribe***", é usando o padrão de mensagem *Publish-Subscribe*. Neste caso, a Plataforma de Inspeção atua como o gerador dos dados, ou seja, *Publisher*, enquanto os sistemas interessados inscrevem-se para recebê-los. Logo, eles atuam como *Subscribers*.

Como discutido na subseção "**4.3.5.4 - Discussão Sobre Protocolos de Mensageria**", o protocolo MQTT foi escolhido neste trabalho para suportar a construção de integrações *Publish-Subscribe*. No caso de uso específico, foi utilizado um *broker online* especializado no protocolo MQTT, conhecido como *HiveMQ*, mas qualquer tipo de *broker* com suporte ao protocolo em questão poderia ser adotado sem maiores problemas, desde que exista conectividade entre os elementos envolvidos (*Publisher -> Broker -> Subscribers*).

Não foi necessário realizar nenhum tipo de configuração no *broker* em si, visto que todos os parâmetros necessários são definidos pelas aplicações nas pontas no momento de estabelecimento da conexão. A única configuração relevante é a qualidade de serviço (QoS), que é configurado tanto na aplicação *Publisher* quanto nas aplicações *Subscribers*. Vale relembrar que o MQTT suporta 3 níveis, que equilibram velocidade *versus* confiabilidade na entrega. Assim, como a velocidade não é o requisito mais crítico, foi adotado um **QoS = 2** na entrega do dado, o que significa

que as mensagens sempre são entregues no máximo uma vez, correspondendo ao mais alto nível de confiabilidade.

Por fim, vale ressaltar que o padrão de mensagem *Publish-Subscribe* usando o protocolo MQTT é aplicável quando os dados em questão não envolvem publicação de vídeo, áudio ou imagens, ou seja, dados brutos e não tratados. Sua aplicabilidade é factível apenas quando os dados já foram processados e tornam-se mensagens de tamanhos reduzidos, que é o caso de uso em questão neste protótipo.

#### 5.4.1.3 Sistema de Controle da Inspeção

Com objetivo de prover acompanhamento de forma remota e em tempo real das atividades envolvendo a Plataforma de Inspeção, foi desenvolvido uma aplicação *WEB* utilizando Angular. A mesma situa-se na camada de **Sistemas Corporativos** e é apenas um protótipo, logo, não foram desenvolvidas todas funcionalidades esperadas para uma aplicação desse tipo, apenas um dos casos de uso fundamentais para suportar a validação da Arquitetura Integrada.

Isto posto, o sistema em questão tem como objetivo permitir que usuários numa sala de controle consigam acompanhar, remotamente, diferentes aspectos relativos à inspeção de rolos de TCs usando a Plataforma de Inspeção. Por exemplo, a posição atual da plataforma e alguns dados de telemetria e sensores, como velocidade, status dos motores, etc. A captura e publicação destes dados é realizada pela aplicação móvel, que atua como um *Publisher*. Já o **Sistema de Controle da Inspeção** funciona como um *Subscriber*, que inscreve-se nos tópicos disponíveis no *broker* MQTT e, assim, recebe as atualizações da Plataforma de Inspeção à medida que são disponibilizadas, praticamente em tempo real.

A Figura 62 apresenta a interface da aplicação. A parte superior tem em destaque um mapa da área onde a plataforma está operando. Ele é atualizado à medida que novas informações de posição são enviadas pela Plataforma de Inspeção. A posição e demais dados de telemetria são apresentados em formato de lista na parte inferior esquerda da tela. Foram adotadas cores em alguns casos, como a indicação em verde representando se os motores do VANT estão armados ou em vermelho se o mesmo encontra-se no chão.

Por fim, além de apresentar dados recebidos da Plataforma de Inspeção, a ideia é que a aplicação também possa enviar comandos ou mensagens a ela. Isso é representado pela barra ao centro da tela, onde consta uma opção para impedir a decolagem e um campo para envio de mensagens de texto. Neste caso, que não foi desenvolvido no protótipo, ressalta-se a importância de usuários da Sala de Controle conseguirem interagir com o inspetor ou com a Plataforma de Inspe-

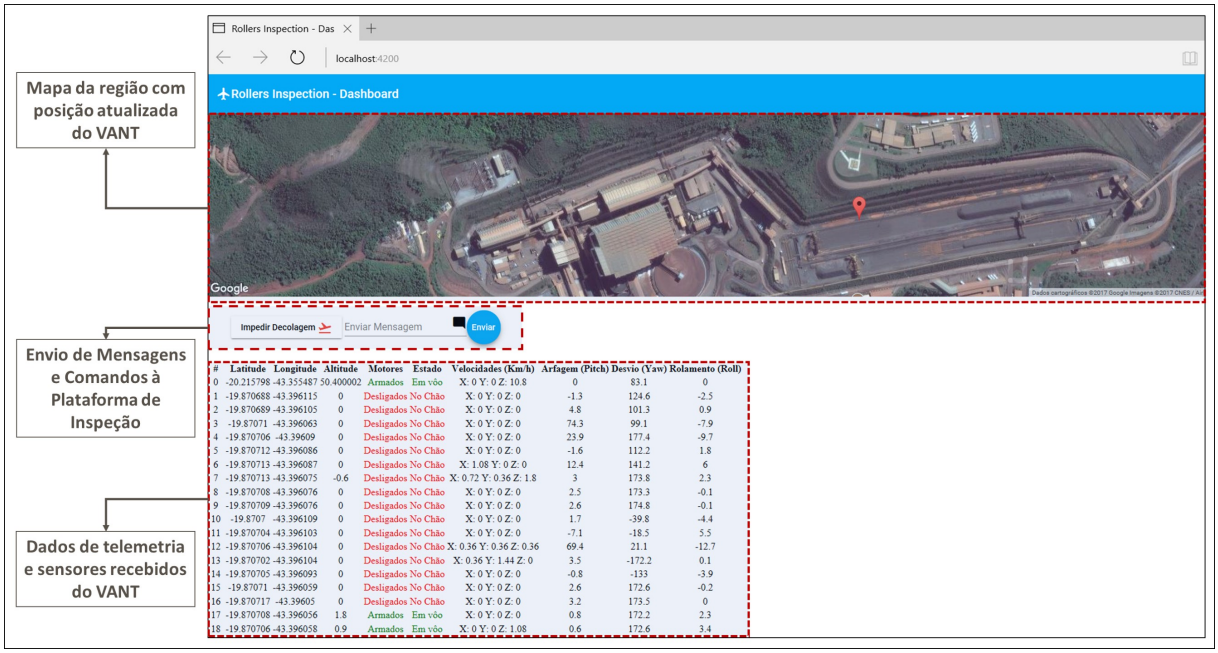



Figura 62 – Visão geral do Sistema de Controle da Inspeção. Fonte: autor

ção diretamente, seja por meio do envio de alertas, mensagens de texto e mesmo configurações atualizadas para a plataforma.

Um caso de uso possível para este último item refere-se à calibração ou configuração remota dos sensores da plataforma. Por exemplo, a termografia é afetada por diferentes parâmetros, dentre eles condições ambientais, como temperatura ambiente e umidade relativa do ar (RAMIREZ-NUNEZ et al., 2016), e condições operacionais, como taxa de operação do TC inspecionado. O **Sistema de Controle de Inspeção** poderia monitorar esses fatores e, remotamente, enviar atualizações para que a Plataforma de Inspeção reconfigure automaticamente a câmera térmica. Isso também pode ser feito para outros tipos de sensores para os quais o inspetor em campo não possui todas as informações necessárias.

É evidente que a versão atual da aplicação é um protótipo ainda limitado e, dessa forma, foi modelada apenas para Plataformas de Inspeção baseadas em portadores do tipo VANT. Entretanto, conceitualmente, a mesma é válida para qualquer arranjo da plataforma, desde que a mesma possua sensores capazes de capturar sua posição, equipamento de comunicação embarcado e conectividade para enviar e receber atualizações para um *Broker*.

### 5.4.2 Protótipo 2 - Envio de defeitos da UPE para o Sistema de Manutenção

Conforme discutido na seção “**5.2.3 - Extração dos Dados de Sensores**”, nem sempre é possível processar todos os defeitos em campo, diretamente pelos equi-

pamentos embarcados ou transportados pelo portador. Isso pode ocorrer por conta do requisito de processamento, incompatível com o disponível na **Unidade de Processamento Móvel (UPM)**, ou por ausência de conectividade da plataforma com redes móveis ou corporativas. Por conta disso, a **Unidade de Processamento Externa (UPE)** é responsável por realizar o processamento que não pôde ser executado na UPM. Ainda que, fisicamente, esteja separada dos componentes operando em campo, a UPE é considerada uma parte integrante da Plataforma de Inspeção.

O processamento automático de defeitos é de extrema importância, já que é a partir dele que o processo de inspeção e monitoramento dos rolos com uma Plataforma de Inspeção móvel passa a ser viável. Como resultado deste processamento, os defeitos identificados devem ser enviados para o Sistema de Manutenção (ou CMMS, do inglês *Computerized Management Maintenance System*). Ele é responsável por todo o processo de gestão da manutenção, que envolve as fases de Planejamento, Programação e Execução, onde as necessidades são registradas em notas e ordens. A partir delas, as equipes responsáveis programam a execução de serviços corretivos ou preventivos, incluindo a substituição de rolos defeituosos. Diante disso, é essencial que defeitos identificados em campo pela Plataforma de Inspeção sejam reportados e registrados no CMMS o quanto antes, com informações precisas sobre o rolo defeituoso, permitindo a continuidade dos procedimentos para reparo.

Desta forma, com o objetivo de validar a integração da UPE com o Sistema de Manutenção, foi desenvolvida uma integração utilizando o padrão de interação *Request-Response*, que permite a criação de **Notas de Manutenção** no CMMS a partir do processamento realizado pela UPE. Assim, o presente protótipo foi desenvolvido para suportar os seguintes requisitos funcionais:

1. Permitir o envio de defeitos identificados pela Plataforma de Inspeção para o CMMS, usando serviço já existente;
2. Habilitar a comunicação entre a Plataforma de Inspeção e o CMMS de forma independente de conectividade, ou seja, não restringir a comunicação à rede corporativa;
3. Permitir que o componente de envio possa ser reaproveitado pela UPM ou mesmo por outras aplicações.

As próximas subseções apresentam um descritivo dos componentes principais. Já a seção “**6.2 - Protótipo 2**” descreve os testes realizados e discute seus resultados.

#### 5.4.2.1 Componentes da Unidade de Processamento Externa

A primeira parte da UPE para este protótipo é um conjunto de algoritmos de processamento que permite identificar defeitos em imagens térmicas de rolos de transportadores de correia. Os algoritmos não serão detalhados, pois fazem parte do trabalho de outro autor, mas de forma resumida, eles são executados no MATLAB e conseguem ler arquivos de imagens em um diretório local. A partir disso, aplicam um método para identificar os rolos nas imagens (regiões de interesse) e obter o valor de temperatura apenas nessas regiões para cada um dos rolos encontrados. As temperatura é comparada com limites pré-estabelecidos, onde avalia-se se representam situações normais, defeitos em andamento ou falhas críticas. Finalmente, quando é identificado algum problema, a posição GNSS registrada no arquivo de imagem é avaliada para identificar em um mapa qual a posição do defeito.

Assim, a saída dos algoritmos em questão é a informação de cada um dos defeitos, o que inclui a temperatura registrada e a posição do rolo defeituoso (latitude e longitude). Tendo como base essas informações de entrada, foi desenvolvida uma função na linguagem *Python* que permite formatar uma mensagem em JSON e enviá-la para uma API REST, desenvolvida para suportar a criação de notas de manutenção no CMMS, melhor explicada na próxima subseção. Além de encapsular detalhes da integração com o CMMS, como a URI a ser chamada, chave de acesso e outras questões, esta função é importante pois também realiza tratamento dos dados, complementando as informações e permitindo que o algoritmo no MATLAB resuma-se a identificar os defeitos e acioná-la. Assim, ela se encarrega da comunicação com a API REST, retornando ao algoritmo chamador o status de processamento, seja ele o número da nota de manutenção criada ou uma mensagem de erro do CMMS ou dos componentes da integração.

É importante destacar que a função desenvolvida em *Python* é independente e, por conta disso, pode ser usada por outras aplicações. Isso permite, por exemplo, incorporá-la diretamente à UPM, caso ela consiga avaliar os defeitos de forma embarcada. Esta abordagem simplifica a evolução da Plataforma de Inspeção, uma vez que uma única função pode ser usada para enviar qualquer tipo de defeito ao CMMS, tendo sido ele identificado pela UPM ou pela UPE, com base em qualquer tipo de sinal (térmico, acústico ou vibração) capturado por algum dos sensores. Isso também reforça os motivadores da escolha da linguagem *Python* para essa função, já que ela é simples, robusta, compatível com alguns *frameworks* de robótica, como o ROS, e facilmente integrável a outras aplicações e pacotes de mercado, como o MATLAB.

Outra questão que facilita o reuso é a própria definição das APIs e *web services* para comunicação com o CMMS. Esse tipo de abordagem, bem como o papel de facilitador realizado pela camada de **Integração**, é explicado na seção a seguir.

#### 5.4.2.2 Componentes de Integração

Conforme explicado na subseção "**5.2.4 - Integração**", quanto maior a complexidade embutida nesta camada, mais simples se torna a comunicação entre as aplicações e sua integração. Parte disso vem dos conceitos de Arquitetura Orientada a Serviços (SOA), apresentados na subseção "**4.3.1 - Arquitetura Orientada a Serviços**", onde é possível expor funções de negócios por meio de serviços que podem ser reusados por diferentes aplicações. A outra parte que justifica a afirmação anterior é que a camada de integração, por meio de Barramentos de Integração (*Brokers* ou *Enterprise Service Bus* - ESB), traz pra si a responsabilidade de absorver diferenças entre as aplicações, como linguagem de programação, protocolos de comunicação distintos e mesmo algumas questões de conectividade.

Dessa forma, a Figura 63 apresenta uma visão geral do protótipo. Do lado esquerdo, encontra-se a camada de **Extração de Dados dos Sensores**, com a UPE desempenhando o papel descrito na subseção anterior de detecção de defeitos e acionamento de uma API para envio. No lado direito, está posicionado o **CMMS**, que é o Sistema Corporativo que necessita receber os defeitos para dar prosseguimento ao processo de manutenção. Ele possui um serviço de **Criação de Notas de Manutenção**, que permite que outros sistemas reportem defeitos.

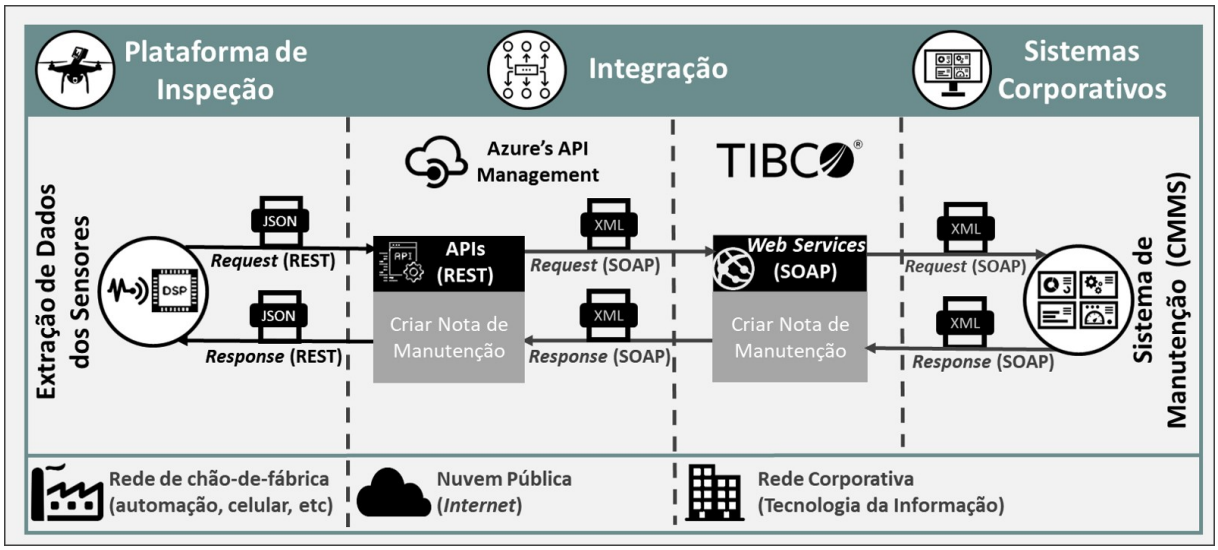



Figura 63 – Papel da camada de Integração para viabilizar a conectividade da Plataforma de Inspeção e conversão REST x SOAP. Fonte: autor

Ainda com base na Figura 63, percebe-se que a camada de **Integração**, ao cen-

tro, é composta por duas ferramentas distintas que se combinam para endereçar aspectos inicialmente incompatíveis entre as pontas. O primeiro deles é a conectividade, uma vez que a Plataforma de Inspeção está em campo e, muitas vezes, em áreas sem cobertura da rede corporativa de TI. O segundo aspecto refere-se à adaptação de protocolos para viabilizar a comunicação. O serviço já existente no CMMS utiliza o protocolo SOAP, que é muito comum na comunicação entre aplicações corporativas, dada sua robustez. Por sua vez, a Plataforma de Inspeção, é baseada no estilo de arquitetura REST, mais simples e leve, que utiliza o protocolo HTTP para transporte. Assim, cabe às ferramentas da camada de **Integração** cuidar destes aspectos e viabilizar a conectividade e adaptação de protocolos.

Uma destas ferramentas, o *TIBCO Business Works*[2], é utilizado na empresa para expor os *web services* SOAP, promovendo o reuso e facilitando a administração, além de outros aspectos já abordados na seção "**4.3.2 - Barramento de Integração**". Dessa forma, os principais serviços disponíveis nos **Sistemas Corporativos** são expostos neste Barramento de Integração para que possam ser consumidos por outras aplicações. Porém, o *TIBCO Business Works* está localizado dentro da rede corporativa de TI e, atualmente, sua comunicação com aplicações externas, via internet, é limitada ao protocolo SOAP.

Por conta disso, uma segunda ferramenta foi utilizada, o *API Management*[3], disponível na plataforma de computação em nuvem da Microsoft, a *Azure*. Além de ser acessível via internet, essa ferramenta possui conectividade com a rede corporativa de TI por meio de uma rede privada (VPN, do inglês *Virtual Private Network*), o que viabiliza a comunicação de aplicações externas com sistemas internos. Isso é feito por meio de APIs, que tornam acessíveis serviços publicados em um Barramento de Integração, como o *TIBCO Business Works*, ou diretamente pelos Sistemas Corporativos.

Tendo como base o *Azure's API Management*, foi criada uma nova API em REST para expor o serviço (ou *web service*) já existente para **Criação de Notas de Manutenção**. Esta ferramenta permite ainda converter o protocolo SOAP do serviço original para o estilo REST. Assim, a Figura 64 apresenta, em alto nível, como essa conversão é feita. Na parte superior, é possível ver que a URI do Recurso em REST é convertida em uma URL da operação SOAP, parte do cabeçalho da mensagem. Além disso, como o *web service* SOAP existente é baseado em XML, na figura é possível perceber algumas etapas que reescrevem o corpo da mensagem (*body*) de JSON para XML. Por fim, na parte inferior, é realizado o processo reverso, onde

2 https://www.tibco.com/products/tibco-businessworks

3 https://docs.microsoft.com/en-us/azure/api-management/api-management-key-concepts

a resposta recebida do *web service* é reescrita de XML para JSON.

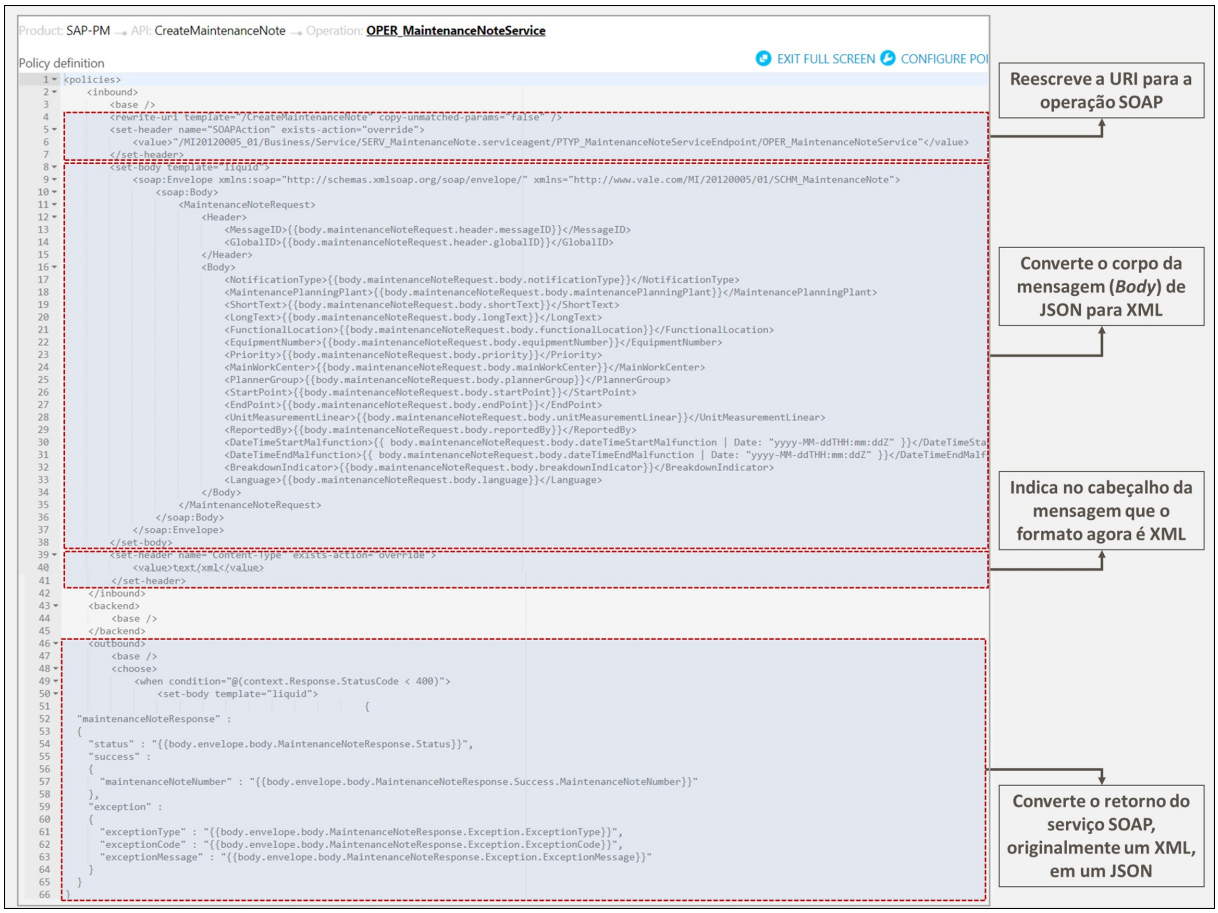



Figura 64 – Macroetapas da conversão de protocolos e de formato (XML x JSON) realizada no *API Management*. Fonte: autor

Assim, com o uso desse ferramental, foi possível construir um processo de integração no estilo *Request-Response* que viabiliza a comunicação da Plataforma de Inspeção com o CMMS para envio dos defeitos identificados. É importante lembrar que essa abordagem vale para qualquer tipo de serviço existente, alterando-se apenas o ferramental envolvido e, evidentemente, os sistemas. Portanto, a subseção "**6.2 - Protótipo 2**" apresenta os testes e discute os resultados deste protótipo.

## 5.5 Simulação

Conforme discutido no capítulo Metodologia, a simulação é uma parte importante do trabalho, particularmente nos casos de uso que envolvem um portador do tipo VANT. Por conta disso, as próximas subseções apresentam como ela foi utilizada. O texto da seção foi organizado da seguinte forma:

- A subseção "**5.5.1 - Simulação de um Ambiente de Porto**" apresenta o ambi-

ente virtual que foi criado para simular algumas das condições encontradas em um porto de materiais a granel;

- Na sequência, a subseção "**5.5.2 - Controle Remoto do VANT e Obtenção de Dados de Sensores**" demonstra um conjunto de aplicações e um esquema de comunicação entre elas para suportar o controle remoto do VANT no simulador;
- Por fim, a subseção "**5.5.3 - *Streaming* de Vídeo do Simulador**" aborda um caso de uso para obtenção de vídeo em tempo real por uma aplicação externa à partir do simulador.

### 5.5.1 Simulação de um Ambiente de Porto

Conforme descrito na seção "**4.4.2.1 - Simulador Microsoft AirSim**", o Microsoft AirSim foi escolhido para ser usado no presente trabalho por conta de sua extensibilidade e capacidade de interação por meio de APIs (*AirLib*), o que habilita o desenvolvimento de aplicações avançadas e a possibilidade de modelagem de ambientes customizados usando a Unreal Engine.

Baseando-se nela, foi desenvolvido um ambiente virtual que procura simular algumas das condições encontradas em um porto, como transportadores de correia, recuperadoras, empilhadeiras e pilhas de material. O porto foi escolhido por ser, na indústria da mineração, a etapa do processo que concentra a maior quantidade e extensão de transportadores de correia, sendo, portanto, um grande candidato para operação da Plataforma de Inspeção. A Figura 65 apresenta uma comparação do ambiente real com o simulado. À esquerda, está uma pequena parte do Pátio de Estocagem do Terminal Marítimo de Ponta da Madeira, enquanto à direita é possível ver uma renderização do ambiente simulado que busca apresentar a essência do ambiente real.

Isto posto, a Figura 66 apresenta o ambiente de desenvolvimento da Unreal Engine, onde é mostrada uma visão aérea de parte do ambiente de porto criado para a simulação. Procurou-se modelar uma pequena parte de um pátio de materiais, que dentro do porto, é uma área aberta com maior possibilidade de aplicação de uma Plataforma de Inspeção baseada em um portador do tipo VANT. Dessa forma, a parte central da imagem mostra uma pilha de materiais, enquanto à esquerda é possível visualizar um transportador de correias e, ao fundo, uma empilhadeira.

Uma vez finalizada a criação do ambiente, é possível executar a simulação e controlar o VANT no ambiente desenvolvido. A Figura 67 mostra a simulação em execução, com o VANT voando ao lado de um transportador de correia. Ainda que

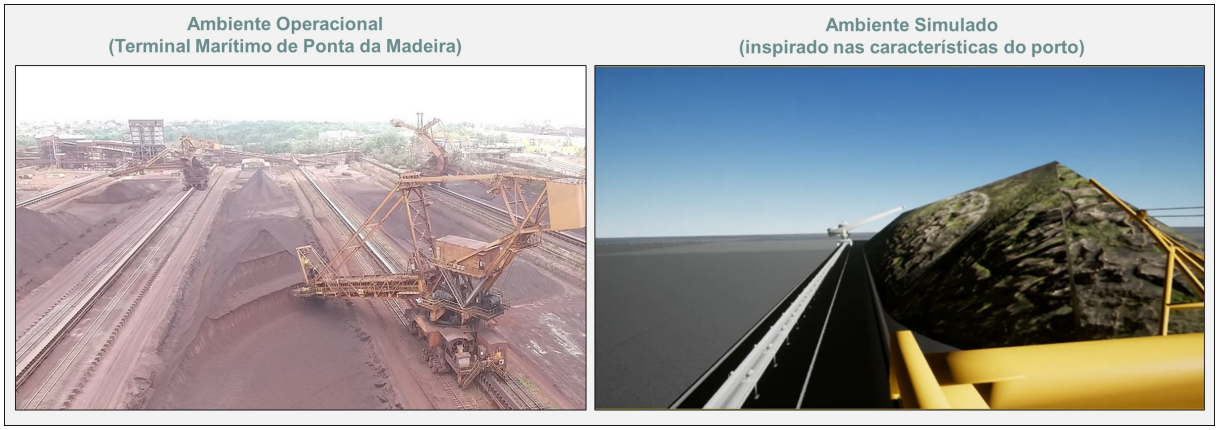



Figura 65 – Comparativo de um ambiente operacional de porto com sua recriação simplificada no simulador. Fonte: autor

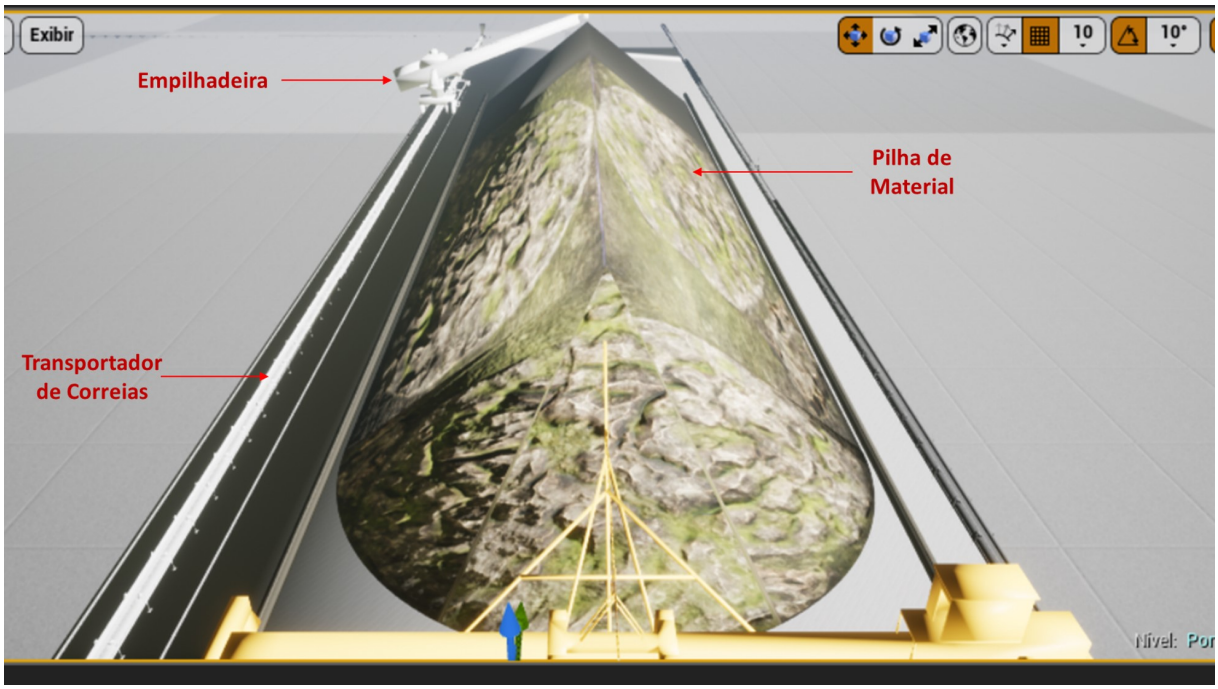



Figura 66 – Interface de desenvolvimento da Unreal Engine mostrando parte do ambiente criado. Fonte: autor

não tenham sido aplicadas texturas de alta definição para melhorar a qualidade gráfica, é possível visualizar os rolos, tanto de carga quanto de retorno, que são os objetos a serem inspecionados. Na parte inferior direita da imagem, é possível também visualizar a câmera FPV do VANT. Outras visualizações em tempo real também poderiam ter sido adicionadas, como segmentação e profundidade, para citar dois dos exemplos possíveis.

Ainda que o foco nesta etapa não tenha sido a criação de modelos fotorrealistas ou fidedignos, a Unreal Engine é capaz de recriar um ambiente de forma tão realista quanto a desejada, mas exige esforço de modelagem 3D. Por conta disso, o modelo criado pode ser evoluído e expandido, permitindo, por exemplo, a simulação

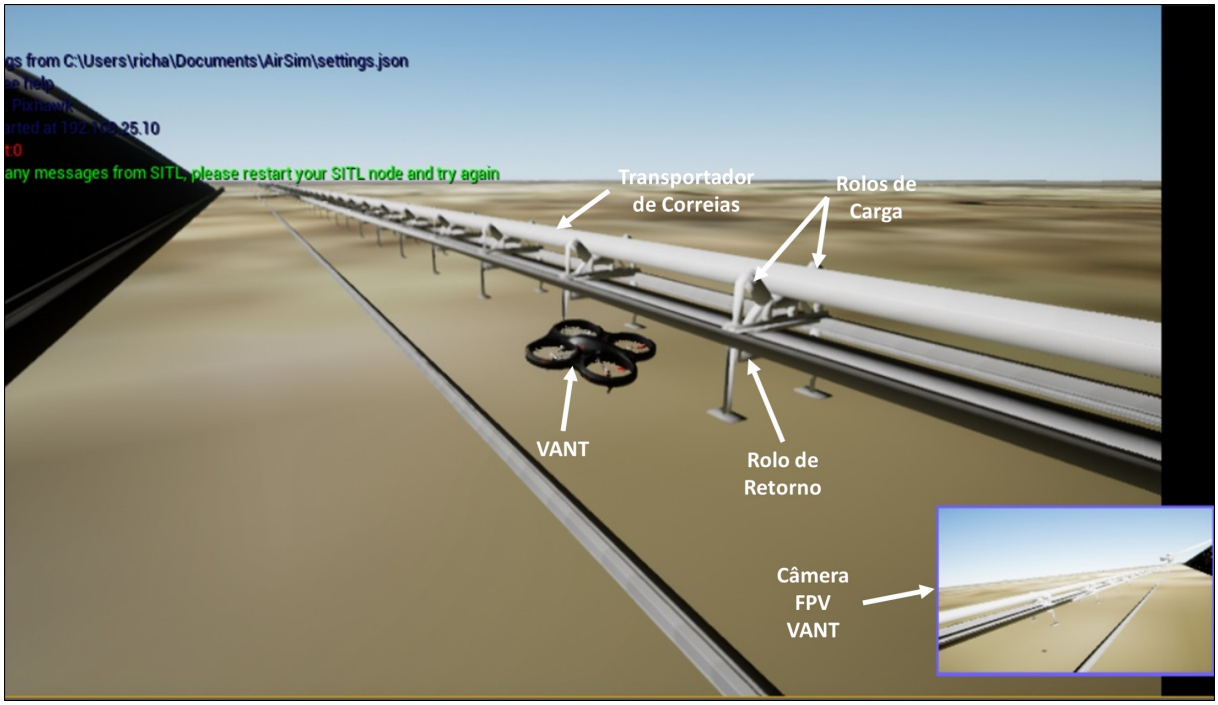



Figura 67 – Simulação com o VANT voando ao lado de um TC. O quadro em destaque é a câmera FPV. Fonte: autor

dinâmica de ventos no porto, a recriação de outras áreas e mesmo a movimentação de ativos e pessoas nos pátios, suportando a coleta de dados para treinar algoritmos para voos autônomos (*reinforcement learning*), caso, futuramente, sejam liberados pela legislação brasileira. Além disso, outra aplicação possível, é o uso do simulador como suporte ao treinamento dos inspetores, dada a introdução de uma nova tecnologia em sua atividade atual. Particularmente no caso de VANTs, ter um contato em ambiente simulado para familiarizar-se com os controles pode ser essencial para que as etapas de capacitação na área operacional sejam mais seguras.

Desta forma, como ponto de partida da avaliação do uso do simulador para este tipo de utilização, foram criados alguns protótipos iniciais, que são descritos na sequência.

### 5.5.2 Controle Remoto do VANT e Obtenção de Dados de Sensores

O AirSim permite diferentes formas de controle do VANT, o que inclui um rádio-controle tradicional, uso de *joysticks* de videogames, como Xbox 360 e Xbox One, e mesmo o controle de alguns fabricantes, como a DJI. Em alguns casos, é necessário realizar a instalação de elementos adicionais para viabilizar a comunicação, como *drivers* e *firmwares*, entretanto, isso não diminui a versatilidade do simulador,

que aceita todas estas possibilidades.

Adicionalmente, o AirSim ainda oferece uma API que permite enviar diversos tipos de instruções para o VANT durante a simulação. Isso vai de comandos simples, como “Decolar” e “Aterrissar”, a instruções complexas, como efetuar o controle e navegação do VANT a partir das imagens do ambiente e dados de seus sensores, como GNSS e atitude (acelerômetro e giroscópio). Dessa forma, por meio de aplicações externas, é possível customizar o controle do VANT no simulador, o que é útil caso as opções de controle oferecidas não atendam.

Dessa forma, o caso de uso desenvolvido procurou explorar dois aspectos do simulador: a coleta de dados e o controle do VANT no ambiente simulado por meio dessa API. Para isso, foram desenvolvidas duas aplicações em *Python*. A primeira delas, daqui em diante chamada **Aplicação de Controle**, possui uma interface gráfica simplificada para envio de comandos pré-estabelecidos ao simulador e visualização de *logs* dos resultados. Por sua vez, a segunda aplicação, chamada ***Handler*** é executada em segundo plano, sem interação com o usuário, tendo como responsabilidade a comunicação com o simulador para receber os comandos e traduzi-los em instruções da API do simulador.

Com base nessa estrutura, o ***Handler*** precisa ser executado em um computador que possua conectividade com a máquina do simulador AirSim, já que necessita estabelecer conexão com ele. Em contrapartida, a **Aplicação de Controle** pode ser executada de qualquer local, sem restrições de conectividade com o AirSim. Isso ocorre porque o protótipo foi montado de modo que os comandos da **Aplicação de Controle** para o ***Handler*** sejam enviados via um *broker* MQTT, com arquitetura *publish-subscribe*. Assim, a aplicação ***Handler*** funciona como um *subscriber* do tópico de **Comandos**, aguardando instruções enviadas por uma aplicação de controle ( *publisher*). A Figura 68 representa essa forma de interação entre as aplicações utilizando os elementos desenvolvidos.

Ainda que numa primeira análise este esquema de comunicação pareça complexo, ele favorece o desacoplamento, permitindo que as pontas evoluam de forma independente. Um exemplo disso é que o ***Handler*** não precisa saber se o comando foi enviado pela Aplicação de Controle desenvolvida ou por outra origem qualquer. Da mesma forma, se a API do AirSim evoluir e tiver métodos alterados, apenas o ***Handler*** precisa ser ajustado, não sendo necessário que as diferentes aplicações de controle sejam também modificadas.

Os testes e resultados deste tipo de controle são apresentados na seção “**6.3 - Testes de Controle e Obtenção de Dados do Simulador**”, enquanto a sequência do texto discute a obtenção de vídeo a partir do simulador.

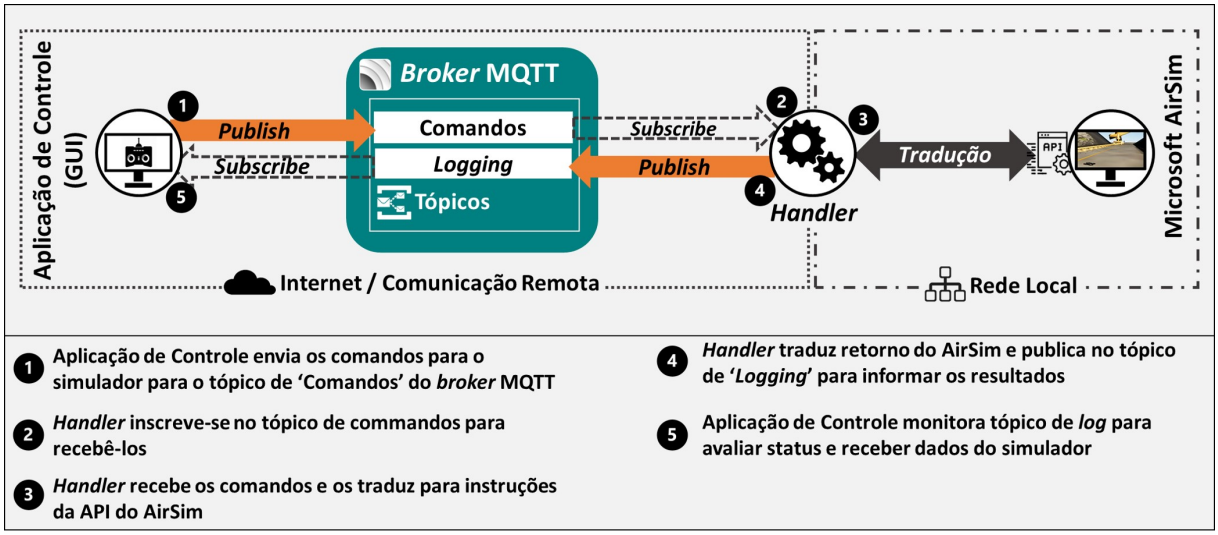



Figura 68 – Comunicação entre a Aplicação de Controle e o Simulador, passando pelo *Handler*. Fonte: autor

### 5.5.3 *Streaming* de Vídeo do Simulador

Tendo como base o ambiente de porto desenvolvido, descrito na subseção "**5.5.1 - Simulação de um Ambiente de Porto**", foi criada uma aplicação em *Python* capaz de comunicar-se com o Microsoft AirSim, obter sequências de imagens na resolução desejada, e enviá-las para dispositivos clientes por meio de uma conexão de rede, utilizando o protocolo UDP. Os clientes podem ser, virtualmente, qualquer dispositivo, como computadores, *tablets*, *smartphones* e mesmo óculos de realidade virtual ou aumentada com processamento embarcado.

A Figura 69 apresenta um esquema desse processo. Por definição, um vídeo é uma sequência de quadros, sendo cada um deles uma imagem. Logo, na aplicação desenvolvida, elas são obtidas do Microsoft AirSim utilizando a API existente, sendo possível capturar variações de imagens do simulador, chamadas "Visualizações", como FPV, segmentação e profundidade. Para cada imagem capturada, é realizado um processo de tratamento, decodificando o formato retornado pelo AirSim em um *array* de *bytes*, redimensionando a imagem para que possa ser transmitida num pacote UDP, e, finalmente, codificando-a em um formato qualquer, por exemplo, "*jpeg*". Por fim, após esse tratamento, cada uma das imagens é transmitida em uma rede local para uma Aplicação de Visualização, que recebe os pacotes UDP, obtém seu conteúdo e o adapta para apresentação de acordo com o tipo de dispositivo de exibição.

Foram feitos alguns testes para validar o funcionamento do esquema apresentado na Figura 69, o que é detalhado na seção "**6.4 - Testes do *Streaming* de Vídeo do Simulador**".

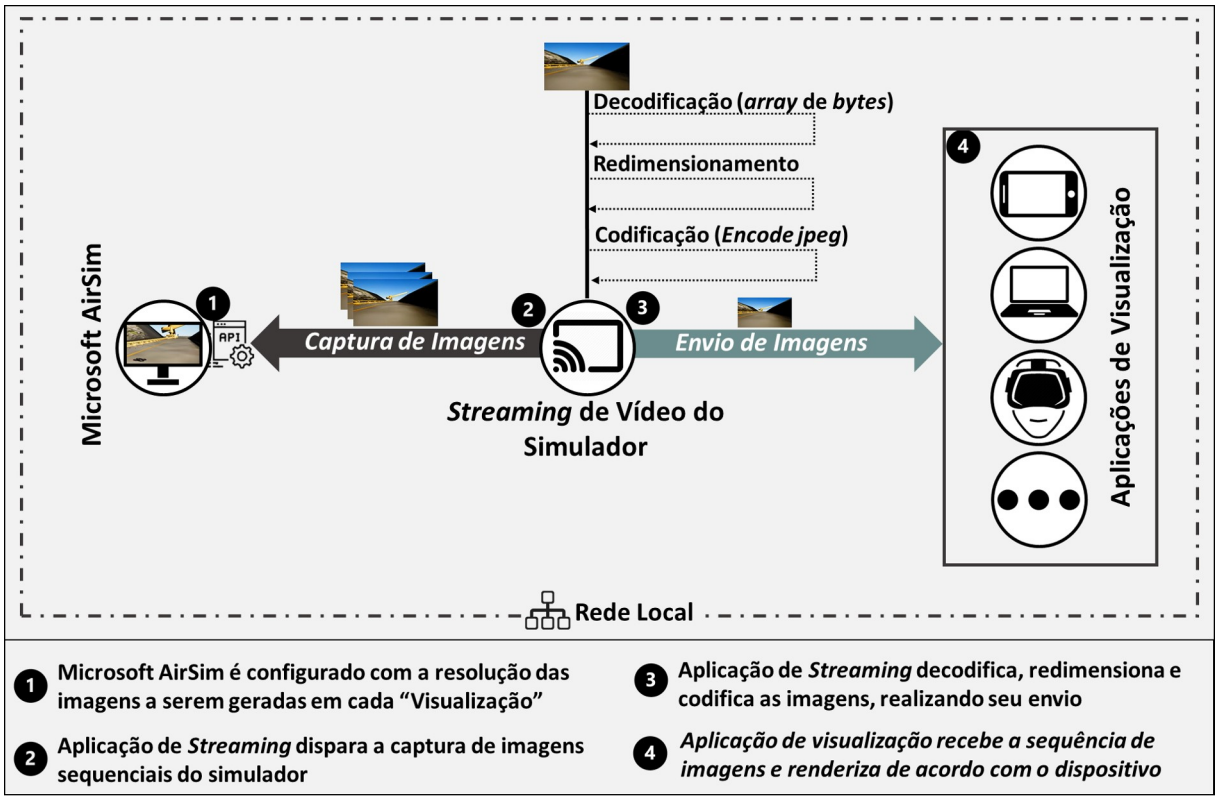



Figura 69 – Esquema de funcionamento da aplicação para captura e *streaming* de vídeo a partir do simulador. Fonte: autor

# 6 Testes e Resultados

A primeira parte deste capítulo descreve os testes realizados para suportar a validação dos principais aspectos da Arquitetura Integrada e discute seus resultados. Para tanto, ela foi organizada da seguinte forma:

- A seção "**6.1 - Protótipo 1**" descreve os testes realizados para validar o fluxo de dados de Sensores e Telemetria entre a Plataforma de Inspeção e Sistemas Corporativos, além do controle do portador usando rotas de inspeção (*waypoints*);
- Já a seção "**6.2 - Protótipo 2**" descreve os testes para envio de defeitos detectados pela Unidade de Processamento Externa (UPE) para o Sistema de Manutenção.

Na sequência, são apresentados os experimentos realizados com o simulador de VANTs, o portador adotado no trabalho. Esta parte foi organizada em duas seções:

- A seção "**6.3 - Testes de Controle e Obtenção de Dados do Simulador**" descreve os testes realizados para avaliar o controle externo do simulador e obter dados de alguns sensores à partir da simulação;
- Por sua vez, a seção "**6.4 - Testes do *Streaming* de Vídeo do Simulador**" descreve os experimentos para captura e transmissão de vídeo do simulador para um aplicativo de realidade virtual.

## 6.1 Protótipo 1

Este protótipo foi detalhado na subseção "**5.4.1 - Protótipo 1 - Aplicativo Móvel e Sistema de Controle da Inspeção**" e a Figura 70 apresenta um resumo, destacando os objetivos e requisitos funcionais que ele deve cumprir. Ela também relaciona os principais componentes utilizados com as respectivas camadas da Arquitetura Integrada. Com exceção da camada de ***Insights* e Conhecimento**, todas as demais participam do protótipo.

Foram realizados diferentes tipos de testes para validar os componentes discutidos nas últimas subseções, tanto de forma individual quanto integrada, visto que o objetivo é a validação do protótipo como um todo, que consiste justamente na

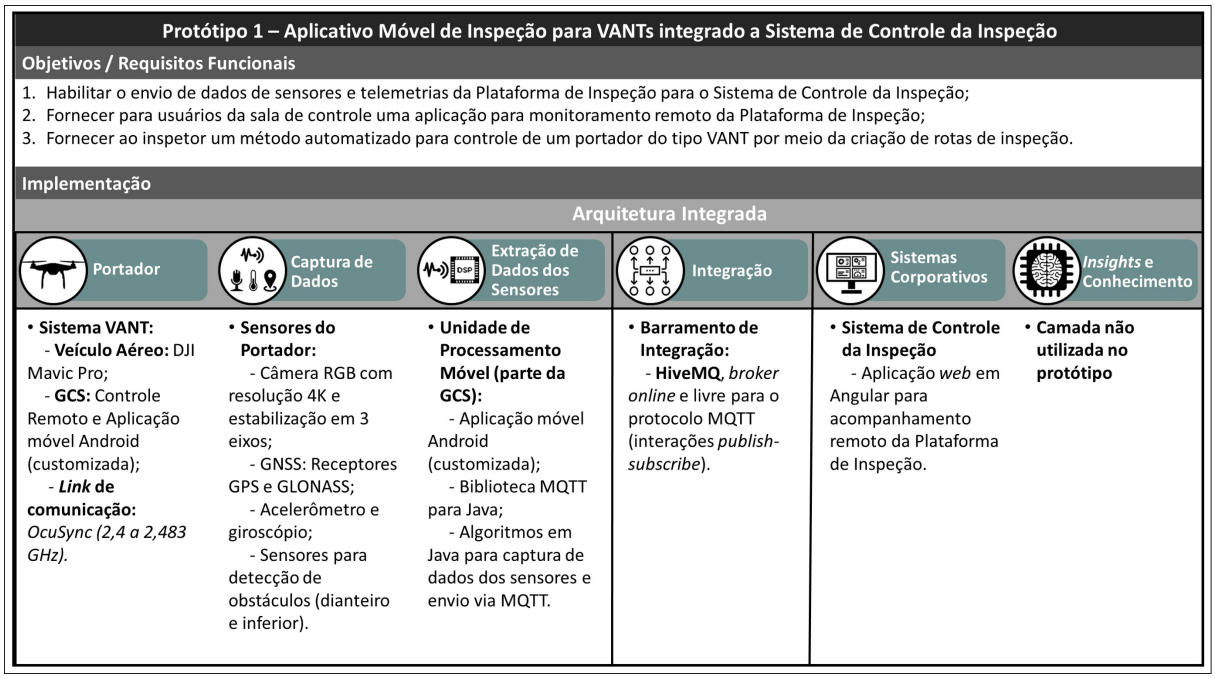



Figura 70 – Principais componentes do primeiro protótipo de acordo com as camadas da Arquitetura Integrada. Fonte: autor

interação entre estes elementos. Tais testes, especialmente os envolvendo o Sistema VANT, foram inicialmente conduzidos em ambientes controlados, visto que eventuais falhas e defeitos tem grande potencial para danificar o equipamento em si e bens de terceiros.

Concluída esta etapa, em que se atingiu um nível suficiente de confiabilidade nas aplicações desenvolvidas, foram executados estes em uma área operacional, ambiente onde de fato a Plataforma de Inspeção irá operar. Ao contrário dos ambientes controlados, usados na fase anterior, uma área operacional apresenta diversas variáveis que estão fora de controle nos testes, como equipamentos móveis, trabalhadores, poeira, projeção de material, ausência de link de comunicação estável, etc., o que, por si só, traz desafios para o seu planejamento e execução.

Dessa forma, a etapa de testes em área operacional foi conduzida na Mina de Brucutu, localizada no município de São Gonçalo do Rio Abaixo. Conforme informações da empresa Vale (2016b), Brucutu foi inaugurada em 2006 e é considerada a mina com maior capacidade inicial de produção do mundo. Ainda segundo a mineradora, é a maior mina de minério de ferro do estado de Minas Gerais e a segunda maior do Brasil, produzindo em torno de 30 milhões de toneladas por ano, o que corresponde, aproximadamente, a 9% de toda produção da empresa.

Por si só, tais características demonstram o porte e a importância da operação escolhida para estes testes. O tamanho da operação se reflete na quantidade de rolos, mais de 50.000, instalados nos vários TCs que transportam o material ao longo das diferentes etapas do processo produtivo, como britagem, peneiramento, homo-

geneização, concentração e expedição. Cada uma dessas etapas possui diferentes características, como áreas abertas (pátios), áreas fechadas (chutes), TCs em altura e outras variações que são bastante representativas dos variados ambientes esperados para operação da Plataforma de Inspeção.

A Figura 71 apresenta uma visão parcial da Mina de Brucutu. As áreas destacadas em laranja são as regiões que concentram a maior extensão de TCs e, por consequência, dos rolos desta operação. Visto que se tratava da primeira execução de testes do protótipo em uma área operacional, priorizou-se a execução em uma região aberta e relativamente livre de obstáculos. Tal área está pontilhada em branco na imagem e corresponde a uma extensão de aproximadamente 200 m, uma parte do TC de 620 m escolhido para os testes no Pátio de Homogeneização.


Figura 71 – Visão geral da área onde os testes foram realizados, com destaque para o transportador escolhido. Fonte: adaptado de Google (2018)

Como esperado, a Plataforma de Inspeção foi levada a campo e posicionada na área de testes mencionada. Neste protótipo, a plataforma foi composta de um veículo aéreo DJI Mavic Pro, que possui uma câmera tradicional com resolução 4K, seu controle remoto padrão, que se comunica com o VANT por meio de um *link* de comunicação proprietário *OcuSync* nas frequências de 2,4 a 2,483 GHZ, e a aplicação móvel de **Missões por *Waypoint***, executada em um Samsung Galaxy S8. O *smartphone* possui um processador *Snapdragon* com oito núcleos, quatro operando a 2,3 GHZ e quatro a 1,7 GHZ, além de 4 GB de memória RAM. Ele foi conectado à rede móvel da operadora Vivo, que demonstrou possuir cobertura de dados irregular no local dos testes, uma vez que não havia sinal em algumas ocasiões e na

maior parte do tempo a conexão foi do tipo “HSPA+” (do inglês, *Evolved High Speed Packet Access*).

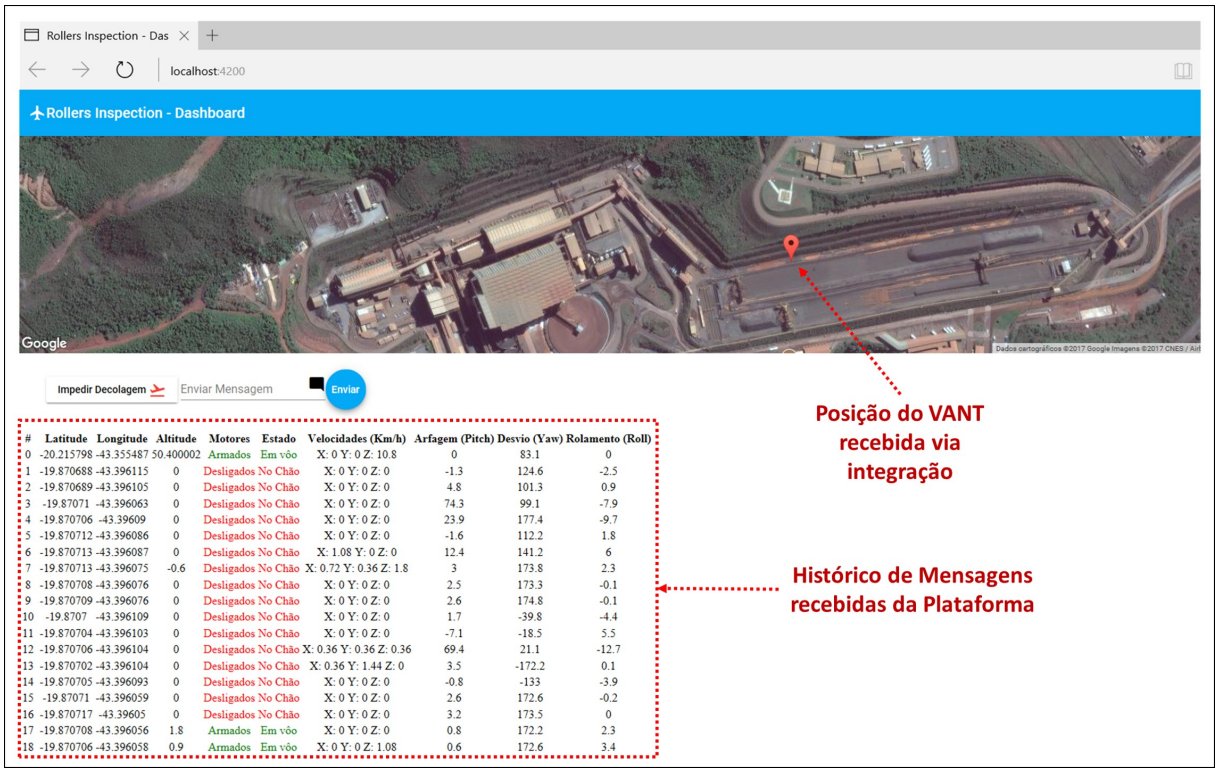
Por sua vez, o **Sistema de Controle da Inspeção** foi executado em um navegador *Microsoft Edge* rodando no sistema operacional Windows 10 em um notebook com processador Intel Core i7 6700HQ, memória RAM de 16 GB e placa de vídeo NVIDIA GeForce 960M (com 4GB de memória). O notebook estava fisicamente em um dos prédios da Usina de Concentração e conectado à rede *Ethernet* cabeada disponível na localidade, que possui velocidade de 100 Mbps e acesso à internet.

Com base nessa configuração, o primeiro teste realizado foi verificar o envio de mensagens de telemetria e sensores da Plataforma de Inspeção em campo para o Sistema de Controle da Inspeção usando o protocolo MQTT e o *broker online HiveMQ*. A aplicação móvel de Missões por Waypoints foi configurada para publicar dados de posição, velocidade, status dos motores e atitude (arfagem, desvio e rolagem) a cada 1 segundo, enquanto o sistema de inspeção monitorava em tempo real o tópico do *Broker* e atualizava a interface do usuário a cada mensagem recebida.

Conforme pode ser visto na Figura 72, o sistema recebeu e atualizou corretamente a posição da Plataforma de Inspeção, vide mapa na parte superior. Além disso, diversas outras mensagens foram recebidas ao longo dos testes, conforme lista apresentada na parte inferior da imagem. Nesta primeira etapa não foi realizada tomada de tempo e contagem de mensagens para avaliar eventuais registros perdidos e o atraso entre envio e recebimento, mas a quantidade de mensagens recebidas no Sistema foi compatível com o período que a Plataforma de Inspeção esteve ativa em campo.

O segundo teste executado refere-se à criação de rotas de inspeção pela aplicação móvel e execução destas missões pelo VANT de forma automática. Uma vez iniciada a missão, o VANT segue as instruções previamente definidas, mas o operador ainda pode tomar ações nos controles para interromper ou alterar a direção da aeronave. Assim, para este teste, foram criadas três missões, todas em distâncias curtas e com no máximo três *waypoints* por vez, espaçados a menos de 6 metros entre eles.

Todas as rotas puderam ser criadas corretamente em relação aos parâmetros de velocidade, altitude e orientação do VANT sendo seguidas corretamente durante o voo. Outro recurso testado foi o controle do VANT usando seu *joystick* durante a missão, com a possibilidade de parar seu avanço ao ponto seguinte e comandá-lo manualmente para retroceder ao anterior. Também foi possível interromper uma missão diretamente pela aplicação desenvolvida, usando para isso um comando criado especificamente para esse fim. A Figura 73 apresenta a tela da aplicação móvel durante os testes, onde é possível visualizar os *waypoints* definidos para a




| # | Latitude | Longitude | Altitude | Motores | Estado | Velocidades (Km/h) | Arfagem (Pitch) | Desvio (Yaw) | Rolamento (Roll) |
|---|---|---|---|---|---|---|---|---|---|
| 0 | -20.215798 | -43.355487 | 50.400002 | Armados | Em vôo | X: 0 Y: 0 Z: 10.8 | 0 | 83.1 | 0 |
| 1 | -19.870688 | -43.396115 | 0 | Desligados | No Chão | X: 0 Y: 0 Z: 0 | -1.3 | 124.6 | -2.5 |
| 2 | -19.870689 | -43.396105 | 0 | Desligados | No Chão | X: 0 Y: 0 Z: 0 | 4.8 | 101.3 | 0.9 |
| 3 | -19.87071 | -43.396063 | 0 | Desligados | No Chão | X: 0 Y: 0 Z: 0 | 74.3 | 99.1 | -7.9 |
| 4 | -19.870706 | -43.39609 | 0 | Desligados | No Chão | X: 0 Y: 0 Z: 0 | 23.9 | 177.4 | -9.7 |
| 5 | -19.870712 | -43.396086 | 0 | Desligados | No Chão | X: 0 Y: 0 Z: 0 | -1.6 | 112.2 | 1.8 |
| 6 | -19.870713 | -43.396087 | 0 | Desligados | No Chão | X: 1.08 Y: 0 Z: 0 | 12.4 | 141.2 | 6 |
| 7 | -19.870713 | -43.396075 | -0.6 | Desligados | No Chão | X: 0.72 Y: 0.36 Z: 1.8 | 3 | 173.8 | 2.3 |
| 8 | -19.870708 | -43.396076 | 0 | Desligados | No Chão | X: 0 Y: 0 Z: 0 | 2.5 | 173.3 | -0.1 |
| 9 | -19.870709 | -43.396076 | 0 | Desligados | No Chão | X: 0 Y: 0 Z: 0 | 2.6 | 174.8 | -0.1 |
| 10 | -19.8707 | -43.396109 | 0 | Desligados | No Chão | X: 0 Y: 0 Z: 0 | 1.7 | -39.8 | -4.4 |
| 11 | -19.870704 | -43.396103 | 0 | Desligados | No Chão | X: 0 Y: 0 Z: 0 | -7.1 | -18.5 | 5.5 |
| 12 | -19.870706 | -43.396104 | 0 | Desligados | No Chão | X: 0.36 Y: 0.36 Z: 0.36 | 69.4 | 21.1 | -12.7 |
| 13 | -19.870702 | -43.396104 | 0 | Desligados | No Chão | X: 0.36 Y: 1.44 Z: 0 | 3.5 | -172.2 | 0.1 |
| 14 | -19.870705 | -43.396093 | 0 | Desligados | No Chão | X: 0 Y: 0 Z: 0 | -0.8 | -133 | -3.9 |
| 15 | -19.87071 | -43.396059 | 0 | Desligados | No Chão | X: 0 Y: 0 Z: 0 | 2.6 | 172.6 | -0.2 |
| 16 | -19.870717 | -43.39605 | 0 | Desligados | No Chão | X: 0 Y: 0 Z: 0 | 3.2 | 173.5 | 0 |
| 17 | -19.870708 | -43.396056 | 1.8 | Armados | Em vôo | X: 0 Y: 0 Z: 0 | 0.8 | 172.2 | 2.3 |
| 18 | -19.870706 | -43.396058 | 0.9 | Armados | Em vôo | X: 0 Y: 0 Z: 1.08 | 0.6 | 172.6 | 3.4 |

Figura 72 – Mensagens recebidas via integração no Sistema de Controle da Inspeção. Fonte: autor

missão na área de testes. Também é possível ver no canto inferior direito a câmera FPV do VANT, que está ampliada na Figura 74. Nela, pode-se observar uma parte do Pátio de Homogeneização, onde aparece o TC inspecionado, uma empilhadeira em operação e uma pilha de materiais. Também é possível visualizar nessa imagem que a área livre disponível para voo do portador era de aproximadamente 8 metros.

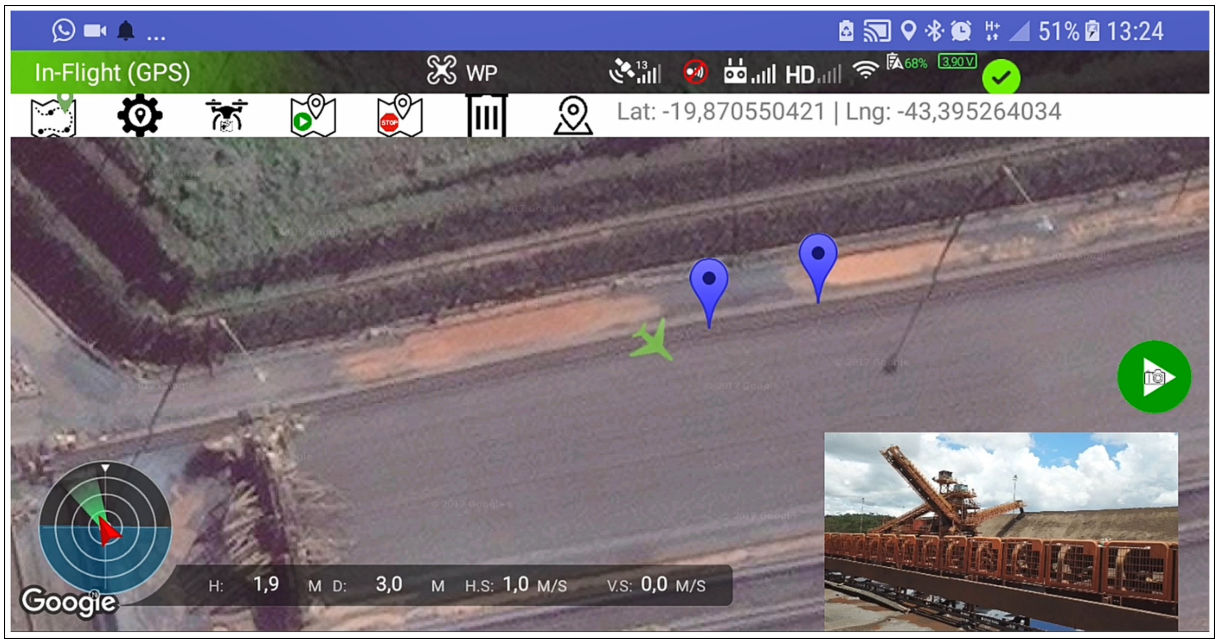



Figura 73 – Tela da aplicação móvel durante a execução de uma missão de inspeção automática. Fonte: autor


Figura 74 – Câmera FPV do VANT durante a execução da missão. Fonte: autor

De forma geral, o protótipo mostrou ser viável a definição de rotas de inspeção usando uma aplicação móvel, mas deve-se ter em mente os problemas relativos à imprecisão, tanto do mapa disponível quanto do equipamento de navegação do VANT. Em relação ao primeiro item, foi utilizado um componente do *Google Maps* na aplicação Android que apresenta ao usuário a visualização de satélite da região, bastando que ele clique na tela para marcar o *waypoint* desejado. Porém, o nível de *zoom* é limitado e nem sempre a coordenada capturada (latitude e longitude) corresponde de fato à posição exata da imagem de satélite que o usuário visualiza (imprecisão do provedor de mapas). Já o segundo caso ocorre porque o VANT utilizado nos testes possui receptores GNSS tradicionais, sem nenhum tipo de correção. Por exemplo, o GPS, um dos receptores disponíveis, possui precisão na ordem de 4,9 metros em áreas abertas (U.S. GOVERNMENT, 2017).

Esta imprecisão, já esperada, foi comprovada durante os testes. Em uma das rotas criadas, foram selecionados pontos que, na imagem de satélite, correspondiam à área aberta na lateral do transportador, a pelo menos 3 metros do TC. Quando a missão foi iniciada, o VANT dirigiu-se para o TC, chegando a voar a menos de um metro dele, o que levou à interrupção da missão por questões de segurança. Assim, como trabalho futuro, é importante realizar o mapeamento das áreas onde será feita a operação da Plataforma de Inspeção por meio de *softwares* e equipamentos especializados, gerando mapas com grande precisão. Por sua vez, os portadores também precisam utilizar equipamentos e técnicas de correção, como *Real Time Kinematic* (RTK) (TALBOT; ALLISON; NICHOLS, 1994), que pode elevar a precisão a centímetros, mais que suficiente para este tipo de uso.

Além disso, cabe destacar que o ambiente onde foram realizados os testes é inadequado a portadores e demais equipamentos que não possuam certificação adequada contra poeira. Durante os testes, o gimbal da câmera do VANT utilizado travou numa posição fixa e não conseguia estabilizar a imagem. Posteriormente, foi constatada a presença de partículas muito finas de poeira em seu interior. Feita a limpeza, ele voltou a funcionar normalmente, mas a longo prazo, esse tipo de situação pode danificar o equipamento de forma definitiva. Por fim, também ocorreram problemas devido a interferência magnética, possivelmente, por conta da grande quantidade de estruturas metálicas e à presença de minério de ferro de alto teor nas camadas de poeira acumulada no chão. Isso impediu que o VANT decolasse em algumas ocasiões e gerou alertas durante alguns voos.

Tendo isso em mente, pode-se concluir que a execução dos testes em campo para este protótipo atingiu os seus objetivos e demonstrou que a integração entre **Sistemas Corporativos** e a **Plataforma de Inspeção** usando o protocolo MQTT (*Publish-Subscribe*) é viável. Também foi possível avaliar que o uso de rotas de inspeção pré-definidas só é totalmente seguro com o uso de equipamentos e mapas de alta precisão para orientação correta do portador.

## 6.2 Protótipo 2

Esta seção descreve os testes realizados no segundo protótipo, apresentado na seção "**5.4.2 - Protótipo 2 - Envio de defeitos da UPE para o Sistema de Manutenção**". A Figura 75 destaca os requisitos funcionais avaliados e distribui os componentes utilizados na construção do protótipo nas camadas da Arquitetura Integrada. Ele procura dar destaque à camada de Integração, que absorve a maior parte da complexidade para facilitar a interação entre as pontas, o que é evidenciado nos testes e resultados aqui discutidos.

Os testes para validar a comunicação da UPE com o CMMS foram divididos em duas etapas principais: captura das imagens em campo e o processamento *offline* delas, com envio dos defeitos identificados para o CMMS. A primeira fase foi realizada por outro autor no Pátio de Estocagem do Terminal Marítimo de Ponta da Madeira, em São Luís-MA, que também pertence à mineradora Vale. Atualmente, é o maior porto do Brasil em volume de movimentação de cargas, com previsão de 230 milhões de toneladas de minério em 2018, o que também o caracteriza como o maior terminal de minério do mundo (VALE, 2016a). Com mais de 600 mil metros de área, o porto possui uma grande extensão de transportadores de correia para movimentar o material desde a região de descarga até o píer para embarque em

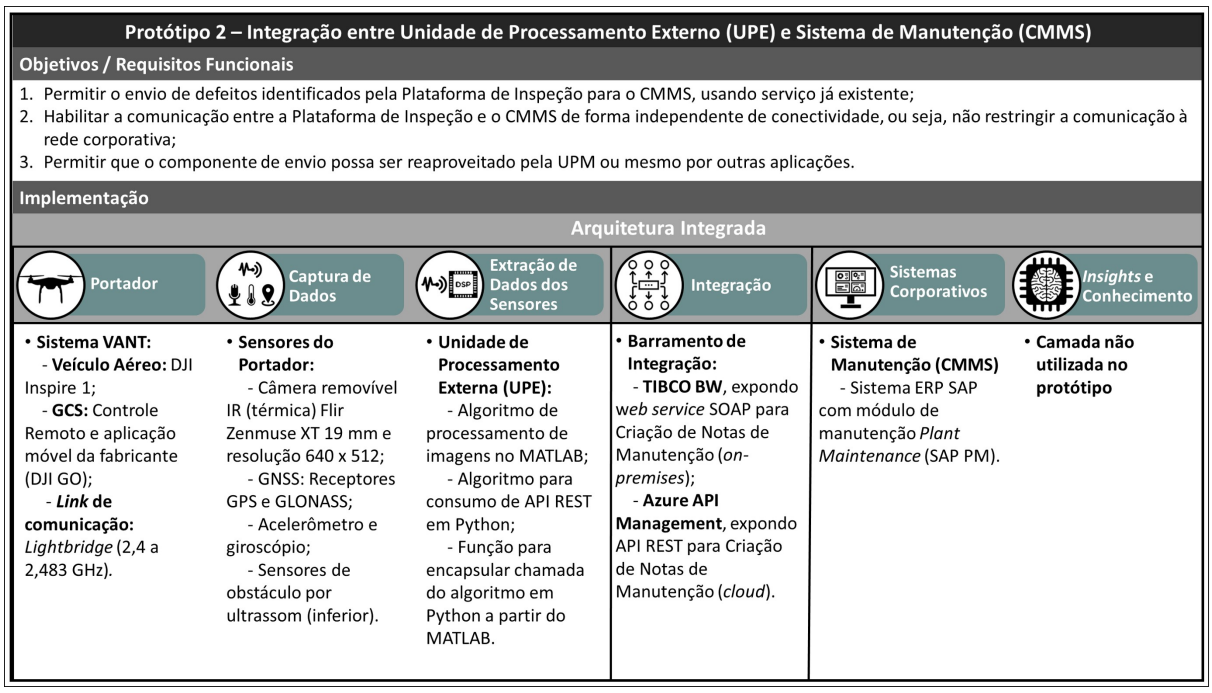



Figura 75 – Principais componentes do segundo protótipo de acordo com as camadas da Arquitetura Integrada. Fonte: autor

navios de até 400 mil toneladas. Por conta disso, o porto possui mais de 200.000 rolos em seus TCs, o que reforça sua relevância para os testes e aplicabilidade para o monitoramento de condição com a Plataforma de Inspeção.

Nesse contexto, a Figura 76 apresenta uma imagem aérea do Pátio de Estocagem do porto, onde é possível visualizar a área onde a captura das imagens foi realizada e o transportador escolhido para testes, que possui aproximadamente 1350 metros de extensão. Os voos para obtenção das imagens térmicas dos rolos foram executados por Carvalho (2018), usando uma Plataforma de Inspeção baseada em um **Portador** DJI Inspire 1. Por sua vez, a camada de **Captura de Dados** empregou uma câmera térmica Flir Zenmuse XT, compatível com o portador e capaz de obter imagens com resolução de 640 x 512 a uma taxa máxima de 9 Hz.

As imagens capturadas ao longo dos testes, que foram feitos em horários e dias diversos, incluindo coletas de dados à noite, foram armazenadas em um cartão de memória conectado à câmera da camada de **Captura de Dados**. Isso foi necessário porque a Plataforma de Inspeção utilizada não tinha conectividade com nenhuma rede para transmissão dos dados, nem possuía algoritmos embarcados como parte de uma Unidade de Processamento Móvel (UPM) para detecção *online* de defeitos. Assim, de forma *offline*, os arquivos de imagens com dados radiométricos gerados pela câmera térmica foram transferidos para um notebook com processador Intel Core i7 6700HQ, memória RAM de 16 GB, placa de vídeo NVIDIA GeForce 960M com 4GB de memória dedicada e sistema operacional Windows 10, conectado à

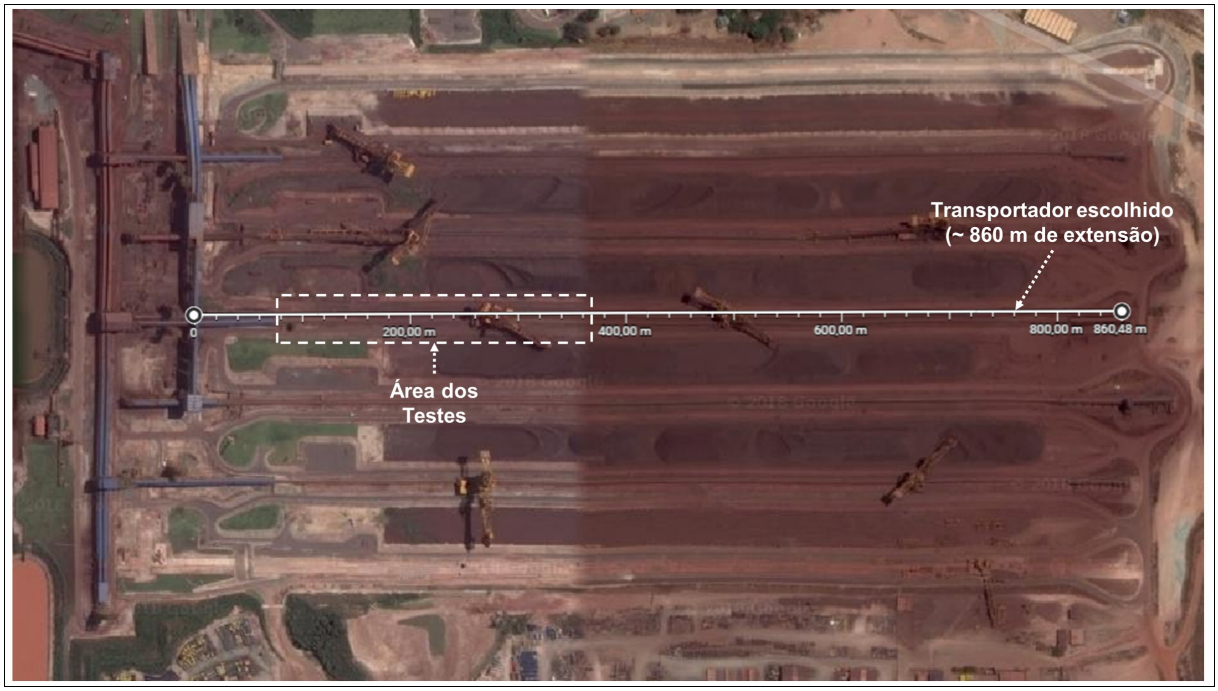



Figura 76 – Pátio de Estocagem do Terminal Marítimo de Ponta da Madeira, onde foram capturadas as imagens para testes. Fonte: adaptado de Google (2017)

internet por meio de uma rede sem fio 802.11n e com velocidade nominal de 60 Mbps.

Concluída a transferência, iniciou-se a segunda etapa dos testes, conforme mostrado na Figura 77. De forma resumida, o algoritmo de processamento da UPE (2), desenvolvido por outro autor, foi aplicado em todo o conjunto de imagens do transportador capturadas pela Plataforma de Inspeção (1), tendo como resultado a delimitação de regiões com temperatura elevada (3). Para cada um destes defeitos, o algoritmo no MATLAB acionou a função desenvolvida em *Python*, que comunicou-se com a API para **Criação de Notas de Manutenção** (4) e, com isso, registrou notas de manutenção no CMMS para cada defeito (5).

Ainda com base na Figura 77, é possível notar que a prioridade das notas foi enviada como “Baixa”, visto que os valores de temperatura, exibidos na descrição da nota, não foram classificados como graves ou emergenciais. Considerando que o CMMS utilizado na empresa não está preparado para receber em campos específicos as coordenadas de posição dos rolos, as informações foram enviadas como parte da descrição longa da nota de manutenção, conforme pode ser visto na Figura 78. Além disso, na mesma imagem, é possível observar outros dados relevantes, como o Notificador, indicando que a nota foi gerada automaticamente pela UPE do MATLAB. Caso o defeito fosse processado por outro meio, por exemplo, pela UPM, essa informação seria alterada para auxiliar a equipe de manutenção na gestão da carteira de serviços.

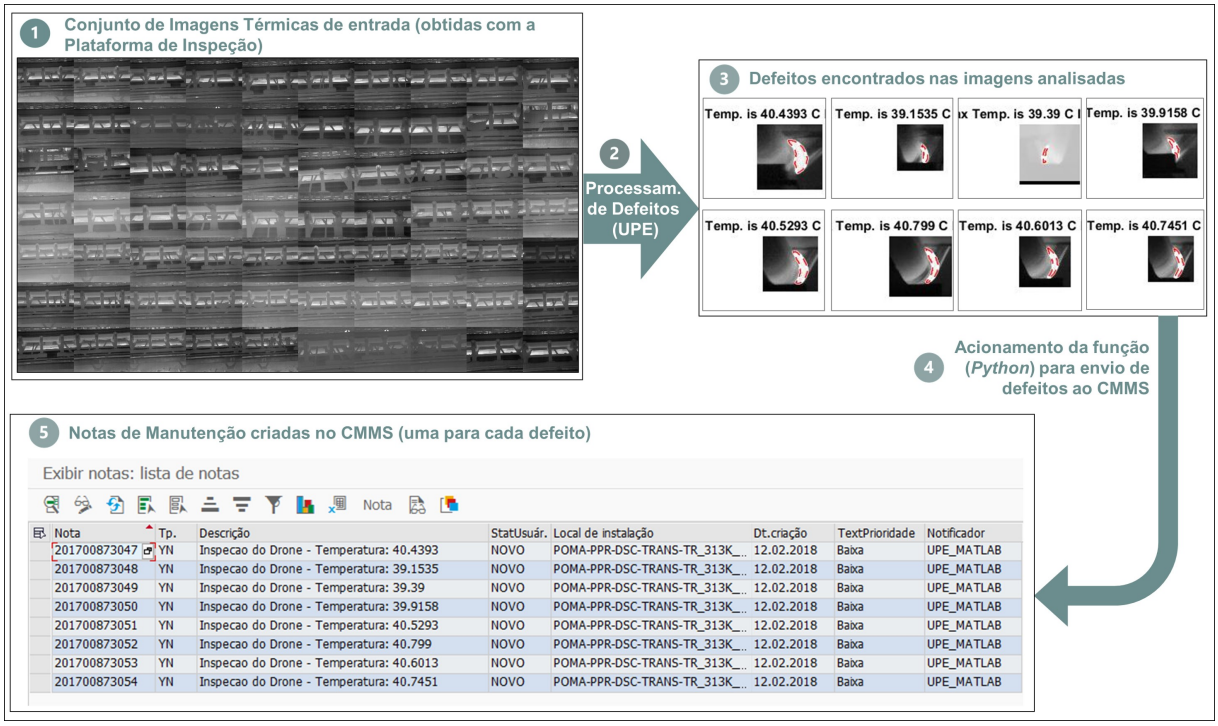



Figura 77 – Etapas de processamento de defeitos e envio para o CMMS executados durante os testes. Fonte: autor, a partir de arquivos internos do projeto

Isto posto, pode-se considerar que a interação no estilo *Request-Response* para envio de defeitos da Plataforma de Inspeção para o CMMS foi realizada com sucesso. Mesmo passando por diferentes etapas de processamento em múltiplas ferramentas, os oito defeitos identificados no lote de imagens analisado levaram 34 segundos para serem registrados no CMMS, uma média de 4,25 segundos por defeito. Além disso, as diferenças de implementação entre o CMMS e a UPE, como protocolos de comunicação incompatíveis e conectividade, foram superadas, já que diferentes requisições REST disparadas de um computador fora da rede corporativa de TI foram processadas com sucesso por um serviço SOAP, localizado na rede interna da empresa. Tais características demonstram, na prática, os benefícios de um barramento de integração e validam o seu papel como elemento central da arquitetura proposta.

## 6.3 Testes de Controle e Obtenção de Dados do Simulador

A Figura 79 apresenta uma execução do simulador sendo controlada pela **Aplicação de Controle** desenvolvida em *Python*, mostrada no canto superior direito. Nela é possível visualizar botões que representam as ações de controle (Decolar, Seguir Rota, Aterrissar) e o botão para obter dados de voo, como posição e atitude

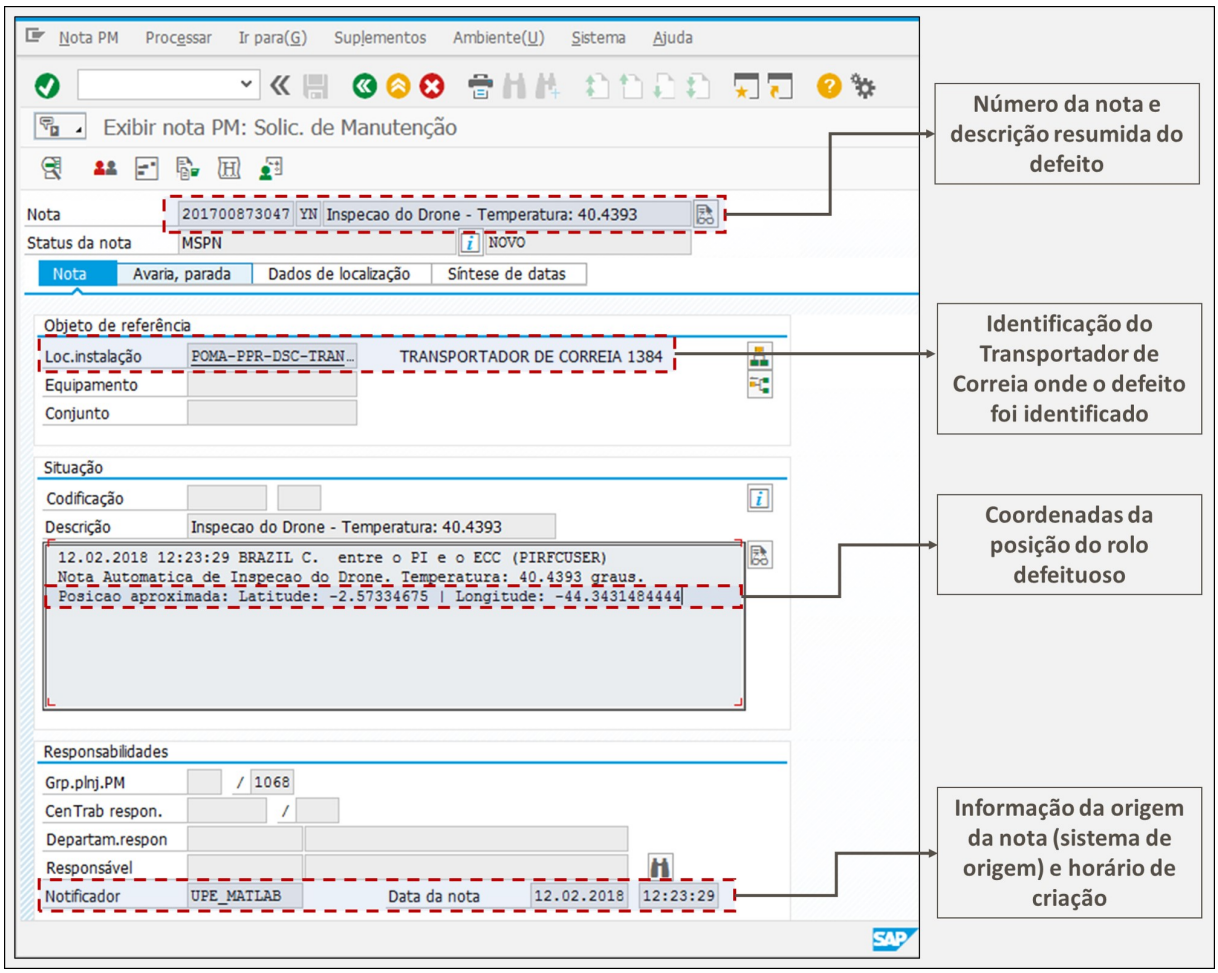



Figura 78 – Nota de Manutenção criada no CMMS com as informações enviadas pela UPE. Fonte: autor

do VANT, além de uma caixa de texto com as informações de *log* recebidas da aplicação ***Handler***. Na mesma figura, no canto inferior direito, é possível visualizar algumas mensagens do *broker* MQTT nos tópicos de **Comando** e ***Logging***. Por fim, ocupando toda a parte esquerda da imagem, está a interface da Unreal Engine com a execução do *plugin* AirSim no ambiente de porto desenvolvido.

Com base nessa estrutura, foi possível realizar o controle do simulador a partir de uma aplicação externa, passando por uma camada de integração que permitiu desacoplar a comunicação. É importante destacar que, neste momento, apenas comandos pré-estabelecidos foram testados, sendo necessário, como trabalho futuro, avaliar o comportamento dessa arquitetura em teleoperação de tempo real, por exemplo, para controle contínuo do VANT. Sabe-se, de antemão, que a API existente permite isso, porém, a latência na comunicação usando o *broker* MQTT pode prejudicar ou mesmo inviabilizar o controle. Entretanto, para uso em ambientes controlados, como no caso de uso para instrução de operadores, entende-se que esta é uma abordagem válida.

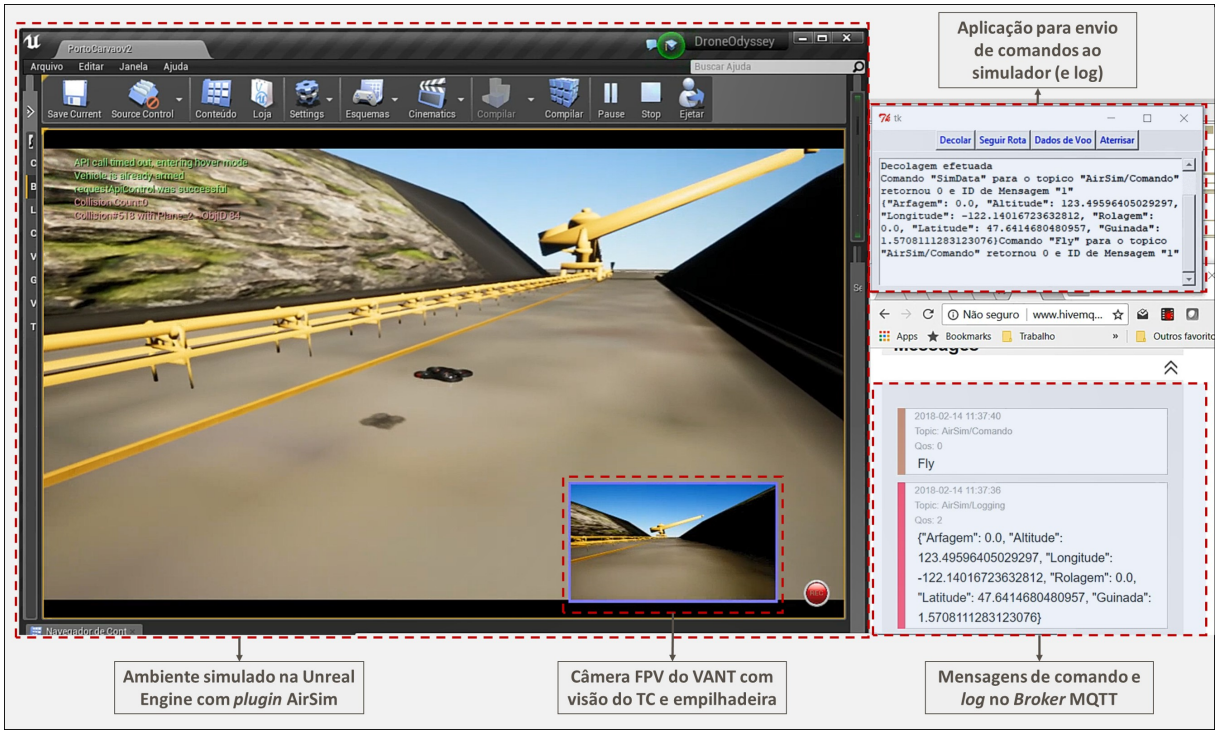



Figura 79 – Esquema dos elementos envolvidos na aplicação para controle externo do simulador. Fonte: autor

Por fim, também foi possível obter dados do simulador, como posição GNSS e atitude. Diversos outros sensores estão disponíveis no simulador e poderiam ser obtidos de forma contínua pelo ***Handler*** e disponibilizados no *broker*, o que valida os testes realizados para esse objetivo. Por outro lado, isso não é verdade para a câmera FPV do VANT, já que o MQTT não é apropriado para os requisitos de latência para transmissão de vídeo próxima de tempo real. Por conta disso, a captura de vídeo, que complementa a experiência de teleoperação, é abordada na subseção a seguir.

## 6.4 Testes do *Streaming* de Vídeo do Simulador

Para validar o funcionamento da captura e transmissão de imagens do simulador, foi utilizada uma Aplicação de Visualização desenvolvida por Lonczynski (2018) com o motor gráfico Unity e compatível com o *Google Cardboard*, daqui em diante chamada **Aplicativo VR**. Ela foi executada em um *smartphone* Samsung Galaxy S8, com processador *Snapdragon* de oito núcleos, quatro operando a 2,3 GHZ e quatro a 1,7 GHZ, 4 GB de memória RAM e sistema operacional Android 7.0 *Nougat*. Como ela foi desenvolvida por outro autor, e usada aqui apenas para fins de validação do processo de captura e *streaming* de vídeo, não será feito nenhum tipo detalhamento de sua arquitetura, apenas os resultados da execução.

Já a aplicação ***Streaming* de Vídeo do Simulador** foi executada na mesma máquina do simulador (Microsoft AirSim), um *notebook* com processador Intel Core i7 6700HQ, 16 GB DDR3, placa de vídeo NVIDIA GeForce 960M com 4 GB DDR5 e sistema operacional Windows 10. Tanto o *smartphone* com o **Aplicativo VR** quanto o *notebook* foram conectados a uma rede sem fio 802.11n operando na frequência de 5,8 GHz, menos suscetível a interferências. Com essa configuração, foram executadas diferentes rodadas de testes para validar o processo mostrado na Figura 69.

Ao tentar capturar imagens com resoluções elevadas do simulador, inicialmente 1024 x 768, percebeu-se duas limitações. A primeira delas é a própria *performance* do simulador durante a captura de quadros, já que a placa de vídeo precisa renderizar a "Visualização" antes que a captura seja feita. Assim, a cada novo tipo de "Visualização" adicionado à simulação, diminui-se a taxa de quadros (FPS, do inglês *Frames per Second*) e aumenta o consumo de recursos, particularmente de processamento (gráfico e principal) (SHAH, 2017). O gráfico mostrado na Figura 80 apresenta a taxa de quadros inicial de 60 FPS sendo afetada pela adição de novas visualizações e, posteriormente, pela captura de quadros via API, que apresenta um pico quando ativada, estabilizando a taxa de quadros na sequência.

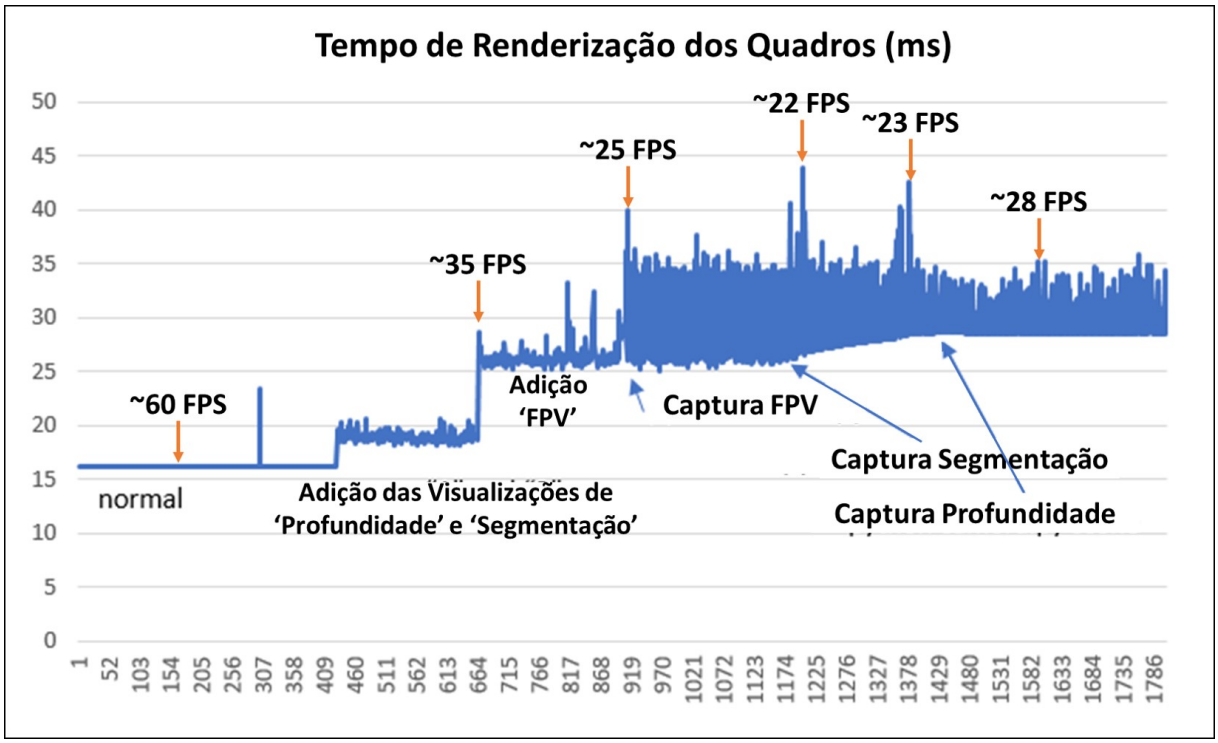



Figura 80 – Impacto da adição de novas "Visualizações" na taxa de quadros da simulação com o AirSim. Fonte: adaptado de (SHAH, 2017)

Já a segunda situação que limita a resolução está relacionada à transmissão dos dados. Durante os testes da aplicação de ***Streaming* de Vídeo do Simulador**,

verificou-se que ela gerava erros quando a imagem já redimensionada sendo transmitida possuía tamanho superior a 65 Kb. Durante a avaliação, verificou-se que o motivo é que protocolo UDP possui um limite de 65.507 *bytes* que podem ser transportados por pacote (FAIRHURST, 2008). Além de avaliar protocolos mais robustos para *streaming*, como o *Real Time Protocol* (RTP), outra forma de tratar essa limitação, como trabalho futuro, é a divisão de cada imagem em múltiplos pacotes. Visto que o maior objetivo da aplicação em questão é provar a viabilidade da captura e transmissão de vídeo oriundo do simulador, essa solução não foi implementada e diferentes resoluções foram verificadas até que se encontrasse a maior possível transportável em um pacote UDP.

Nesse contexto, verificou-se que 480 x 240 *pixels* foi a maior resolução possível utilizando a estrutura existente da aplicação. Diante disso, com o objetivo de avaliar a viabilidade do *streaming* de vídeo do simulador, foram executadas algumas rodadas de testes para comparar a taxa de captura e transmissão de quadros do simulador e a taxa de quadros recebidos e mostrados em formato de vídeo pelo **Aplicativo VR**. Além da resolução de 480 x 240, também foram executados testes com resoluções inferiores. Porém, abaixo da resolução de 384 x 256, as imagens obtidas não possuíam um mínimo de definição e foram desconsideradas. A Figura 81 apresenta capturas do Aplicativo VR[1] de uma execução com as duas resoluções consideradas. É possível perceber que a 384 x 256, na parte esquerda da figura, a definição da imagem já é comprometida.


Figura 81 – Definição das imagens em cada uma das resoluções testadas para o Aplicativo VR. Fonte: autor

Considerando tais resoluções, a Figura 82 apresenta o resultado dos testes em relação às taxas de *streaming* de vídeo. Para cada resolução considerada, é possível observar a taxa de quadros da captura e transmissão de imagens, apresentada como ***Streaming* Simulador**, e a taxa de quadros recebida e renderizada pelo **Aplicativo VR**.

1 As imagens em cada resolução aparecem duplicadas pois o aplicativo é usado com um óculos VR, que junta ambas partes para formar uma imagem com profundidade

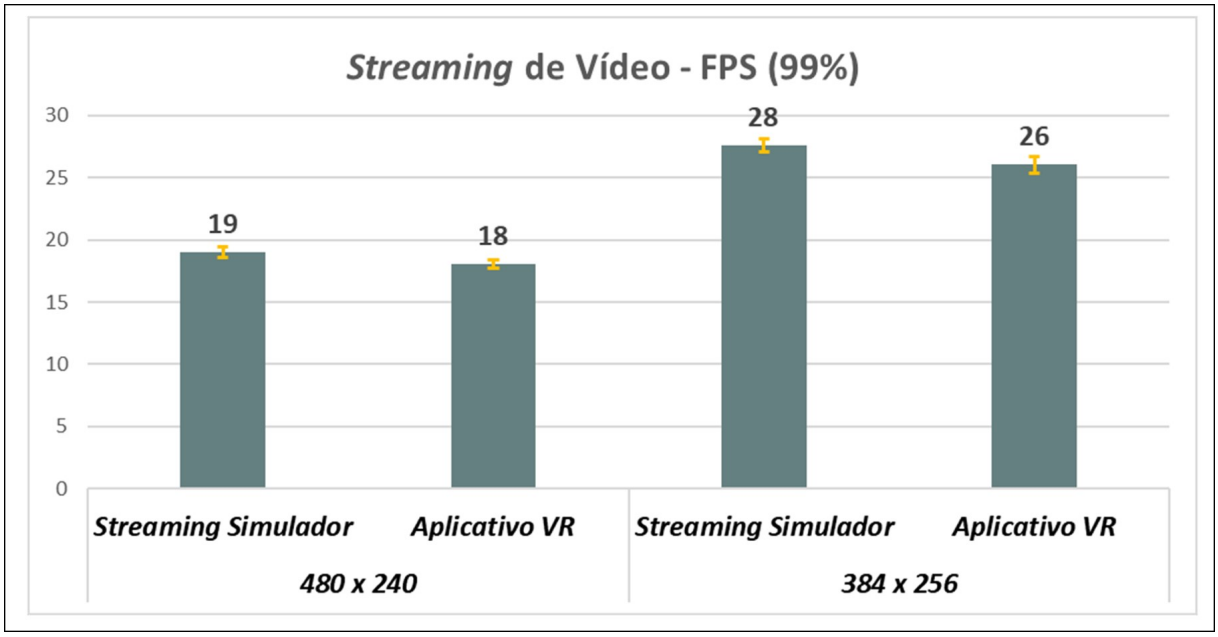



Figura 82 – Resultado da captura e transmissão de imagens do simulador para o Aplicativo VR. Fonte: autor

Com base nesse gráfico, percebe-se que a resolução influencia fortemente a taxa de captura e *streaming* de imagens, dada a discrepância nos resultados entre as duas resoluções. Por outro lado, não há perda significativa de quadros na transmissão e, por conta disso, o **Aplicativo VR** consegue manter uma taxa de apresentação dos quadros muito próxima à da captura. Assim, percebe-se que a captura no simulador é o maior problema nas resoluções avaliadas.

Uma das explicações possíveis para estes resultados remete ao impacto que a adição da visualização FPV, que serve como origem para as imagens capturadas via API, tem na taxa de quadros da simulação, mostrado na Figura 80. Potencialmente, vide discussão em (SHAH, 2017), em uma máquina com maior disponibilidade de recursos para processamento, as taxas de captura obtidas pela aplicação de ***Streaming* de Vídeo do Simulador** seriam maiores. Porém, quando se avalia o uso de vídeo para realidade virtual, questões relativas à experiência do usuário e imersão são extremamente relevantes. O ideal é que a taxa de quadros seja próxima a 90 FPS e a resolução a mais alta possível (HECHT, 2016), o que está bem distante dos resultados obtidos.

Assim, conclui-se que a abordagem adotada para obtenção de vídeo do simulador é válida, mas é limitada e não suporta os requisitos de todos os tipos de Aplicação de Visualização, particularmente os de realidade virtual. Por conta disso, como trabalho futuro, deve-se avaliar formas mais eficientes de obtenção de vídeo do simulador, que não afete a simulação em si. A transmissão das imagens, que limitou a resolução adotada por conta do tamanho máximo de cada pacote, tam-

bém pode ser evoluída. Nesse caso, deve-se investigar algoritmos mais robustos para compressão e segmentação de imagens e protocolos mais adequados para transmissão desse tipo de conteúdo, como o RTP.

# 7 Considerações Finais

## 7.1 Conclusões

O presente trabalhou propôs uma arquitetura para suportar a montagem de Plataformas de Inspeção para rolos de transportadores de correias, um dos grandes desafios da indústria da mineração. A arquitetura permite a evolução independente dos elementos de cada camada, ao mesmo tempo que a combinação de suas partes resulta em plataformas adaptadas a diferentes contextos. Por exemplo, uma plataforma baseada em portadores VANTs com sensores termográficos é apropriada para a inspeção em pátios, enquanto um robô terrestre equipado com acelerômetros pode acessar e medir a vibração de rolos em áreas fechadas.

Além deste aspecto, o trabalho propôs a integração da Plataforma de Inspeção a Sistemas Corporativos. Além de minimizar uma das grandes desvantagens de plataformas móveis, que é o tempo entre a detecção do defeito e seu registro, tal integração também permite que a plataforma receba dados destes sistemas para suportar a inspeção. Adicionalmente, os dados gerados por ela passam a ser armazenados em repositórios de *analytics*, onde cientistas de dados podem analisá-los em conjunto com informações de outros sistemas e, assim, propor novos métodos de detecção de falhas, retroalimentando o processo.

Para suportar tais fluxos de dados, a arquitetura proposta posicionou uma camada de Integração como elemento central, ondem foram avaliadas diferentes técnicas e protocolos apropriados aos tipos de interações entre a Plataforma de Inspeção e Sistemas Corporativos: *Request-Response* e *Publish-Subscribe*. Em ambos os casos, defendeu-se o uso de um Barramento de Integração para absorver grande parte da complexidade de comunicação entre essas partes, como as diferenças de protocolos e de conectividade.

A arquitetura em questão utilizou protótipos para suportar sua validação. O primeiro protótipo avaliou em campo o envio de dados de telemetria e sensores da plataforma para um sistema corporativo e o controle do portador VANT usando missões do tipo *waypoint*. Tal protótipo validou a escolha do protocolo MQTT neste caso de uso, pois mesmo diante de instabilidades da rede móvel em campo, as mensagens foram entregues ao sistema de inspeção, permitindo seu acompanhamento remoto. Por outro lado, o controle do portador via aplicativo móvel reforçou a necessidade do uso de equipamentos de RTK e mapeamento prévio das áreas inspecionadas, já que a precisão usando a API do Google Maps e os sensores

GNSS disponíveis no VANT não permitem uma operação segura no modo *waypoint*. Ademais, recomenda-se o uso de equipamentos certificados para operar em ambientes agressivos, já que o VANT utilizado teve alguns travamentos em seu gimbal por conta do excesso de poeira durante os testes.

Por sua vez, o segundo protótipo demonstrou a viabilidade do processamento de defeitos pela Plataforma de Inspeção e sua integração ao Sistema de Manutenção. Nele, foi empregado um portador com câmera térmica e algoritmos especializados para análise *offline* das imagens, além de uma função para comunicação com o barramento de integração. Por conta dos mecanismos desenvolvidos na camada de Integração, os testes demonstraram uma performance adequada ao envio de defeitos e simplicidade no desenvolvimento, o que reforça a importância desta camada na arquitetura.

O trabalho também avaliou o uso de simulação, visto que foram empregados VANTs como principal portador. Foi desenvolvido um ambiente virtual de um porto de minério com potencial para suportar aplicações diversas, incluindo treinamento de operadores. Para validar essa abordagem, foi criada uma aplicação que permitiu controlar externamente um portador VANT neste ambiente e obter dados de sensores e telemetria. Também foi desenvolvida uma aplicação para obtenção de vídeo do simulador e transmissão em tempo real via rede. Enquanto a primeira aplicação mostrou-se viável, a segunda esbarrou em limitações de performance devido ao *hardware* do simulador e ao protocolo de rede utilizado (UDP). Esse resultado demonstra a necessidade de uma outra abordagem para clientes que demandem alta resolução e taxa de quadros, como realidade virtual.

Por fim, considerando que o trabalho dos demais autores envolvidos nessa iniciativa tiveram bons resultados no processamento de defeitos e que este trabalho empregou componentes de integração já usados na empresa, ela abriu um projeto para implementar essa iniciativa em escala industrial, o que demonstra a aplicabilidade da arquitetura aqui proposta e a viabilidade de Plataformas de Inspeção móveis para a inspeção de rolos de transportadores de correia.

## 7.2 Trabalhos Futuros

Os protótipos apresentados são um esforço inicial para demonstrar a viabilidade da arquitetura para suportar Plataformas de Inspeção móveis, baseados em portadores e sensores comerciais que não são adaptados a esta aplicação. Dessa forma, as próximas etapas da pesquisa devem usar a arquitetura definida como referência para construir uma Plataforma de Inspeção totalmente adaptada ao monitoramento

de condições dos rolos, utilizando componentes certificados para o ambiente agressivo e com sensores de alta precisão (RTK) para navegação. Também deve ser dado foco ao mapeamento das áreas a serem inspecionadas, favorecendo portadores com capacidade de navegação por *waypoints* ou autônoma.

Em relação à simulação, é evidente o seu potencial para o treinamento dos inspetores que irão operar a Plataforma de Inspeção, especialmente as baseadas em VANTs. Portanto, a pesquisa deve focar em métodos mais eficientes para captura e transmissão de vídeo do simulador, que permitam uma alta taxa de quadros e resolução adequadas a aplicações de realidade virtual. Também deve-se avaliar se o método de controle proposto para o simulador é capaz de suportar teleoperação em tempo real.

# Reconhecimento

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior, Brasil (CAPES), Código de Financiamento 001; do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq); da Fundação de Amparo à Pesquisa do Estado de Minas Gerais (FAPEMIG); e da Vale SA.

# REFERÊNCIAS


3D TOWER. *Sandwich Conveyor*. 2017. Disponível em: <https://www.turbosquid.com/3d-models/lwo-sandwich-conveyor/477507>. Acesso em: 26 ago. 2017. Citado na página 31.

ABB Technology AG. **Conveyor Inspection With Unmanned Vehicle Carying Sensor Structure**. USPTO, 2014. Disponível em: <https://patents.google.com/patent-/US20160152416A1/en>. Acesso em: 29 nov. 2016. Citado 2 vezes nas páginas 10 e 48.

AGÊNCIA NACIONAL DE AVIAÇÃO CIVIL. **Regras da ANAC para uso de drones entram em vigor**. 2017. Disponível em: <http://www.anac.gov.br/noticias-/2017/regras-da-anac-para-uso-de-drones-entram-em-vigor/release_drone.pdf>. Acesso em: 03 nov. 2017. Citado na página 83.

____. **Requisitos Gerais para Aeronaves Não Tripuladas de Uso Civil**. 2017. Disponível em: <http://www.anac.gov.br/assuntos/legislacao/legislacao-1/rbha-e-rbac/rbac/rbac-e-94-emd-00>. Acesso em: 03 nov. 2017. Citado 2 vezes nas páginas 81 e 82.

ALSHAMMARI, F.; ADDALI, A. Bearing Condition Monitoring with Acoustic Emission Techniques. **International Journal of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering**, 2015. World Academy of Science, Engineering and Technology, v. 9, n. 12, p. 2073–2077, 2015. Citado na página 39.

AP SENSING. **AP Sensing Fire Detection**. 2017. Disponível em: <https://www.ap-sensing.com/application/fire-detection/>. Acesso em: 11 jan. 2017. Citado na página 46.

APACHE SOFTWARE FOUNDATION. **Apache ServiceMix**. 2017. Disponível em: <http://servicemix.apache.org/>. Acesso em: 19 nov. 2017. Citado na página 91.

ARMBRUST, M. et al. A View of Cloud Computing Clearing the clouds away from the true potential and obstacles posed by this computing capability. **Communications of the ACM**, 2010. v. 53, n. 4, p. 50–58, 2010. Citado 2 vezes nas páginas 82 e 83.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **NBR 6177: Transportadores contínuos: Transportadores de correia: Terminologia**. Rio de Janeiro, 2016. 45 p. Citado 3 vezes nas páginas 9, 35 e 36.

BANKS, A.; GUPTA, R. (Ed.). **MQTT Version 3.1.1**. 2014. Disponível em: <http:/-/docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html>. Acesso em: 22 nov. 2017. Citado na página 102.

BONOMI, F.; MILITO, R.; ZHU, J.; ADDEPALLI, S. Fog Computing and Its Role in the Internet of Things. In: **Proceedings of the first edition of the MCC workshop on Mobile Cloud Computing**. Helsinki: ACM, 2012. p. 13–16. Citado na página 84.

BORMANN, C.; CASTELLANI, A. P.; SHELBY, Z. CoAP: An application protocol for billions of tiny internet nodes. **IEEE Internet Computing**, 2012. v. 16, n. 2, p. 62–67, 2012. Citado 3 vezes nas páginas 11, 105 e 106.

BRESCIANI, T. Mestrado em Automação e Controle, **Modelling, Identification and Control of a Quadrotor Helicopter**. Lund: [s.n.], 2008. 180 p. Citado na página 72.

BRISTEAU, P.-J.; CALLOU, F.; VISSIÈRE, D.; PETIT, N. The Navigation and Control technology inside the AR.Drone micro UAV. In: **Preprints of the 18th IFAC World Congress**. Milano: [s.n.], 2011. p. 8. Citado 4 vezes nas páginas 10, 77, 78 e 79.

BROWN, T. X.; MCHENRY, M.; JAROONVANICHKUL, S. Cognitive Radio Architectures for Unmanned Aircraft Systems. In: VALAVANIS, K. P.; VACHTSEVANOS, G. J. (Ed.). **Handbook of Unmanned Aerial Vehicles**. [S.l.]: Springer Publishing Company, Incorporated, 2014. cap. 35, p. 813–844. Citado na página 66.

CARMO, T. d. R. e. **Uso do padrão AMQP para transporte de mensagens entre atores remotos**. 74 p. Dissertação (Mestrado em Ciência da Computação) — Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2012. Citado na página 104.

CARO, N. D.; COLITTI, W.; STEENHAUT, K.; MANGINO, G.; REALI, G. Comparison of two lightweight protocols for smartphone-based sensing. In: **20th Symposium on Communications and Vehicular Technology in the Benelux**. Namur: IEEE, 2013. p. 1–6. Citado 2 vezes nas páginas 108 e 109.

CARVALHO, R. **Processamento Digital de Imagens para a Identificação Automática de Falhas em Rolos de Transportadores de Correias**. 118 p. Dissertação (Mestrado em Automação) — Escola de Minas, Universidade Federal de Ouro Preto, Ouro Preto, 2018. Citado na página 163.

CHAPPEL, D. A. **Enterprise Service Bus: Theory in Practice**. 1. ed. Sebastopol: O'Reilly, 2004. 248 p. Citado na página 90.

CIVIL AVIATION SAFETY AUTHORITY. **NFRM 1309OS - Remotely piloted aircraft systems**. 2017. Disponível em: <https://www.casa.gov.au/standard-page-/nfrm-1309os-remotely-piloted-aircraft-systems>. Acesso em: 03 nov. 2017. Citado na página 81.

COIFMAN, R. R.; WICKERHAUSER, M. V. Entropy-based algorithms for best basis selection. **IEEE Transactions on Information Theory**, 1992. v. 38, n. 2, p. 713–718, 1992. Citado na página 44.

CONNER, M. **Ikhana Unmanned Science and Research Aircraft System**. 2015. Disponível em: <https://www.nasa.gov/centers/armstrong/news/FactSheets/FS-097-DFRC.html>. Acesso em: 29 out. 2017. Citado na página 59.

CONVEYOR EQUIPMENT MANUFACTURERS ASSOCIATION. **Belt Conveyors for Bulk Materials**. 7th. ed. Naples: Conveyor Equipment Manufacturers Association, 2014. 829 p. Citado 6 vezes nas páginas 9, 29, 30, 31, 32 e 33.

CORTES, C.; VAPNIK, V. Support-vector networks. **Machine Learning**, 1995. v. 20, n. 3, p. 273–297, 1995. Citado na página 44.

CREAGH, B. Flying into the future of mining. **Australian Mining**, 2017. v. 1, n. 6, p. 2, 2017. Citado na página 54.

CUADRA, A.; WHITLOCK, C. **How drones are controlled**. 2014. Disponível em: <http://www.washingtonpost.com/wp-srv/special/national/drone-crashes/how-drones-work/>. Acesso em: 16 nov. 2017. Citado 2 vezes nas páginas 10 e 67.

CURBERA, F.; DUFTLER, M.; KHALAF, R.; NAGY, W. Unraveling the Web services web: an introduction to SOAP, WSDL, and UDDI. **IEEE Internet Computing**, 2002. v. 6, n. April, p. 86–93, 2002. Citado na página 95.

DEPARTMENT OF DEFENSE. Joint Publication 1-02 - Department of Defense Dictionary of Military and Associated Terms. 2001. p. 770, 2001. Disponível em: <https://marineparents.com/downloads/dod-terms.pdf>. Acesso em: 29 out. 2017. Citado na página 54.

DJI. **Inspire 1 - Specs, FAQ, manual, video tutorials and DJI GO**. 2017. Disponível em: <https://www.dji.com/inspire-1/info>. Acesso em: 13 nov. 2017. Citado na página 68.

____. **Matrice 600**. 2017. Disponível em: <https://www.dji.com/matrice600>. Acesso em: 02 nov. 2017. Citado 2 vezes nas páginas 61 e 139.

____. **Mavic Pro - Specs, FAQ, Tutorials and Downloads**. 2017. Disponível em: <https://www.dji.com/mavic/info>. Acesso em: 13 nov. 2017. Citado 3 vezes nas páginas 65, 66 e 68.

____. **Mobile SDK Introduction**. 2017. Disponível em: <https://developer.dji.com/mobile-sdk/documentation/introduction/mobile_sdk_introductio>. Acesso em: 24 dez. 2017. Citado 2 vezes nas páginas 111 e 112.

____. **Onboard SDK Introduction**. 2017. Disponível em: <https://developer.dji.com/onboard-sdk/documentation/introduction/onboard-sdk-introduction.html>. Acesso em: 25 dez. 2017. Citado 2 vezes nas páginas 111 e 113.

____. **SDK Architectural Overview**. 2017. Disponível em: <https://developer.dji.com/mobile-sdk/documentation/introduction/sdk_architectural_overvie>. Acesso em: 24 dez. 2017. Citado 3 vezes nas páginas 11, 113 e 114.

____. **Testing, Profiling and Debugging - DJI Mobile SDK Documentation**. 2017. Disponível em: <https://developer.dji.com/mobile-sdk/documentation/application-development-workflow/workflow-testing.html>. Acesso em: 09 set. 2017. Citado 4 vezes nas páginas 11, 120, 121 e 122.

DOBROKHODOV, V. Kinematics and Dynamics of Fixed-Wing UAVs. In: VALAVANIS, K. P.; VACHTSEVANOS, G. J. (Ed.). **Handbook of Unmanned Aerial Vehicles**. 1. ed. [S.l.]: Springer Publishing Company, Incorporated, 2014. cap. 14, p. 244–277. Citado 2 vezes nas páginas 62 e 64.

DOGLIO, F. Rest 101. In: **Pro REST API Development with Node.js**. 1. ed. Berkeley, CA: Apress, 2015. cap. 1, p. 1–24. Citado 4 vezes nas páginas 97, 98, 99 e 100.

DOS SANTOS INTERNATIONAL. **Sandwich Belt High Angle Conveyors**. 2017. Disponível em: <http://www.dossantosintl.com/sandconv.php>. Acesso em: 26 ago. 2017. Citado na página 31.

DUNPHY, G. et al. **BizTalk 2009**. New York: Apress, 2009. 750 p. Citado 2 vezes nas páginas 89 e 90.

ELSTON, J.; STACHURA, M.; DIXON, C.; ARGROW, B.; FREW, E. W. Layered approach to networked: Command and control of complex uas. In: VALAVANIS, K. P.; VACHTSEVANOS, G. J. (Ed.). **Handbook of Unmanned Aerial Vehicles**. [S.l.]: Springer Publishing Company, Incorporated, 2014. cap. 34, p. 781–811. Citado na página 67.

ENDREI, M. et al. **Patterns: Service-Oriented Architecture and Web Services**. 1. ed. [S.l.]: IBM Redbooks, 2004. 364 p. Citado na página 90.

EPIC GAMES. **What is Unreal Engine 4**. 2017. Disponível em: <https://www.unrealengine.com/what-is-unreal-engine-4>. Acesso em: 12 jul. 2017. Citado na página 116.

ERL, T. **SOA: Principles of Service Design**. 1. ed. Upper Saddle River: Prentice Hall, 2008. 608 p. Citado 2 vezes nas páginas 87 e 93.

EUROPEAN AVIATION SAFETY AGENCY. **Notice of Proposed Amendment 2017-05**. 2017. 128 p. Disponível em: <https://www.easa.europa.eu/document-library-/notices-of-proposed-amendment/npa-2017-05>. Acesso em: 03 nov. 2017. Citado na página 81.

EUROPEAN PARLIAMENT. **Regulation (EC) No 216/2008**. 2008. 49 p. Disponível em: <http://eur-lex.europa.eu/legal-content/EN/TXT/PDF-/?uri=CELEX:32008R0216>. Acesso em: 03 nov. 2017. Citado na página 80.

FAHLSTROM, P. G.; GLEASON, T. J. **Introduction to UAV Systems**. 4th. ed. Chichester: Wiley, 2012. 312 p. Acesso em: 03 nov. 2017. Citado 5 vezes nas páginas 10, 61, 62, 63 e 66.

FAIRHURST, G. **The User Datagram Protocol (UDP)**. 2008. Disponível em: <http://www.erg.abdn.ac.uk/users/gorry/course/inet-pages/udp.html>. Acesso em: 15 ago. 2017. Citado na página 169.

FEDERAL AVIATION ADMINISTRATION. **AC 107-2 - Small Unmanned Aircraft Systems (sUAS)**. 2016. 52 p. Disponível em: <https://www.faa.gov-/regulations_policies/advisory_circulars/index.cfm/go/document.information/documentID/1019962>. Citado na página 81.

FERNANDO, C. **Rethinking Integration Solutions: From SOA to the Future**. 2016. Disponível em: <https://dzone.com/articles/building-integration-solutions-a-rethink>. Acesso em: 18 out. 2017. Citado 2 vezes nas páginas 11 e 87.

FIELDING, R. T. **Architectural Styles and the Design of Network-based Software Architectures**. 162 p. Tese (Doutorado em Informática e Ciências da Computação) — University of California, Irvine, 2000. Citado na página 98.

FLEUREAU, J.; GALVANE, Q.; TARIOLLE, F.-L.; GUILLOTEL, P. Generic Drone Control Platform for Autonomous Capture of Cinema Scenes. In: **Proceedings of the 2nd Workshop on Micro Aerial Vehicle Networks, Systems, and Applications for Civilian Use**. New York: ACM, 2016. p. 35–40. Citado na página 53.

FLIR. **Termovisores Série E**. 2017. Disponível em: <http://www.flir.com.br/instruments/display/?id=56911>. Acesso em: 30 ago. 2017. Citado na página 38.

FRANCA, A.; FERNANDES, J. **Sistema robótico para inspeção de rolos de transportadores de correia é testado no porto de Tubarão**. 2017. Disponível em: <https://valeinformar.valeglobal.net/BR/Paginas/Home-17-08-17.aspx>. Citado na página 130.

G1. **Incêndio atinge unidade da Vale Fertilizantes em Cubatão, SP**. 2017. Disponível em: <http://g1.globo.com/sp/santos-regiao/noticia/2017/01/incendio-atinge-unidade-da-vale-fertilizantes-em-cubatao-sp.html>. Acesso em: 10 fev. 2017. Citado na página 24.

GAO, Y. **Why Have Drones Suddenly Become a Big Thing?** 2015. Disponível em: <http://www.newsweek.com/quora-question-why-have-drones-suddenly-become-big-thing-312589>. Acesso em: 02 nov. 2017. Citado na página 59.

GIRDHAR, P.; SCHEFFER, C. Machinery fault diagnosis using vibration analysis. In: **Practical Machinery Vibration Analysis and Predictive Maintenance**. Oxford: Newnes, 2004. cap. 5, p. 89–133. Citado na página 39.

GONCALVES, A. SOAP Web Services. In: **Beginning Java EE 7**. Berkeley, CA: Apress, 2013. cap. 14, p. 455–494. Citado na página 97.

GONZALEZ-DUGO, V. et al. Using high resolution UAV thermal imagery to assess the variability in the water status of five fruit tree species within a commercial orchard. **Precision Agriculture**, 2013. v. 14, n. 6, p. 660–678, 2013. Citado na página 53.

GOOGLE. **Google Maps**. 2017. Disponível em: <https://goo.gl/RL9ayX>. Acesso em: 12 out. 2017. Citado 2 vezes nas páginas 13 e 164.

____. ____. 2018. Disponível em: <https://goo.gl/MEFJuV>. Acesso em: 12 jan. 2018. Citado 2 vezes nas páginas 12 e 158.

Google Trends. **MQTT, CoAP, AMQP - Google Trends**. 2018. Disponível em: <https://goo.gl/XtLRLd>. Acesso em: 04 abr. 2018. Citado 2 vezes nas páginas 11 e 110.

HAWMAN, M.; GALINAITIS, W. Acoustic emission monitoring of rolling element bearings. **IEEE 1988 Ultrasonics Symposium Proceedings**, 1988. v. 2, p. 885–889, 1988. Citado na página 38.

HCD CONVEYORS. **HCD Steel Super HD Rollers**. 2017. Disponível em: <http://www.hcdconveyors.co.nz/product/hcd-steel-super-hd-rollers/>. Acesso em: 28 ago. 2017. Citado 2 vezes nas páginas 9 e 37.

HECHT, J. Optical Dreams, Virtual Reality. **Optics & Photonics News**, 2016. n. 27, p. 24–31, 2016. Citado na página 170.

HECKE, B. V.; HE, D.; QU, Y. On the Use of Spectral Averaging of Acoustic Emission Signals for Bearing Fault Diagnostics. **Journal of Vibration and Acoustics**, 2014. ASME, v. 136, n. 6, p. 61009–61013, sep 2014. Citado na página 39.

HEMMATI, F.; ORFALI, W.; GADALA, M. S. Roller bearing acoustic signature extraction by wavelet packet transform, applications in fault detection and size estimation. **Applied Acoustics**, 2016. v. 104, p. 101–118, mar 2016. Citado na página 38.

HICKE, K.; HUSSELS, M.-T.; EISERMANN, R.; CHRUSCICKI, S.; KREBBER, K. Condition monitoring of industrial infrastructures using distributed fibre optic acoustic sensors. In: **25th Optical Fiber Sensors Conference**. Jeju: SPIE, 2017. v. 10323, p. 1–4. Citado na página 46.

HOHPE, G.; WOOLF, B. **Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions**. Boston: Addison-Wesley Professional, 2012. 735 p. Citado 2 vezes nas páginas 93 e 102.

HU, C. et al. Applications Study of Distributed Optical Fiber Sensor System in Coal Mine. **2011 Symposium on Photonics and Optoelectronics (SOPO)**, 2011. n. 6, p. 1–4, 2011. Citado na página 45.

HU, Y. C.; PATEL, M.; SABELLA, D.; SPRECHER, N.; YOUNG, V. Mobile Edge Computing - A key technology towards 5G. **ETSI White Paper**, 2015. v. 11, n. 11, p. 1–16, 2015. Citado na página 84.

IMIRANTE.COM. **Incêndio é registrado na Vale na madrugada desta segunda**. 2015. Disponível em: <http://imirante.com/sao-luis/noticias/2015/01/26/incendio-e-registrado-na-vale-na-madrugada-desta-segunda.shtml>. Acesso em: 12 out. 2016. Citado na página 24.

INDRASIRI, K. **Beginning WSO2 ESB**. 1. ed. San Jose: Apress, 2016. 285 p. Citado 3 vezes nas páginas 87, 89 e 90.

INGENIUM DESIGN. **Conveyor Belt Design Engineering**. 2017. Disponível em: <http://www.ingeniumdesign.us/services/conveyor-belt-design-engineering/>. Acesso em: 27 ago. 2017. Citado na página 30.

INTERNET ENGINEERING TASK FORCE. **Hypertext Transfer Protocol (HTTP) Status Code Registry**. 2017. Disponível em: <http://www.ietf.org/assignments/http-status-codes/http-status-codes.xml>. Acesso em: 28 nov. 2017. Citado na página 98.

JIANG, X.-p.; CAO, G.-g. Belt conveyor roller fault audio detection based on the wavelet neural network. In: **2015 11th International Conference on Natural Computation**. Zhangjiajie: IEEE, 2015. p. 954–958. Citado na página 44.

KEANE, J. F.; CARR, S. S. A Brief History of Early Unmanned Aircraft. **John Hopkins APL Technical Digest**, 2013. v. 32, n. 3, p. 558–571, 2013. Citado 3 vezes nas páginas 55, 56 e 57.

KEEN, M. et al. **Patterns: Implementing an SOA Using an Enterprise Service Bus**. 1. ed. [S.l.]: IBM Redbooks, 2004. 380 p. Citado na página 90.

KING, J. P. et al. Development of a Coherent OTDR Instrument. **Journal of Lightwave Technology**, 1987. v. 5, n. 4, p. 616–624, 1987. Citado na página 45.

KUSHLEYEV, A.; MELLINGER, D.; POWERS, C.; KUMAR, V. Towards a swarm of agile micro quadrotors. **Autonomous Robots**, 2013. v. 35, n. 4, p. 287–300, 2013. Citado 2 vezes nas páginas 60 e 61.

LAMPKIN, V. et al. **Building Smarter Planet Solutions with MQTT and IBM WebSphere MQ Telemetry**. 1. ed. [S.l.]: IBM Redbooks, 2012. 270 p. Citado 2 vezes nas páginas 103 e 109.

LEU, C. **The Secret History of World War II-Era Drones**. 2015. Disponível em: <https://www.wired.com/2015/12/the-secret-history-of-world-war-ii-era-drones/>. Acesso em: 02 nov. 2017. Citado 2 vezes nas páginas 10 e 56.

LI, W.; WANG, Z.; ZHU, Z.; ZHOU, G.; CHEN, G. Design of online monitoring and fault diagnosis system for belt conveyors based on wavelet packet decomposition and support vector machine. **Advances in Mechanical Engineering**, 2013. v. 2013, p. 10, 2013. Citado 3 vezes nas páginas 10, 44 e 45.

LIANG, B. Mobile Edge Computing. In: WONG, V. W. S.; SCHOBER, R.; NG, D. W. kwan; WANG, L.-C. (Ed.). **Key Technologies for 5G Wireless Systems**. 1. ed. Cambridge: Cambridge University Press, 2017. cap. 4, p. 76–91. Citado na página 84.

LIANG, O. **Quadcopter Hardware Overview: Every Component Explained**. 2018. Disponível em: <https://oscarliang.com/quadcopter-hardware-overview/>. Acesso em: 06 jun. 2018. Citado 2 vezes nas páginas 10 e 69.

LINDBORG, J. **About XMPP-IoT**. 2017. Disponível em: <http://www.xmpp-iot.org-/about/>. Acesso em: 06 jan. 2018. Citado na página 101.

LIOS TECHNOLOGY. **LIOS Technology - Distributed Temperature Sensing**. 2017. Disponível em: <http://www.lios-tech.com/Menu/Technology/Distributed+Temperature+Sensing/Distributed+Temperature+Sensing>. Acesso em: 21 jul. 2017. Citado na página 46.

LODEWIJKS, G. Strategies for automated maintenance of belt conveyor systems. **Bulk Solids Handling**, 2004. v. 24, n. 1, p. 16–24, 2004. Citado na página 22.

LODEWIJKS, G.; LI, W.; PANG, Y.; JIANG, X. An Application of the IoT in Belt Conveyor Systems. In: **9th International Conference on Internet and Distributed Computing Systems**. Wuhan: Springer, 2016. p. 340–351. Citado 2 vezes nas páginas 21 e 43.

LONCZYNSKI, J. P. **Aplicação de Realidade Virtual para Inspeção de Correias Transportadoras de Minério**. 56 p. Monografia (Graduação em Ciências da Computação) — Instituto de Ciências Exatas e Biológicas, Universidade Federal de Ouro Preto, Ouro Preto, 2018. Citado na página 167.

LOVETT, C. **AirLib on a Real Drone**. 2017. Disponível em: <https://github.com/Microsoft/AirSim/blob/master/docs/custom_drone>. Acesso em: 02 nov. 2017. Citado 3 vezes nas páginas 11, 118 e 119.

LUQUE-VEGA, L. F.; CASTILLO-TOLEDO, B.; LOUKIANOV, A.; GONZALEZ-JIMENEZ, L. E. Power line inspection via an unmanned aerial system based on the quadrotor helicopter. In: **Proceedings of the Mediterranean Electrotechnical Conference**. Beirut: IEEE, 2014. p. 393–397. Citado na página 53.

LUZURIAGA, J. E. et al. A comparative evaluation of AMQP and MQTT protocols over unstable and mobile networks. In: **12th Annual IEEE Consumer Communications and Networking Conference**. Las Vegas: IEEE, 2015. p. 931–936. Citado 3 vezes nas páginas 105, 108 e 109.

MARTINS, W. et al. Communication Models for Distributed Acoustic Sensing for Telemetry: A Tutorial. **IEEE Sensors Journal**, 2017. v. 17, n. 15, p. 4677–4688, 2017. Citado na página 46.

MCGUIRE, P. M. **Conveyors: Application, Selection, and Integration**. Boca Raton: CRC Press, 2009. 210 p. Citado 2 vezes nas páginas 29 e 32.

MEIER, L.; HONEGGER, D.; POLLEFEYS, M. **Pixhawk Flight Controller Hardware Project**. 2017. Disponível em: <https://pixhawk.org/>. Acesso em: 04 mar. 2017. Citado na página 118.

MEIJER, G. (Ed.). **Smart sensor systems**. New Delhi: John Wiley & Sons, 2008. 404 p. Citado na página 131.

MELLO, K. **Drones 'dançam' ao som de orquestra e fazem desenhos no céu do Rock in Rio**. 2017. Disponível em: <https://g1.globo.com/musica/rock-in-rio/2017-/noticia/drones-dancam-ao-som-de-orquestra-e-fazem-desenhos-no-ceu-do-rock-in-rio.ghtml>. Acesso em: 02 nov. 2017. Citado na página 61.

MESAS-CARRASCOSA, F.; SANTANO, D. V.; PORRAS, F. P.; MEROÑO-LARRIVA, J.; GARCÍA-FERRER, A. The Development of an Open Hardware and Software System Onboard Unmanned Aerial Vehicles to Monitor Concentrated Solar Power Plants. **Sensors**, 2017. Multidisciplinary Digital Publishing Institute, v. 17, n. 6, p. 1329, jun 2017. Citado na página 53.

METSO. **Conveyor Belts**. 2017. Disponível em: <http://www.metso.com/services-/spare-wear-parts-conveyors/conveyor-belts/>. Acesso em: 27 ago. 2017. Citado na página 30.

MICROSOFT. **GitHub - AirSim**. 2018. Disponível em: <https://github.com/Microsoft-/AirSim>. Acesso em: 28 jan. 2018. Citado na página 119.

MUEHLEN, M. Z.; NICKERSON, J. V.; SWENSON, K. D. Developing web services choreography standards - The case of REST vs. SOAP. **Decision Support Systems**, 2005. v. 40, n. 1, p. 9–29, 2005. Citado na página 100.

MUELLER, J. **Understanding SOAP and REST Basics And Differences**. 2013. Disponível em: <https://blog.smartbear.com/apis/understanding-soap-and-rest-basics>. Acesso em: 27 nov. 2017. Citado 2 vezes nas páginas 97 e 100.

MULESOFT. **Anypoint Platform: Enterprise Hybrid Integration**. 2017. Disponível em: <https://www.mulesoft.com/platform/enterprise-integration>. Acesso em: 19 nov. 2017. Citado na página 91.

MUMBAIKAR, S.; PADIYA, P. Web Services Based On SOAP and REST Principles. **International Journal of Scientific and Research Publications**, 2013. v. 3, n. 5, p. 3–6, 2013. Citado na página 100.

NAIK, N. Choice of Effective Messaging Protocols for IoT Systems : MQTT, CoAP, AMQP and HTTP. In: **2017 IEEE International Systems Engineering Symposium**. Vienna: IEEE, 2017. p. 1–7. Citado 5 vezes nas páginas 103, 105, 106, 107 e 108.

NAUGHTON, R. **Remote Piloted Aerial Vehicles - The Radioplane Target Drone**. 2006. Disponível em: <http://www.ctie.monash.edu.au/hargrave/rpav_radioplane-.h>. Acesso em: 28 out. 2017. Citado na página 57.

NEWMARCH, J. REST. In: **Network Programming with Go: Essential Skill for Using and Securing Networks**. 1. ed. [S.l.]: Apress, 2017. cap. 14, p. 221–245. Citado 2 vezes nas páginas 99 e 100.

NORRIS, R. D.; MOUTZOURIS, P. **An idler, a method for monitoring a plurality of idlers, and a conveyor system**. 2014. Disponível em: <https://patents.google.com-/patent/WO2015042661A2>. Acesso em: 10 jan. 2017. Citado na página 43.

OLIVER, D. **UAV: Info, History, Timeline**. 2017. Disponível em: <http://www.the-flightbay.com/uav/>. Acesso em: 06 jun. 2018. Citado na página 78.

ORACLE. **Overview of the JMS API**. 2013. Disponível em: <https://docs.oracle-.com/javaee/6/tutorial/doc/bncdr.html>. Acesso em: 06 jan. 2018. Citado na página 101.

PANG, H. H.; TAN, K. L. Authenticating query results in edge computing. In: **Proceedings - International Conference on Data Engineering**. Boston: IEEE, 2004. p. 560–571. Citado na página 84.

PARROT. **Swing Quadcopter Minidrone**. 2017. Disponível em: <https://www.parrot.com/global/minidrones/parrot-swing>. Acesso em: 02 nov. 2017. Citado na página 61.

PCE INSTRUMENTS. **PCE-VT 2700S**. 2017. Disponível em: <https://www.pce-instruments.com/english/measuring-instruments/test-meters/vibration-meter-pce-instruments-handheld-vibration-meter-pce-vt-2700s>. Acesso em: 30 ago. 2017. Citado na página 39.

PERTEL, V. M.; SATURNO, M.; DESCHAMPS, F.; LOURES, E. d. F. R. Analysis of IT Standards and Protocols for Industry 4.0. In: **24th International Conference on Production Research**. Poznan: DEStech Publications, 2017. p. 6. Citado na página 109.

PINKHAM, R. **REST 101: An Introduction to RESTful APIs**. 2016. Disponível em: <https://blog.smartbear.com/api-testing/an-introduction-to-restful-apis-infographic/>. Acesso em: 27 nov. 2017. Citado 2 vezes nas páginas 98 e 100.

POPE, M. T.; CUTKOSKY, M. R. Thrust-Assisted Perching and Climbing for a Bioinspired UAV. In: **5th International Conference on Biomimetic and Biohybrid Systems: Living Machines**. Edinburgh: Springer, 2016. p. 288–296. Citado na página 60.

POWERS, C.; MELLINGER, D.; KUMAR, V. Handbook of Unmanned Aerial Vehicles. In: VALAVANIS, K. P.; VACHTSEVANOS, G. J. (Ed.). **Handbook of Unmanned Aerial Vehicles**. 1. ed. Dordrecht: Springer, 2015. cap. 16, p. 307–328. Citado 2 vezes nas páginas 63 e 64.

PROBELT GLOBAL. **Pipe Conveyor Belt**. 2017. Disponível em: <http://www.pgl-.com.tw/en/2-2251-52023/product/Pipe-Conveyor-Belt-Manufacturers-id300244-.html>. Acesso em: 26 ago. 2017. Citado na página 32.

RAMBALDI, G. **Drone Governance: A Scan of Policies, Laws and Regulations Governing the Use of Unmanned Aerial Vehicles (UAVs) in 79 ACP Countries**. 2016. 22 p. (ICTs for agriculture). Disponível em: <https://publications.cta.int/media/publications/downloads/1971_PDF.pdf>. Acesso em: 03 nov. 2017. Citado na página 80.

RAMIREZ-NUNEZ, J. A.; MORALES-HERNANDEZ, L. A.; OSORNIO-RIOS, R. A.; ANTONINO-DAVIU, J. A.; ROMERO-TRONCOSO, R. J. Self-adjustment methodology of a thermal camera for detecting faults in industrial machinery. In: **Proceedings of Industrial Electronics Conference**. Florence: IEEE, 2016. p. 7119–7124. Citado na página 144.

REICKS, A. Belt conveyor idler roll behaviors. In: **Bulk Material Handling by Conveyor Belt**. 7th. ed. Littleton: Society for Mining Metallurgy & Exploration, 2008. cap. 1, p. 35–40. Citado na página 37.

RIBEIRO, L. A. **Espalhamento Raman em Fibras Ópticas com Aplicação em Sensores Distribuídos de Temperatura**. 218 p. Tese (Doutorado em Ciência e Tecnologia de Materiais e Sensores) — Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2011. Citado na página 45.

RODRÍGUEZ-DOMÍNGUEZ, C. et al. A communication model to integrate the Request-Response and the Publish-Subscribe paradigms into ubiquitous systems. **Sensors**, 2012. MDPI, v. 12, n. 6, p. 7648–68, 2012. Citado na página 93.

ROGERS, A. J. Distributed optical-fibre sensors for the measurement of pressure, strain and temperature. **Physics Reports**, 1988. v. 169, n. 2, p. 99–143, 1988. Citado na página 45.

ROSOLEM, J. B.; BASSAN, F. R.; FREITAS, D. E. de; SALGADO, F. C. Raman DTS Based on OTDR Improved by Using Gain-Controlled EDFA and Pre-Shaped Simplex Code. **IEEE Sensors Journal**, 2017. v. 17, n. 11, p. 3346–3353, jun 2017. Citado na página 46.

RUFF, T.; COLEMAN, P.; MARTINI, L. Machine-related injuries in the US mining industry and priorities for safety research. **International Journal of Injury Control and Safety Promotion**, 2011. Taylor & Francis, v. 18, n. 1, p. 11–20, 2011. Citado 3 vezes nas páginas 9, 25 e 26.

SANTOS, J. A. D. **High angle conveyor**. 1983. Disponível em: <https://www.google-.com/patents/US4609097>. Acesso em: 26 ago. 2017. Citado na página 30.

SCHEVE, T. **How the MQ-9 Reaper Works**. 2008. Disponível em: <https://science-.howstuffworks.com/reaper.htm>. Acesso em: 13 nov. 2017. Citado na página 65.

SHAH, S. **AirSim - Camera Views**. 2017. Disponível em: <https://github.com/Microsoft/AirSim/blob/master/docs/camera_views>. Acesso em: 22 ago. 2017. Citado 3 vezes nas páginas 13, 168 e 170.

SHAH, S.; DEY, D.; LOVETT, C.; KAPOOR, A. AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles. In: **Field and Service Robotics**. Zürich: [s.n.], 2017. p. 14. Citado 4 vezes nas páginas 11, 115, 116 e 117.

SHI, W.; CAO, J.; ZHANG, Q.; LI, Y.; XU, L. Edge Computing: Vision and Challenges. **IEEE Internet of Things Journal**, 2016. v. 3, n. 5, p. 637–646, 2016. Citado na página 84.

SILVA, H. M. da. **Simulação com Hardware In The Loop aplicada a Veículos Submarinos**. 92 p. Dissertação (Mestrado em Engenharia Mecatrônica) — Escola Politécnica da Universidade de São Paulo, São Paulo, 2008. Citado na página 115.

SKONNARD, A. **Understanding SOAP**. 2003. Disponível em: <https://msdn.microsoft.com/en-us/library/ms995800.aspx>. Acesso em: 25 nov. 2017. Citado 2 vezes nas páginas 96 e 97.

SONG, H.; KIM, B.-W.; MUKHERJEE, B. Long-reach optical access networks: A survey of research challenges, demonstrations, and bandwidth assignment mechanisms. **IEEE Communications Surveys & Tutorials**, 2010. v. 12, n. 1, p. 112–123, 2010. Citado na página 84.

SPOTNITZ, A. **Moving the Cloud to the Edge**. 2017. Disponível em: <https://www-.pubnub.com/blog/moving-the-cloud-to-the-edge-computing/>. Acesso em: 17 jan. 2018. Citado 2 vezes nas páginas 11 e 85.

STEIN, B.; MORRISON, A. **The Enterprise Data Lake: Better Integration and Deeper Analytics**. 2014. Disponível em: <http://www.smallake.kr/wp-content/uploads-/2017/03/20170313_074222.pdf>. Acesso em: 11 nov. 2017. Citado na página 136.

STOMP. **STOMP Protocol Specification, Version 1.2**. 2012. Disponível em: <https:-//stomp.github.io/stomp-specification-1.2.html>. Acesso em: 06 jan. 2018. Citado na página 101.

SUBBARAMAN, N. **In the virtual cockpit: What it takes to fly a drone**. 2013. Disponível em: <https://www.nbcnews.com/technology/virtual-cockpit-what-it-takes-fly-drone-1C9319684>. Acesso em: 13 nov. 2017. Citado na página 65.

SWINDERMAN, R. T.; MARTI, A. D.; GOLDBECK, L. J.; MARSHALL, D.; STREBEL, M. G. **Foundations - Guia Prático para um Controle mais Limpo, Seguro e Produtivo de Pó e Material a Granel**. 4. ed. Neponset: Martin Engineering Ltda., 2009. 562 p. Citado 3 vezes nas páginas 9, 30 e 34.

TALBOT, N. C.; ALLISON, M. T.; NICHOLS, M. E. **Centimeter accurate global positioning system receiver for on-the-fly real-time kinematic measurement and control**. USPTO, 1994. Disponível em: <https://patents.google.com/patent-/US5519620A/en>. Acesso em: 30 jan. 2018. Citado na página 161.

TAN, J.; LU, W.; AN, J.; WAN, X. Fault diagnosis method study in roller bearing based on wavelet transform and stacked auto-encoder. In: **27th Chinese Control and Decision Conference**. Qingdao: IEEE, 2015. p. 4608–4613. Citado na página 44.

TANDON, N.; CHOUDHURY, A. Review of vibration and acoustic measurement methods for the detection of defects in rolling element bearings. **Tribology International**, 1999. v. 32, n. 8, p. 469–480, 1999. Citado na página 39.

TARANTOLA, A. **This Flying Bomb Failure Was America's WWI Cruise Missile**. 2013. Disponível em: <https://gizmodo.com/this-flying-bomb-failure-was-americas-wwi-cruise-missi-1184824802>. Acesso em: 21 out. 2017. Citado na página 55.

THANGAVEL, D.; MA, X.; VALERA, A.; TAN, H.-X.; TAN, C. K.-Y. Performance evaluation of MQTT and CoAP via a common middleware. In: **Proceedings of 9th International Conference on Intelligent Sensors, Sensor Networks and Information Processing**. Cingapura: IEEE, 2014. p. 1–6. Citado 2 vezes nas páginas 101 e 106.

THE ECONOMIST. **Suddenly, there are drones everywhere**. 2015. Disponível em: <https://www.economist.com/news/science-and-technology/21684685-and-pilots-are-reporting-ever-more-close-encounters-them-suddenly-there-are>. Acesso em: 01 nov. 2017. Citado na página 59.

THU, K. M.; GAVRILOV, A. I. Designing and Modeling of Quadcopter Control System Using L1 Adaptive Control. In: **Procedia Computer Science**. Moscow: [s.n.], 2017. v. 103, p. 528–535. Citado 3 vezes nas páginas 73, 74 e 76.

TIBCO SOFTWARE. **TIBCO BusinessWorks**. 2017. Disponível em: <https://www-.tibco.com/products/tibco-businessworks>. Acesso em: 19 nov. 2017. Citado na página 91.

TOMKINS, R. **Predator drones surpass 4 million flight hours**. 2016. Disponível em: <https://www.upi.com/Defense-News/2016/09/21/Predator-drones-surpass-4-million-flight-hours/1361474467936/>. Acesso em: 29 out. 2017. Citado 2 vezes nas páginas 58 e 59.

UE SYSTEMS. **Ultraprobe 10,000**. 2017. Disponível em: <http://www.uesystems-.com/products/state-of-the-art-ultrasound-detectors/ultraprobe-10000>. Acesso em: 30 ago. 2017. Citado na página 38.

U.S. GOVERNMENT. **GPS Accuracy**. 2017. Disponível em: <https://www.gps.gov-/systems/gps/performance/accuracy/>. Acesso em: 30 jan. 2018. Citado na página 161.

VALAVANIS, K.; VACHTSEVANOS, G. **Handbook of Unmanned Aerial Vehicles**. 1. ed. Dordrecht: Springer, 2015. 3022 p. Citado 4 vezes nas páginas 10, 55, 57 e 58.

VALE. **30 anos do Terminal Marítimo de Ponta da Madeira**. 2016. Disponível em: <http://www.vale.com/brasil/PT/initiatives/innovation/30-years-marine-terminal/Paginas/default.aspx>. Acesso em: 15 nov. 2017. Citado na página 162.

____. **Vale comemora 10 anos da Mina de Brucutu**. 2016. Disponível em: <http://www.vale.com/brasil/PT/aboutvale/news/Paginas/cale-comemora-10-anos-mina-brucutu.aspx>. Acesso em: 24 dez. 2017. Citado na página 157.

VAQUERO, L. M.; RODERO-MERINO, L. Finding your way in the fog: Towards a comprehensive definition of fog computing. **ACM SIGCOMM Computer Communication Review**, 2014. ACM, v. 44, n. 5, p. 27–32, 2014. Citado na página 84.

VAYERON PTY. **Smart Idler Technology**. 2016. Disponível em: <http://vayeron-.com.au/tech/>. Acesso em: 13 jan. 2017. Citado 2 vezes nas páginas 9 e 43.

VINOSKI, S. Advanced message queuing protocol. **IEEE Internet Computing**, 2006. v. 10, n. 6, p. 87–89, nov 2006. Citado na página 104.

VINTAGE WINGS OF CANADA. **The Mother of All Drones**. 2013. Disponível em: <http://www.vintagewings.ca/VintageNews/Stories/tabid/116/articleType-/ArticleView/articleId/484/The-Mother-of-All-Drones.aspx>. Acesso em: 28 out. 2017. Citado na página 57.

WANG, D. **Quadcopter Parts: What are they and what do they do?** 2015. Disponível em: <http://www.quadcopteracademy.com/quadcopter-parts-what-are-they-and-what-do-they-do/>. Acesso em: 06 jun. 2018. Citado na página 69.

WILSON, P. Conveyor condition monitoring with fibre optic acoustics. **Australian Bulk Handling Review**, 2017. Rozelle, v. 22, n. 3, p. 17–20, 2017. Citado na página 46.

WU, G.; TALWAR, S.; JOHNSSON, K.; HIMAYAT, N.; JOHNSON, K. D. M2M: From mobile to embedded internet. **IEEE Communications Magazine**, 2011. v. 49, n. 4, p. 36–43, 2011. Citado na página 101.

YALI, Y.; YUANXI, W. Controller Design of Quadrotor Aerial Robot. **Physics Procedia International Conference on Medical Physics and Biomedical Engineering**, 2012. v. 33, p. 1254–1260, 2012. Citado 3 vezes nas páginas 72, 74 e 75.

YANG, B. **Fibre optic conveyor monitoring system**. 113 p. Dissertação (Mestrado em Engenharia Mecânica) — School of Mechanical and Engineering, The University of Queensland, Queensland, 2014. Citado 6 vezes nas páginas 10, 21, 38, 39, 45 e 46.

YANG, W.; ZHANG, X.; MA, H. An inspection robot using infrared thermography for belt conveyor. In: **2016 13th International Conference on Ubiquitous Robots and Ambient Intelligence**. Xi'an: IEEE, 2016. p. 400–404. Citado 2 vezes nas páginas 10 e 47.

YONG, R.; GONG, W.; SHEN, M. Z.; GUOAN, G. **Belt conveyor automatic inspection system and method based on multi-rotor unmanned aerial vehicle**. 2014. 8 p. Disponível em: <https://patents.google.com/patent/CN103869819B>. Acesso em: 06 dez. 2016. Citado na página 48.

ZOOMLINE. **Pipe Belt Conveyor**. 2017. Disponível em: <http://www.beltconveyorcn.com/en/beltconveyorseries/2017/04-26/28.html>. Acesso em: 26 ago. 2017. Citado na página 32.