



# Digitalização de Ativos Industriais baseados em Asset Administration Shells

**Iury Michel Treuk - 57014**

Dissertação apresentada para a Escola Superior de Tecnologia e Gestão de Bragança para obter o diploma de Mestrado em Engenharia Eletrotécnica e de Computadores. Trabalho desenvolvido durante o programa de dupla diplomação entre o Instituto Politécnico de Bragança (IPB) e a Universidade Tecnológica Federal do Paraná (UTFPR).

Trabalho orientado por:
Prof. PhD. Paulo Jorge Pinto Leitão
Prof. PhD. Joaquim de Mira Junior

Bragança
2025



# Digitalização de Ativos Industriais baseados em Asset Administration Shells

**Iury Michel Treuk - 57014**

Dissertação apresentada para a Escola Superior de Tecnologia e Gestão de Bragança para obter o diploma de Mestrado em Engenharia Eletrotécnica e de Computadores. Trabalho desenvolvido durante o programa de dupla diplomação entre o Instituto Politécnico de Bragança (IPB) e a Universidade Tecnológica Federal do Paraná (UTFPR).

Trabalho orientado por:
Prof. PhD. Paulo Jorge Pinto Leitão
Prof. PhD. Joaquim de Mira Junior

Bragança
2025

# Dedicatória

Dedico este trabalho aos meus pais, por acreditarem em mim e oferecerem todo o suporte necessário ao longo da minha trajetória. Dedico também às minhas irmãs e amigos, que estiveram presentes durante esta jornada, tornando-a mais leve e cheia de momentos descontraídos.

# Agradecimentos

Primeiramente, gostaria de agradecer a minha família, meu pai Velislei, minha mãe Janete, minhas irmãs Pâmela e Brenda. Muito obrigado! O apoio de vocês foi essencial para eu atingir meus objetivos na implementação e finalização deste projeto.

Eu quero agradecer ao meu orientador do Instituto Politécnico de Bragança (IPB), Professor PhD. Paulo Leitão, por sua orientação, paciência e apoio durante todo o desenvolvimento da dissertação. Agradeço também pelo voto de confiança e por todos os ensinamentos compartilhados, que contribuíram para o meu crescimento acadêmico e profissional. Os momentos de incentivo e discussão de ideias foram essenciais para a realização deste trabalho.

Também sou grato ao meu co-orientador da Universidade Tecnológica Federal do Paraná (UTFPR), Professor PhD. Joaquim de Mira, por todo o conhecimento e apoio durante o desenvolvimento deste projeto. Sou grato também pela oportunidade e pelas orientações recebidas ao longo do caminho da dupla diplomação durante meu período de intercâmbio.

Eu também quero agradecer ao meu colega de pesquisa Alexandre, obrigado por todos os direcionamentos e conhecimentos passados, pela parceria e apoio constante durante todo o processo de pesquisa.

Por fim, quero agradecer aos meus amigos Arthur, Yasmin, Vini, Dalila, Felipe, Nathan, Lais e Matheus, que estiveram presentes durante toda esta jornada e tornaram este momento muito mais divertido.

# Resumo

Impulsionados por novas tecnologias digitais, o desenvolvimento em sistemas de manufatura industrial está transformando os métodos tradicionais de operação adotados pelas empresas, otimizando os processos produtivos e a capacidade de inovação. A implementação destas tecnologias permitiu a digitalização de ativos industriais, possibilitando o monitoramento em tempo real, a análise preditiva e a tomada de decisões mais rápidas quando comparadas às tecnologias anteriores. Assim, emergindo na Indústria 4.0, o Asset Administration Shell aparece como uma forma de representação digital de um ativo, informando suas características técnicas e de identificação, bem como seus dados operacionais, por meio de uma estrutura de elementos modulares e hierárquicos, garantindo desse modo a padronização de componentes e a interoperabilidade entre sistemas. Neste trabalho será apresentada uma abordagem baseada em um cenário real de linha de montagem automotiva, com o intuito de implementar e digitalizar os equipamentos de medição geométrica presentes na fábrica, utilizando os conceitos e normas estabelecidos pelo Asset Administration Shell. Com suporte da ferramenta de interface gráfica AASX Package Explorer, será desenvolvido o modelo passivo do ativo e, posteriormente, o modelo reativo, este que utilizará o *framework* FastAPI para o gerenciamento em tempo real dos dados obtidos junto ao modelo da estrutura. Os desafios enfrentados e os resultados obtidos no estudo serão discutidos de forma crítica, relacionando-os com as tecnologias atuais aplicadas no cenário industrial.

**Palavras-chave:** Asset Administration Shell; Indústria 4.0; RAMI 4.0; Digitalização; Sistemas ciberfísicos; AASX Package Explorer; FastAPI.

# Abstract

Driven by new digital technologies, developments in industrial manufacturing systems are transforming the traditional operating methods adopted by companies, optimizing production processes and innovation capacity. The implementation of these technologies has made it possible to digitize industrial assets, enabling real-time monitoring, predictive analysis and faster decision-making compared to previous technologies. Emerging in Industry 4.0, the Asset Administration Shell appears as a form of digital representation of an asset, informing its technical and identification characteristics, as well as its operational data, through an architecture of modular and hierarchical elements, ensuring the standardization of components and interoperability between systems. This paper will present an approach based on a real automotive assembly line scenario, with the purpose of implementing and digitizing the geometric measuring equipment present in the factory, using the concepts and standards established by the Asset Administration Shell. With the support of the AASX Package Explorer graphical interface tool, the passive model of the asset will be developed, followed by the reactive model, which will use the FastAPI framework for real-time management of the data obtained alongside the model. The challenges faced and the results obtained in the study will be discussed critically, relating them to the theories and methods used.

**Keywords:** Asset Administration Shell; Industry 4.0; RAMI 4.0; Digitization; Cyber-physical systems; AASX Package Explorer; FastAPI.

# Conteúdo

# Lista de Tabelas

# Lista de Figuras

# Acrônimos

**AAS** *Asset Administration Shell.*

**API** *Application Programming Interface.*

**CPES** *Cyber-Physical Enterprise System.*

**CPS** *Cyber-physical Systems.*

**DF** *Digital Factory.*

**DT** *Digital Twin.*

**HTTP** *Hypertext Transfer Protocol.*

**I4.0** Indústria 4.0.

**IDTA** *Industrial Digital Twin Association.*

**IEC** *International Electrotechnical Commission.*

**IIRA** *Industrial Internet Reference Architecture.*

**IoT** *Internet of Things.*

**IoT-A** *Internet of Thing - Architecture.*

**IPB** Instituto Politécnico de Bragança.

**IRDI** *International Registration Data Identifier.*

**IRI** *Internationalized Resource Identifier*.

**JSON** *JavaScript Object Notation*.

**MQTT** *Message Queuing Telemetry Transport*.

**OPC-UA** *Open Platform Communications-Unified Architecture*.

**RAMI 4.0** *Reference Architectural Model Industrie 4.0*.

**REST** *Representational State Transfer*.

**SGAM** *Smart Grid Architecture Model*.

**UTFPR** Universidade Tecnológica Federal do Paraná.

**XML** *Extensible Markup Language*.

**ZVEI** *German Electrical and Electronic Manufacturers Association*.

# Capítulo 1

# Introdução

O desenvolvimento repentino das sociedades nos quesitos tecnológicos e de produções em rede, viabiliza a aplicabilidade da Indústria 4.0 (I4.0) em sistemas inovadores, onde este atua como um modelo de manufatura avançado composto de diversos conjuntos de tecnologias e informações integrados entre si [1]. Caracterizado principalmente pelo seu alto desempenho em áreas de virtualização e modularidade, a I4.0 permitiu a integração entre sistemas e ativos físicos, assim como a interoperabilidade através de seus sistemas e a digitalização de processos, exibindo sistemas avançados de controle e uma maior adaptabilidade ao avanço tecnológico presente no meio industrial [2].

No presente cenário, diversas companhias iniciam seu processo de investimento em modelos de tecnologia que suportem a interoperabilidade entre diversas infraestruturas, buscando flexibilidade comunicativa e agilidade operacional entre sistemas pertencentes a empresas parceiras [3]. A ausência da integração destes sistemas tende a reduzir o alcance de resultados bem-sucedidos e pré-determinados anteriormente, podendo esta ser uma consequência de diferentes protocolos de comunicação utilizados durante a troca de informação entre os sistemas ou até mesmo o surgimento de problemas de interoperabilidade entre dispositivos conjuntos pertencentes a um mesmo ambiente [4].

Desta forma, com o intuito de solucionar e amenizar estes desafios, os *Cyber-physical Systems* (CPS) surgem sendo capazes de integrar elementos dos mundos físico e digital por intermédio de tecnologias que viabilizam o uso de equipamentos autônomos e confiáveis

na I4.0 [5]. Além disso, o modelo tridimensional de seis camadas, também conhecido como *Reference Architectural Model Industrie 4.0* (RAMI 4.0), permite a aquisição de dados em tempo real destes sistemas, dando suporte para aplicações que realizam a digitalização de processos e produtos no atual setor industrial [6].

Desse modo, o *Asset Administration Shell* (AAS) emerge como uma representação digital de ativos físicos constituídos na I4.0, garantindo a padronização de componentes em sua representação digital, assim como a interoperabilidade de sistemas no meio tecnológico. Com seus submodelos que informam e descrevem os dados e funcionalidades de um ativo, o AAS é classificado em três diferentes tipos de acordo com seu nível de interação. Primeiramente, o modelo passivo (AAS Tipo 1) é dado como um ficheiro estático que representa um ativo físico. Para o segundo caso, o modelo reativo (AAS Tipo 2), ao contrário do primeiro modelo, estabelece uma comunicação entre os ativos e os modelos do AAS por meio de uma *Application Programming Interface* (API). Por fim, o modelo proativo (AAS Tipo 3) permite a interação autônoma e inteligente entre modelos do AAS [7], podendo ser implementado através dos denominados sistemas multiagentes [8].

Neste contexto, este trabalho tem como objetivo apresentar um caso de estudo em um cenário real de linha de montagem automotiva, permitindo a digitalização de ativos ao utilizar os conceitos e padrões do AAS. Como parte do escopo do *Horizon Europe project openZDM* (*openZDM*), o desenvolvimento do projeto conta primeiramente com o uso da ferramenta *AASX Package Explorer* para a criação do AAS Tipo 1 do sistema de medição geométrico. Após isso, será possível observar o desenvolvimento do AAS Tipo 2 para o controle em tempo real dos dados coletados pelo sistema, visando à detecção de falhas e análises dos dados obtidos durante o funcionamento do arranjo. Por questões de confidencialidade, algumas designações de dispositivos, estações e parâmetros serão alteradas, utilizando dados fictícios para garantir a proteção das informações sem comprometer a análise do projeto.

## 1.1 Objetivos

O principal objetivo desta dissertação é demonstrar, por intermédio de parte do escopo do projeto *openZDM*, o processo adotado para realizar a implementação e digitalização de ativos industriais na área automobilística. Desenvolvida com base em uma linha de montagem automotiva de um cenário real, será mostrado o progresso da criação dos modelos passivo e reativo de equipamentos de medição industrial, tendo como suporte a ferramenta de interface gráfica *AASX Package Explorer*, assim como as bibliotecas e estruturas disponibilizadas pela linguagem de programação Python.

A metodologia para alcançar este objetivo compreende a execução das seguintes etapas:

- Estudo e investigação dos CPSs, analisando as possíveis ferramentas da arquitetura RAMI 4.0 em um cenário industrial ao garantir a interoperabilidade e padronização entre sistemas.

- Implementação do AAS Tipo 1 de equipamentos de medição geométricos por meio da ferramenta de interface gráfica e edição de submodelo *AASX Package Explorer*.

- Desenvolvimento do AAS Tipo 2 para o gerenciamento em tempo real dos dados obtidos dos dispositivos de medição, visando a análise e detecção de falha com base em uma *Representational State Transfer* (REST) API.

## 1.2 Estrutura do Documento

Este documento é dividido em 6 partes para descrever as especificações realizadas ao longo da elaboração da tese.

A Introdução é descrita no Capítulo 1, mostrando os desafios presentes na I4.0 quanto à relação de interoperabilidade entre sistemas e as consequências desta na integração de dispositivos, sendo proposto desta maneira a implementação dos modelos passivo e reativo do AAS para a digitalização de ativos em um cenário real da indústria automotiva.

O Capítulo 2 apresenta o Estado da Arte sobre os CPSs, ressaltando a arquitetura RAMI 4.0 e sua influência para com o surgimento das definições do AAS, informando as dificuldades e possibilidades de interoperabilidade da ferramenta *AASX Package Explorer* para a aplicação em ativos industriais. Também será tratado sobre a semântica do AAS, mostrando a importância e flexibilidade por meio de algumas pesquisas que utilizam os conceitos e especificações da estrutura do AAS.

O Caso de Estudo será mostrado no Capítulo 3, contextualizando o projeto *openZDM* ao detalhar os principais objetivos durante a implementação do trabalho, assim como os meios e métodos necessários para realizar a construção e desenvolvimento do estudo em um cenário automotivo.

O Capítulo 4 demonstra a metodologia para a implementação dos sistemas de medição geométricos, onde será realizada a digitalização de ativos desses sistemas ao apresentar os desenvolvimentos dos modelos passivo e reativo disponibilizados pela estrutura do AAS.

O Capítulo 5 irá discutir os resultados obtidos durante o desenvolvimento dos modelos passivo e reativo dos equipamentos de medição geométricos, além das dificuldades durante a implementação dos ativos.

Por fim, o Capítulo 6 irá descrever as conclusões sobre o estudo e os possíveis desenvolvimentos e trabalhos futuros em um cenário futuro.

# Capítulo 2

# Estado da Arte

Este capítulo irá fazer uma breve contextualização sobre o AAS, apresentando os atuais desenvolvimentos de pesquisa na área, assim como os principais desafios relacionados à interoperabilidade, aplicabilidade e viabilidade de sistemas utilizando os conceitos e ferramentas do AAS.

## 2.1 Sistemas Ciberfísicos

O surgimento da I4.0 introduziu técnicas revolucionárias no desenvolvimento de inovações tecnológicas, integrando sistemas e otimizando processos, além de permitir uma maior flexibilidade e automação de processos produtivos com o uso de eletrônicos e tecnologias de informação no contexto do setor industrial [9]. Com a manifestação destas invenções, conceitos como interoperabilidade e padronização tornam-se amplamente empregadas em recentes pesquisas vinculadas ao AAS, assim como em estudos que também buscam alcançar ambientes inteligentes e eficientes de sistemas inovadores [10]

Neste contexto de indústrias inteligentes, os CPSs permitem uma conexão e coordenação entre dispositivos colaborativos de grande capacidade computacional e ativos presentes no mundo físico, administrando a interoperabilidade entre sistemas ao utilizar e prover serviços de comunicação de dados e processamento disponíveis de forma simultânea na infraestrutura de rede digital [11] [12]. Apesar da complexidade devido à necessidade

da integração entre diferentes tecnologias e protocolos de comunicação, sua aplicação vem crescendo no setor de manufatura, sendo um sistema de benefícios para a melhoria contínua da qualidade do produto, maior confiabilidade do sistema e uma melhor gestão industrial [13].

Dessa forma, designado e introduzido como Arquitetura 5C, é estabelecido um modelo piramidal de 5 níveis que tem como intuito uma maior precisão durante a implementação de CPSs no setor de manufatura, informando detalhadamente os passos para cada um de seus níveis presentes na configuração, sendo eles: conexão, conversão, cibernético, cognição e configuração [11] [12] (Figura 2.1).

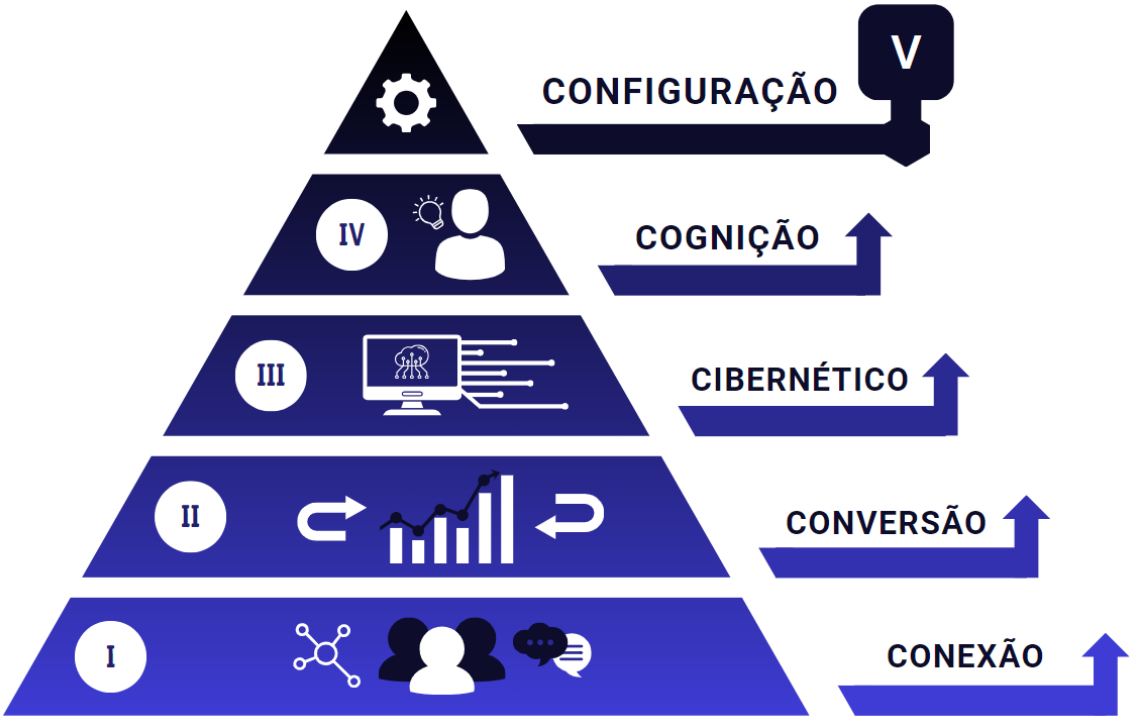


Figura 2.1: Arquitetura 5C (Adaptado de [12]).

Portanto, em um cenário empresarial de manufatura, o termo *Cyber-Physical Enterprise System* (CPES) apresenta suas capacidades operacionais básicas de computação, comunicação e controle suportadas pelas informações da estrutura piramidal, em aspectos de monitoramento e controle dos processos físicos da companhia [14]. Isto permite-nos avaliar o CPES como um sistema de auxílio para converter seus processos em redes

interconectadas e vastamente distribuídas, contribuindo de forma significativa para a interoperabilidade de sistemas.

## 2.2 RAMI 4.0

A demanda pela otimização e digitalização de processos na I4.0, junto à necessidade de modelos de padronização no setor industrial, torna-se um obstáculo para sistemas interoperáveis que desejam transferir e receber cada vez mais dados consistentes de outros sistemas presentes no mesmo ambiente digital. Abordagens promissoras sobre arquiteturas de referência da I4.0 surgem com o intuito de eliminar este problema, buscando estruturar, classificar e organizar elementos técnicos por meio de regras e conjuntos de diretrizes pré-definidas [15].

Nesta tentativa de padronização e simplificação no desenvolvimento de sistemas, três modelos estruturais principais são criados no setor da I4.0, dentre os quais *Industrial Internet Reference Architecture* (IIRA), *Internet of Thing - Architecture* (IoT-A) e RAMI 4.0 [16]. Cada uma das arquiteturas apresenta sua peculiaridade e área de maior enfoque para um mesmo tópico, por exemplo, a IIRA irá auxiliar setores da indústria que necessitam de um maior nível de detalhes na implementação de soluções da Internet das Coisas, comumente conhecido em inglês como *Internet of Things* (IoT), enquanto a IoT-A possui uma ênfase maior no desenvolvimento sobre a tecnologia da informação da I4.0. Já o modelo RAMI 4.0, em comparação aos outros dois modelos, irá solucionar de forma mais adequada o desafio mencionado anteriormente, visto que a arquitetura visa implementar otimizações com base nos conceitos da I4.0, adotando elementos de produção e logística industrial [16] [17].

Neste sentido, apresentado pela *Plattform Industrie 4.0* (Plattform Industrie 4.0) e *German Electrical and Electronic Manufacturers Association* (ZVEI) [18], o RAMI 4.0 é um modelo de arquitetura constituído de um sistema de coordenadas tridimensionais, que busca as padronizações essenciais em casos de estudo da I4.0 [19]. Adaptando e expandindo as informações da estrutura *Smart Grid Architecture Model* (SGAM) ao descrever

os aspectos fundamentais do setor, como os conceitos, conformidades e interações dentro da I4.0, o RAMI 4.0 fornece uma visão arquitetônica dos principais componentes utilizados em um ativo, aplicando um modelo de níveis composto de três eixos: *Hierarchy Levels*, *Life Cycle & Value Stream* e *Layers* [20], observado na Figura 2.2.

- *Hierarchy Levels*: indicado na região direita da Figura 2.2, este eixo representa os níveis hierárquicos definidos pela IEC 62264 [21] e IEC 61512-1 [22], estas que são responsáveis pelos série de padrões internacionais para gerenciamento de sistemas e tecnologia de informação corporativa. Ela é contida de sete elementos distintos, onde inicialmente eram pré-denominados quatro camadas (*Control Device, Station, Work Centers, Enterprise*), na intenção de agregar e classificar de forma assertiva todos os setores da indústria automotiva até o processo industrial. Estes elementos definidos de acordo com suas funcionalidades acabam por receber mais 3 camadas na arquitetura RAMI 4.0, onde inicialmente o Produto (*Product*) irá levar em conta a homogeneidade do produto a ser produzido, por segundo o Dispositivo de Campo *(Field Device)* irá permitir o controle inteligente de sistemas, e por fim o componente Mundo Conectado *(Connected World)* estará responsável pela conexão externa com a IoT por intermédio de seus diversos serviços colaborativos [23].

- *Life Cycle & Value Stream*: presente na região horizontal esquerda, o eixo baseado na IEC 62890 [24] irá representar o processo de vida útil de instalações e produtos, desde o início de seu desenvolvimento até o seu atual desenvolvimento, produção e manutenção. Ela é classificada ao diferenciar os ciclos em tipos e instâncias *(types and instances)* em sua arquitetura, onde o primeiro estará responsável pela fase do desenvolvimento e protótipo de um novo produto, enquanto o segundo terá como ideia a mudança do momento deste primeiro processo, ou seja, do tipo de fase para o tipo de instância em que o produto já estará prototipado e inicia sua real fabricação [25].

- *Layers*: constitúida de seis camadas verticais, este eixo descreve as propriedades da arquitetura de uma entidade, fornecendo em formato de camadas as definições

de suas funções e dados provenientes de funções específicas [26]. De cima para baixo, na Figura 2.2 é possível observar suas camadas separadas de acordo com suas definições, sendo elas: Negócios *(Business)*, responsável pelos requerimentos de regulação e legislação de um ativo; por segundo o Funcional *(Functional)*, encarregado de realizar a integração horizontal entre diversas funções; em terceiro a Informação *(Information)*, que representa a informação de um ativo; Comunicação *(Communication)*, envia e recebe informações de outros componentes da I4.0; após temos a Integração *(Integration)*, considerada uma camada de transição e permite conversões de dados provindos do mundo real para o mundo digital; e por último, na base da estrutura, o Ativo *(Asset)* representa uma entidade do mundo físico, sendo um processo ou objeto de grande importância para uma organização [26] [27].

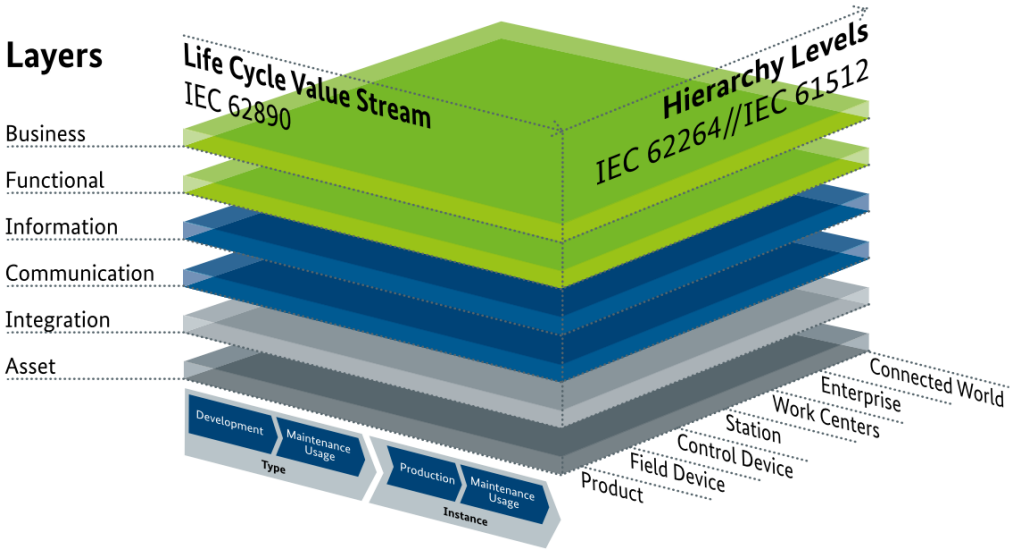


Figura 2.2: Arquitetura RAMI 4.0 [20].

Nesta contextualização, os estudos realizados em [27] e [28] mencionam os benefícios do uso da estrutura em indústrias inteligentes, destacando o uso da Arquitetura Orientada a Serviços e a associação dos elementos da tecnologia da informação para facilitar a comunicação e processos no ambiente industrial. Por exemplo, em [28] os autores implementam, baseado no RAMI 4.0, um dispositivo de controle de código aberto para a I4.0

e mencionam o estudo como uma prova de que o uso do RAMI 4.0 é coerente com este tipo de dispositivo, permitindo a sua aplicabilidade flexível em sistemas da I4.0.

Portanto, é possível imaginar as diversas possibilidades que esta arquitetura impulsionou no desenvolvimento e adaptações da I4.0, sendo desenvolvida principalmente em ambientes que visam a indústria inteligente, assim como em produtos inteligentes de grande movimento financeiro [25].

## 2.3 *Asset Administration Shell*

Esta seção analisará o AAS e sua contribuição para os processos desenvolvidos em ambientes industriais, descrevendo os conceitos e especificações utilizados para a construção da estrutura, assim como os elementos que o constituem e as definições e funções de cada componente existente em seus modelos.

### 2.3.1 Abordagem do AAS

O grande desenvolvimento tecnológico acerca da I4.0, tal como a necessidade de digitalização de objetos e processos industriais, exigiu uma nova forma de padronização comunicativa entre diferentes sistemas presentes neste meio inteligente. Com a complexidade existente nos processos de manufatura, junto à ideia de interoperabilidade inteligente entre diferentes sistemas, viabiliza o desenvolvimento de dispositivos que apresentem a habilidade de auto-descrição para um maior entendimento entre diferentes máquinas e sistemas, visando sempre uma descrição semântica própria dos dispositivos [29].

Dessa forma, com o intuito de alcançar essa padronização e permitir a interoperabilidade entre sistemas da I4.0, a arquitetura RAMI 4.0 inclui o AAS como conceito fundamental para suportar os pré-requisitos da I4.0 [30]. O AAS é definido como uma representação digital e virtual de todas as funções, particularidades e características presentes em um ativo físico, se portando como uma conexão entre objetos físicos e o mundo digital distribuído [30] [31]. Ele permite estabelecer uma comunicação entre os componentes da I4.0, auxiliando na implementação de *Digital Twin* (DT), assim como na solução

de interoperabilidade em diferentes sistemas dados por fornecedores.

Neste sentido, como parte do arranjo do AAS, o ativo é descrito por normas como um objeto físico ou lógico pertencente a uma organização, que possui um valor substancial para o seu proprietário. Ele representa ideias materiais, como é o caso de máquinas, sensores e computadores, ou até mesmo imateriais, como por exemplo processos pré-definidos, receitas e definições de equipamentos, apresentando naturezas como softwares, documentos, informações e até mesmo serviços prestados pela humanidade [26]. Quando em conjunto com o AAS, um ativo é de extrema importância para o funcionamento da arquitetura pré-definida pelo RAMI 4.0.

Portanto, o AAS possui sua construção voltada para permitir uma melhor adaptação de componentes da I4.0 no setor fabril. Baseadas no manifesto de arquitetura padronizada do AAS, mais especificamente referenciada em *International Electrotechnical Commission* (IEC) *Techinical Specification* 62832-1 [32], sua estrutura é dividida em *Digital Factory* (DF) *Header* (cabeçalho) e DF *Body* (corpo), sendo vista na Figura 2.3, onde o corpo possui informações sobre o ativo respectivo, como por exemplo seus tipos, submodelos e dicionários, e o cabeçalho diz respeito ao uso do ativo dentro do sistema, ou seja, informações para identificação e administração, além de suas funcionalidades e visualizações [33] [34]. Visando uma aquisição de dados em tempo real pelos ativos físicos, esta estrutura é definida como um dos requerimentos técnicos essenciais para a construção do AAS [35].

O AAS é distribuído e classificado em três diferentes tipos de acordo com os seus níveis de interação e possibilidades dentro de um sistema da I4.0, como observado na Figura 2.4. Para o primeiro caso, o AAS Tipo 1, também chamado de modelo passivo do AAS, irá se comportar apenas como um ficheiro estático ao representar um ativo. O modelo irá descrever todos os componentes presentes no ativo, desde as suas especificações técnicas e documentações até suas estruturas de dados. Seus ficheiros apresentam os formatos *Extensible Markup Language* (XML), *JavaScript Object Notation* (JSON) e AASX, tendo como intuito a transmissão destes dados entre parceiros distintos para a conclusão de um processo em comum e de valor entre eles [7].

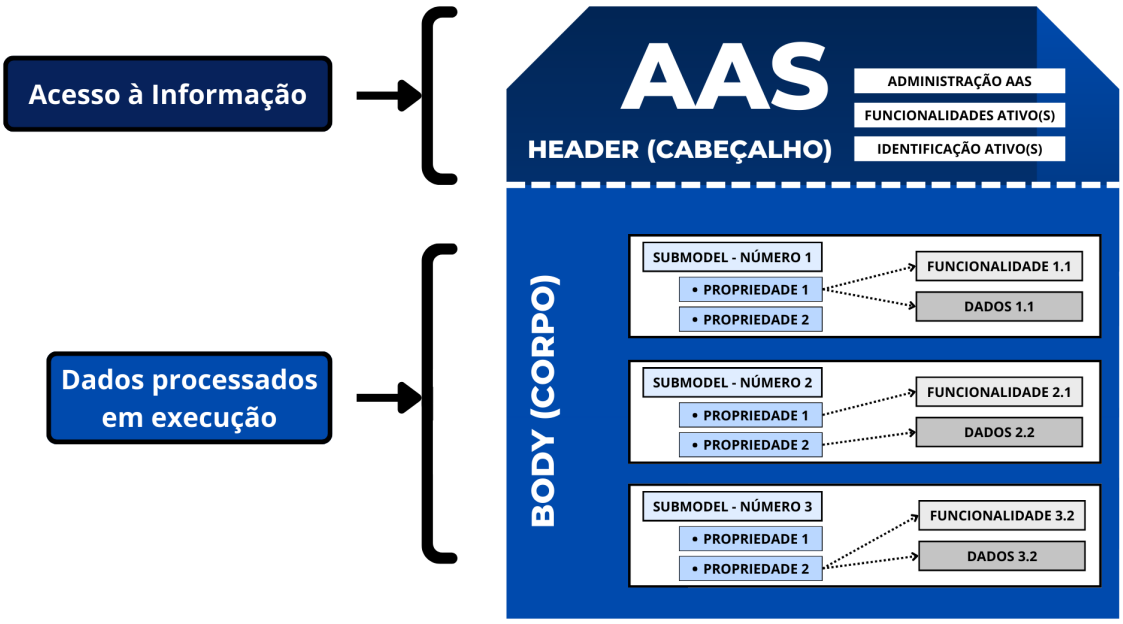


Figura 2.3: Estrutura do DF *Header* e DF *Body* (Adaptado de [26]).

Já para o modelo reativo (AAS Tipo 2), o AAS deixa de ser apenas um ficheiro estático e estabelece uma comunicação entre o ativo físico e a representação do AAS por meio de APIs, como por exemplo o servidor REST API, com auxílio dos protocolos de comunicação *Open Platform Communications-Unified Architecture* (OPC-UA), *Hypertext Transfer Protocol* (HTTP) e *Message Queuing Telemetry Transport* (MQTT). Por sua vez, o modelo ativo (AAS Tipo 3) permite uma maior autonomia dos AASs, estes que apresentarão a habilidade de se comunicar e trocar informações entre eles de forma inteligente, seguindo interfaces padronizadas com bases semânticas para permitir processos e previsões de atuação de um ativo no sistema [36] [18].

Com o intuito de tornar a tecnologia mais acessível às companhias, a plataforma *Industrial Digital Twin Association* (IDTA) nos capacita a viabilizar o uso do AAS ao padronizar sintaxes e semânticas da estrutura, disponibilizando e desenvolvendo com a comunidade normas em sua plataforma digital de acordo com as necessidades que surgem na indústria [38] [39]. A plataforma possibilita-nos acessar especificações da estrutura do AAS para uma troca significativa de informações através de seus documentos, agindo

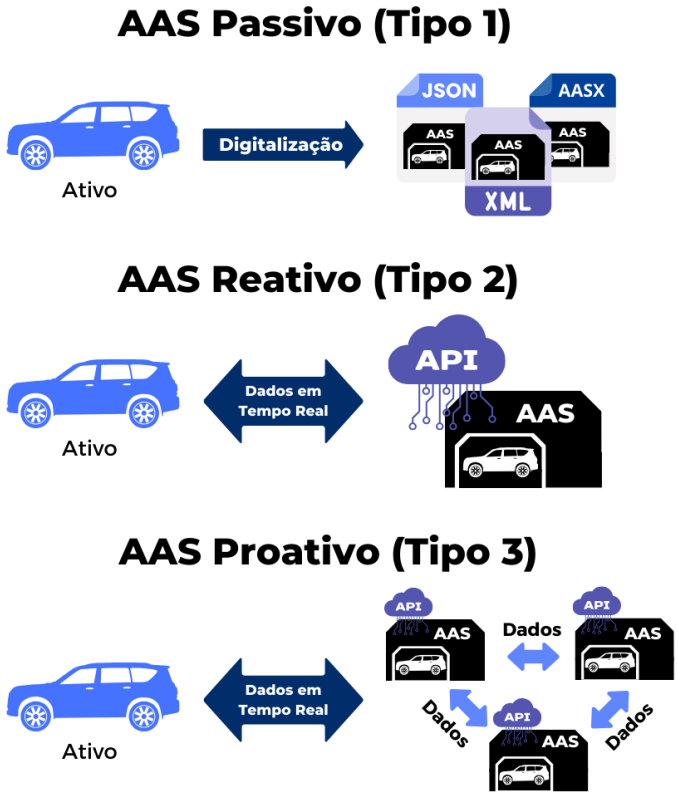


Figura 2.4: Descrição dos AASs passivo, reativo e ativo (Adaptado de [37]).

como um ponto central de contato para diversos grupos de interesse.

Na literatura, os estudos em [40] e [30] avaliam a aplicabilidade do AAS semântico ao utilizar ferramentas disponibilizadas na plataforma IDTA, assim como a interoperabilidade existente entre elementos do AAS. Mais especificamente em [40], os autores abordam as contribuições dos AASs semânticos, estas que modelam as camadas presentes na estrutura RAMI 4.0, buscando uma análise das tecnologias presentes no setor industrial. Permitindo a troca de informações entre equipamentos, eles apontam e viabilizam o uso do submodelo baseado no *Resource Description Framework* como a de maior relevância para a padronização de metadados no sistema. Contudo, como conclusão, os pesquisadores afirmam que, para um cenário futuro, ainda há a necessidade da padronização internacional destes componentes, para uma maior adaptação e padronização dos modelos de informação durante a aplicabilidade dos AASs semânticos.

### 2.3.2 *AASX Package Explorer* e *AASX Server*

O investimento na padronização de componentes na indústria, buscando a interoperabilidade de sistemas pela plataforma IDTA, tal como o surgimento dos conceitos da arquitetura RAMI 4.0, carece de uma ferramenta para comportar e aplicar os conceitos definidos pelo AAS. Desse modo, atendendo aos requisitos e sendo muito utilizada na literatura, a ferramenta *AASX Package Explorer* surge como uma interface gráfica com habilidade de edição de elementos e submodelos do AAS [41], como é vista na Figura 2.5.

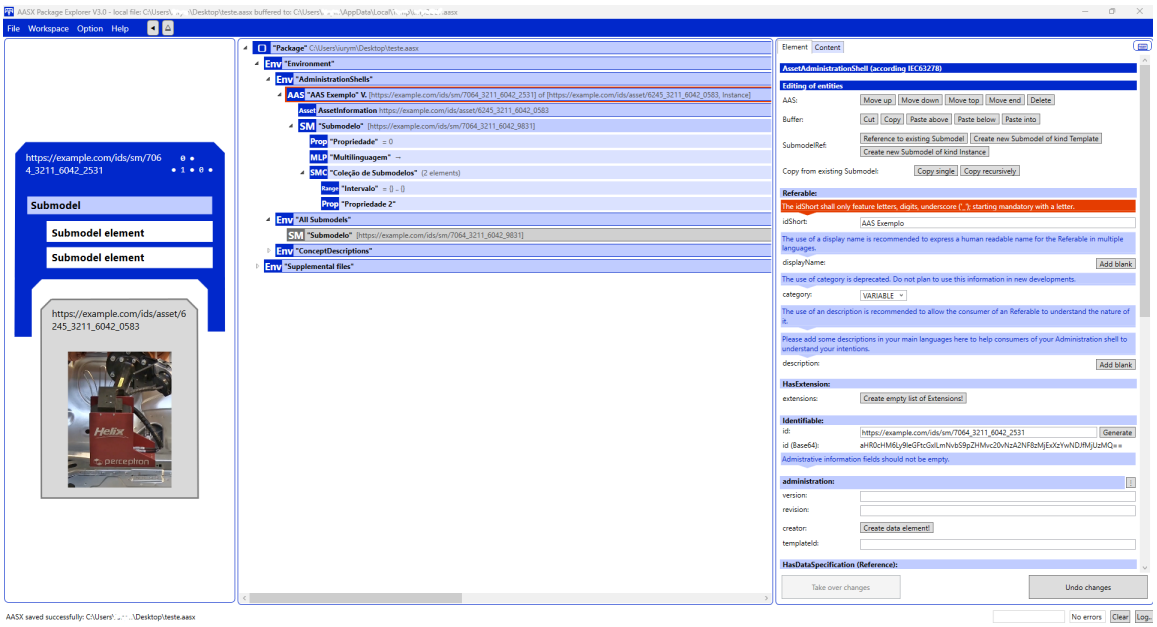


Figura 2.5: Inteface da ferramenta *AASX Package Explorer*.

A ferramenta, utilizada para a implementação de modelos passivos do AAS, possui em sua interface interativa blocos de informação que permitem a análise dos dados referentes ao ativo contido na estrutura *Administration Shell*. Sua hierarquia inicia com o ambiente do Administration Shell, onde estarão contidas em seu interior todas as informações do ativo, assim como o elemento do próprio ativo.

Abaixo dela, seguindo a ordem da estrutura, surge o componente *Submodel*, sendo caracterizado como um container de *SubmodelElements* que irá definir a estrutura hierárquica do modelo do AAS. Já os *SubmodelElements* são elementos presentes em submodelos, demonstrando diferentes tipos de segmentos de dados, por exemplo *Property*, *Range*,

*MultiLanguageProperty* e *ReferenceElement*, contendo uma definição semântica (*semanticID*) para garantir a interpretação padronizada de dados [42]. Essa estrutura é observada com mais detalhes na Figura 2.6, onde 0 e 1 indicam, respectivamente, a necessidade e a obrigatoriedade do elemento em um ativo.

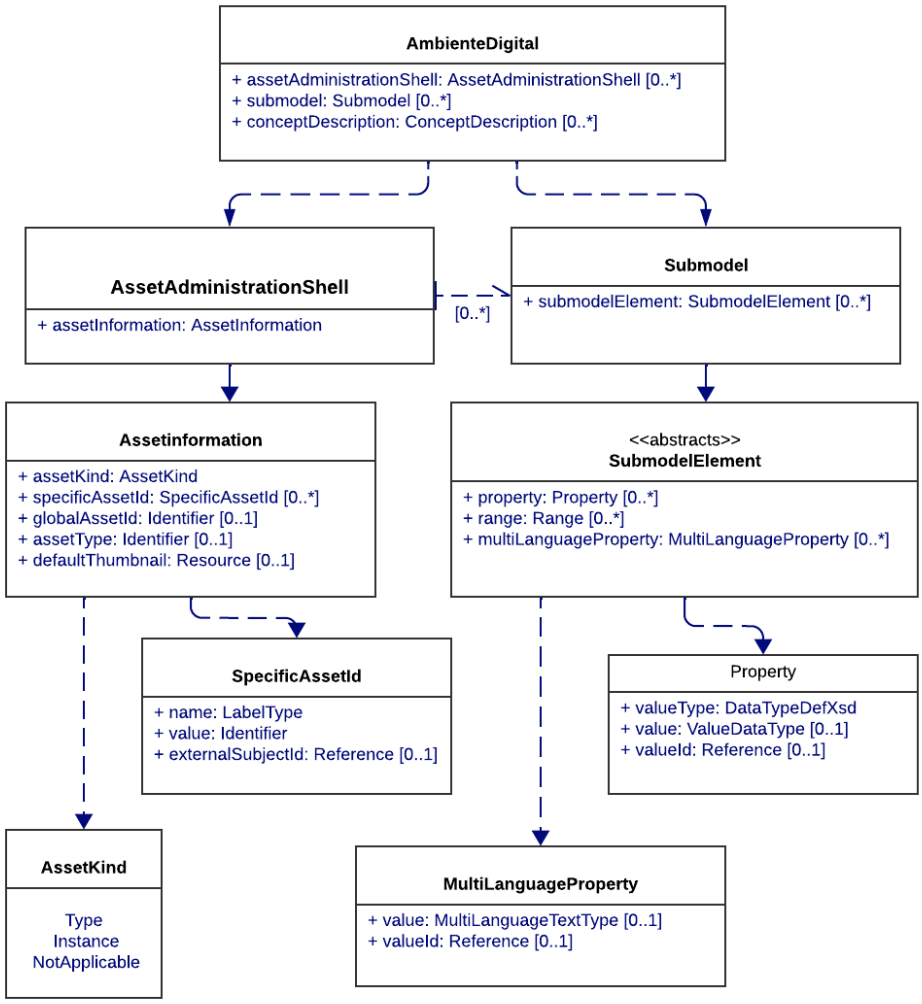


Figura 2.6: Estrutura e funcionalidades do AAS em ambiente digital (Adaptado de [42]).

O primeiro componente *Property* é um elemento de dados que contém valores simples, tais como *xs:string*, *xs:int*, *xs:date*, *xs:year*, entre outros que facilitam a classificação do tipo de dado provido para o sistema. De acordo com a especificação da IEC 61360, ele é capaz de fornecer as informações e funções do *Administration Shell*, sendo necessário um número mínimo de propriedades para uma boa definição do AAS. Além deste, outros

elementos são retratados no interior do componente *Submodel*, auxiliando na descrição e transmissão de informações do ativo de forma mais precisa e adequada, conforme o esquema mostrado na Figura 2.7 [42].

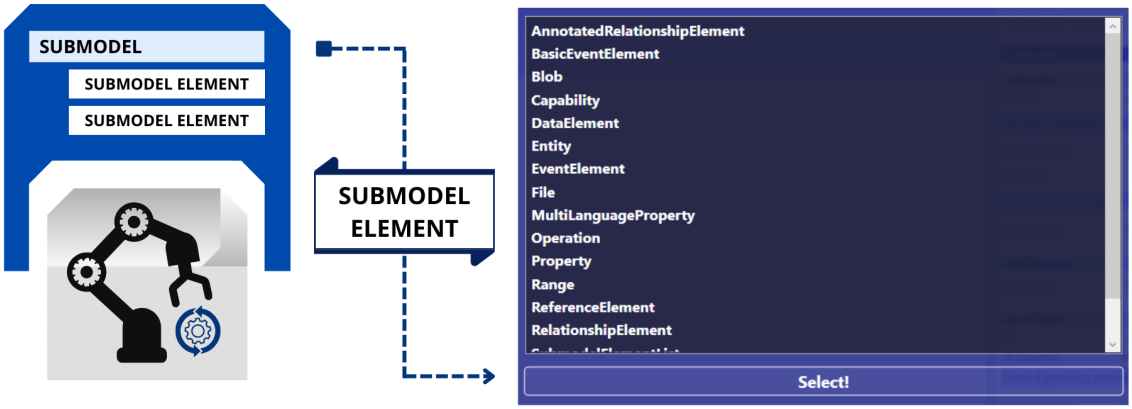


Figura 2.7: Elementos de um submodelo (Adaptado de [42]).

Para cada um dos elementos construídos na ferramenta, com exceção dos *Submodels*, tais como o AAS e *ConceptDescriptions*, assim como o próprio ativo, é preciso uma identificação global única, ou seja, um único identificador (ID), onde o *Administration Shell* estará responsável por representar apenas um ativo deste modelo. Para isso, atualmente existem dois identificadores classificados e padronizados pelas normas ISO29002-5, ISO IEC 6523 e ISO IEC 11179-6 [43], sendo eles o *International Registration Data Identifier* (IRDI) e *Internationalized Resource Identifier* (IRI). O IRDI é tipicamente utilizado para reconhecer características específicas de produtos e componentes em um sistema de classificação, tais como seus atributos, descrições e capacidades de conexão. Ao contrário deste, o IRI é empregado para identificar de forma única os elementos, produtos e serviços na estrutura do AAS, permitindo que diferentes entidades descrevam os mesmos recursos de forma a manter a interoperabilidade.

Por último, localizados ao fim da estrutura do AAS, os *ConceptDescriptions* são a base para definir os submodelos, sendo derivadas externamente de outra definição de propriedade de um padrão externo ou internamente, como parte do ambiente AAS. Assim, definindo as propriedades e os dados utilizados no AAS, eles utilizam diversas categorias

em sua estrutura, como listado na Figura 2.8, permitindo uma compreensão coerente dos dados entre sistemas distintos e seus componentes [42].

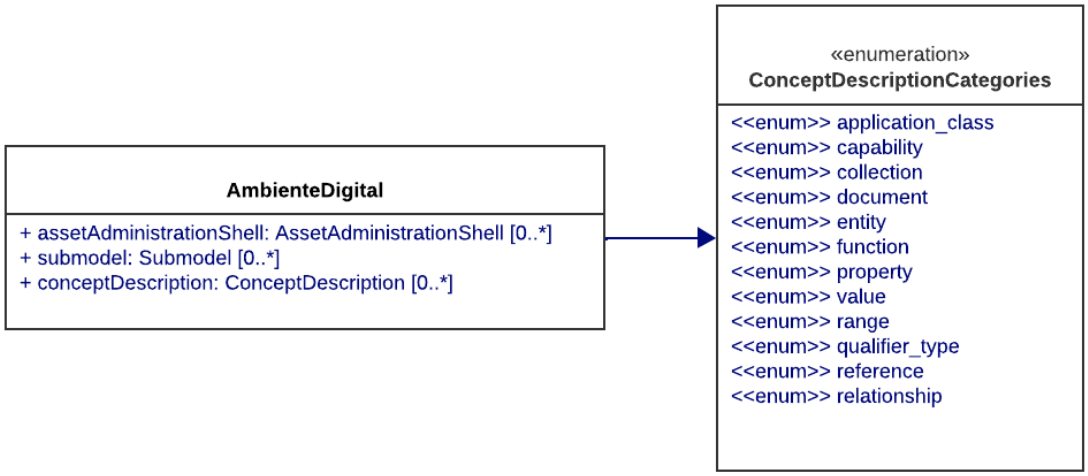


Figura 2.8: Categorias de *Concept Descriptions* (Adaptado de [42]).

Neste contexto, o *AASX Server* é uma ferramenta baseada em C# que possui a capacidade de implementar um servidor para armazenar informações contidas no AAS, realizando sua leitura e assim permitindo o desenvolvimento de modelos reativos do AAS [44]. Tendo grande compatibilidade com o *AASX Package Explorer*, este servidor utiliza os protocolos de comunicação MQTT, OPC-UA e HTTP, sendo uma boa escolha para atender às especificações da plataforma IDTA.

Com a junção e conhecimento de todos estes elementos, é possível construir os modelos do AAS, definindo de forma estável as informações de uma entidade industrial e permitindo a interoperabilidade entre sistemas que envolvam a estrutura. Ela proporciona uma base padronizada que organiza os dados de forma acessível e integrada, atendendo às demandas de conectividade e eficiência da I4.0.

Assim, algumas pesquisas que utilizam o *AASX Package Explorer* como elemento base foram desenvolvidas. Em [7] os autores buscam uma solução baseada no *AASX Package Explorer* que disponibiliza a troca de informações entre níveis de aplicativos empresariais e de controle sem perdas de dados durante o processo. Eles mencionam a habilidade de

conexão do *AASX Package Explorer*, destacando o agrupamento de informações do AAS, tais como submodelos e propriedades, além de sua capacidade de integração de dados em nível de aplicações empresariais para promover a transmissão de dados entre estes aplicativos. Baseando-se nos ficheiros providos pela IDTA, os pesquisadores validam a conversão de dados ao desenvolver um modelo AAS do controle de um motor, apontando a necessidade de operadores para finalizar o processo de conversão de dados.

Utilizando o *AASX Package Explorer* como ferramenta principal, em [45] os autores explicam a geração semi-automática dos submodelos de gerenciamento de ciclo de vida de produtos e de aplicações de software, e como realizar a integração entre estes dois elementos. Com o intuito de desenvolver uma estratégia definida como *Plm4AAS* utilizando submodelos concedidos pelo AAS, eles mencionam o uso do *AASX Package Explorer* apenas para visualizar os dados, provendo uma funcionalidade de edição básica dos componentes. Por fim, os escritores viabilizam o uso do *AASX Package Explorer* para a conexão entre os dois componentes de gerenciamento de ciclo de vida, destacando a necessidade de significados semânticos padronizados para a correlação entre submodelos do AAS para este caso de estudo.

Assim, o uso do *AASX Package Explorer* para a construção de novos projetos vem sendo cada vez mais desenvolvido para testar a sua viabilidade e possível padronização de elementos industriais de forma a manter uma interoperabilidade entre componentes da estrutura. Outros trabalhos relacionados que utilizam o *AASX Package Explorer* para desenvolvimento serão vistos no próximo tópico, onde os projetos *Eclipse BaSyx* e *NOVA Asset Administration Shell (NOVAAS)* possuem grande impacto na literatura atual.

### 2.3.3 Implementações Open-Source

A implementação do AAS vem sendo de grande pauta nos setores de pesquisa que relacionam a I4.0 e a digitalização de processos neste ambiente revolucionário, permitindo a criação de novos projetos e um maior investimento em projetos anteriores. Atualmente, as implementações open-source do AAS de grande relevância, atendendo aos requisitos de aplicação para as especificações fornecidas pela plataforma IDTA, são designadas como *AASX Server*, *Eclipse BaSyX*, *FA³ST Service* e *NOVAAS*.

A plataforma Eclipse BaSyx fornece implementações de modelos reativos do AAS (Type 2), sendo instaurada em diferentes linguagens de programação, como: C#, C++, Rust, Python e Java. Também aplicada para AASs reativos, a plataforma *FA³ST Service* é baseada em Java, que emprega uma estrutura aberta viabilizando uma fácil extensão e customização do AAS [36]. Como último ponto, desenvolvido pela *NOVA School of Science and Technology*, o *NOVAAS* é uma plataforma de implementação e execução que oferece uma interface gráfica de usuário baseada na web, possibilitando a usuários não experientes a criação de painéis de controle.

Dentro desta perspectiva, as pesquisas em [46] e [47] abordam a utilização da plataforma *open-source Eclipse BaSyx* durante a implementação de um AAS reativo no cenário da I4.0. Em [46], os desenvolvedores buscam uma maior experiência prática ao ampliar as concepções de inteligência artificial no AAS, utilizando um caso do mundo real como forma de caso de estudo. Eles afirmam que a plataforma *Eclipse BaSyx* demonstrou fácil acessibilidade aos submodelos específicos contidos na implementação do AAS, obtendo os meios necessários para a aplicação e seguimento deste caso de estudo. Contudo, os autores chamam a atenção para a compatibilidade do uso da plataforma *BaSyx* entre diferentes estruturas do projeto, ressaltando a necessidade de maiores abordagens neste tema para que as informações possam ser analisadas em diferentes plataformas providas pela arquitetura *BaSyx*.

Além disso, o desenvolvimento em [36] mostra a implementação do conceito do modelo reativo do AAS (Type 2) ao utilizar a plataforma *FA³ST* (*Fraunhofer Advanced Asset*

*Administration Shell Tools)* Service, sendo possível observar os detalhes de sua estrutura, assim como a sincronização de ativos e as suas limitações. Fazendo parte do ecossistema *FA³ST*, que busca prover ferramentas para o ciclo de vida de um DT, os autores afirmam que o *FA³ST Service* permite a sincronização de um DT com um ativo subjacente. Ainda, a pesquisa informa diferentes normas e especificações para um DT, assim como suas relações conceituais com a IoT. Os pesquisadores finalizam argumentando sobre os meios de busca de interoperabilidade pela plataforma IDTA, mencionando também a falta de linguagens oficiais disponíveis na I4.0 para a implementação do modelo proativo do AAS (Type 3) na arquitetura FA³ST Service.

Os autores em [48] possuem como intuito apresentar uma implementação de referência de um AAS definido como *NOVAAS*, este que usa como base as últimas tecnologias de programas e desenvolvimentos web disponíveis na rede. Contribuindo para o cenário, a pesquisa prevê duas entidades principais para a construção da arquitetura do NOVAAS, sendo primeiramente o manifesto, que fornece o DF *Header* e *Body* para a estrutura do AAS, e também o componente administrativo, que administra junto ao manifesto os modelos e conexões de informações do AAS presentes no sistema. Os autores viabilizam em curto prazo o uso do AAS ao citar que a implementação foi madura o suficiente para o desenvolvimento de equipamentos industriais. Para médio e longo prazo, eles definem a necessidade de otimização do *NOVAAS* para o gerenciamento de redes, mencionando para pesquisas futuras a investigação da plataforma *System of-System of AAS* junto ao funcionamento das interações entre sistemas constituídos de AASs.

Portanto, a literatura existente utilizando diferentes plataformas de implementação open-source viabiliza o desenvolvimento de novos projetos que visam o uso do modelo reativo do AAS, permitindo uma maior flexibilidade ao atender os pré-requisitos das especificações presentes na plataforma IDTA.

### 2.3.4 Protocolos de Comunicação

A padronização dos protocolos de comunicação em cenário industrial é de extrema importância para manter a interoperabilidade entre sistemas de controle e automação. Em termos de dispositivos IoT, os protocolos de comunicação MQTT, OPC-UA e HTTP são frequentemente utilizados durante o desenvolvimento de projetos que envolvem este ambiente fabril [49] [50], sendo adaptados para pesquisas que utilizam o AAS como estrutura principal de um sistema.

O MQTT é um protocolo de mensagens baseado no modelo de publicação-assinatura, que tem como intuito facilitar a comunicação assíncrona entre dispositivos. Ele é constituído de um servidor broker, utilizado como intermediário para envio de mensagens entre dois elementos, um cliente publicador e um cliente assinante. Através do servidor broker, os dois clientes irão interagir de forma assíncrona, onde o cliente publicador irá enviar uma mensagem para o servidor broker e, por meio deste, será possível observar pelo cliente assinante a mensagem enviada pelo cliente publicador [51].

Além disso, desenvolvido para facilitar a interoperabilidade entre sistemas, o OPC-UA é um padrão de comunicação industrial que fornece uma arquitetura de comunicação confiável ao realizar trocas de dados em tempo real entre diferentes sistemas e aplicativos da I4.0 [52].

Na literatura, o desenvolvimento em [53] utiliza o protocolo OPC-UA para a implementação de um AAS, criando um sensor inteligente como um CPS específico da I4.0. Os pesquisadores afirmam que o protocolo de comunicação OPC-UA, junto aos conceitos do AAS, em termos de comunicação e semântica, permite a criação de um dispositivo incorporado interoperável com os componentes da I4.0. Assim, os autores afirmam que o protocolo OPC-UA atende aos requisitos para a troca de dados nesta implementação, sendo de grande contribuição para impulsionar a digitalização no setor industrial.

Ainda, os estudos em [54] e [55] discutem sobre o uso do AAS em termos de estrutura e componentes, assim como suas especificações, utilizando os protocolos de comunicação OPC-UA, MQTT e HTTP para validar modelos comuns existentes na arquitetura. Sendo

assim, em [54] os autores comparam quatro implementações *open-source* (*AASX Server*, *Eclipse BaSyx*, *FA³ST Service*, *NOVAAS*) do AAS modelo reativo para a criação de DT, objetivando identificar ideias de melhoria para implementações futuras na indústria. Ao utilizar três protocolos de comunicação (HTTP, MQTT, OPC-UA), devido às menções nas especificações do AAS, eles criam ativos testes, fornecendo funcionalidades suficientes para comparar e testar as três interações, referindo a falta de funcionalidade por parte do HTTP para inscrição e do protocolo MQTT para leitura e execução. Assim, eles finalizam a pesquisa afirmando que as implementações do AAS suportam as funções mínimas, como leitura, escrita e comunicação via HTTP, enfatizando a falta de compatibilidade das especificações do AAS no uso de diferentes implementações.

Por fim, com o suporte do protocolo de comunicação HTTP, este que transfere dados pela *World Wide Web*, a interface de programação REST utiliza uma arquitetura que permite a comunicação entre cliente e servidor nos sistemas computador e internet [56]. Ao auxiliar na escalabilidade da implementação de componentes presentes em sistemas de multimídia interativa, este serviço torna-se de grande ajuda na automação de processos e integração de sistemas da I4.0.

# Capítulo 3

# Caso de Estudo

Este capítulo contextualiza o caso de estudo do projeto automobilístico *Horizon Europe project openZDM*, mencionando os seus principais objetivos e ambições buscados durante o desenvolvimento e planejamento da proposta. Também serão vistos os obstáculos para a execução deste no cenário automotivo, esclarecendo as etapas necessárias para a implementação e, por fim,a obtenção dos dados adquiridos pelos equipamentos de medição geométricos.

O projeto *openZDM*, combinado com pesquisas atuais e avanços da tecnologia, tem como principal intuito disponibilizar uma plataforma aberta para auxiliar nos processos de zero defeitos na produção (ZDM). Reunindo as soluções de pesquisa e desenvolvimento já existentes e formando uma solução integrada, inovadora e de ponta, a proposta visa testar e aprimorar as ferramentas tecnológicas através de cinco pilotos industriais em condições reais de operação, facilitando assim a adoção da solução do *openZDM* (Figura 3.1) no ambiente fabril [57].

A execução deste projeto segue o modelo de uma linha de produção de veículos, sendo essencial que a disposição geométrica dos componentes do carro apresente um alinhamento adequado e preciso conforme dados definidos, permitindo uma maior qualidade no produto final da linha de montagem. Com o propósito de alcançar esse objetivo, é necessário medir e monitorar as coordenadas tridimensionais X, Y e Z, e os pontos *Gap* (distância entre as superfícies) e *Flush* (alinhamentos entre superfícies) em diversas etapas do processo de

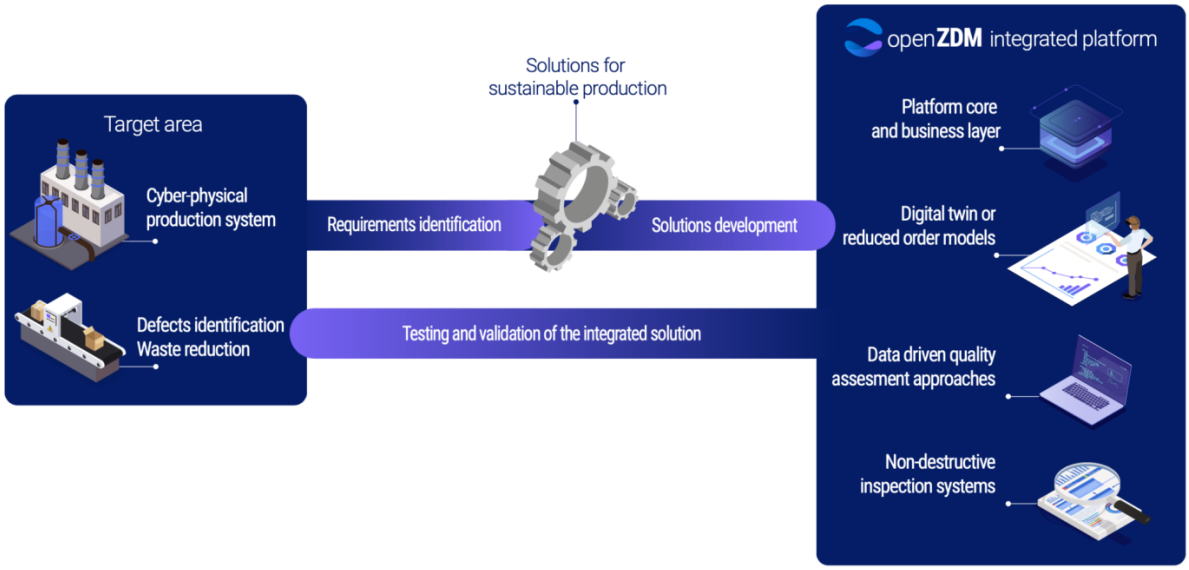


Figura 3.1: Abordagem do projeto *openZDM* [57].

produção, estes sendo representados de forma visual na Figura 3.2.

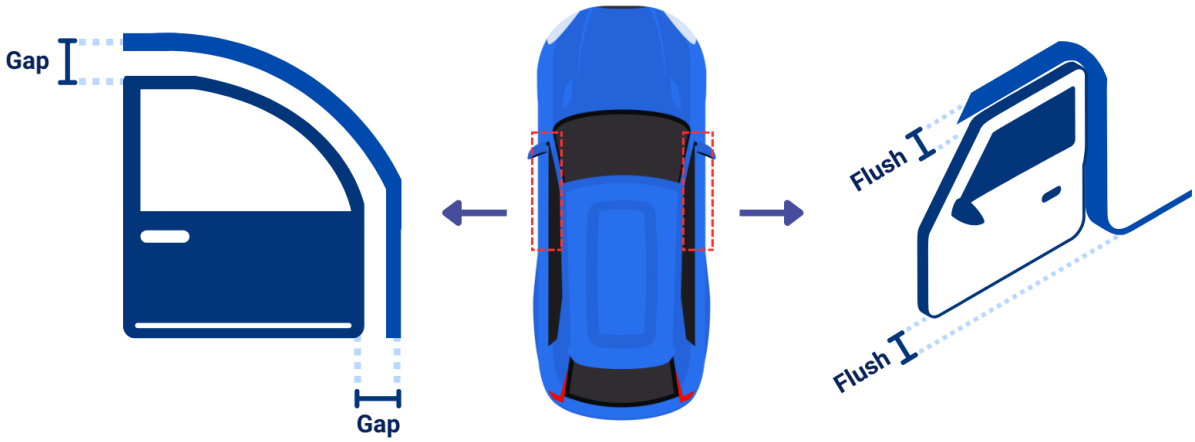


Figura 3.2: Espaçamentos *Gap* e *Flush*.

Para o início do processo de produção, os sistemas robóticos automatizados, junto aos equipamentos de medição geométricos (EMG) a laser, irão realizar a medição dos pontos *Gap* e *Flush* em diferentes estações das linhas de montagem inicial (*body shop*) e montagem final (*final assembly*). Os dados obtidos nesta etapa de medição serão inspecionados utilizando diferentes ferramentas de análise de dados, identificando a diferença de

alinhamento entre os componentes do veículo, assim como a influência e a relação entre os pontos e limites de tolerância em estações distintas da linha de montagem.

Portanto, os EMGs fixos, integrados a braços robóticos da linha de produção, estarão localizados no *bodyshop* onde atuarão na fase automatizada da construção do veículo. Por outro lado, os EMGs portáteis, operados diretamente pelos trabalhadores da fábrica, serão utilizados no *final assembly*, auxiliando nos ajustes e fixação de peças do produto. Ambos os dispositivos serão responsáveis pela coleta de dados e seu envio ao banco de dados da empresa automobilística, permitindo análises posteriores por técnicos e engenheiros encarregados pelo monitoramento do processo.

De maneira gráfica, na Figura 3.3 é possível observar as três estações da linha de montagem inicial e as estações de trabalho (*workstation*) em que o carro irá passar ao ser montado durante a esteira de produção. Primeiramente, na estação de medição 1, será medido por meio do EMG fixo a estrutura do chassi do carro, analisando os pontos tridimensionais X, Y e Z desta região do carro. Já na estação de medição 2, o EMG fixo irá verificar os pontos *Gap* e *Flush* presentes nas portas dianteiras e traseiras do carro em relação ao monobloco (estrutura superior) do veículo. Por fim, na estação de medição 3 também serão medidos os pontos *Gap* e *Flush* do automóvel, comparando-os com o capô (porta frontal) e porta-malas (porta traseira) em relação ao monobloco.

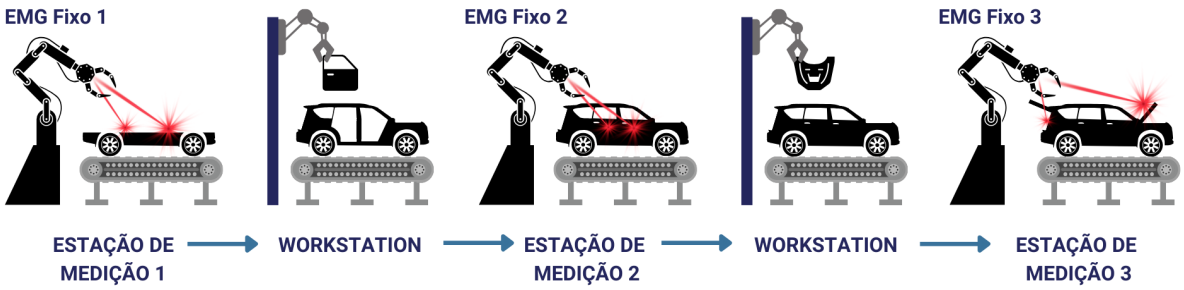


Figura 3.3: Aplicação do EMG fixo nas estações 1, 2 e 3 da linha de montagem inicial.

Adicionado a isso, na Figura 3.4 nota-se o manuseio dos equipamentos pelos operadores nas estações 4 a 23 da linha de montagem final, ocorrendo após a conclusão do processo de pintura do veículo. Para cada uma dessas estações em específico, os operadores irão utilizar o EMG portátil em diversos pontos do carro, enviando os dados obtidos

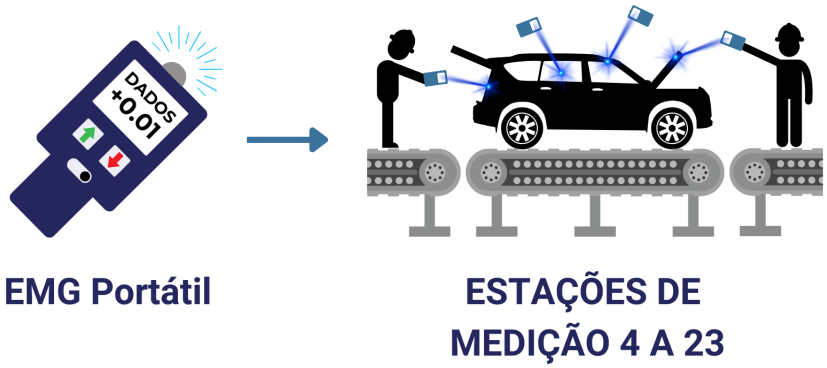


Figura 3.4: Aplicação do EMG portátil nas estações 4 a 23 da linha de montagem final.

por intermédio de um protocolo de comunicação de telemetria ao sistema da empresa automobilística.

O gerenciamento e acesso às informações dos pontos medidos durante a abordagem do projeto *openZDM* serão de grande auxílio para mitigar desafios futuros durante a montagem do veículo, reduzindo os custos de retrabalho na produção e o número de itens que deverão ser inspecionados para correção, tal como o tempo de trabalho dedicado por um operador a esta tarefa.

# Capítulo 4

# Desenvolvimento do AAS

Esse capítulo apresenta o processo de implementação da digitalização de ativos de diferentes sistemas de medição geométricos, tendo como base um cenário real de linha de montagem automotiva. Uma visão geral do desenvolvimento dos modelos passivo e reativo será dada, utilizando como referência as especificações do AAS definidas pela IDTA e suas disponibilidades de protocolos de comunicação.

## 4.1 Implementação do AAS Tipo 1

A construção e design da estrutura para o modelo passivo do AAS foram realizados utilizando a ferramenta *AASX Package Explorer*, esta que irá suportar todos os ativos representando os EMGs durante as vinte e três diferentes estações nas linhas de montagem e produção final automotiva. Com o intuito de estruturar o AAS Tipo 1, foram adotados exemplos do repositório da ferramenta *AASX Package Explorer*, assim como as normas de padronização e interoperabilidade disponibilizadas pela plataforma IDTA.

Alguns dos conceitos do modelo do AAS citados anteriormente também são utilizados, seguindo as especificações definidas nas normas *Part 1: Metamodel - IDTA Number: 01001-3-0* [58] e *Part 3a: Data Specification - IEC 61360* [42], assim como as especificações das ISOs citadas no estado da arte, visando manter a padronização da estrutura de nossos dados em ambiente industrial.

### 4.1.1 EMGs Fixos e Portáteis

A implementação do modelo passivo do AAS é realizada para os EMGs fixos 1, 2 e 3, e portátil 4, com cada dispositivo recebendo seu modelo AAS individual. Os modelos de estrutura do AAS de ambos os dispositivos serão semelhantes, contendo informações distintas conforme especificações técnicas e pontos de medição do sistema de medição.

Assim, para as três primeiras estações da linha de produção automotiva, cada EMG fixo será vinculado a um modelo individual do AAS, resultando em um total de três modelos distintos. Já a partir da estação de medição 4 até a estação de medição 23, será construído um único modelo do AAS, associado ao EMG portátil.

Nas estações 5 a 23, os dispositivos portáteis adotarão como referência o modelo passivo do EMG portátil 4, considerando que todas essas estações utilizam o mesmo tipo de equipamento de medição. Dessa forma, os dados técnicos permanecem idênticos e os pontos de medição são compartilhados ao longo da linha de montagem final.

A distribuição dos modelos do AAS será organizada da seguinte forma:

- EMG Fixo 1 – associado à estação de medição 1 (*bodyshop*);
- EMG Fixo 2 – associado à estação de medição 2 (*bodyshop*);
- EMG Fixo 3 – associado à estação de medição 3 (*bodyshop*);
- EMG Portátil 4 – associado à estação de medição 4 e utilizado nas estações de medição 5 a 23 (*final assembly*).

Sendo assim, cada dispositivo apresenta seus próprios submodelos *Identification*, *Technical Data*, *Operational Data*, *Capabilities* e *Asset Interface Description* de forma individual. Esses elementos serão descritos na implementação de forma conjunta, com o intuito de garantir uma compreensão clara e completa de suas funcionalidades e interconexões.

### 4.1.2 Modelo do AAS e Ativo

O modelo do AAS estruturado para receber os dados obtidos pelos dispositivos de medição pode ser visualizado na Figura 4.1 em forma de um fluxograma, onde inicialmente

o elemento "Asset Administration Shell" irá conter todos os demais componentes da estrutura em seu interior ao se basear na norma da IEC 63278 [59], nomeando de forma distinta todas as estações da linha de montagem.

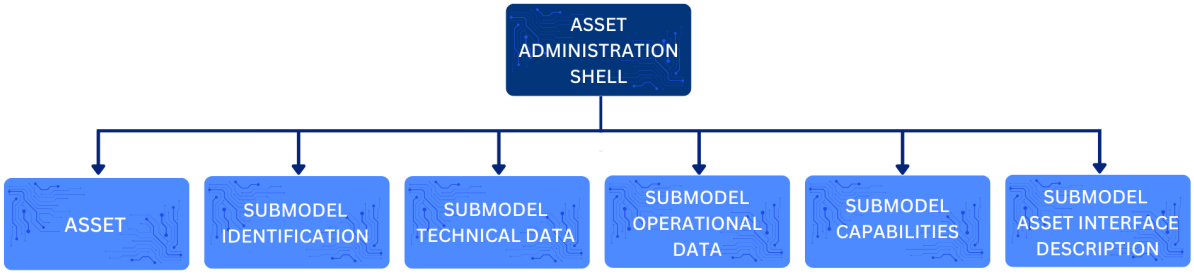


Figura 4.1: Fluxograma do modelo do AAS.

Compondo um nível abaixo da base do modelo do AAS, o primeiro elemento é o *Asset*, sendo criado automaticamente pelo AAS como uma representação digital do ativo físico. Com sua implementação baseada nos dados inseridos na plataforma *AASX Package Explorer*, presentes na Figura 4.2, este elemento permite estabelecer a conexão entre o mundo físico e o digital, sendo responsável por identificar o nosso ativo e estabelecer a integração com os modelos e submodelos presentes no AAS, diferenciando este das demais estações em que o equipamento de medição se encontra.

Portanto, a definição do AAS irá apresentar a nomenclatura de nossos EMGs fixo 1, 2 e 3, e portátil 4, conforme a estação de medição presente ao utilizar o campo *idShort*, assim como a escolha da categoria *parameter* ao representar um valor configurável antes do início da produção da linha de montagem. Suas últimas especificações contam com a definição da identificação no campo *id* e a existência dos elementos *assetInformation* e *submodel* em seu modelo.

Além disso, o elemento *Asset* irá ser definido com o tipo *Instance* devido à necessidade da descrição de um EMG físico individual, com um identificador único para o distinguir de outros que estão no mesmo sistema da empresa automotiva. Tendo em conta isso, o campo *specificAssetId* descreverá o tipo de ativo que estamos digitalizando, auxiliado pela identificação única do ativo presente no campo *globalAssetId*. Suas informações impostas em *assetType* e *defaultThumbnail* ajudam a definir o tipo de elemento físico utilizado no

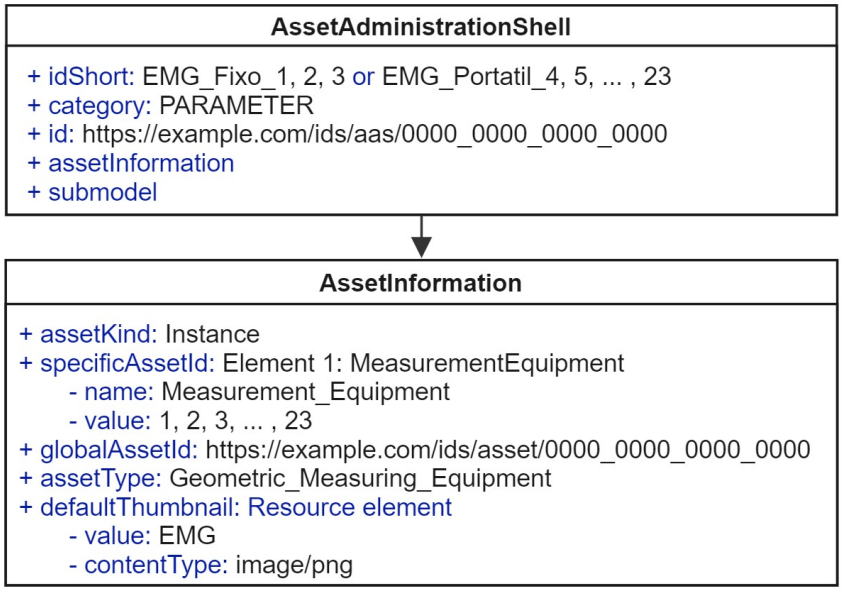


Figura 4.2: Diagrama de construção do elemento *Asset*.

projeto, por exemplo, o equipamento de medição geométrico, assim como o formato da imagem utilizada dentro da plataforma.

Por último, são criados os componentes *Submodels* que possuem como principal objetivo informar e descrever as funções e características do ativo, tais como: *Identification, Technical Data, Operational Data, Capabilities* e *Asset Interface Description*, sendo implementados de forma semelhante para ambos os EMGs e todas as estações utilizadas no projeto.

### 4.1.3 Submodelo *Identification*

O primeiro submodelo chamado de *Identification* é criado para apresentar os dados de identificação de nossos EMGs fixo e portátil presentes nas estações da linha de montagem. Podendo ser visto na Figura 4.3, a sua construção é dada ao adicionar primeiramente um *idShort*, onde é definida a nomenclatura de nosso submodelo, e um campo *category* com o valor constante, levando-se em consideração a invariabilidade dos dados de identificação de nossos EMGs. A identificação *id* do submodelo também é estruturada, utilizando valores numéricos para conceder a individualidade dos EMGs.

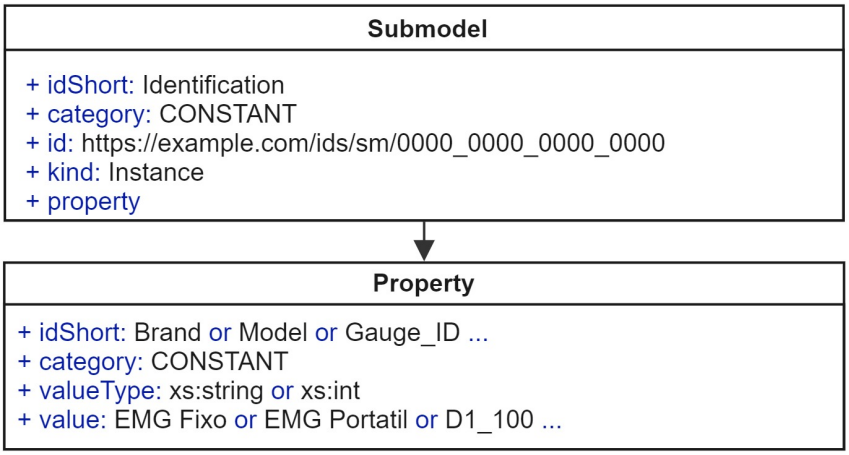


Figura 4.3: Diagrama de construção do elemento *Identification*.

Um nível abaixo é criado o elemento *Property*, podendo ser implementado para ambos os dispositivos de medição citados anteriormente e consequentemente, para todas as estações do caso de estudo. Com o intuito de reconhecer o ativo, as propriedades deste submodelo são construídas de modo a esclarecer suas especificações de modelo, estabelecendo, por exemplo, a marca e identificação do EMG, assim como o seu ano de fabricação e localização na linha de produção. Analisando a Figura 4.3, a construção das propriedades é feita com a adição dos dados nos campos *idShort*, *category*, *valueType* e *value*, onde as duas últimas estarão responsáveis por especificar o valor numérico inteiro *xs:int* ou textual (*xs:string*) de nossa propriedade.

### 4.1.4 Submodelo *Technical Data*

Demonstrado na Figura 4.4 o segundo submodelo denominado *Technical Data* utiliza como base alguns componentes presentes no modelo *IDTA 02003-1-2 Generic Frame for Technical Data for Industrial Equipment in Manufacturing* para a sua criação. Este elemento estará responsável por informar os dados técnicos de nosso equipamento de medição, apresentando diferentes estruturas conforme as especificações de nosso EMG.

Inicialmente, o campo *idShort* determina o nome do submodelo como *Technical Data*,

apresentando o campo *category* como constante devido à inalterabilidade após uma primeira configuração. Os campos de identificação, assim como ocorre no submodelo *Identification*, são adicionados para estabelecer a individualidade de identidade e instância única de nosso ativo, facilitando o monitoramento de dados em tempo real provenientes dos EMGs.

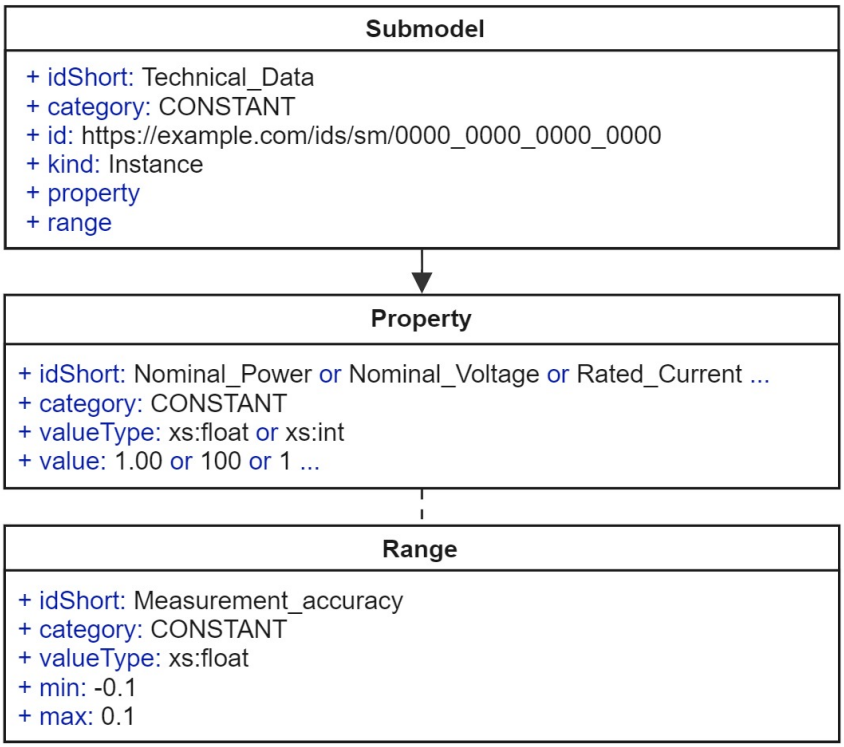


Figura 4.4: Diagrama de construção do elemento *Technical Data*.

A seguir, uma camada abaixo, tem-se a criação do elemento propriedade, adequando o EMG fixo ao utilizar elementos para informar a tensão nominal e a corrente nominal, assim como a precisão de medição do equipamento. Com uma construção semelhante para o EMG portátil, as propriedades irão descrever as dimensões e o peso do equipamento de medição, facilitando a compreensão do equipamento durante a sua utilização.

Para elaborar o elemento propriedade foram utilizados os campos *idShort*, para a nomenclatura, *category*, para estabelecer se há ou não mudança nas informações no ativo e, por fim, os campos *valueType* e *value*, especificando os valores de nossa propriedade. Também é criado o elemento *Range*, este que irá definir o valor mínimo e máximo de

precisão de medição de nossos EMGs de acordo com as informações pré-definidas pela empresa automotiva.

### 4.1.5 Submodelo *Operational Data*

Como terceiro submodelo do AAS de nossos equipamentos de medição, o elemento *Operational Data* estará responsável por estruturar os dados provenientes da indústria automobilística, fornecendo uma estrutura organizada e eficiente para o desenvolvimento do AAS Tipo 2.

Ao contrário de outros submodelos que já recebem um valor atribuído *string* ou *int* na própria ferramenta digital, como visto anteriormente em *Identification* e *Technical Data*, o *Operational Data* apresenta exclusivamente o campo abreviado do ativo, mantendo sempre a lacuna "valor" em branco para posterior adição de dados.

Neste sentido, podemos analisar por meio da Figura 4.5 a construção do elemento *Operational Data*, este que é estruturado para comportar os dados de todos os EMGs da linha de produção automotiva. Assim como em submodelos anteriores, serão criados o campo de nomenclatura *idShort*, o campo *category*, este que será determinado como variável em razão da mudança constante de valores originados dos EMGs, e por fim os campos de identificação *id* e *kind*.

Após isso, o elemento propriedade é criado com o intuito de descrever as informações do carro no momento de medição dos pontos na estação. A propriedade *JSN* é implementada com o objetivo de informar o código de identificação único do carro analisado, sendo incrementada pela propriedade seguinte *Timestamp*, que irá identificar o momento exato de análise dos pontos do carro no período de medição, concedendo um carimbo de data e hora em formatos padronizados. Outros elementos também são desenvolvidos com o intuito de informar as características dos dados enviados pelos EMGs na linha de produção.

Ainda, no mesmo nível da camada das propriedades, o elemento *SubmodelElementList* é desenvolvido, comportando uma lista ordenada de pontos que serão medidos pelos

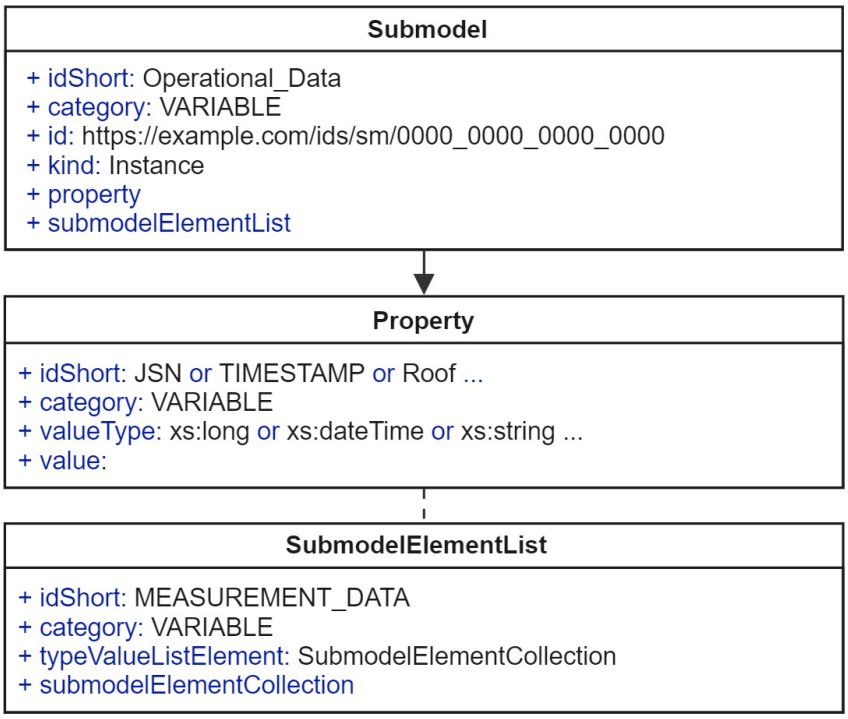


Figura 4.5: Diagrama inicial de construção do elemento *Operational Data*.

EMGs. Com seus valores variáveis, este elemento inclui também o campo *typeValue-ListElement*, responsável pelo atendimento de pontos coletados pelo EMG, agrupando e organizando os dados de forma estruturada.

Após isso, a adição do elemento *SubmodelElementCollection*, podendo ser vista na Figura 4.6, este que recebe o nome *Measurement Point* no campo *idShort*, é fundamental para atender aos requisitos de nosso projeto, visto que esta irá possuir diversos pontos listados em ordem conforme envio de informações dos dados do carro. No interior de seu componente, são inseridos os pontos que servem como modelo base para comportar os dados do EMG.

Portanto, enquanto a propriedade *Name* é criada e utilizada para receber o nome do ponto medido na estação, os critérios dos pontos do elemento *SubmodelElementCollection*, D, US, LS, UR, LR, UT, LT e *Feedback*, definidos no campo *shortID* e detalhados na Tabela 4.1, estarão responsáveis por armazenar os dados coletados anteriormente.

Na camada a seguir, mais especificamente no elemento *Property*, são elaborados os

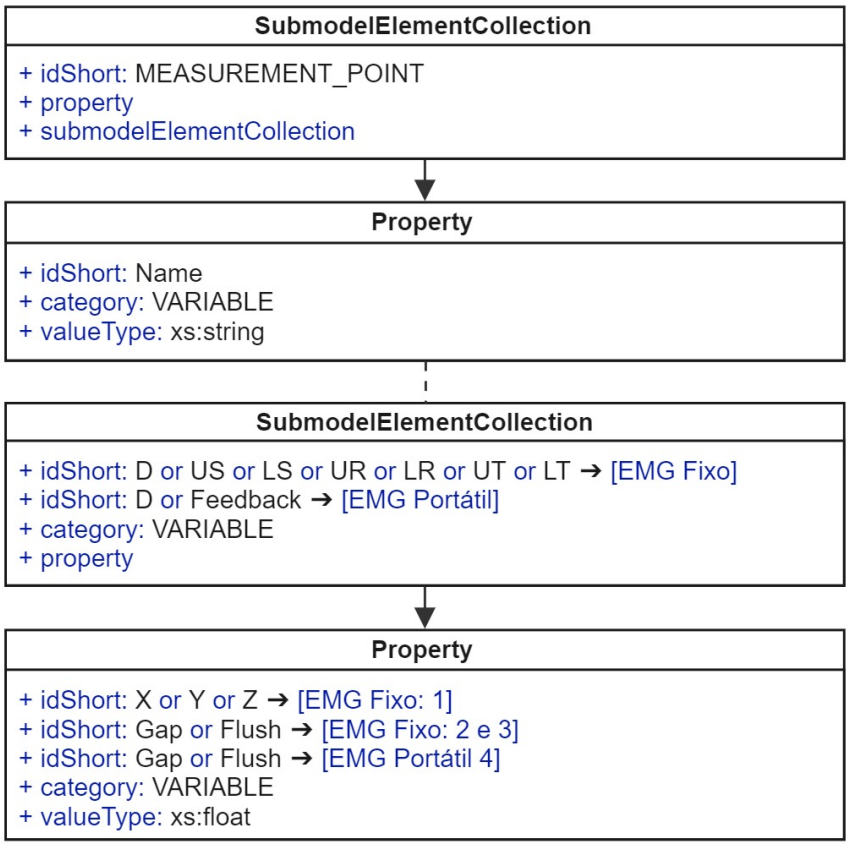


Figura 4.6: Diagrama final de construção do elemento *Operational Data*.

pontos responsáveis pelo armazenamento dos dados medidos pelos EMGs. Por meio do campo *idShort* são determinados os pontos de medição de cada estação da linha de produção, estes sendo construídos para todos os EMGs do caso de estudo, apresentando diferença conforme o tipo de ponto medido e a qual estação pertence cada equipamento de medição.

Logo, o submodelo *Operational Data* estará designado em estruturar elementos de forma a receber dados dos pontos medidos, sendo necessária a extração dos modelos do AAS no formato JSON para posterior implementação no modelo reativo de nossos ativos. Esta extração é feita por intermédio da ferramenta *AASX Package Explorer*, que irá manter a estrutura da hierarquia de acordo com os modelos implementados na interface gráfica.

| Ponto | Descrição |
|---|---|
| D | Medida do Ponto |
| US | Limite Máximo na Especificação |
| LS | Limite Mínimo na Especificação |
| UR | Limite Máximo de Rejeição |
| LR | Limite Mínimo de Rejeição |
| UT | Limite Máximo de Tolerância Limite |
| LT | Limite Mínimo de Tolerância Limite |
| Feedback | Valor Booleano de Limites Máximo e Mínimo |

Tabela 4.1: Descrição dos critérios para determinar as faixas de tolerância e rejeição dos pontos presentes na estrutura do *Operational Data*.

### 4.1.6 Submodelo *Capabilities*

Para as próximas etapas, serão adicionados os submodelos *Capabilities* e *Asset Interface Description* na estrutura do AAS, tendo como intuito complementar os dados disponíveis pelos submodelos *Identification* e *Technical Data* mostrados anteriormente, onde o diagrama de construção das capacidades pode ser visto na Figura 4.7.

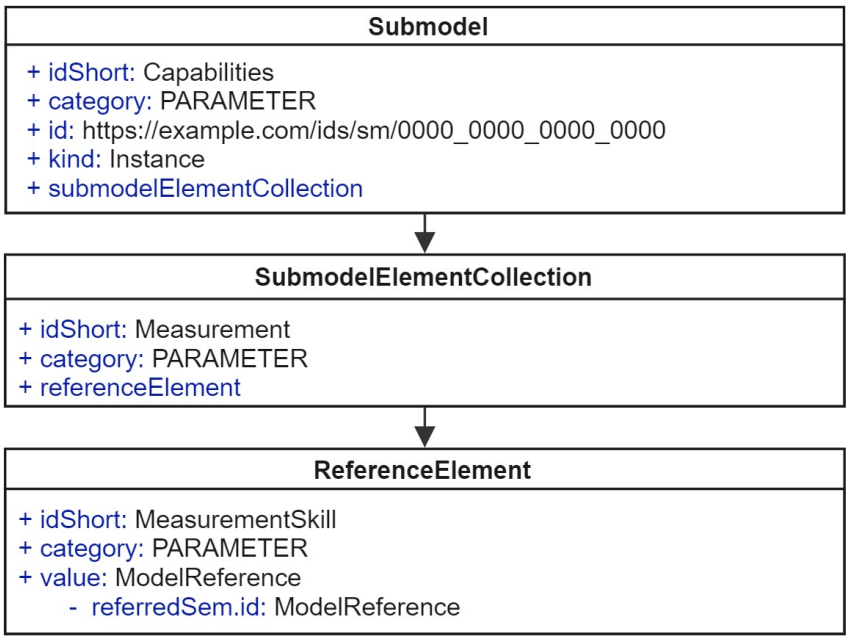


Figura 4.7: Diagrama de construção do elemento *Capabilities*.

A adição do submodelo *Capabilities* irá informar as habilidades de nosso equipamento

de medição, apresentando o elemento *SubmodelElementCollection* em seu interior. Este elemento de coleção, definido como *Measurement* no campo *idShort*, servirá como base para a construção do elemento *ReferenceElement*. A criação deste elemento de referência permite definir os campos *idShort*, *category* e *value*, este último utilizado para vincular o elemento de referência à instância *Asset* do modelo do AAS, associando o ativo à habilidade de medição no sistema.

### 4.1.7 Submodelo *Asset Interface Description*

Complementando o submodelo *Capabilities*, cria-se o último submodelo de nossa estrutura AAS chamado de *Asset Interface Description*, que realizará a descrição das interfaces dos ativos. Este submodelo se interconectará com o submodelo *Capabilities* por intermédio do elemento *Reference* (*idshort: MeasurementSkill*), permitindo uma correlação entre as capacidades e o serviço de comunicação utilizados por nosso ativo.

Assim, tendo em conta a Figura 4.8, o elemento *SubmodelElementCollection* é descrito como *Interface For Communication* no campo *idShort*, com a categoria *parameter* devido à configuração dos dados de comunicação. Outros elementos, como as propriedades, estão inclusos neste submodelo, definindo as especificações do protocolo de comunicação e as configurações de segurança utilizadas.

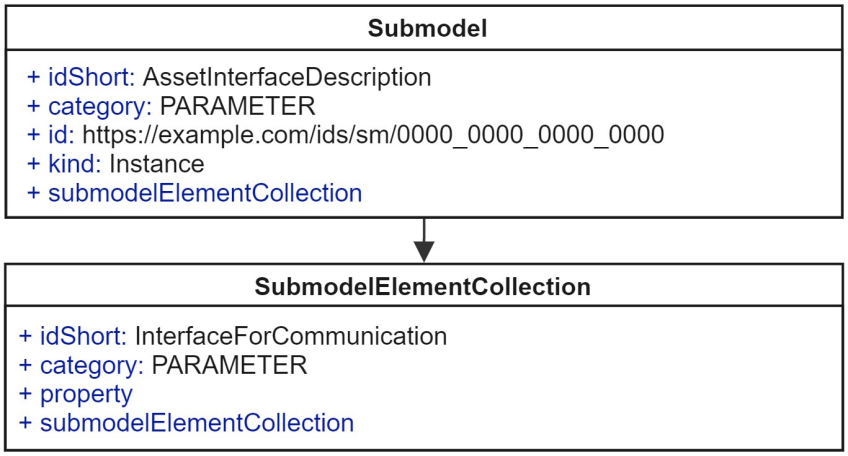


Figura 4.8: Diagrama de construção do elemento *Asset Interface Description*.

### 4.1.8 Conversão de Submodelos

Os submodelos construídos na ferramenta *AASX Package Explorer* possuem originalmente seu formato *.aasx*, havendo a possibilidade de conversão para outros formatos. A plataforma oferece compatibilidade de extração desses ficheiros para os formatos XML e JSON, facilitando a interoperabilidade entre diferentes sistemas industriais.

Com o objetivo de implementar o modelo reativo do AAS, todos os submodelos do formato *.aasx* são exportados para o formato *.json*. Essa conversão simplifica a leitura do modelo do AAS em ambiente de codificação Python, onde cada EMG da estação de medição da linha de montagem apresentará sua estrutura do AAS e, consequentemente, seu ficheiro JSON correspondente. Esses ficheiros apresentarão um destinatário em comum, sendo utilizados posteriormente no desenvolvimento do AAS Tipo 2.

Portanto, a construção destes submodelos para integrar os modelos do AAS permite uma integração eficaz dos ativos, além de atender às necessidades de comunicação específicas em todas as estações do sistema de produção. Com o modelo passivo estruturado, torna-se possível implementar o modelo reativo descrito no próximo capítulo, aplicado ao nosso caso de estudo.

## 4.2 Implementação do AAS Tipo 2

O desenvolvimento do modelo reativo para todas as estações da linha de montagem é realizado através do trabalho conjunto de ferramentas de programação em linguagem Python e ficheiros no formato CSV, criando um servidor para receber os dados dos EMGs no formato adequado do modelo passivo do AAS.

### 4.2.1 Fluxograma do Modelo Reativo

O fluxo de implementação é descrito na Figura 4.9, onde primeiramente ocorre a integração dos dados da empresa automotiva com o sistema AAS, demonstrando os meios para o envio das funcionalidades e dados do EMG durante a conexão com o servidor AAS. Os ficheiros

armazenados em um banco de dados da empresa automotiva são enviados para a pasta compartilhada e automaticamente identificados pelo *filewatcher* como novos ficheiros.

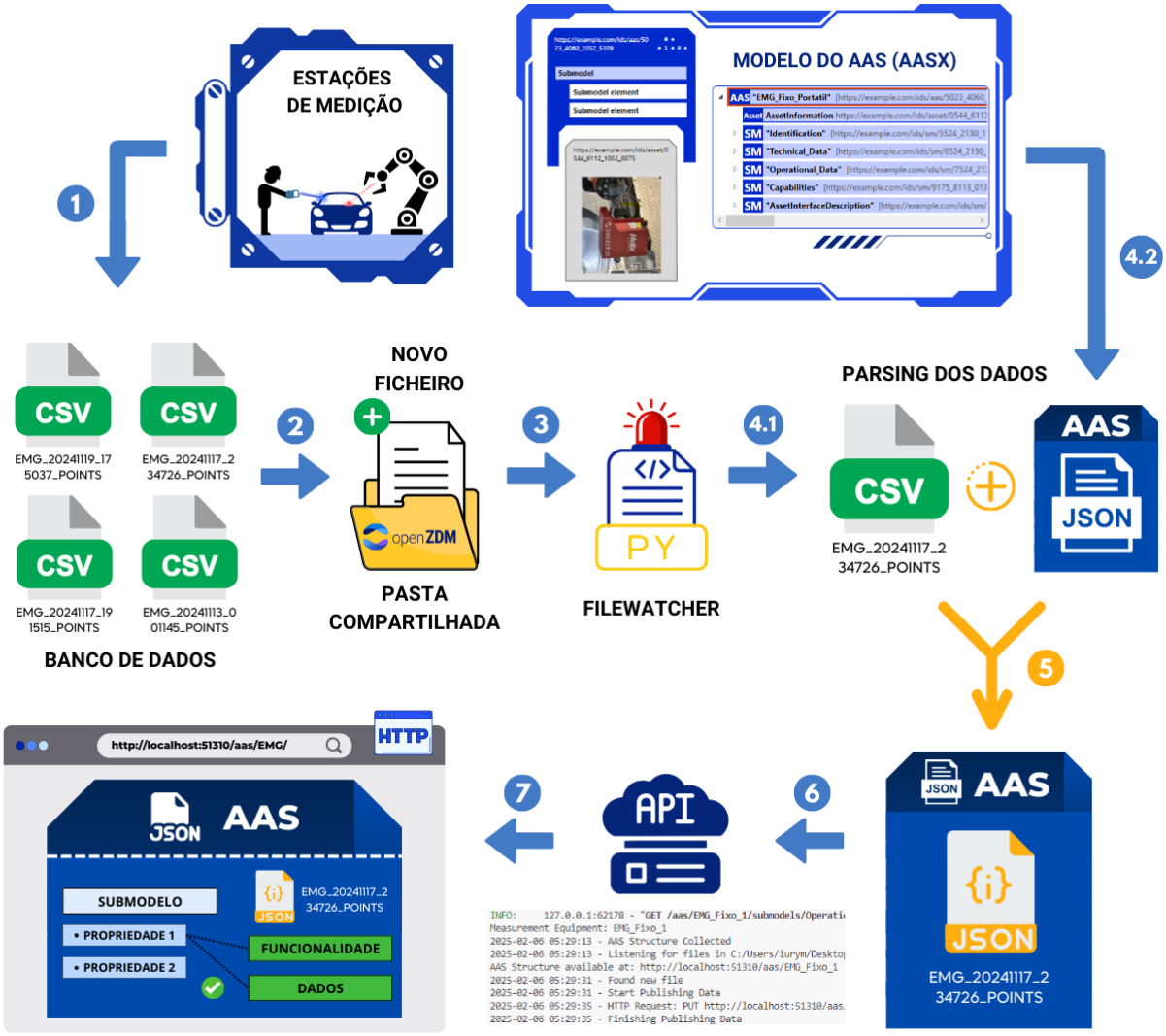


Figura 4.9: Fluxograma do AAS Tipo 2 através de uma *FastAPI*.

Em seguida, é realizado o *parsing* dos dados, convertendo o ficheiro CSV para o formato JSON, atendendo aos requisitos necessários para a integração com a estrutura JSON do AAS. Por fim, utilizando a estrutura *FastAPI*, os dados do ficheiro JSON são incorporados ao modelo do AAS, permitindo a visualização das funcionalidades e informações enviadas pela empresa parceira.

Estes métodos irão fornecer o acesso ao modelo passivo do AAS, onde o método GET

requisita a estrutura dos submodelos e o PUT atualiza os dados processados no servidor, o que possibilita o uso dos submodelos presentes no modelo do ativo de forma interoperável com os dados que chegam na pasta compartilhada.

Com o fim da conversão de dados, o servidor AAS deixa de atuar como um HTTP de método de requisição GET e passa a atuar como um HTTP de método de requisição PUT, o que torna os dados acessíveis por meio do *endpoint* da plataforma openZDM, esta que é encarregada de coordenar as atividades de análise de dados, ajustando os serviços de acordo com a estrutura dos dados recebidos. Os dados fornecidos pelo servidor AAS são armazenados em um banco de dados, podendo ser posteriormente analisados por intermédio de ferramentas de análise de dados que respondem a eventos.

### 4.2.2 Bibliotecas e Pacotes

As importações de bibliotecas para o funcionamento do sistema são estruturadas na seção inicial do código, onde as bibliotecas *os*, *json* e *pandas* são utilizadas para as manipulações de ficheiros CSV, permitindo também a conversão de dados de formato JSON em Python que serão realizadas durante a operação do servidor. Além disso, a biblioteca *watchdog* é fundamental para a implementação dos dados enviados pela empresa, sendo esta responsável por monitorar alterações em ficheiros CSV dentro de uma pasta compartilhada, definindo assim novas ações de processamento após a chegada de novos CSVs no diretório determinado.

Ainda, a biblioteca *requests* é implementada para o envio de dados obtidos ao servidor HTTP criado. Os pacotes *time* e *datetime* são utilizados para trabalhar com datas e horários, definindo intervalos de tempo e estabelecendo *timestamps* para os dados que chegam no servidor. Por último, o módulo *logger* estará registrando as mensagens e informações de execução de nosso sistema, informando o momento em que a estrutura do AAS foi coletada, assim como a espera por novos ficheiros pelo serviço *filewatcher*.

### 4.2.3 Coleta de Dados

A coleta de dados no sistema disponibilizado pela empresa automotiva é realizada para cada veículo que percorre a linha de produção, estes que apresentam seus dados extraídos através dos EMGs fixo e portátil. No entanto, a forma como os dados chegam na pasta compartilhada ocorre de forma distinta quando estes são obtidos pelo EMG portátil, sendo necessário intervir com alguns ajustes para comportar a chegada dos dados no sistema.

Assim, com a conexão direta entre o EMG fixo e o sistema global de gestão da empresa, os dados obtidos através destes dispositivos fixos serão enviados diretamente ao sistema sem a necessidade de ajustes. Com os dados integrados, podendo ser visualizados na Figura 4.10, serão criados através de automatizações os ficheiros no formato CSV, direcionando-os para uma pasta compartilhada contida na rede interna da empresa.

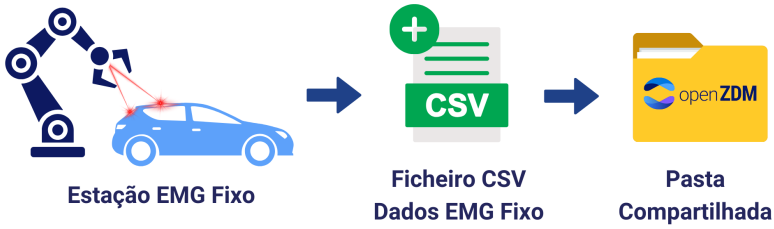


Figura 4.10: Envio de dados por meio do EMG fixo para a pasta compartilhada.

Contrário a estes, os EMGs portáteis, que não possuem integração com o sistema, utilizam um intermediário para o envio de dados ao sistema de gestão da empresa, conhecido como ferramenta *Node-RED*. Esta ferramenta, utilizando blocos de fácil interação, enviará os dados via MQTT ao código Python para posterior interação com o servidor AAS, como ilustrado no esquema da Figura 4.11. Para isso, são configuradas definições que permitem a comunicação MQTT integrar-se ao restante do código implementado, além da definição de utilizador e senha para garantir a segurança dos dados enviados.

Assim, a biblioteca *paho.mqtt.cliente* é adicionada com o intuito de enviar os dados obtidos do EMG portátil ao servidor via protocolo MQTT, enquanto o módulo *threading* é utilizado para o monitoramento e comunicação em tempo real do sistema, permitindo o processamento e a transferência de dados sem interromper a execução principal do

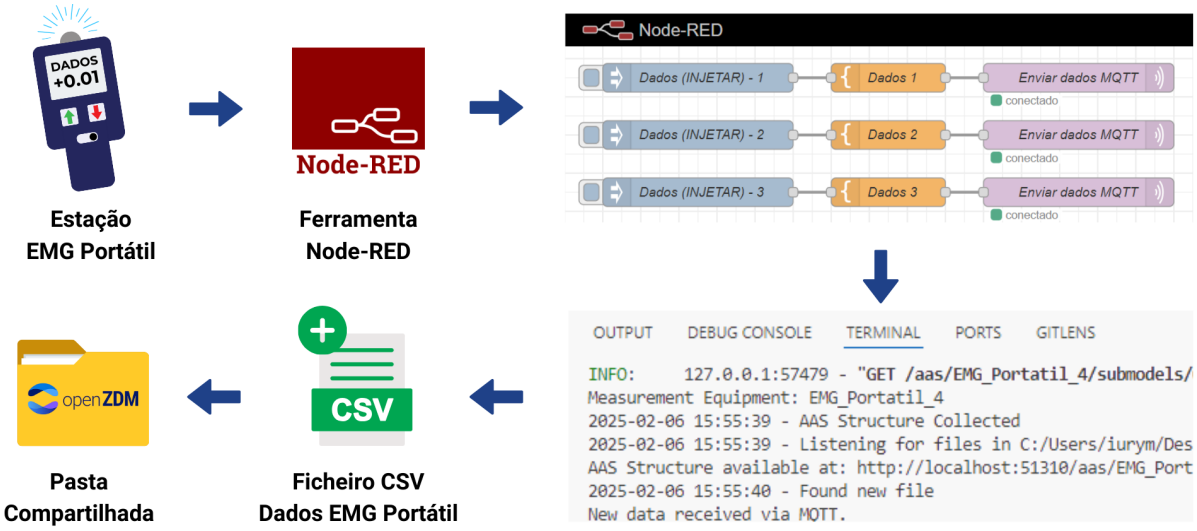


Figura 4.11: Envio de dados por meio do EMG portátil para a pasta compartilhada.

programa.

Os parâmetros adotados para este fluxo de dados via MQTT têm como auxílio o *broker Mosquitto*, sendo utilizado para a publicação e assinatura de mensagens enviadas em ambiente de simulação privado. Para o caso real, será utilizado um servidor interno para estabelecer estes eventos. Assim, como podemos ver no Algoritmo 4.1, a variável *mqtt broker* especifica o endereço do servidor MQTT, gerenciando a troca de mensagens entre os EMGs portáteis e o servidor AAS. Enquanto a variável *mqtt port* estará definindo a porta utilizada para conectar o *broker* MQTT, a configuração em *mqtt topic* define o tópico MQTT específico no qual os dados dos EMGs portáteis serão publicados.

```
config = {
    "mqtt_broker": "localhost",
    "mqtt_port": 1883,
    "mqtt_topic": "/topic/example", }
```

Algoritmo 4.1: Configurações globais do MQTT.

Sendo assim, é criada a função *initialize client*, responsável por gerenciar a conexão do cliente MQTT com o *broker*, conforme demonstrado no Algoritmo 4.2. Quando conectado, este cliente será automaticamente inscrito no tópico *topic* para receber novos dados.

Assim, sempre que uma nova mensagem for publicada nesse tópico, a função *on message callback* será chamada para processar a informação. Além disso, a conexão é mantida ativa em segundo plano por meio do método *loop start*, garantindo a recepção contínua dos dados pelo cliente MQTT.

```
1 def initialize_client(mqtt_client, broker, port, topic,
     on_message_callback):
2     def on_connect(client, userdata, flags, rc):
3         client.subscribe(topic)
4 
5     mqtt_client.on_connect = on_connect
6     mqtt_client.on_message = on_message_callback
7     mqtt_client.connect(broker, port)
8     mqtt_client.loop_start()
```

Algoritmo 4.2: Comunicação do protocolo MQTT.

Após a implementação do MQTT para adequar a chegada de novos dados, com o intuito de padronizar o processamento dos dados, é criado um ficheiro CSV com os dados do EMG portátil, este que, da mesma forma que ocorre nas estações do EMG fixo, será colocado na pasta compartilhada da empresa automotiva, possibilitando o processamento dos dados obtidos pelo dispositivo de medição.

### 4.2.4 Monitoramento dos Ficheiros

Em seguida a inserção do novo ficheiro no diretório determinado, este será detectado por meio do serviço *filewatcher*, baseado em Python, atuando como o ponto inicial para a construção do servidor AAS deste caso de estudo. Este serviço irá monitorar constantemente o diretório em busca de novos ficheiros, que, ao chegarem, acionam um manipulador de eventos fornecido pelo *filewatcher*. O manipulador de eventos estará responsável por processar o novo ficheiro dentro do caminho específico, como mostrado na Figura 4.12, para posterior tratamento dos dados contidos no ficheiro.

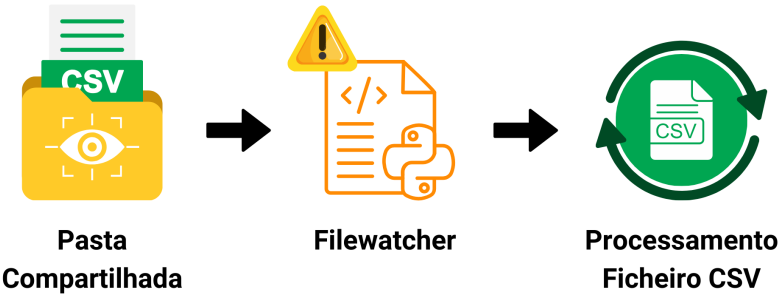


Figura 4.12: Monitoramento da pasta compartilhada pelo serviço *filewatcher*.

A sua implementação pode ser vista no Algoritmo 4.3, onde a classe *WatcherEvent* estará responsável por monitorar a pasta compartilhada em busca de novos ficheiros CSV.

Quando um novo ficheiro CSV é identificado na pasta compartilhada, o método *on created* é acionado, enviando uma mensagem que indica este novo evento e chamando o método *process event* para o processamento do ficheiro no sistema. O processamento utiliza a biblioteca *pandas* para a leitura do ficheiro CSV, este que possui a separação de colunas no formato tabular. Após a leitura do ficheiro, uma nova mensagem é enviada indicando o fim do processamento de dados, apresentando também um *logger* para envio de mensagem em casos de existência de erros durante o processo.

```
1  class WatcherEvent:
2      def __init__(self, process_callback):
3          self.process_callback = process_callback
4
5      def on_created(self, event):
6          if not event.is_directory:
7              self.process_event(event.src_path)
8
9      def process_event(self, path):
10         self.process_callback(path)
```

Algoritmo 4.3: Monitoramento e processamento de ficheiros.

Portanto, ao observar os dados continuamente e em tempo real, sempre que um novo ficheiro é adicionado na pasta, o evento é disparado novamente, mantendo o sistema sempre

atualizado conforme a adição de novos dados de medição feitos pelo EMG que chegam na pasta compartilhada. Isso garante uma sincronização eficiente entre o sistema de medição e o servidor AAS criado, sem a necessidade de correções manuais das informações.

### 4.2.5 Reestruturação dos Dados

Antes de iniciar a reestruturação de dados, é realizada a validação de submodelos, garantindo que os dados recebidos pelos dispositivos de medição estão alinhados com a estrutura do modelo passivo do AAS. Assim, são carregadas as estruturas dos submodelos contidas em ficheiros JSON, verificando a compatibilidade entre os campos da estrutura AAS e dos ficheiros CSV obtidos dos dispositivos de medição.

Além disso, com o intuito de preparar o processamento dos dados obtidos pelos dispositivos de medição, são reconstruídos os submodelos do modelo passivo do AAS. Este processo é essencial para estruturar o mapeamento entre elementos do AAS, sendo posteriormente utilizado pela função de reestruturação de dados dos dispositivos de medição.

Agora, com a validação de submodelo e a estrutura do modelo do AAS completa, é iniciado o processo de reestruturação dos dados. Quando um novo ficheiro é identificado, o sistema irá realizar o *parsing* dos dados do ficheiro, modificando-o para o formato especificado e definido pelo submodelo *Operational Data* da estrutura do modelo passivo do AAS. Cada ponto de medição realizado pelo EMG é convertido e organizado de acordo com a estrutura do submodelo, transformando as informações do ficheiro CSV em um formato compatível com o modelo JSON do AAS, como visualizado na Figura 4.13.

Com base no Algoritmo 4.4, o modelo AAS contido no ficheiro JSON, determina o formato ideal dos dados no sistema, sendo utilizado como base para comparação com os campos do ficheiro CSV. Nesta implementação, o principal objetivo é processar cada linha de dados do ficheiro CSV e verificar se o identificador único (*idShort*) do ponto já existe no dicionário.

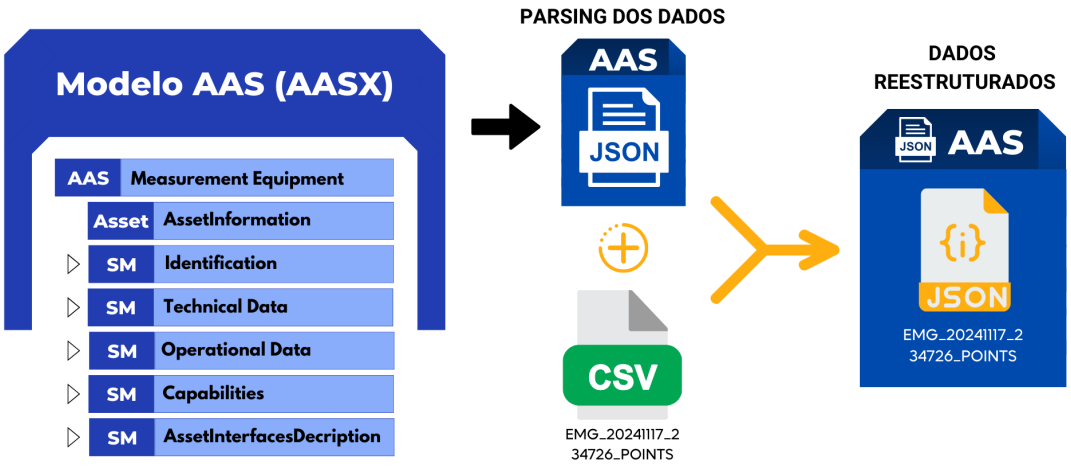


Figura 4.13: Reestruturação dos dados do EMG.

```
def restructure_data(data_structure, input_data):
    unique_ids = []

    for entry in input_data:
        identifier = entry["id"]
        if identifier not in unique_ids:
            unique_ids.append(identifier)
            new_item = deepcopy(data_structure["items"][0])
            new_item["id"] = identifier
            for key in ["A", "B", "C", "D", "E"]:
                new_item["values"][key] = entry.get(key, None)
            data_structure["items"].append(new_item)
        else:
            index = unique_ids.index(identifier)
            for key in ["A", "B", "C", "D", "E"]:
                data_structure["items"][index]["values"][key] =
                    entry.get(
                     key, data_structure["items"][index]["values"].
                         get(key))
    return data_structure
```

Algoritmo 4.4: Reestruturação de dados.

Como o submodelo *Operational Data* foi desenvolvido exercendo o papel de uma matriz no elemento *SubmodelElementList*, este será capaz de alocar os dados de apenas um ponto obtido pelos dispositivos de medição. Dessa forma, para comportar todos os pontos obtidos pelos EMGs, é realizada a replicação do submodelo *Operational Data* para múltiplos pontos, onde este será recriado até integrar todos os pontos distintos do ficheiro CSV.

Portanto, quando um novo ponto é identificado no dicionário *ids*, tem-se a replicação da estrutura base do submodelo *Operational Data* ao utilizar a biblioteca *deepcopy*, preenchendo-o com novos valores do ficheiro CSV. Caso o ponto identificado já exista no dicionário *ids*, este será atualizado, sobrescrevendo os valores com os dados mais recentes obtidos. Isto permite que dados de novos ficheiros CSV possam ser publicados no servidor AAS.

Com o processo de reestruturação concluído, cria-se um ficheiro JSON dos dados obtidos pelo dispositivo de medição, este que agora está adequado ao formato definido pelo modelo passivo do AAS, iniciando desse modo o processo de envio de dados ao servidor.

### 4.2.6 Implementação do Servidor

O desenvolvimento do servidor por meio da estrutura *FastAPI* [60] permite integrar os dados do EMG com o servidor HTTP. Assim, com base na estrutura, ele é implementado utilizando um *endpoint* REST e seus métodos de requisição HTTP GET e PUT disponibilizados para garantir a coleta eficiente e contínua dos dados do EMG.

A construção da lógica, mostrada na classe *Server* do Algoritmo 4.5, define inicialmente os caminhos para os diretórios de origem e temporário. Com os caminhos definidos, cria-se os *endpoints* do servidor AAS, onde o GET irá verificar se o servidor está em funcionamento e, logo após, irá ler e retornar o ficheiro JSON do modelo do AAS. O *endpoint* PUT, por outro lado, estará responsável por atualizar ou criar um ficheiro JSON no diretório temporário com os dados processados no servidor.

```
class Server:
    def __init__(self, source_dir: str, temp_dir: str):
        self.app = FastAPI()
        self.source_dir = Path(source_dir)
        self.temp_dir = Path(temp_dir)
        self._initialize_temp()

        @self.app.get("/")
        async def root():
            return {"message": "Server is running"}

        @self.app.get("/{item_id}")
        async def get_item(item_id: str):
            return self._read_item(item_id)

        @self.app.put("/{item_id}")
        async def update_item(item_id: str, content: str):
            return self._write_item(item_id, content)
```

Algoritmo 4.5: Servidor *FastAPI* para manipulação de ficheiros.

Agora, levando em conta a continuação da lógica no Algoritmo 4.6, utiliza-se a função *intialize temp* para preparar o ambiente de trabalho do servidor, copiando os ficheiros JSON do modelo do AAS para um diretório temporário de manipulação, permitindo, dessa forma, preservar os dados originais do ficheiro. Logo após, são implementadas as funções *read item* e *write item*, que, respectivamente, realizam a leitura de ficheiros JSON presentes no diretório temporário, permitindo criar ou atualizar ficheiros JSON, como explicado anteriormente.

Com estes pontos definidos, será possível enviar e receber dados dos dispositivos de medição em nosso servidor AAS, mantendo de forma interoperável a comunicação entre os sistemas envolvidos. Assim, a Figura 4.14 demonstra um exemplo onde o ficheiro JSON reestruturado enviará por meio do servidor AAS, implementado com a estrutura *FastAPI*,

os dados de medição do EMG. Estas informações enviadas poderão ser visualizadas em uma interface HTTP, visando à aquisição de dados em tempo real.

```
19      def _initialize_temp(self):
20          self.temp_dir.mkdir(parents=True, exist_ok=True)
21          for file in self.source_dir.iterdir():
22              if file.is_file():
23                  (self.temp_dir / file.name).write_bytes(file.
                        read_bytes())
24
25      def _read_item(self, item_id: str):
26          file_path = self.temp_dir / item_id
27          if not file_path.exists():
28              return Response(content="Item not found", status_code
                    =404)
29          return Response(content=file_path.read_text(), media_type="
                text/plain")
30
31      def _write_item(self, item_id: str, content: str):
32          (self.temp_dir / item_id).write_text(content)
33          return Response(content="Item updated", media_type="text/
                plain")
```

Algoritmo 4.6: Servidor *FastAPI* para diretório, leitura e escrita dos ficheiros.

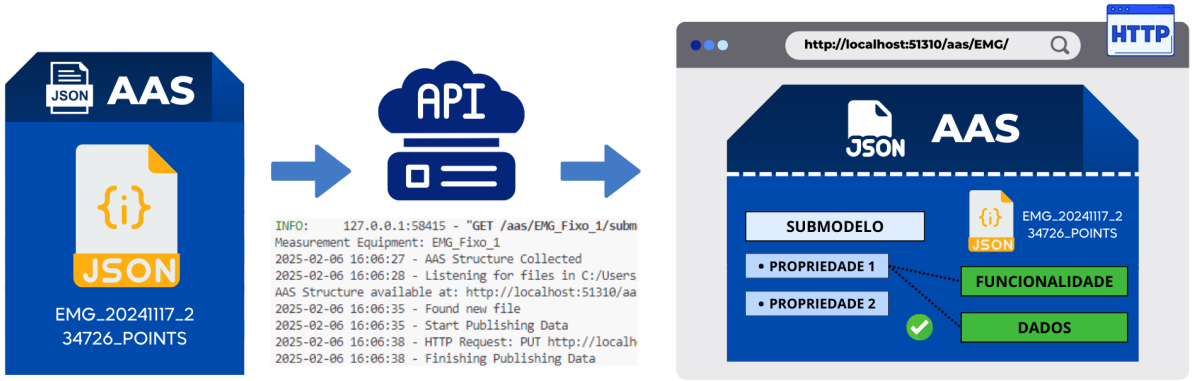


Figura 4.14: Envio de dados do EMG ao servidor AAS.

# Capítulo 5

# Resultados e Discussão

Este capítulo irá apresentar os resultados obtidos através da implementação dos modelos passivo e reativo para a digitalização dos sistemas de medição da indústria automobilística. Os principais desafios existentes ao digitalizar ativos com o uso da ferramenta *AASX Package Explorer* com o intuito da interoperabilidade de sistemas, além dos pontos relacionados ao *framework* utilizados no projeto, serão discutidos de forma a esclarecer a viabilidade do uso da ferramenta para futuras implementações em linhas de produção.

Ainda, em razão da política de confidencialidade de informações sensíveis da empresa parceira, são apresentados dados fictícios junto aos resultados das implementações do modelo passivo e reativo. Essa estratégia foi adotada com o objetivo de assegurar a proteção das informações reais, mantendo a privacidade e a integridade dos dados da indústria, sem comprometer a análise e os resultados apresentados.

## 5.1 Resultados do AAS Tipo 1

Os resultados obtidos da implementação do AAS Tipo 1 consistem em elementos estáticos construídos no modelo passivo do AAS, utilizando a interface disponível na ferramenta *AASX Package Explorer*. Estes elementos, quando em conjunto, estabelecem as características e funcionalidades do ativo de forma clara e padronizada, viabilizando seu uso em ambientes industriais.

### 5.1.1 Modelos do AAS para o EMG Fixo

As estruturas dos dispositivos de medição fixo, representados como ativos por intermédio da ferramenta AAS, mostram-se viáveis para o caso de estudo. Com um conjunto de elementos representativos de um *Administration Shell*, a ferramenta fornece uma estrutura completa para incluir os dados adquiridos pelos EMGs fixos das estações de medição 1, 2 e 3 da linha de montagem inicial.

#### EMG Fixo 1

A Figura 5.1 mostra os resultados obtidos da implementação para a EMG Fixo 1 do sistema, onde a região esquerda possui o painel de exibição do ativo, esta sendo uma representação estática do ativo que possui interação indireta com os submodelos criados no painel de navegação, ilustrado na região direita da figura. Este painel de exibição estará, por outro lado, diretamente conectado à identificação dos elementos AAS e ativo da estrutura criada, refletindo as mudanças feitas nesses elementos do modelo.

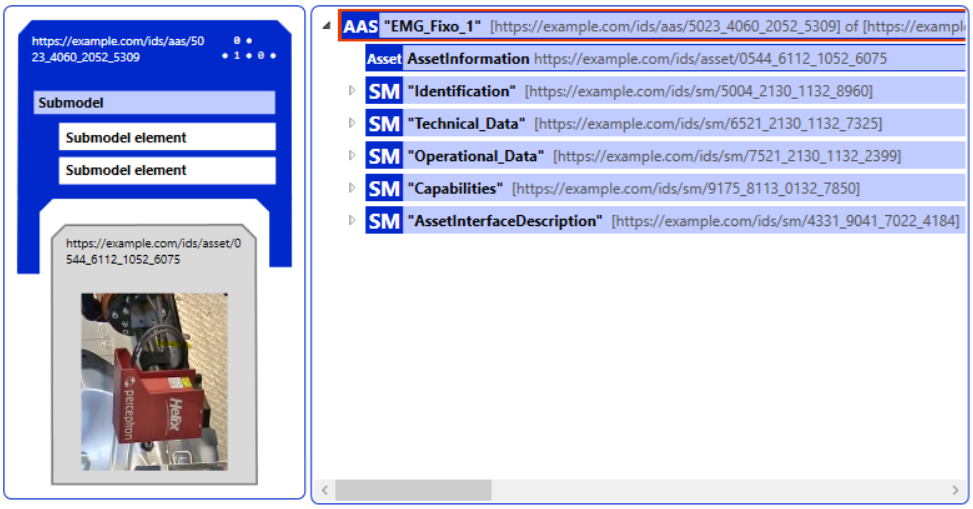


Figura 5.1: Interface do modelo passivo do AAS para o EMG fixo 1.

Neste contexto, o painel de navegação demonstra o elemento AAS do sistema, contendo todos os demais componentes da estrutura em seu interior ao se basear na norma da IEC 63278 [59], exemplificando o EMG Fixo 1 do estudo. O ativo do AAS, representado pelo

elemento *Asset (AssetInformation)*, apresenta uma identificação única global que o difere de outros componentes do sistema, garantindo a rastreabilidade do EMG em diferentes sistemas.

Em conjunto a isso, também é possível visualizar na Figura 5.2 a interface da ferramenta *AASX Package Explorer* com as definições do elemento AAS para o ativo da estação de medição 1. O painel possui as informações de identificação de nosso AAS, assim como as configurações para a visualização gráfica do ativo.

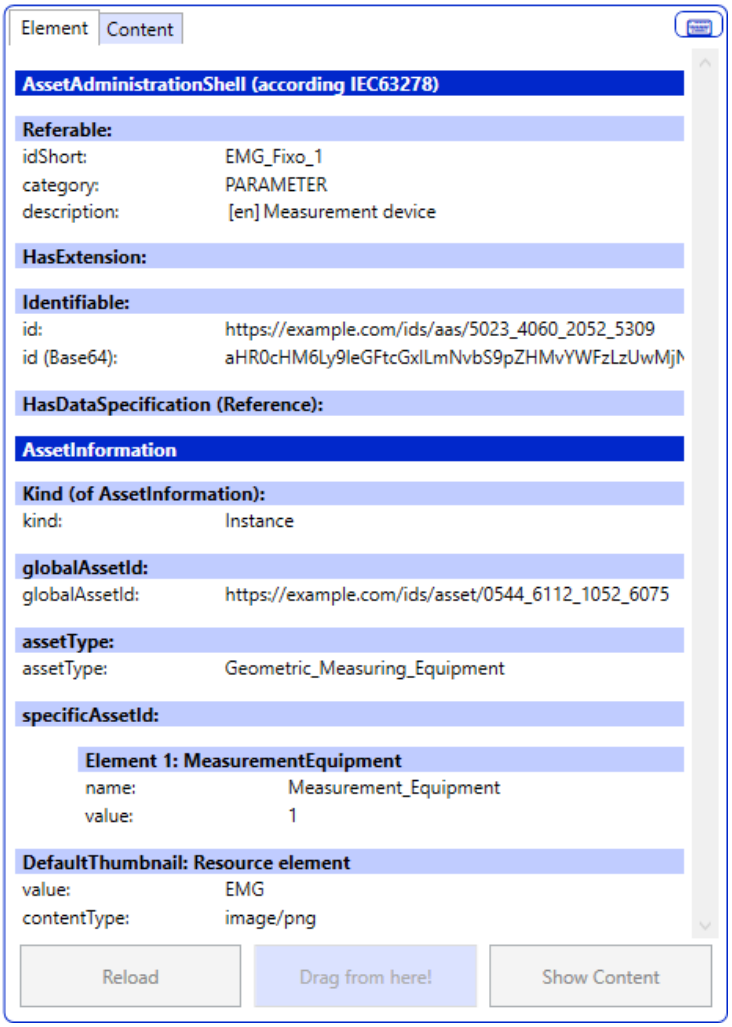


Figura 5.2: Interface das definições do AAS para o EMG fixo 1.

Os resultados do primeiro submodelo *Identification* implementado podem ser vistos na Figura 5.3, apresentando os dados de identificação de nosso ativo por meio dos elementos

propriedade da ferramenta. A hierarquia mostra a marca e o modelo do equipamento, o número de identificação do dispositivo, o ano de fabricação e a localização do ativo na linha de produção, permitindo facilitar o processo de digitalização de nosso ativo industrial.

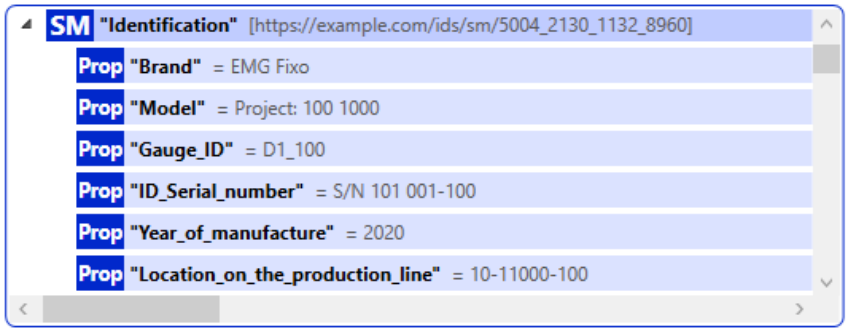


Figura 5.3: Interface do submodelo *Identification* para o EMG fixo 1.

Compondo este, o submodelo *Technical Data* na Figura 5.4 revela os dados técnicos do EMG fixo, sendo possível observar os valores de potência nominal, tensão nominal, corrente nominal, frequência e precisão de medição do dispositivo de medição. Como mencionado no início do capítulo, estes resultados utilizam dados fictícios, não interferindo no entendimento do submodelo.

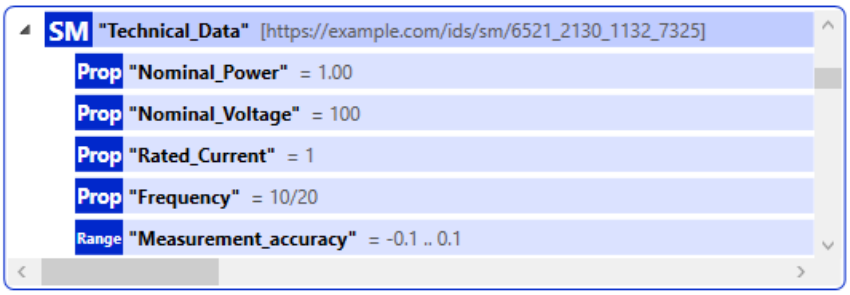


Figura 5.4: Interface do submodelo *Technical Data* para o EMG fixo 1.

Além disso, o submodelo *Operational Data* exibe a estrutura base para integrar os dados dos pontos medidos pelos EMGs fixos. Este elemento possui 6 propriedades iniciais, como mostrado na Figura 5.5, que especificam as nomenclaturas necessárias para o processo de implementação do modelo reativo do AAS.

Seu elemento *SubmodelElementList (Measurement Data)* resulta em um arranjo organizado de propriedades e *SubmodelElementCollections*, responsáveis por comportar os

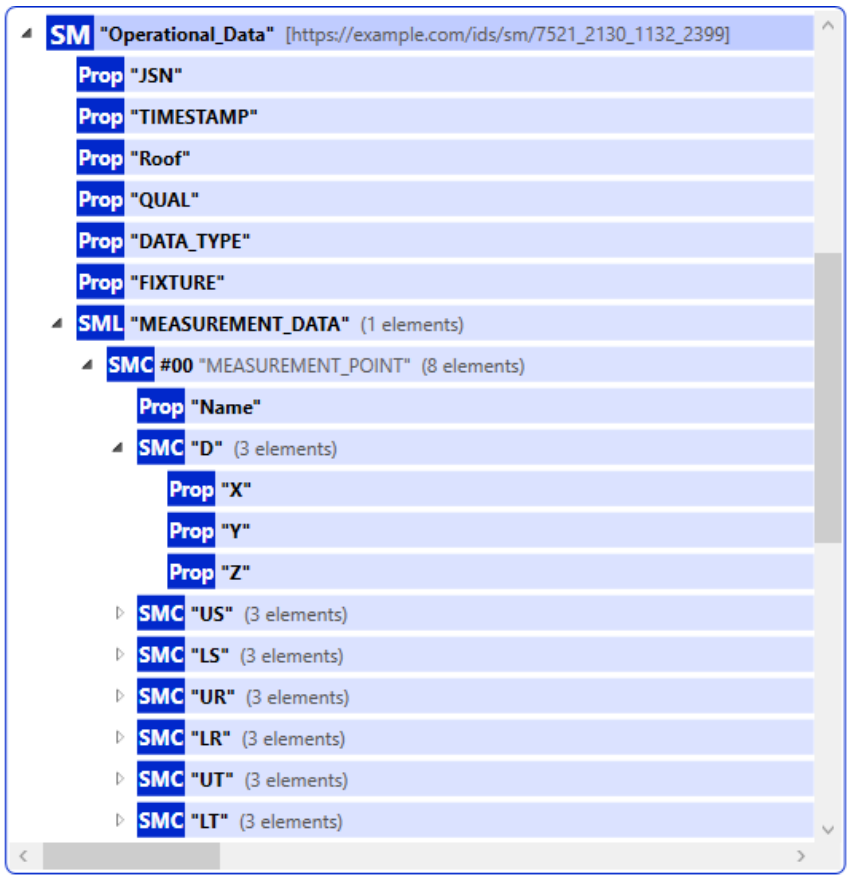


Figura 5.5: Interface do submodelo *Operational Data* para o EMG fixo 1.

dados de critérios adquiridos pelo EMG. Seu elemento *SubmodelList* ordena os dados em formato de lista, mostrando-se viável ao estruturar os pontos de forma sequencial e padronizada, atendendo às necessidades do modelo requerido para o funcionamento. Portanto, os critérios de definição de limite desenvolvidos, como D, US, LS, UR, LR, UT e LT, possuem em seu interior eixos tridimensionais X, Y e Z da estação de medição 1, não apresentando um valor associado no painel de navegação.

Como a funcionalidade da ferramenta *AASX Package Explorer* não exige um valor associado às propriedades, o que provém do submodelo *Operational Data* é particularmente útil para nosso modelo reativo, tendo em conta que estas serão preenchidas posteriormente por dados extraídos no servidor AAS, facilitando a coleta de informações enviadas pela indústria automotiva.

A estrutura designada para as capacidades de nosso ativo pode ser vista na Figura 5.6,

onde o elemento *SubmodelElementCollection* possui a referência *MeasurementSkill* descrevendo a habilidade de medição do ativo. Assim, este elemento interconecta o submodelo *Capabilities* ao elemento *Asset* do ambiente AAS, permitindo uma representação mais detalhada do ativo.

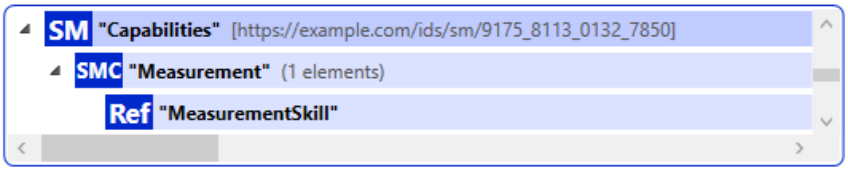


Figura 5.6: Interface do submodelo *Capabilities* para o EMG fixo 1.

Por último, o submodelo *Asset Interfaces Description* observado na Figura 5.7 complementa as informações obtidas nos resultados dos submodelos *Identification*, *Technical Data* e *Capabilities*, descrevendo as interfaces do ativo. Os resultados deste submodelo definem o protocolo de comunicação genérico de nosso ativo, especificando o *endpoint* para conexão com o ativo por meio do endereço base http://127.0.0.1.51310/, assim como o formato de serialização JSON dos dados no *endpoint*.

Este submodelo também tem como resultado o esquema de segurança *oauth2 sc* aplicado à interface do ativo, indicando que o painel de controle exige autenticação durante o acesso. O elemento *scheme* utiliza o modo de autenticação *oauth2*, seguindo o protocolo *OAuth 2.0 Security Scheme*, garantindo segurança, controle de acesso e integração eficiente com o sistema da indústria automotiva.

Além disso, o elemento *token* especifica o endereço do *endpoint* onde seu acesso será obtido, enquanto a propriedade *flow* determina como o sistema obtém o token, que neste caso *clientCredentials* representa uma comunicação entre sistemas sem usuário final. Os resultados desses elementos são significativos para comportar um modelo padronizado e seguro do AAS, que permita interoperabilidade entre sistemas.

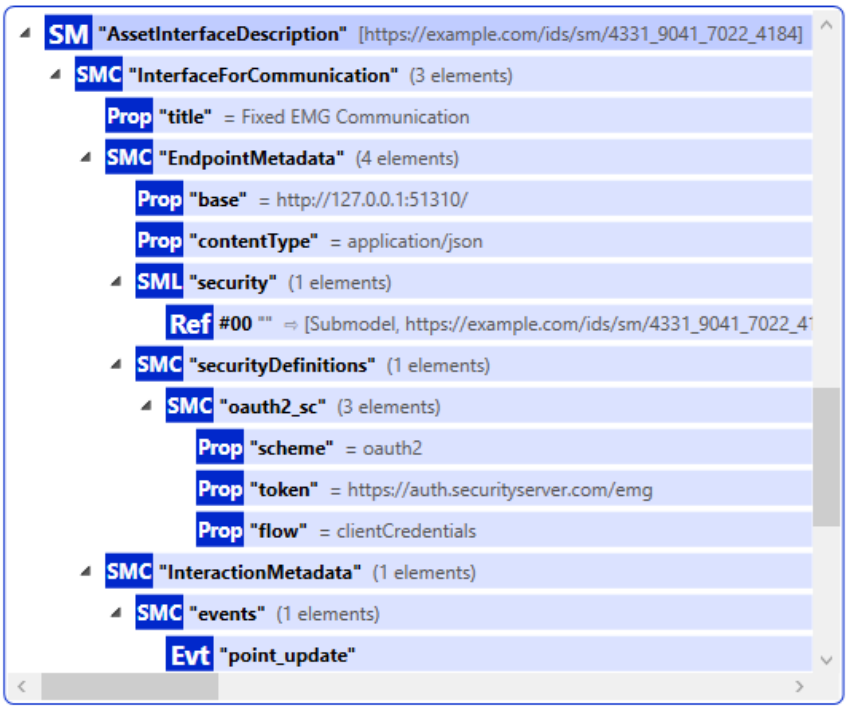


Figura 5.7: Interface do submodelo *Asset Interfaces Description* para o EMG fixo 1.

### EMGs Fixos 2 e 3

OS EMGs fixos 2 e 3 possuem resultados semelhantes ao EMG fixo 1, proporcionando um sistema padronizado no modelo passivo do AAS. Assim, a Figura 5.8 mostra a estrutura do AAS para o EMG fixo 2 da linha automotiva.

Devido a isso, os resultados dos submodelos *Identification*, *Technical Data*, *Operational Data* e *Asset Interfaces Description* são semelhantes aos resultados do EMG Fixo 1, demonstrando as mesmas finalidades explicadas no ativo. O submodelo *Operational Data* por sua vez, apresenta a estrutura ajustada para os EMGs fixos 2 e 3, visando comportar diferentes pontos provenientes dos dispositivos de medição.

Assim, a Figura 5.8 mostra o submodelo *Operational Data* para o EMG fixo 2, este que, quando comparado com o EMG fixo 1, possui o elemento *SubmodelElementList* ajustado para comportar os pontos de outros componentes do carro. Assim, ele fornece as propriedades *Gap* e *Flush* sem um valor associado para todos os pontos de critérios limite, como D e US, permitindo a atribuição de dados medidos pelo dispositivo nas estações.

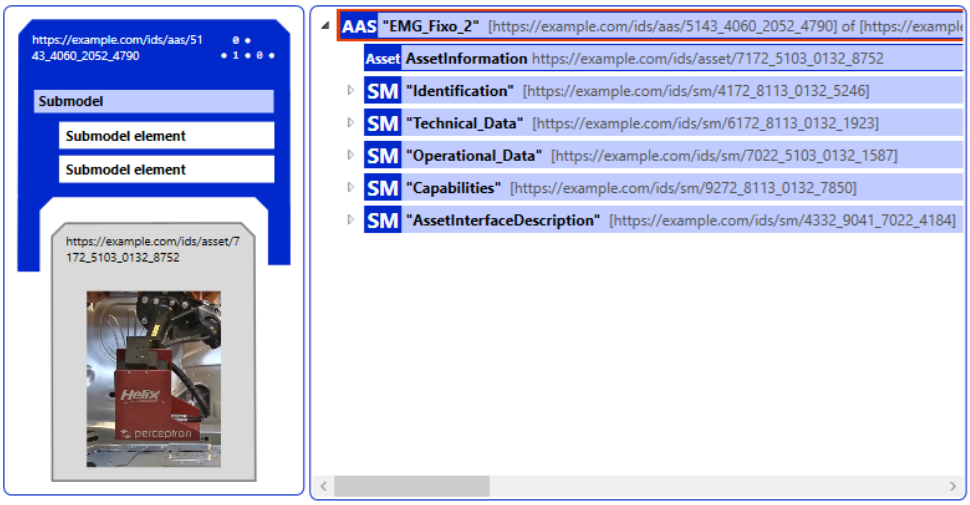


Figura 5.8: Interface do modelo passivo do AAS para o EMG fixo 2.

Apesar de medirem regiões diferentes do carro, ambos os EMGs fixos 2 e 3 apresentam a estrutura do submodelo *Operational Data* mostrada na Figura 5.9 como resultado, onde o elemento construído para o EMG fixo 3 apresenta a mesma estrutura definida no EMG fixo 2. No entanto, como dito anteriormente, cada dispositivo possui seu próprio modelo representando de forma digital o ativo, diferenciando-os no sistema por meio do identificador único disponibilizado na ferramenta digital.

A funcionalidade de padronização de elementos da ferramenta *AASX Package Explorer*, mesmo em caso de medições de pontos em estações e componentes distintos do carro, garante a uniformidade na estrutura dos submodelos construídos, facilitando a interoperabilidade e o processamento dos dados adquiridos pelos diferentes dispositivos de medição.

Portanto, os resultados dos modelos do AAS dos três dispositivos de medição exibem uma hierarquia viável para o funcionamento do projeto, permitindo a integração eficiente dos dados em um ambiente digital. A implementação dos modelos AAS demonstra ser uma solução eficaz para a gestão e monitoramento das informações no contexto do trabalho.

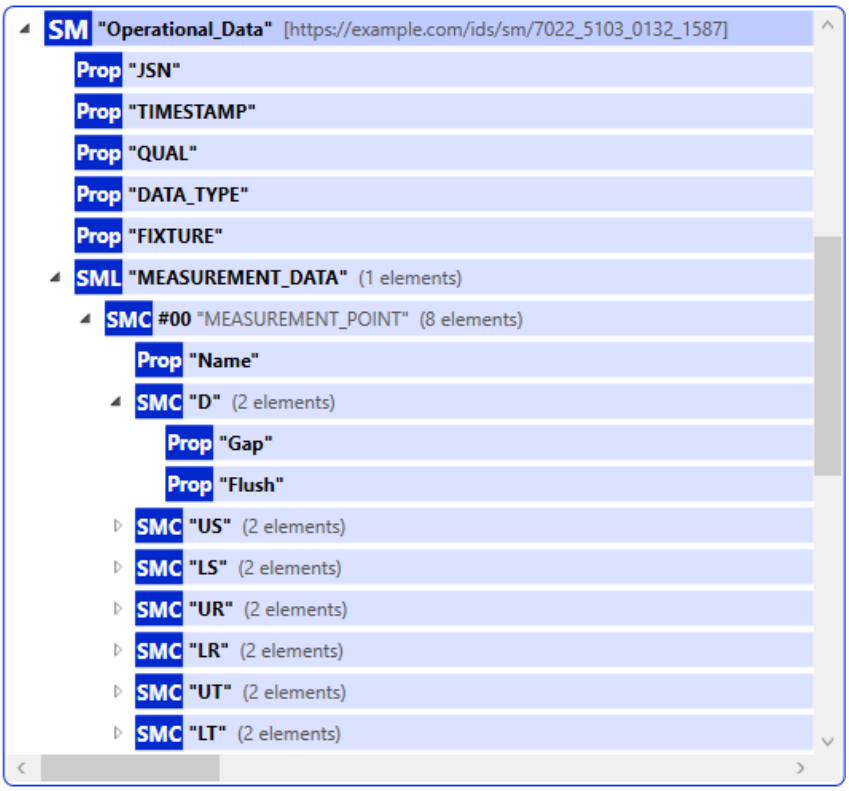


Figura 5.9: Interface do submodelo *Operational Data* para o EMG fixo 2.

### 5.1.2 Modelos do AAS para o EMG Portátil

Os modelos do AAS construídos para os dispositivos de medição portáteis apresentaram resultados satisfatórios, evidenciando a funcionalidade da ferramenta digital ao fornecer elementos que permitem a estruturação de ativos em ambiente industrial. O modelo do AAS criado para o EMG portátil 4 possui resultados de sua estrutura similares às estruturas de AAS dos EMGs fixos, evidenciando a padronização dos submodelos e a consistência na integração dos dados adquiridos, independentemente do tipo de equipamento utilizado.

Portanto, os resultados dos painéis de exibição do ativo e navegação podem ser vistos na Figura 5.10, exibindo de forma gráfica o EMG portátil, assim como a estrutura hierárquica construída para o ativo. A interface das definições do AAS, quando comparada ao EMG fixo, apresenta um layout semelhante, manifestando diferenças na nomenclatura e identificador único do ativo.

Agora, representado na Figura 5.11, os resultados do submodelo *Operational Data*

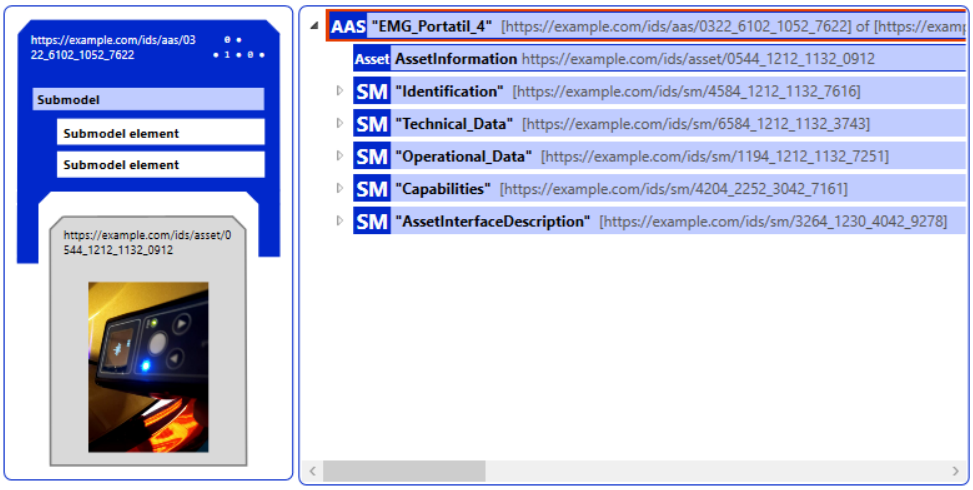


Figura 5.10: Interface do modelo passivo do AAS para o EMG portátil 4.

para o EMG portátil 4 exibem 5 propriedades de valores vazios, capacitando a alocação de dados do dispositivo no modelo reativo. Isto permite que informações como a cor e a estação de medição do carro sejam definidas durante a transmissão de dados no servidor AAS, viabilizando o uso para o modelo reativo do sistema.

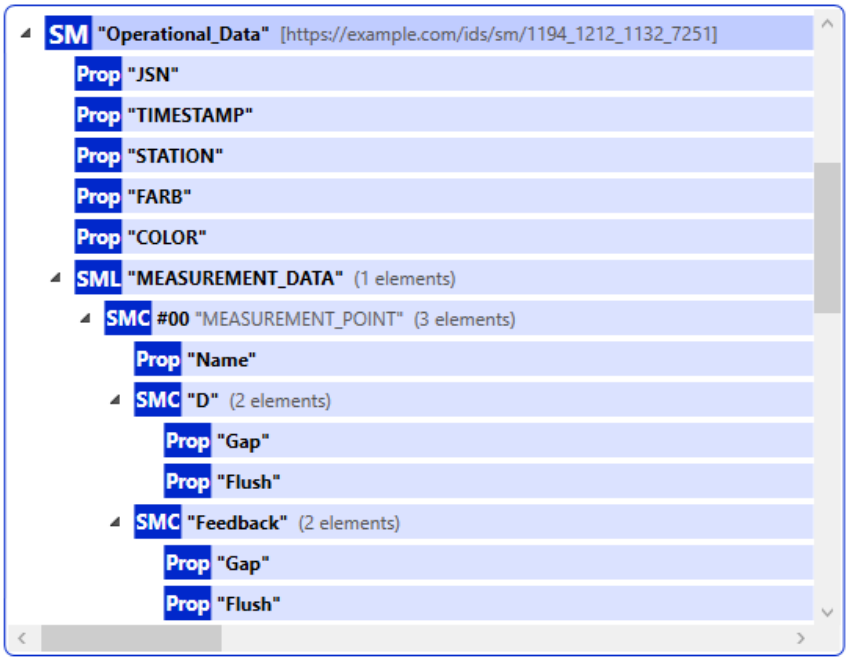


Figura 5.11: Interface do submodelo *Operational Data* para o EMG portátil 4.

Além disso, o elemento *SubmodelElementList* possui como resultado uma estrutura

com 2 critérios principais: D e Feedback, os quais irão atribuir os valores de *Gap* e *Flush* dos equipamentos de medição em seus respectivos campos. Com a propriedade *Name*, que permite o armazenamento do nome do ponto medido, valida-se o uso desta estrutura para comportar os múltiplos pontos existentes nos dados do ficheiro do dispositivo de medição.

Neste caso de estudo, devido às grandes semelhanças, os EMGs portáteis presentes nas estações 5 a 23 utilizam o modelo AAS do EMG Portátil 4, apresentando distinção apenas no identificador único da estrutura.

## 5.2 Resultados do AAS Tipo 2

Os resultados da implementação do modelo reativo do AAS utilizam as estruturas do modelo passivo do AAS, interconectando o servidor baseado em *FastAPI* aos dados de medição obtidos pelos EMGs fixos e portáteis. Dessa forma, é possível visualizar, por meio do protocolo de comunicação HTTP, os modelos AAS dos ativos com seus dados devidamente integrados.

### 5.2.1 Dados do EMG Fixo no Servidor AAS

Os dispositivos de medição fixos, quando relacionados com a estrutura do AAS, demonstram graficamente os dados de seus pontos obtidos nas estações, reestruturados de acordo com a hierarquia do submodelo *Operational Data*. Isto resulta em um ambiente padronizado e organizado via HTTP, permitindo uma análise mais precisa e rápida dos dados, assim como a integração desses a outros sistemas da indústria.

Os resultados do modelo reativo para as estações de medição 1, 2 e 3 permitem analisar as características do ativo, como sua identificação e dados técnicos apresentados, além de suas capacidades e protocolos de comunicação utilizados. Também são exibidos seus dados operacionais, estes enviados por meio dos EMGs fixos presentes na linha de montagem inicial. Nesse contexto, os resultados obtidos para o EMG fixo 1 são aplicáveis aos resultados dos EMGs fixos 2 e 3, que possuem modelos passivos do AAS construídos de forma semelhante, com exceção da estrutura do submodelo *Operational Data*, que

apresenta coordenadas de pontos medidos distintas em cada estação.

A inicialização do servidor AAS, podendo ser vista na Figura 5.12, exibe detalhes de conexão e informações da estrutura AAS. Determinando o EMG conectado e seu diretório no HTTP, a solicitação de dados do submodelo *Operational Data* através do método de requisição GET do HTTP também é demonstrada, assegurando que a comunicação entre o servidor AAS e o EMG fixo esteja funcionando corretamente.

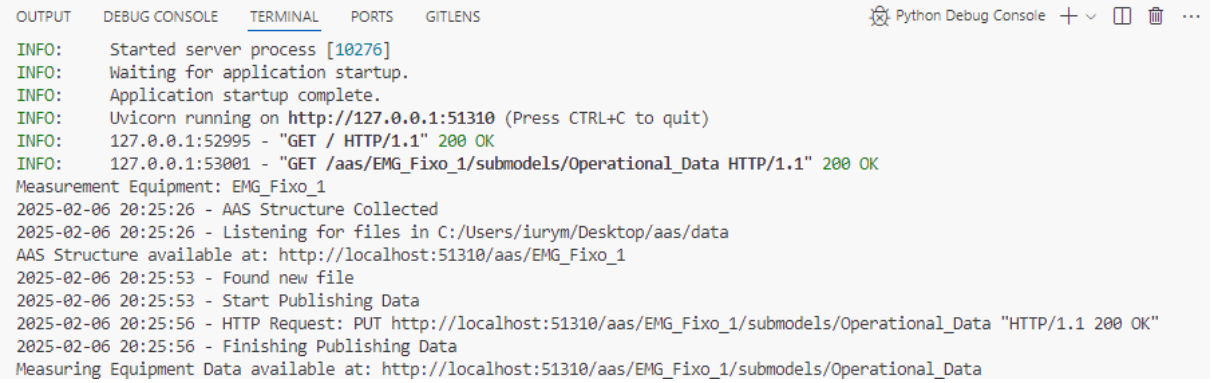


Figura 5.12: Terminal da inicialização e publicação de dados do EMG fixo no servidor AAS.

O terminal também apresenta o processo de início de publicação de dados, ocorrendo após um novo ficheiro ser identificado na pasta compartilhada. Assim, o servidor recebe uma requisição HTTP do tipo PUT, utilizada para atualizar ou enviar novos dados de medição obtidos pelos EMGs fixos. Estas informações, assim como no caso anterior, garantem um processamento correto dos dados disponibilizados pelo dispositivo.

Dessa forma, nas Figura 5.13 e Figura 5.14, observam-se respectivamente os resultados dos submodelos *Identification* e *Technical Data* para o EMG fixo 1 no servidor AAS. Com base nos modelos passivos do AAS, suas estruturas revelam os dados técnicos e de identificação do ativo, permitindo a análise detalhada das características do ativo, bem como o monitoramento contínuo de seu desempenho. Ressalta-se que os dados apresentados são fictícios, assegurando a proteção das informações do ativo da empresa parceira.


Figura 5.13: Submodelo *Identification* do EMG fixo 1 no servidor AAS.

Portanto, no elemento *Identification* a estrutura esclarece particularidades como a marca, modelo e código de identificação do ativo, sendo também possível observar no elemento *Technical Data* suas dimensões e peso, incluindo o tempo de duração da bateria do ativo utilizado. Tais funcionalidades também serão exibidas em outras estações que utilizam o EMG fixo, com uma estrutura semelhante e valores distintos, conforme as características específicas de cada ativo.


Figura 5.14: Submodelo *Technical Data* do EMG fixo 1 no servidor AAS.

Adicionalmente, é evidenciado o caminho específico de acesso aos submodelos do EMG fixo 1, em que cada estação possui seu diretório determinado, o qual pode ser alterado manualmente no código de configuração implementado para o caso de estudo. Isso garante a flexibilidade na modificação dos diretórios de acordo com as exigências do sistema, possibilitando uma organização adaptável dos dados do ativo.

Em sequência a isto, o servidor AAS visualizado na Figura 5.15 exibe os resultados do submodelo *Operational Data*, sendo o principal elemento de validação neste projeto. O submodelo informa os parâmetros de qualidade, definidos como *Roof*, *Qual*, *Data Type*

e *Fixture*, assegurando a coerência das informações para a análise dos pontos do EMG.


Figura 5.15: Submodelo *Operational Data* do EMG fixo 1 no servidor AAS.

Esta estrutura mostra também, por meio da propriedade *JSN*, o código de identificação único do carro em análise, sendo incrementada pela propriedade *Timestamp*, que identifica o momento exato da análise dos pontos do carro durante a medição, concedendo um carimbo de data e hora em formatos padronizados. Esses elementos possibilitam o rastreamento preciso do carro, viabilizando a análise detalhada e organizada ao longo das estações de medição.


Figura 5.16: Submodelo *Measurement Data* do EMG fixo 1 no servidor AAS.

Além disso, para análises em tempo real, o submodelo de lista *Measurement Data* do EMG fixo 1, presente na Figura 5.16, exibe os valores dos critérios D/US/LS/UR/LR/UT/LT, coletados pelo dispositivo de medição, para as coordenadas tridimensionais X, Y e Z, localizadas na estrutura do chassi do carro. Devido à confidencialidade de dados, alguns valores foram censurados, não comprometendo o entendimento dos dados enviados ao servidor.

Os EMGs fixos 2 e 3, por sua vez, exibem os mesmos critérios, mas para os pontos *Gap* e *Flush*, localizados respectivamente nas portas dianteiras e traseiras do carro, em relação ao monobloco (estrutura superior) do veículo, podendo ser visto na Figura 5.17. O nome de

cada ponto medido também é revelado na propriedade *Name*, permitindo identificar cada ponto associado. Com auxílio do elemento *SubmodelList*, essas estruturas são reproduzidas sequencialmente, de modo a comportar todos os pontos existentes durante a execução do servidor AAS, permitindo integrar todos os dados dos EMGs fixos.


Figura 5.17: Submodelo *Measurement Data* do EMG fixo 2 no servidor AAS.

Por fim, podendo ser visto na Figura 5.18, e na Figura 5.19 até a Figura 5.23 , os submodelos *Capabilities* e *Asset Interface Description*, assim como os submodelos *Identification* e *Technical Data*, fornecem informações detalhadas sobre as características do ativo. Sua capacidade é exibida pela habilidade de medição, enquanto o elemento de interface apresenta os protocolos de comunicação e segurança utilizados, além de descrever os elementos interconectados no sistema por meio de identificadores únicos.


Figura 5.18: Submodelo *Capabilities* do EMG fixo 1 no servidor AAS.


Figura 5.19: Parte 1 do Submodelo *Asset Interface Description* do EMG fixo 1 no servidor AAS.


Figura 5.20: Parte 2 do Submodelo *Asset Interface Description* do EMG fixo 1 no servidor AAS.


Figura 5.21: Parte 3 do Submodelo *Asset Interface Description* do EMG fixo 1 no servidor AAS.


Figura 5.22: Parte 4 do Submodelo *Asset Interface Description* do EMG fixo 1 no servidor AAS.


Figura 5.23: Parte 5 do Submodelo *Asset Interface Description* do EMG fixo 1 no servidor AAS.

### 5.2.2 Dados do EMG Portátil no Servidor AAS

A integração de dados dos dispositivos de medição portáteis no servidor AAS, nas estações de medição 4 a 23, apresenta resultados semelhantes à integração dos dados do EMG fixo, devido à implementação padronizada do modelo passivo do AAS. Por meio da interface construída via RestAPI, são visualizadas as aquisições de dados em tempo real dos EMGs portáteis da linha automotiva.

Os resultados do AAS Tipo 2 para estes dispositivos permitem a visualização e análise das informações do ativo de acordo com a estrutura dos submodelos apresentadas no AAS Tipo 1 do trabalho. Assim, serão exibidos os dados obtidos no servidor AAS do EMG portátil 4, estes aplicáveis como resultado das estações 5 a 23 devido ao alto nível de similaridade.

Assim, mostrada na Figura 5.24, o terminal de inicialização do servidor demonstra os detalhes de integração do servidor com o modelo do AAS. Como ocorria no terminal do servidor para o EMG fixo, a solicitação de dados do submodelo *Operational Data* por meio do método de requisição GET do HTTP é apresentada, determinando a estação de medição vinculada e seu respectivo endereço HTTP. Esse método permite consultar dados operacionais do dispositivo de medição, assegurando a comunicação correta entre o servidor AAS e os EMGs portáteis.

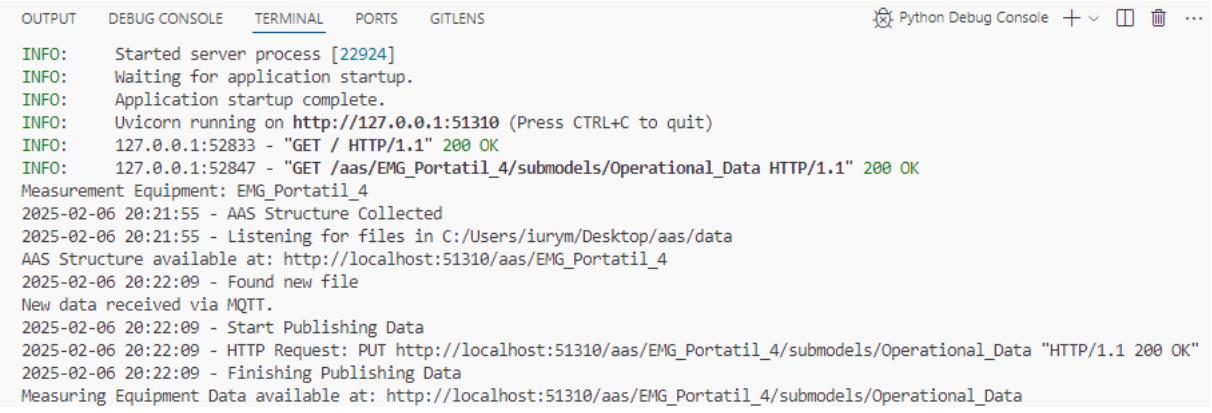


Figura 5.24: Terminal da inicialização e publicação de dados do EMG portátil no servidor AAS.

Além disso, por meio de uma mensagem, é mostrado o registro de dados recebidos via

MQTT, validando a chegada correta das informações por meio desse protocolo de comunicação. Com o fim da publicação de dados no servidor AAS, os diretórios para acessar à estrutura do AAS e dos dados do dispositivo de medição são registrados, agilizando o processo de acesso à interface do servidor AAS.

Desse modo, os resultados do submodelo *Identication* podem ser vistos na Figura 5.25, disponibilizando as características informativas do ativo. Entre elas, destacam-se a marca e o modelo do dispositivo de medição, além do identificador único da interface de rede, conhecido como *Mac Adress*, e o nome atribuído ao dispositivo, o *Host Name*.


Figura 5.25: Submodelo *Identification* do EMG portátil 4 no servidor AAS.

Junto a este, o submodelo *Technical Data* fornece como resultado as informações técnicas do ativo. Ilustrado na Figura 5.26, as especificações de interface, peso, dimensões e autonomia da bateria garantem a adequação do dispositivo às necessidades operacionais e industriais. Estes dois submodelos contribuem com a atribuição dos dados em tempo real, permitindo o rastreamento preciso dos ativos e a transparência das informações transmitidas no servidor AAS.


Figura 5.26: Submodelo *Technical Data* do EMG portátil 4 no servidor AAS.

Em seguida, mostrado na Figura 5.27, o servidor AAS apresenta os resultados do elemento *Operational Data*, descrevendo inicialmente, por meio de propriedades, a estação do equipamento de medição e seu código de identificação única, respectivamente nomeados *Station* e *JSN*. Em adição a estas, o elemento *Timestamp* atribui o momento de medição do equipamento, permitindo visualizar o carimbo de data e hora e, consequentemente,

registrar o avanço de cada unidade no processo. Os elementos *Color* e *Farb* exibem por fim a cor e seu código utilizados, servindo como uma referência visual para a identificação da pintura do veículo.


Figura 5.27: Submodelo *Operational Data* do EMG portátil 4 no servidor AAS.

Integrados no *Operational Data*, o submodelo de lista *Measurement Data*, para os EMGs portáteis, exibe os dados obtidos dos critérios D e Feedback, como mostra na Figura 5.28. O critério D atribui os valores de *Gap*, ou seja, a distância entre as superfícies, e de *Flush*, o alinhamento entre os componentes, enquanto o *Feedback* retorna os valores verdadeiro (1) e falso (0), conforme a concordância dos pontos nos limites de intervalos pré-definidos pela empresa. Por fim, a propriedade *Name* mostra o nome do ponto medido

pelo dispositivo de medição, identificando qual região apresenta esses valores.


Figura 5.28: Submodelo *Measurement Data* do EMG portátil 4 no servidor AAS.

Por último, visualizado nas Figura 5.29 e Figura 5.30, os resultados do submodelo *Capabilities* apresentam os dados técnicos de medição do dispositivo e sua habilidade de medição, informando a distância de operação do laser em suas coordenadas, bem como a velocidade de medição. Em cooperação, o submodelo *Asset Interface Description* nas Figura 5.31, Figura 5.32, Figura 5.33 e Figura 5.34, demonstra as descrições de interface do ativo, apresentando as informações de conexão e especificações do protocolo de comunicação MQTT, além de suas configurações de segurança, simulando os dados em tempo real pela empresa automotiva.


Figura 5.29: Parte 1 do submodelo *Capabilities* do EMG portátil 4 no servidor AAS.

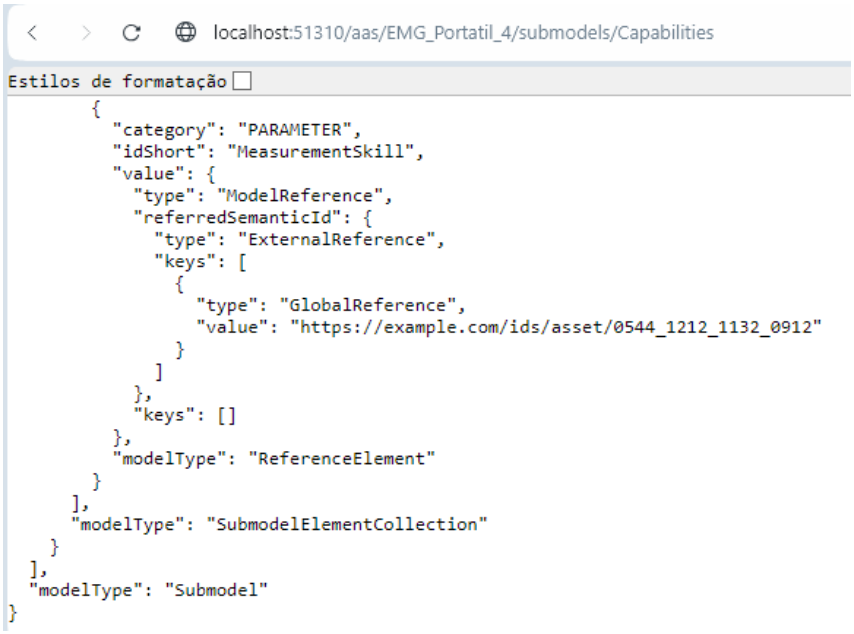


Figura 5.30: Parte 2 do submodelo *Capabilities* do EMG portátil 4 no servidor AAS.


Figura 5.31: Parte 1 do submodelo *Asset Interface Description* do EMG portátil 4 no servidor AAS.


Figura 5.32: Parte 2 do submodelo *Asset Interface Description* do EMG portátil 4 no servidor AAS.


Figura 5.33: Parte 3 do submodelo *Asset Interface Description* do EMG portátil 4 no servidor AAS.


Figura 5.34: Parte 4 do submodelo *Asset Interfaces Description* do EMG portátil 4 no servidor AAS.

## 5.3 Inicialização e Envio de Dados no Servidor AAS

A análise dos tempos de inicialização do servidor AAS e da transferência de dados dos dispositivos de medição ao servidor AAS são essenciais para determinar a existência de uma aquisição de dados em tempo real no cenário industrial. Assim, os resultados obtidos em ambiente de simulação foram influenciados pelas condições operacionais descritas na Tabela 5.1, onde o sistema operacional *Windows 11* foi executado em conjunto com um processador Intel Core i7-11800H em modo de alto desempenho, garantindo uma latência reduzida na transmissão de dados.

| **Característica** | **Descrição** |
|---|---|
| Sistema Operacional | Microsoft Windows 11 |
| Versão do SO | 10.0.22631 Compilação 22631 |
| Arquitetura | 64 bits |
| Processador (CPU) | Intel Core i7-11800H @ 2.30GHz, 2304 Mhz |
| Memória RAM | 16GB DDR4 3200MHz |
| Tipo de Armazenamento | SSD 480GB |
| Observações Extras | Execução em modo de alto desempenho |

Tabela 5.1: Sistema operacional do ambiente simulado.

### Desempenho da Inicialização do Servidor AAS

Inicialmente, podendo ser visto na Tabela 5.2, os resultados dos tempos de inicialização do servidor foram obtidos por meio de 10 amostras, nas quais foi medido o tempo total desde o acionamento do servidor AAS até o momento de estabilização dos processos definidos no código de programação.

Tendo isso em vista, esses tempos determinam a duração da inicialização da *infraestrutura* e o carregamento dos serviços até a total disponibilidade do sistema para operação, considerando possíveis variações devido a fatores como carga computacional e distribuição de recursos. Vale destacar que esses tempos não possuem relação com os dados obtidos dos EMGs fixos e portáteis, exibindo apenas o tempo necessário para que o servidor esteja completamente operacional e pronto para o processamento dos dados.

| Simulação | Tempos (s) |
|---|---|
| 1 | 4,27 |
| 2 | 4,14 |
| 3 | 4,12 |
| 4 | 4,15 |
| 5 | 4,16 |
| 6 | 4,14 |
| 7 | 4,16 |
| 8 | 4,17 |
| 9 | 4,14 |
| 10 | 4,16 |
| **Média** | $4,\overline{161}$ |
| **Desvio Padrão** | $0,041$ |

Tabela 5.2: Tempos de inicialização do servidor AAS.

Assim, os valores coletados foram analisados estatisticamente, determinando a média e o desvio padrão dos tempos obtidos nas simulações. Com um tempo médio de 4,161 segundos para a inicialização do servidor AAS, o sistema demonstra bom desempenho em ambiente industrial, tendo em vista que, em necessidade de reinicializações imprevistas, este possui como vantagem uma rápida recuperação, reduzindo impactos na coleta e análise de dados nas linhas de produção.

Além disso, o cálculo do desvio padrão exibe um valor de 0,041 segundos, apresentando um desempenho estável na inicialização do servidor AAS. Este fator nos permite determinar o nível de consistência do sistema em um cenário automotivo, garantindo o processamento de dados do servidor AAS em tempo real de forma confiável. Em trabalhos futuros, a redução destes valores é dada com a otimização do código de programação do servidor implementado, eliminando redundâncias e utilizando estruturas mais eficientes em seu desenvolvimento.

### Eficiência do Envio de Dados para o Servidor AAS

Com a coleta dos dados apresentados pelos EMGs fixos e portáteis, é possível visualizar os resultados dos tempos de envio de dados ao servidor AAS. Utilizando 10 amostras de simulação para cada dispositivo de medição implementado no projeto. O número de

10 amostras é justificado pelo elevado tempo necessário para gerar dados fictícios nas simulações, considerando que cada estação de medição dos EMGs possui milhares de pontos coletados ao longo da linha de montagem.

| Simulação | Tempos EMG Fixo 1 (s) | Tempos EMG Fixo 2 (s) | Tempos EMG Fixo 3 (s) | Tempos EMG Portátil 4 (s) |
|---|---|---|---|---|
| 1 | 3,169 | 0,860 | 0,427 | 0,066 |
| 2 | 3,107 | 0,872 | 0,414 | 0,074 |
| 3 | 3,074 | 0,874 | 0,412 | 0,068 |
| 4 | 3,157 | 0,863 | 0,435 | 0,067 |
| 5 | 3,075 | 0,865 | 0,409 | 0,073 |
| 6 | 3,084 | 0,937 | 0,417 | 0,069 |
| 7 | 3,244 | 0,895 | 0,500 | 0,075 |
| 8 | 3,125 | 0,896 | 0,426 | 0,069 |
| 9 | 3,143 | 0,859 | 0,429 | 0,067 |
| 10 | 3,070 | 0,868 | 0,405 | 0,070 |
| **Média** | 3,1248 | 0,8789 | 0,4274 | 0,0698 |
| **Desvio Padrão** | 0,055 | 0,024 | 0,027 | 0,003 |

Tabela 5.3: Tempo de envio de dados dos EMGs ao servidor AAS.

Dessa forma, conforme observado na Tabela 5.3 e na Figura 5.35, os resultados dos tempos de transferência de dados dos dispositivos de medição possuem valores abaixo de 4 segundos. A média de tempos de envio de dados é de 3,1248s para o EMG fixo 1, 0,8789s para o EMG fixo 2, 0,4274s para o EMG fixo 3, e 0,0698s para o EMG portátil 4.

As diferenças de tempo entre os EMGs ocorrem devido ao número de processamentos realizados, que varia conforme a quantidade de critérios (D, US, LS, UR, LR, UT, LT, *Feedback*) e coordenadas medidas (X, Y, Z, *Gap*, *Flush*). Por exemplo, o EMG fixo 1 que apresenta um tempo maior precisa processar três coordenadas em sete critérios. Por outro lado, os EMGs fixos 2 e 3 processam duas coordenadas nos mesmos sete critérios, enquanto o EMG portátil 4 realiza o processamento para duas coordenadas, mas apenas dois critérios, resultando em um tempo menor de envio de dados quando comparado ao primeiro dispositivo. Isso demonstra que, quanto menor o número de critérios e coordenadas, mais rápido é o envio de dados e menor o tempo de processamento no servidor.

Essas diferenças de tempo ocorrem também em razão da quantidade de pontos medidos em cada estação. Por exemplo, o EMG fixo 2, que possui os mesmos critérios e coordenadas de medição do EMG fixo 3, apresenta um volume maior de pontos medidos, resultando em um envio mais lento de dados ao servidor AAS quando comparado ao EMG fixo 3. Este fator também é válido para o EMG fixo 1 e o EMG portátil 4, já que a estação de medição do EMG fixo 1 possui um número elevado de pontos medidos em comparação à estação de medição do EMG portátil 4.

Além disso, também são demonstrados os valores de desvio padrão de cada dispositivo, evidenciando a variabilidade no tempo de transferência dos dados ao servidor. Considerando que o número de pontos enviados ao servidor AAS é o mesmo para cada dispositivo individualmente, a diferença nos desvios padrão é atribuída a flutuações no uso de recursos do sistema, como processos em segundo plano que competem pelo uso da CPU e da memória durante os testes.

Vale ressaltar que os desvios padrões não podem ser comparados entre os diferentes dispositivos, pois cada um possui um número distinto de pontos de medição, implicando em volumes de dados diferentes que estão sendo enviados ao sistema.

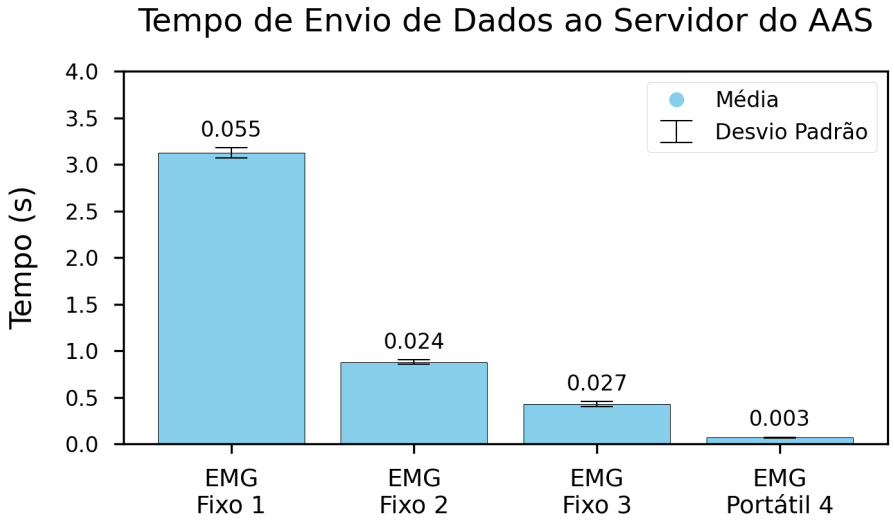


Figura 5.35: Média e desvio padrão do tempo de envio de dados ao servidor AAS.

Por outro lado, como a análise dos dados é realizada enquanto o carro ainda está na esteira de produção, onde o tempo de permanência do veículo em uma estação de medição

varia entre 60 segundos e 120 segundos, o tempo médio de envio de dados dos EMGs ao servidor AAS é relativamente curto, o que permite a aquisição de dados em tempo real para o monitoramento.

Portanto, tendo em conta o ambiente industrial de uma linha automotiva, o tempo médio da transferência de dados ao servidor AAS obtido pelos EMGs apresentou resultados satisfatórios, validando a aquisição de dados em tempo real nesse cenário. Embora a redução do tempo de transmissão seja benéfica em situações que exigem intervenções imediatas, como ajustes automáticos na linha de produção, o objetivo deste projeto foi garantir a interoperabilidade e a padronização na aquisição dos dados, priorizando a integridade do monitoramento em tempo real do sistema.

## 5.4 Considerações do AAS em Ambiente Industrial

A plataforma de interface gráfica *AASX Package Explorer*, quando comparada a outras ferramentas como o *Eclipse BaSyx*, permite a criação de submodelos do tipo AAS de maneira ágil no cenário tecnológico. Seu fácil entendimento e implementação tornam-na uma opção viável para desenvolvedores que procuram soluções eficientes ao integrar e gerenciar dados de ativos industriais. Em adição a isto, por meio da plataforma IDTA, nos é permitido o acesso às instruções em formato de vídeo referentes aos principais conceitos presentes na ferramenta digital, estas que podem ser visualizadas a partir do servidor *Admin Shell Io* (https://admin-shell-io.com) localizado na web.

Contrário a este ponto e consequente ao seu baixo tempo de desenvolvimento, a ferramenta ainda necessita de melhorias com o intuito de tornar a interface amigável ao usuário, apresentando diversos erros e bugs que ocorrem frequentemente ao utilizar a aplicação. O desenvolvimento de novas funcionalidades dentro do *software* se torna essencial para o aumento do leque de possibilidades no momento do uso da ferramenta, estas que a tornariam ainda mais flexível em aplicações industriais.

Na construção do modelo passivo do AAS, utilizando como base as documentações definidas pela *Platform Industrie 4.0* e IDTA, outro ponto notável como um desafio no

desenvolvimento do projeto é a falta de normas implementadas e validadas pela plataforma IDTA. Embora a comunidade apresente atualizações frequentes na plataforma, poucos estudos com aplicação em cenários industriais estão oficialmente publicados, tornando-se uma consequência na limitação de possibilidades de criação de submodelos no projeto que atendessem às normas designadas anteriormente.

Com a atualização da ferramenta *AASX Package Explorer*, mais especificamente a versão 3.0.1 publicada em setembro de 2023, o desenvolvimento do projeto se tornou mais simplificado em termos de criação de submodelos que comportassem os pontos provindos das medições dos EMGs. O elemento SML, que antes era inexistente na ferramenta, permitiu que o método anteriormente utilizado fosse otimizado durante o tratamento de dados dos pontos presentes no submodelo *Operational Data*, estes que para cada ponto de medição durante os testes da ferramenta digital exigiam a criação de um submodelo para comportar seus valores, sendo um dos grandes desafios na integração dos modelos passivo e reativo do AAS durante a implementação do trabalho.

Esta proposta de adição e remoção de um grande número de pontos e parametrizações provindos dos EMGs tornou-se inviável no estudo, exigindo uma grande carga computacional durante a transmissão de dados em tempo real, além da adição e remoção dos submodelos do AAS no decorrer do funcionamento. Em razão disso, comportando todos os submodelos presentes na estrutura do *AASX Package Explorer*, o excesso de métodos de requisições *REST* torna-se necessário ao rodar o servidor, aumentando o tempo de envio de dados ao servidor.

Apesar de contribuir com o desenvolvimento do projeto, as atualizações de grande escala também mostraram pontos desinteressantes ao longo do projeto, complicando a implementação do AAS passivo conforme características do ativo. A falta de compatibilidade entre versões da ferramenta digital mostrou limitações ao acessar ficheiros antigos em novas versões, assim como em novos ficheiros em versões antigas, exigindo a reconstrução do modelo passivo do AAS para continuidade do caso de estudo.

Em contraste, um aspecto vantajoso de grande relevância é a opção de exportar os

submodelos criados dentro da ferramenta nos formatos XML e JSON, assim como a capacidade de transformar esses submodelos em URLs para futuras importações dentro da aplicação, permitindo uma integração eficiente do AAS Tipo 1 e Tipo 2 no sistema.

Ainda, com o uso da estrutura *FastAPI* para o desenvolvimento do AAS em todas as estações, a aplicação que antes demonstrava desempenho limitado apresentou fácil adaptação à estrutura de análise de dados do projeto *openZDM*, revelando-se eficaz na obtenção dos resultados esperados neste caso de estudo. A interface possibilitou a construção do servidor AAS por meio do protocolo de comunicação HTTP, que recebe e responde às requisições feitas dos dados que chegam por meio do monitoramento de ficheiros, viabilizando assim a comunicação entre o ativo e o modelo passivo do AAS.

Junto a isso, o uso do protocolo de comunicação MQTT demonstrou fácil integração com a *FastAPI*, sendo aplicada com o intuito de receber os dados do EMG portátil para posterior análise destes em tempo real. Ao utilizar a ferramenta *Node-RED* e o *broker Mosquitto*, em ambiente de testes e com dados fictícios, foi possível estabelecer uma conexão flexível com o sistema, facilitando a transmissão contínua e segura dos dados do dispositivo de medição, permitindo o envio de dados ao servidor AAS.

Portanto, a ferramenta *AASX Package Explorer* demonstrou grande potencial de interoperabilidade entre sistemas que utilizam ferramentas da engenharia para solucionar problemas da indústria. Com o intuito de buscar a padronização e integração, a implementação do AAS Tipo 1 dos EMGs por intermédio da ferramenta de interface gráfica se mostrou viável, suprindo todas as necessidades no momento da construção da estrutura do AAS. O mesmo transpareceu para o desenvolvimento do AAS Tipo 2, esta que gerenciou em tempo real os dados obtidos pelos dispositivos de medição em todas as estações, permitindo a análise e detecção de falhas dessas informações, validando a proposta inicial de nosso caso de estudo.

# Capítulo 6

# Conclusões e Trabalhos Futuros

## 6.1 Conclusões

Abordando anteriormente problemas relacionados à ausência de interoperabilidade entre sistemas, assim como a necessidade de documentações neste meio, ao decorrer deste projeto são desenvolvidas a digitalização de ativos industriais no setor automotivo, utilizando conceitos e padrões apresentados pelo AAS. Considerando parte do escopo do projeto *openZDM*, o estudo é desenvolvido a partir de um cenário real de uma linha de montagem automobilística, construindo através de ferramentas do AAS e da linguagem de programação Python os modelos passivo e reativo dos equipamentos de medição geométricos presentes na indústria.

Sendo assim, o desenvolvimento levou em conta os estudos da arquitetura RAMI 4.0 e os conceitos fundamentais dos CPSs, buscando promover a integração e padronização entre estes sistemas durante a criação do AAS Tipo 1 e Tipo 2. Para o modelo passivo, foi utilizado o *AASX Package Explorer* de modo a esclarecer as especificações e dados técnicos dos EMGs da indústria parceira, implementando submodelos publicados anteriormente na plataforma IDTA. Por sua vez, o modelo reativo foi elaborado por meio da estrutura *FastAPI*, auxiliando na criação do servidor AAS e na integração em tempo real dos dados com a estrutura do AAS Tipo 1.

Portanto, por intermédio das normas estabelecidas pela IDTA, o modelo passivo mostrou bons resultados no caso de estudo, garantindo que os dados estruturados estejam integrados com o sistema de monitoramento do modelo reativo, além de facilitar a visualização e configuração das propriedades dos equipamentos de medição geométrica. Ainda, a implementação do modelo reativo se mostrou validada, garantindo a interoperabilidade dos dados coletados do ativo com o servidor AAS ao permitir análises futuras para detecção e predição de falhas relacionadas às medições de X, Y, Z, *Gap* e *Flush* das peças automotivas, contribuindo na melhoria dos processos produtivos da empresa.

## 6.2 Trabalhos Futuros

Considerando os resultados obtidos durante o desenvolvimento do caso de estudo, bem como as possíveis melhorias e simulações do processo, os tópicos a seguir descrevem as direções para trabalhos futuros:

- Desenvolvimento dos modelos passivo e reativo do produto do carro, este que irá comportar as informações dos EMGs fixos e portáteis de todas as estações de medição da empresa parceira, permitindo uma análise simplificada e eficiente de dados dos dispositivos.
- Implementação do modelo proativo (AAS Tipo 3), adotando os sistemas multiagentes com o intuito de permitir a interação autônoma e a troca de informações entre os diferentes ativos de um sistema.

# Bibliografia


[1] N. G. P. Carvalho e E. W. Cazarini, "Industry 4.0-What Is It?" Em *Industry 4.0-Current Status and Future Trends*, IntechOpen, 2020.

[2] J. H. Ortiz, *Industry 4.0: Current status and future trends*. IntechOpen, 2020.

[3] PI4.0, "The Asset Administration Shell: Implementing Digital Twins for Use in Industrie 4.0, Platform Industrie 4.0," ZVEI: Die Eletroindustrie.

[4] D. Gürdür e F. Asplund, "A systematic review to merge discourses: Interoperability, integration and cyber-physical systems," *Journal of Industrial information integration*, vol. 9, pp. 14–23, 2018.

[5] R. Alguliyev, Y. Imamverdiyev e L. Sukhostat, "Cyber-physical systems and their security issues," *Computers in Industry*, vol. 100, pp. 212–223, 2018.

[6] M. Hankel e B. Rexroth, "The reference architectural model industrie 4.0 (rami 4.0)," *Zvei*, vol. 2, n.º 2, pp. 4–9, 2015.

[7] X. Ye, S. H. Hong, W. S. Song, Y. C. Kim e X. Zhang, "An industry 4.0 asset administration shell-enabled digital solution for robot-based manufacturing systems," *Ieee Access*, vol. 9, pp. 154 448–154 459, 2021.

[8] V. Siatras, E. Bakopoulos, P. Mavrothalassitis, N. Nikolakis e K. Alexopoulos, "On the use of asset administration shell for modeling and deploying production scheduling agents within a multi-agent system," *Applied Sciences*, vol. 13, n.º 17, p. 9540, 2023.

[9] Y. Lu, "Industry 4.0: A survey on technologies, applications and open research issues," *Journal of industrial information integration*, vol. 6, pp. 1–10, 2017.

[10] L. D. Xu, E. L. Xu e L. Li, "Industry 4.0: state of the art and future trends," *International journal of production research*, vol. 56, n.º 8, pp. 2941–2962, 2018.

[11] J. Lee, B. Bagheri e H.-A. Kao, "A cyber-physical systems architecture for industry 4.0-based manufacturing systems," *Manufacturing letters*, vol. 3, pp. 18–23, 2015.

[12] J.-R. Jiang, "An improved cyber-physical systems architecture for Industry 4.0 smart factories," *Advances in Mechanical Engineering*, vol. 10, n.º 6, p. 1 687 814 018 784 192, 2018.

[13] B. Dafflon, N. Moalla e Y. Ouzrout, "The challenges, approaches, and used techniques of CPS for manufacturing in Industry 4.0: a literature review," *The International Journal of Advanced Manufacturing Technology*, vol. 113, pp. 2395–2412, 2021.

[14] G. Weichhart, H. Panetto e A. Molina, "Interoperability in the cyber-physical manufacturing enterprise," *Annual Reviews in Control*, vol. 51, pp. 346–356, 2021.

[15] D. Mourtzis, E. Vlachou, G. Dimitrakopoulos e V. Zogopoulos, "Cyber-physical systems and education 4.0–the teaching factory 4.0 concept," *Procedia manufacturing*, vol. 23, pp. 129–134, 2018.

[16] D. Mourtzis, A. Gargallis e V. Zogopoulos, "Modelling of customer oriented applications in product lifecycle using RAMI 4.0," *Procedia Manufacturing*, vol. 28, pp. 31–36, 2019.

[17] E. Y. Nakagawa, P. O. Antonino, F. Schnicke, R. Capilla, T. Kuhn e P. Liggesmeyer, "Industry 4.0 reference architectures: State of the art and future trends," *Computers & Industrial Engineering*, vol. 156, p. 107 241, 2021.

[18] A. M. Hosseini, T. Sauter e W. Kastner, "Towards adding safety and security properties to the Industry 4.0 Asset Administration Shell," em *2021 17th IEEE International Conference on Factory Communication Systems (WFCS)*, IEEE, 2021, pp. 41–44.

[19] M. A. Pisching, M. A. Pessoa, F. Junqueira, D. J. dos Santos Filho e P. E. Miyagi, "An architecture based on RAMI 4.0 to discover equipment to process operations required by products," *Computers & Industrial Engineering*, vol. 125, pp. 574–591, 2018.

[20] B. R. Martin Hankel, "Industrie 4.0: The Reference Architectural Model Industrie 4.0 (RAMI 4.0)," v1.0, Apr. 2015, ZVEI: Die Eletroindustrie.

[21] IEC, "Enterprise-control system integration - Part 1: Models and terminology - IEC62264-1," Mai, 2013.

[22] IEC, "Batch control - Part 1: Models and terminology - IEC61512-1," Ago, 1997.

[23] A. Bastos, M. L. S. C. De Andrade, R. T. Yoshino e M. M. D. Santos, "Industry 4.0 readiness assessment method based on RAMI 4.0 standards," *IEEE Access*, vol. 9, pp. 119 778–119 799, 2021.

[24] IEC, "Industrial-process measurement, control and automation - Life-cycle-management for systems and components - IEC62890," Jul, 2020.

[25] A. Corradi, L. Foschini, C. Giannelli et al., "Smart appliances and RAMI 4.0: Management and servitization of ice cream machines," *IEEE Transactions on Industrial Informatics*, vol. 15, n.º 2, pp. 1007–1016, 2018.

[26] N. D. et al., "The Structure of the Administration Shell: TRILATERAL PERSPECTIVES from France, Italy and Germany)," v1.0, Apr. 2015, ZVEI: Die Eletroindustrie.

[27] P. F. S. De Melo e E. P. Godoy, "Controller Interface for Industry 4.0 based on RAMI 4.0 and OPC UA," em *2019 II Workshop on Metrology for Industry 4.0 and IoT (MetroInd4. 0&IoT)*, Ieee, 2019, pp. 229–234.

[28] P. F. Melo, E. P. Godoy, P. Ferrari e E. Sisinni, "Open source control device for industry 4.0 based on RAMI 4.0," *Electronics*, vol. 10, n.º 7, p. 869, 2021.

[29] F. Lelli, "Interoperability of the Time of Industry 4.0 and the Internet of Things," *Future Internet*, vol. 11, n.º 2, p. 36, 2019.

[30] S. Beden, Q. Cao e A. Beckmann, "Semantic asset administration shells in industry 4.0: A survey," em *2021 4th IEEE International Conference on Industrial Cyber-Physical Systems (ICPS)*, IEEE, 2021, pp. 31–38.

[31] J. Fuchs, J. Schmidt, J. Franke, K. Rehman, M. Sauer e S. Karnouskos, "I4. 0-compliant integration of assets utilizing the Asset Administration Shell," em *2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, IEEE, 2019, pp. 1243–1247.

[32] "Industrial-Process Measurement, Control and Automation—Digital Factory Framework —Part 1: General Principles)," v1.0, Dec. 2016, IEC Standard TS 62832-1.

[33] A. Festo, "The AutomationML Component Description in the context of the Asset Administration Shell,"

[34] T. Benešl, V. Kaczmarczyk, T. Sỳkora et al., "Asset Administration Shell-manufacturing processes energy optimization," *IFAC-PapersOnLine*, vol. 55, n.º 4, pp. 334–339, 2022.

[35] X. Ye e S. H. Hong, "Toward industry 4.0 components: Insights into and implementation of asset administration shells," *IEEE Industrial Electronics Magazine*, vol. 13, n.º 1, pp. 13–25, 2019.

[36] M. Jacoby, F. Volz, C. Weißenbacher e J. Müller, "FA 3 ST Service–An Open Source Implementation of the Reactive Asset Administration Shell," em *2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA)*, IEEE, 2022, pp. 1–8.

[37] I. M. Treuk, A. O. Júnior, R. P. Lopes, G. Mota, J. J. Mira e P. Leitao, "Digitalization of Industrial Inspection Assets through the Asset Administration Shell," em *2024 IEEE 7th International Conference on Industrial Cyber-Physical Systems (ICPS)*, IEEE, 2024, pp. 1–6.

[38] M. Platenius-Mohr e S. Grüner, "An analysis of use cases for the asset administration shell in the context of edge computing," em *2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA)*, IEEE, 2022, pp. 1–4.

[39] Z. Fan, D. Shi, O. Meyer, J. Seidelmann e H. Wang, "Asset Adminstration Shell-based Flexible Manufacturing System," em *2023 IEEE 6th International Conference on Industrial Cyber-Physical Systems (ICPS)*, IEEE, 2023, pp. 1–6.

[40] S. R. Bader e M. Maleshkova, "The semantic asset administration shell," em *Semantic Systems. The Power of AI and Knowledge Graphs: 15th Interl Conf., SEMANTiCS 2019*, Springer, 2019, pp. 159–174.

[41] C. Bouter, M. Pourjafarian, L. Simar e R. Wilterdink, "Towards a comprehensive methodology for modelling submodels in the industry 4.0 asset administration shell," em *2021 IEEE 23rd Conference on Business Informatics (CBI)*, IEEE, vol. 2, 2021, pp. 10–19.

[42] IEC, "Specification of the Asset Administration Shell - Part 3a: Data Specification - IEC 61360, Industrial Digital Twin Association," Mar, 2023.

[43] "Industrial automation systems and integration - Exchange of characteristic data - Part 10: Characteristic data exchange format," Technical Specification ISO/TS 29002-10:2009(E), 2009.

[44] X. Ye, W. S. Song, S. H. Hong, Y. C. Kim e N. H. Yoo, "Toward data interoperability of enterprise and control applications via the industry 4.0 asset administration shell," *IEEE Access*, vol. 10, pp. 35 795–35 803, 2022.

[45] A. Deuter e S. Imort, "PLM/ALM integration with the asset administration shell," *Procedia Manufacturing*, vol. 52, pp. 234–240, 2020.

[46] L. Rauh, M. Reichardt e H. D. Schotten, "AI asset management: a case study with the asset administration shell (AAS)," em *2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA)*, IEEE, 2022, pp. 1–8.

[47] W. Motsch, A. Sidorenko, A. David, P. Rübel, A. Wagner e M. Ruskowski, "Electrical energy consumption interface in modular skill-based production systems with the asset administration shell," *Procedia Manufacturing*, vol. 55, pp. 535–542, 2021.

[48] G. Di Orio, P. Maló e J. Barata, "NOVAAS: A Reference Implementation of Industrie4. 0 Asset Administration Shell with best-of-breed practices from IT engineering," em *IECON 2019-45th Annual Conference of the IEEE Industrial Electronics Society*, IEEE, vol. 1, 2019, pp. 5505–5512.

[49] J. Polge, J. Robert e Y. Le Traon, "Assessing the impact of attacks on opc-ua applications in the industry 4.0 era," em *2019 16th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, IEEE, 2019, pp. 1–6.

[50] F. Yallic, Ö. Albayrak e P. Ünal, "Asset Administration Shell Generation and Usage for Digital Twins: A Case Study for Non-destructive Testing.," em *IN4PL*, 2022, pp. 299–306.

[51] S. Lee, H. Kim, D.-k. Hong e H. Ju, "Correlation analysis of MQTT loss and delay according to QoS level," em *The International Conference on Information Networking 2013 (ICOIN)*, IEEE, 2013, pp. 714–717.

[52] P. Drahoš, E. Kučera, O. Haffner e I. Klimo, "Trends in industrial communication and OPC UA," em *2018 cybernetics & informatics (K&I)*, IEEE, 2018, pp. 1–5.

[53] R. Pribiš, L. Beňo e P. Drahoš, "Asset administration shell design methodology using embedded opc unified architecture server," *Electronics*, vol. 10, n.º 20, p. 2520, 2021.

[54] M. Jacoby, M. Baumann, T. Bischoff et al., "Open-source implementations of the reactive asset administration shell: a survey," *Sensors*, vol. 23, n.º 11, p. 5229, 2023.

[55] J. Arm, T. Benesl, P. Marcon et al., "Automated design and integration of asset administration shells in components of industry 4.0," *Sensors*, vol. 21, n.º 6, p. 2004, 2021.

[56] L. Li e W. Chou, "Design and describe REST API without violating REST: A Petri net based approach," em *2011 IEEE International Conference on Web Services*, IEEE, 2011, pp. 508–515.

[57] OpenZDM Consortium, *OpenZDM: Open Zero Defects Manufacturing Platform*, 2025, 2025. URL: `https://www.openzdm.eu`.

[58] IDTA, "Specification of the Asset Administration Shell - Part 1: Metamodel – IDTA Number: 01001-3-0, Industrial Digital Twin Association," 2023.

[59] IEC, "Asset Administration Shell for industrial applications - Part 1: Asset Administration Shell structure - IEC63278-1," Dez, 2023.

[60] Sebastián Ramírez, *FastAPI: A Modern, Fast (High-performance) Web Framework for Building APIs with Python*, 2024, 2024. URL: `https://fastapi.tiangolo.com/`.

# Apêndice A

# Publicações

I. M. Treuk, A. O. Júnior, R. P. Lopes, G. Mota, J. Joaquim Mira and P. Leitao, "Digitalization of Industrial Inspection Assets through the Asset Administration Shell,"2024 IEEE 7th International Conference on Industrial Cyber-Physical Systems (ICPS), St. Louis, MO, USA, 2024, pp. 1-6, doi: 10.1109/ICPS59941.2024.10640027. Keywords: Productivity; Protocols; Digital representation; Inspection; Real-time systems; Servers; Interoperability; Asset Administration Shell; Industry 4.0; RAMI4.0; Digitalization; Cyber-physical System.