UNIVERSIDADE DE SÃO PAULO
ESCOLA POLITÉCNICA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA MECÂNICA

VITOR FURLAN DE OLIVEIRA

ARQUITETURA PARA INTEGRAÇÃO E ANÁLISE DE DADOS BASEADA NO PADRÃO INDÚSTRIA 4.0

São Paulo
2023

VITOR FURLAN DE OLIVEIRA

ARQUITETURA PARA INTEGRAÇÃO E ANÁLISE DE DADOS BASEADA NO PADRÃO INDÚSTRIA 4.0

Versão Corrigida

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo, como parte dos requisitos para obtenção do título de Mestre em Ciências.

Programa: Engenharia Mecânica

Área de concentração: Engenharia de Controle e Automação Mecânica

São Paulo

2023

Este exemplar foi revisado e corrigido em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, 07 de maio de 2023

Assinatura do autor: _____

Assinatura do orientador: _____

Catalogação-na-publicação

Oliveira, Vitor Furlan de
Arquitetura para Integração e Análise de Dados Baseada no Padrão Indústria 4.0 / V. F. Oliveira -- versão corr. -- São Paulo, 2023.
192 p.

Dissertação (Mestrado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos.

1.Indústria 4.0 I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos II.t.

Agradecimentos

A Deus e aos meus pais, por terem proporcionado as oportunidades que me permitiram ingressar e concluir este curso.

A todos e todas que me apoiaram, motivaram, ajudaram, ensinaram, se fizeram presente, foram pacientes e compreensivos durante esta trajetória. Estes relacionamentos foram fundamentais, sobretudo quando a caminhada se tornou exaustiva.

Aos professores e professoras, pelo conhecimento transmitido. Em particular, ao professores Dr. Fabrício Junqueira e Dr. Paulo Eigi Miyagi, pela orientação e por serem fontes de inspiração.

Aos colaboradores da Universidade de São Paulo, instituição na qual eu sempre sonhei em estudar. Aos colaboradores do Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos e do programa de pós-graduação em Engenharia Mecânica, em especial às sras. Marisa Amado e Regianne Amaral. Aos colegas do Laboratório de Sistemas de Automação, em especial ao Dr. Marcosiris Pessoa pelo auxílio durante a pesquisa.

Ao Me. Thadeu Carneiro da Silva e professor Dr. Julio Carlos Teixeira, pela disponibilização dos dados utilizados e pelas contribuições a este trabalho.

Às agências de fomento Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) e Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) por toda a infraestrutura fornecida. Em especial a última, pelo suporte financeiro prestado durante este estudo.

A todos e todas que, de alguma outra maneira, contribuíram para a realização deste trabalho.

Em especial, à Tereza de Lourdes Furlan.

Resumo

OLIVEIRA, V. F. **Arquitetura para Integração e Análise de Dados Baseada no Padrão Indústria 4.0**. 2023. Dissertação (Mestrado em Engenharia Mecânica – Escola Politécnica), Universidade de São Paulo, São Paulo, 2023.

A Indústria 4.0 caracteriza o fenômeno de transformação tecnológica que vem sendo observado desde a última década e que visa a realização da Fábrica Inteligente. É marcada por serviços e processos inteligentes, que se adaptam rapidamente às demandas do mercado e que estabelecem efetiva interconexão entre todas as entidades envolvidas nesses processos. Diretrizes foram propostas para a integração das tecnologias digitais que servem como base para a Indústria 4.0, a fim de promover uma compreensão comum acerca deste fenômeno, contemplar seus principais aspectos e garantir a interoperabilidade entre sistemas. Os dados se mostram um recurso fundamental para a Indústria 4.0 que, por consequência, tem como um pilar tecnológico as tecnologias e métodos compreendidos pelo termo *Big Data & Analytics*. Por esta razão, traçar estratégias para a gestão e análise de dados no contexto em questão é essencial para a implementação da Fábrica Inteligente. O presente trabalho apresenta uma proposta de arquitetura para integração e análise de dados, levando em conta as diretrizes propostas para a Indústria 4.0. Entre estas diretrizes, destaca-se neste trabalho o Modelo de Arquitetura de Referência da Indústria 4.0 (RAMI 4.0), uma proposta de modelo de arquitetura de sistemas que busca contemplar as principais funcionalidades da Indústria 4.0; e o *Asset Administration Shell*, um formato padronizado de representação de artefatos no mundo virtual. O trabalho traz como contribuições: a caracterização dos dados na Indústria 4.0; a identificação de modelos de dados lógicos adequados para diferentes cenários dentro deste contexto; possíveis aplicações industriais de aprendizado de máquina como forma de se extrair valor a partir dos dados; e, como contribuição principal, uma proposta de arquitetura para integração e análise de dados baseada em sistemas de *Data Warehouse*. Por fim, apresenta-se o mapeamento da proposta de arquitetura em um caso real a fim de exemplificar as contribuições deste trabalho.

Palavras-chave: Indústria 4.0, *Asset Administration Shell*, RAMI 4.0, *Data Warehouse*, Banco de Dados, Aprendizado de Máquina

Abstract

OLIVEIRA, V. F. **Architecture for Data Integration and Analysis Based on the Industry 4.0 Standard**. 2023. Dissertação (Mestrado em Engenharia Mecânica – Escola Politécnica), Universidade de São Paulo, São Paulo, 2023.

Industry 4.0 characterizes the phenomenon of technological transformation that has been observed over the last decade and that aims to achieve the Smart Factory. It is denoted by intelligent services and productive processes that quickly adapt to market demands and establish an effective interconnection between all entities involved. Guidelines were proposed for integrating digital technologies that constitute the basis for Industry 4.0 to promote a common understanding of this phenomenon, encompass its main aspects, and ensure interoperability between systems. Data is a fundamental resource for Industry 4.0 so technologies and methods covered by the term Big Data & Analytics emerge as a technological pillar for this phenomenon. For this reason, devising strategies for data management and analysis in the context of Industry 4.0 is fundamental for implementing the Smart Factory. Among these guidelines, the following stand out in this work: The Reference Architecture Model of Industry 4.0, a system architecture model proposal that seeks to cover the main features of Industry 4.0; and the *Asset Administration Shell*, a standardized format for representing artifacts in the virtual world. This work presents a literature review on topics relevant to data management and analysis in the Industry 4.0 context, as well as a description of this phenomenon including the guidelines and standards proposed for it. The work provides as contributions: the characterization of data in Industry 4.0; identification of logical data models suitable for different scenarios within this context; possible industrial applications of machine learning as a way to extract value from data; and an architecture proposal for data integration and analysis based on data warehouse systems as its main contribution. Finally, the mapping of the architectural proposal in a real case is presented in order to exemplify the contributions of this work.

Keywords: Industry 4.0, *Asset Administration Shell*, RAMI 4.0, *Data Warehouse*, Database, Machine Learning

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

AAS	*Asset Administration Shell*
ACID	Atomicidade, Consistência, Isolamento e Durabilidade
AM	Aprendizado de máquina
ANS	Aprendizado não supervisionado
API	*Application programming interface*
AR	Aprendizado por reforço
AS	Aprendizado supervisionado
BASE	Basicamente indisponível, estado flexível e eventualmente consistente
BD	Banco de dados
CAP	Consistência, Disponibilidade e Tolerância à partição
CDD	Dicionário de dados comuns
CECS	Complexo Eólico Corredor do Senandes
CGE	Central Geradora Eólica
DAS	Área de preparação de dados
DM	*Data mart*
DO	Objeto de dados
DT	Árvore de decisão
DW	*Data warehouse*
ETL	Extração, transformação e carregamento
FN	Falso negativo
FP	Falso positivo
GNB	Naive Bayes gaussiano
HOLAP	OLAP híbridos
HTTP	*Hypertext Transfer Protocol*
I4.0	Indústria 4.0
I4.0C	Componente da Indústria 4.0
IA	Inteligência artificial
IIRA	Arquitetura de Referência da Internet Industrial
IRI	Identificador de Recurso Internacionalizado
IVRA	Arquitetura de Referência da Cadeia de Valor Industrial
KNN	K-ésimos vizinhos mais próximos
LN	Nó lógico
MCD	Módulo Consumidor de Dados
MDM	Módulo Data Mart
MDW	Módulo Data Warehouse
MFDO	Módulo Fonte de Dados Operacionais
MGA	Módulo Gerenciador da Arquitetura
MOLAP	OLAP Multidimensionais
MPD	Módulo de Preparação de Dados
OLAP	Processamento analítico *online*
OLTP	Processamento de transações *online*
OPC-UA	*Open Platform Communications - Unified Architecture*
RAMI 4.0	Modelo de Arquitetura de Referência da Indústria 4.0

REST	*Representational State Transfer*
RF	Floresta aleatória
RL	Regressão logística
ROLAP	OLAP relacionais
SGBD	Sistema gerenciador de banco de dados
SI	Sistemas de informação
SOA	Arquitetura orientada a serviços
TCG	Torre Anemométrica Capão Grande
TN	Verdadeiro negativo
TP	Verdadeiro positivo
UML	Linguagem de Modelagem Unificada

Sumário

1 Introdução

Desde a última década, observa-se que os avanços nas tecnologias de informação e comunicação têm promovido uma contínua transformação tecnológica nos sistemas automatizados de manufatura (KAGERMANN *et al*., 2014; NAKAYAMA; SPÍNOLA; SILVA, 2020). Estas tecnologias passaram a integrar os sistemas de manufatura e possibilitaram a representação deles no mundo virtual, resultando nos gêmeos digitais, que correspondem a modelos fidedignos à realidade. A convergência entre o físico e o virtual resultou nos chamados sistemas ciberfísicos, que integram as capacidades de ambos os mundos para a criação de processos e produtos inteligentes (TAO *et al*., 2019). Estas transformações têm gerado mudanças nos modelos de negócios das empresas, bem como nos seus sistemas organizacionais e logísticos, modelos de consumo e formas de trabalho (KAGERMANN *et al*., 2014). Por estas razões, essas transformações passaram a ser reconhecidas por autores como Schwab (2017) como elementos característicos de uma Quarta Revolução Industrial.

Uma vez que a Quarta Revolução Industrial pôde ser identificada como tal desde seus primeiros avanços, certas iniciativas assumiram a responsabilidade de propor diretrizes para a transformação tecnológica associada a este processo. Destaca-se, neste recorte tecnológico da Quarta Revolução Industrial, a Indústria 4.0 (I4.0). Apresentada pela primeira vez durante a feira de Hannover em 2011, a I4.0 tem como objetivo a realização da chamada Fábrica Inteligente (KAGERMANN *et al*., 2014).

Schwab (2017) ressalta a importância da integração das tecnologias na Quarta Revolução Industrial. Assim, uma das tarefas associadas à I4.0 consiste em propor, por meio de padrões, diretrizes para a integração das tecnologias digitais que servem como base para este processo. Para esta finalidade, foi proposto o Modelo de Arquitetura de Referência da Indústria 4.0 (RAMI 4.0, do inglês *Reference Architecture Model Industrie* 4.0) (ADOLPHS; EPPLE, 2015). O RAMI 4.0 busca representar os aspectos principais da I4.0 e indicar diretrizes acerca do "que fazer" para se alcançar a Fábrica Inteligente, enquanto o "como fazer" fica à cargo das organizações que buscam atingir tal objetivo. Por esta razão, questões relacionadas à implementação do RAMI 4.0 encontram- se em aberto, de modo que surgem lacunas a serem exploradas pela comunidade acadêmica e industrial em geral.

Um dos aspectos relevantes do RAMI 4.0 diz respeito aos dados que são gerados, modificados e utilizados por ativos na I4.0. Os dados se caracterizam como recurso fundamental para as transformações associadas à Indústria 4.0, devido às suas características como baixo

custo; aparente inesgotabilidade; além de criar possibilidades para redução de custos, criação de valor e implementação de melhorias nas organizações (KLINGENBERG; BORGES; ANTUNES, 2021). Por estas características, torna-se particularmente importante explorar as lacunas do RAMI 4.0 relacionadas à gestão e análise de dados.

1.1 Objetivos

O objetivo deste trabalho consiste em propor uma arquitetura para integração e análise de dados baseada nas diretrizes propostas para a Indústria 4.0. Esta proposta de arquitetura busca contribuir para a implementação da Fábrica Inteligente tendo os dados como recurso fundamental. Para alcançar o objetivo proposto, foram estabelecidas as seguintes metas:

- A identificação de características dos dados no contexto da I4.0 e as tecnologias de bancos de dados que se adequam a essas características;
- O estudo de tecnologias de integração de dados para fins analíticos;
- O estudo de interfaces e operações padronizadas para disponibilização de dados na I4.0;
- A interpretação de arquitetura de *Data Warehouse* com base nas diretrizes do RAMI 4.0 e do *Asset Administration Shell*;
- O levantamento de possibilidades de aplicações de *aprendizado de máquina* no setor industrial e as classes de técnicas que melhor se adequem a estas aplicações;
- A demonstração da aplicação dos conhecimentos obtidos e desenvolvidos em um exemplo prático.

1.2 Estrutura do Trabalho

O conteúdo deste trabalho está dividido em seis capítulos, além de referências e apêndices. No capítulo 2, apresenta-se o conceito e as características da Indústria 4.0, bem como tópicos relacionados a este fenômeno. O terceiro capítulo contém a fundamentação referente ao presente trabalho. O quarto capítulo, “Resultados e Discussões” discorre a respeito da importância de *Big Data & Analytics* para a I4.0, sobretudo para caracterização da natureza dos dados envolvidos neste processo e a adequação de diferentes modelos lógicos de dados na Indústria 4.0, levando em conta as características dos dados neste contexto. Apresenta-se um mapeamento de classes de técnicas de *aprendizado de máquina* no RAMI 4.0, considerando as

componentes estáticas e dinâmicas de elementos da I4.0, além de seu ciclo de vida e sua cadeia de valor. Por fim, apresenta-se uma proposta de arquitetura baseada em sistemas de *Data Warehouse* para integração e análise de dados, adotando as diretrizes da Indústria 4.0, em particular o RAMI 4.0 como forma de organizar suas funcionalidades e o *Asset Administration Shell* como formato de representação dos dados no mundo digital.

O capítulo 5, “Exemplo de Aplicação”, demonstra a aplicação das contribuições deste trabalho em um caso real. “Conclusões e Trabalhos Futuros” formam o sexto capítulo, que apresenta considerações relacionadas aos resultados obtidos e discussões sobre o trabalho desenvolvido.

2 Indústria 4.0

Até o final do século XX, três grandes transformações tecnológicas que ocorreram principalmente nos processos produtivos e modelos de negócios foram chamadas de revoluções industriais. Hobsbawn (2009) afirma que se pode ter uma ideia da importância e da profundidade da primeira revolução industrial ao tentar imaginar o mundo moderno sem as "fábricas" ou até mesmo "indústrias". Desta maneira, pode-se ter noção do poder transformador dos meios de produção e da importância das Revoluções Industriais.

As três Revoluções Industriais que ocorreram entre os finais dos séculos XVIII e XX são comumente identificadas e referenciadas pelas tecnologias que as caracterizaram. Isto se deve ao fato de que estes processos foram dirigidos pelo crescimento da disponibilidade de determinadas tecnologias e pela interação entre elas. É comum associar a Primeira Revolução Industrial ao maquinário a vapor; a Segunda ao uso da eletricidade e à produção em massa; e a Terceira aos avanços no campo da eletrônica (DATHEIN, 2003; DEUTSCHLAND.DE, 2014). No entanto, além dos pilares tecnológicos de uma Revolução Industrial, sabe-se da importância das condições sociais e até mesmo tecnológicas que as precederam, assim como os desdobramentos destas Revoluções nas estruturas sociais, organizações políticas, na distribuição de poder e recursos, etc. (DATHEIN, 2003). Este trabalho se concentra em paradigmas tecnológicos da Quarta Revolução Industrial.

Dado o entendimento consensual que os sistemas de automação da manufatura estão sendo submetidos a uma contínua transformação de paradigmas tecnológicos desde a última década, diversos autores associam estas transformações ao que identificam como a Quarta Revolução Industrial (NAKAYAMA; SPÍNOLA; SILVA, 2020; SCHWAB, 2017). Tendo em vista a escala mundial destas transformações, várias iniciativas ao redor do mundo buscam estabelecer diretrizes para este processo de transformação tecnológica. Essas iniciativas são referidas neste trabalho como recortes tecnológicos da Quarta Revolução Industrial. A necessidade de haver um guia (ou múltiplos guias equivalentes) para o processo de transformação tecnológica associado à Quarta Revolução Industrial está associada ao fato de que, diferentemente das três primeiras Revoluções Industriais, esta foi identificada como Revolução Industrial já em seus primeiros estágios. Em outras palavras, trata-se de um processo em que o objeto de pesquisa (a Quarta Revolução Industrial) e o pesquisador são contemporâneos. Assim, essas iniciativas se tornam responsáveis por delinear o avanço da

transformação tecnológica da manufatura, propor uma compreensão comum acerca do fenômeno, estabelecer padronizações e etc.

Entre os diferentes recortes tecnológicos mencionados anteriormente, alguns são destacados neste trabalho e maior foco é dado à chamada Indústria 4.0, termo este que é frequentemente utilizado como sinônimo de Quarta Revolução Industrial. Na Alemanha, foi criada a *Plattform Industrie 4.0*[1], um consórcio entre diversas organizações, incluindo indústrias, universidades e o governo alemão, que se propõe a "moldar a transformação digital na manufatura" segundo os preceitos da Indústria 4.0. Nos EUA, o *Industrial Internet Consortium*[2], criado em 2014, possui o intuito de "reunir as organizações e tecnologias necessárias para acelerar o crescimento da Internet Industrial, identificando, montando, testando e promovendo as melhores práticas". Uma iniciativa equivalente foi lançada na China em 2015: o plano *Made in China 2025*, que busca "reestruturar a manufatura da China e os setores de serviços relacionados para alcançar a independência de tecnologia estrangeira; tornar-se a superpotência tecnológica global por meio de avanços revolucionários em setores-chave; elevar a qualidade do produto chinês e ajudar as marcas chinesas a se tornarem globais". A Indústria 4.0, a Internet Industrial, o *Made in China 2025*, entre outras iniciativas são, portanto, propostas de diferentes países para a transformação tecnológica na manufatura associada à Quarta Revolução Industrial. Este trabalho será fundamentado na proposta alemã, isto é, a Indústria 4.0.

O significado do termo "Indústria 4.0" é objeto de análise de diversos pesquisadores. Ao invés de apresentar uma definição, opta-se por descrever o fenômeno em termos das suas caraterísticas: a I4.0 é caracterizada basicamente por processos produtivos que se adaptam rapidamente às demandas do mercado e com efetiva interconexão entre todas as entidades envolvidas nesses processos. O resultado deste fenômeno é a concepção da chamada Fábrica Inteligente, que visa a manufatura baseada em serviços e processos inteligentes (KAGERMANN *et al*., 2014).

As várias iniciativas ao redor do mundo que buscam estabelecer diretrizes para o atual processo de transformação digital o fazem, dentre outras formas, pela elaboração de modelos de arquitetura de referência. Estes modelos buscam reunir e padronizar as funcionalidades necessárias para a realização da Fábrica Inteligente. Os dados, enquanto recurso fundamental

[1] www.plattform-i40.de

[2] www.iiconsortium.org

deste tipo de fábrica, precisam ser armazenados, processados, analisados e disponibilizados a fim de garantir a inteligência dos sistemas de manufatura. Tais tarefas estão contempladas no escopo de funcionalidades nestes modelos que, devido ao fato de estarem em seus estágios iniciais de desenvolvimento, contêm lacunas associadas às soluções tecnológicas e formas de organizá-las para que tais funcionalidades sejam atendidas. Estas lacunas são exploradas neste trabalho.

2.1 Modelo de Arquitetura de Referência da Indústria 4.0

O fato de a Quarta Revolução Industrial ter sido identificada em seus primeiros estágios permitiu que instituições estabelecessem diretrizes para seus avanços. Schwab (2017) ressalta a importância da integração das tecnologias associadas a este fenômeno de maneira apropriada. Assim, uma das tarefas das entidades mencionadas anteriormente consiste em propor, por meio de normas e padrões, diretrizes para a integração das tecnologias digitais que servem como base para o fenômeno em questão. Para esta finalidade, foi proposta uma série de modelos de arquitetura de referência para este processo de transformação tecnológica. Entre as propostas, identifica-se o Modelo de Arquitetura de Referência da Indústria 4.0 (RAMI 4.0) (ADOLPHS; EPPLE, 2015; PISCHING et al., 2018), a Arquitetura de Referência da Internet Industrial (IIRA, do inglês *Industrial Internet Reference Architecture*) (LIN *et al*., 2017), a Arquitetura de Referência da Cadeia de Valor Industrial (IVRA, *Industrial Value-Chain Reference Architecture*) (IVI - INDUSTRIAL VALUE CHAIN INITIATIVE, 2016), entre outros.

Os modelos de arquitetura de referência propostos buscam contemplar os aspectos mais importantes do processo de transformação digital no contexto em que se inserem. É importante destacar que, dado o nível global das transformações tecnológicas, é necessário que exista a possibilidade de interoperabilidade e equivalência entre os modelos. Por conta disso, as organizações que propõem tais arquiteturas também dedicam esforços para encontrar e descrever a correspondência entre os modelos. Tendo em vista que este trabalho tem seu foco voltado para a Indústria 4.0, explora-se em mais detalhes o RAMI 4.0. A seguir, descreve-se algumas das características a serem englobadas por este e pelos outros modelos de arquitetura de referência.

O uso do termo “transformação tecnológica” para descrever a Indústria 4.0 implica que, assim como os processos produtivos, os modelos de negócio e estruturas organizacionais devem ser submetidos a mudanças de cunho tecnológico. Ardolino *et al*. (2018) propõem uma

associação direta entre as tecnologias e as suas implicações nos modelos de negócio e estruturas organizacionais das empresas. Trabalhos como este permitem expressar aspectos organizacionais, de integração interna (integração de ativos dentro de uma mesma organização) e externa (entre os participantes do processo de transformação tecnológica). Assim, aspectos de integração horizontal (entre empresas) e vertical (entre diferentes níveis organizacionais de uma empresa), além da engenharia fim-a-fim (ligação – a partir da digitalização – entre todas as etapas do ciclo de vida de uma entidade, da aquisição de matéria-prima até seu descarte) devem poder ser contemplados por uma arquitetura de referência.

Para a realização da Fábrica Inteligente, propõe-se a integração padronizada de tecnologias que servem como pilares da Indústria 4.0 (SCHWAB, 2017). É sabido que a associação destas tecnologias em um cenário industrial permite elevar consideravelmente o nível de automação de processos e serviços (RUSSMANN *et al*., 2015). Embora a literatura não apresente um consenso com relação a quais tecnologias formam estes pilares da I4.0, um modelo de arquitetura de referência para a I4.0 deve ser genérico suficientemente para que as suas diretrizes possam ser adotadas por instituições que se diferenciam em vários aspectos, como o segmento em que se inserem, nível de automação, porte, etc. Ainda, o modelo deve ser suficientemente abrangente para que até mesmo sistemas legados possam ser inseridos no contexto da I4.0.

Com base no que tem sido apresentado e discutido, pode-se identificar alguns aspectos chaves a serem levados em conta para a condução da Indústria 4.0. Neste sentido, em 2015, foi proposta a integração ou combinação destes aspectos chaves de acordo com o RAMI 4.0 (ADOLPHS; EPPLE, 2015), padronizado pela norma DIN SPEC 91345 (DIN, 2016). Em essência, o RAMI 4.0 busca representar os aspectos principais da I4.0 e assegurar que se desenvolva uma compreensão comum deste processo por parte de todos os seus participantes (DORST *et al*., 2016). Além de contemplar os aspectos chaves da Indústria 4.0, uma característica importante do RAMI 4.0 é a sua abrangência. Assim, o RAMI 4.0 busca mapear os aspectos mais relevantes da Indústria 4.0 sem restringir as suas possibilidades de aplicação a determinadas tecnologias ou arquiteturas de implementação específicas.

2.1.1 Estrutura

O RAMI 4.0 é comumente ilustrado por meio de uma representação tridimensional, conforme apresenta a Figura 1. Nesta representação, os três eixos contemplam os aspectos

fundamentais da Indústria 4.0 mencionados anteriormente. Para descrever estes eixos, utiliza-se o termo Componente da Indústria 4.0, abreviado por I4.0C, que será definido na Seção 2.2. Neste momento, limita-se a descrevê-lo como um artefato físico ou não, que tem valor para a organização, e que é mapeado no mundo digital. Todo componente da Indústria 4.0 pode ser mapeado no espaço tridimensional compreendido entre os eixos do RAMI 4.0. Cada uma destas dimensões é descrita nos parágrafos seguintes.

O eixo horizontal à direita da Figura 1 designa níveis de hierarquia dos I4.0Cs nas organizações. Este eixo de "Níveis Hierárquicos" representa, de maneira padronizada, a classificação funcional dos I4.0Cs, sendo um aspecto importante para integração vertical das organizações (ADOLPHS; EPPLE, 2015). Ao longo do eixo de "Níveis Hierárquicos" são representadas as classificações dos I4.0Cs quanto às suas funções (sobretudo de controle e gerenciamento). A representação horizontal destas hierarquias serve para ressaltar a ideia de que, na Indústria 4.0, pode haver comunicação direta entre diferentes níveis hierárquicos, de modo que o fluxo de dados pode ocorrer de maneira otimizada. Isto também implica que este eixo do RAMI 4.0 não busca estabelecer padrões quanto a aspectos de implementação da hierarquia propriamente dita, mas padroniza a atribuição funcional dos I4.0Cs.

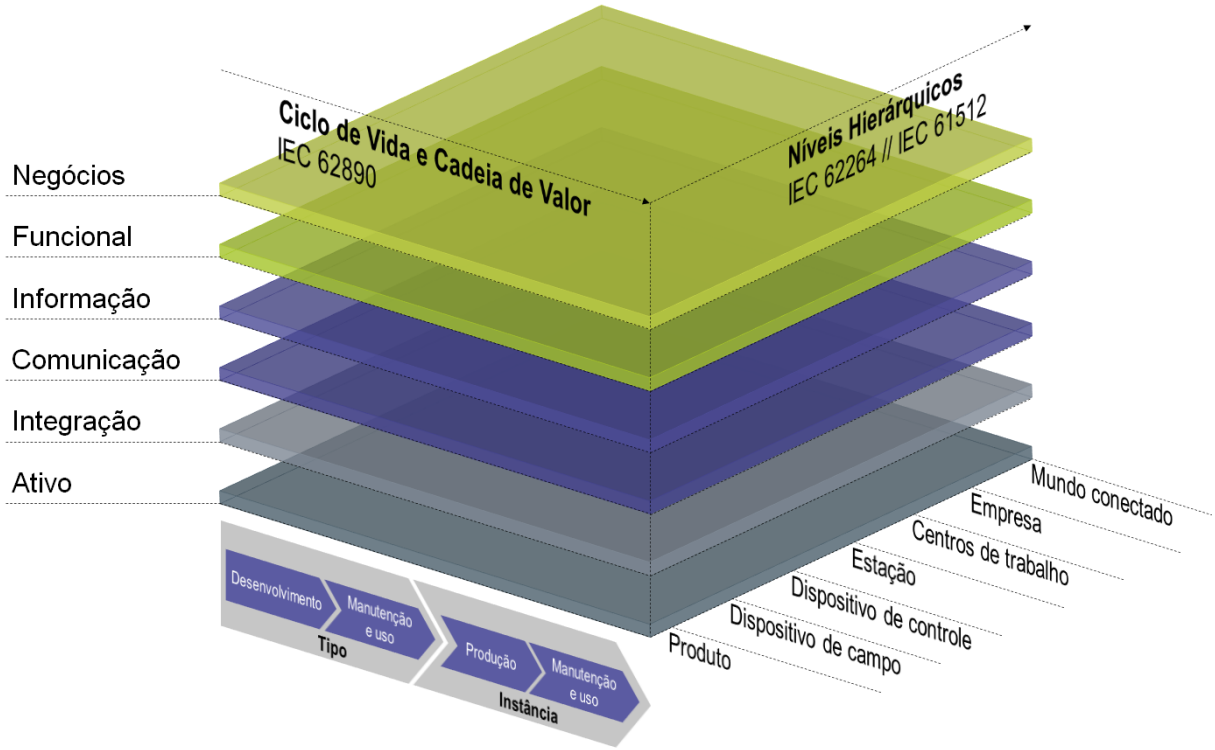


Figura 1 - Representação tridimensional do RAMI 4.0.

Fonte: adaptado de Adolphs et al. (2016).

O eixo “Níveis Hierárquicos” do RAMI 4.0 é inspirado na norma ISA 95, cujas camadas hierárquicas são comumente representadas por meio do que se convencionou chamar “pirâmide da automação” (DORST *et al*., 2016; DOUGLAS *et al*., 2018). No caso do RAMI 4.0, além dos níveis contemplados pela norma ISA 95, foram incluídos três outros: “Produto”, “Dispositivo de Campo” e “Mundo Conectado”. O primeiro permite representar considerações relevantes acerca do produto de uma Fábrica Inteligente (ADOLPHS; EPPLE, 2015), sendo útil, por exemplo, para fins de controle de qualidade. O segundo nível adicionado – “Dispositivo de Campo” – destaca a importância não apenas do dispositivo de (realização de) controle, mas também do objeto de controle, do qual se pode monitorar o estado, por exemplo, de um produto ou do ambiente. Por fim, o nível “Mundo Conectado”, que serve para representar um I4.0C cuja atribuição funcional é realizar a gestão entre empresas (PISCHING, 2018).

O eixo horizontal à esquerda representa o “Ciclo de Vida e Cadeia de Valor” de I4.0Cs por meio de uma representação derivada da norma IEC 62890 (BADER *et al*., 2019). Por meio deste eixo, descreve-se o estado de um I4.0C em determinado ponto no tempo de sua vida útil (DIN, 2016), isto é, é possível mapear e gerenciar um I4.0C desde a sua concepção até o seu descomissionamento e descarte. Este mapeamento é feito não apenas no tempo, mas também no espaço, levando em conta as organizações pelas quais o I4.0C passa ao longo de sua vida útil. Desde modo, o eixo em questão permite contemplar aspectos de integração horizontal e de engenharia fim-a-fim destes I4.0Cs.

O mapeamento e gerenciamento de um I4.0C ao longo de seu ciclo de vida e da sua cadeia de valor podem trazer grandes benefícios às organizações. Possibilita que as informações geradas ao longo deste período possam ser aproveitadas para realizar melhorias no próprio I4.0C, bem como nas versões futuras deste I4.0C. Estas possibilidades serão exploradas em maiores detalhes na Seção 4.2.

O eixo “Ciclo de Vida e Cadeia de Valor” é dividido em duas fases: “Tipo” e “Instância”. Cada uma destas fases é dividida em duas etapas: “Desenvolvimento” e “Manutenção e Uso” para a fase “Tipo”, “Produção” e “Manutenção e Uso” para a fase “Instância”. Estas fases e etapas são descritas a seguir e ilustradas na Figura 2. Para ilustrar estas etapas e as conversões entre “Tipo” e “Instância”, utiliza-se a Figura 3, em que um I4.0C “peça” é utilizado como item constituinte de um I4.0C “máquina”, que por sua vez é empregada em um I4.0C “fábrica”.

Figura 2 - Eixo “Ciclo de Vida e Cadeia de Valor” do RAMI 4.0.

Fonte: autoria própria.

Um I4.0C, seja ele uma peça, uma máquina ou uma fábrica, passa a existir na fase de “Tipo” desde que é idealizado, na etapa de “Desenvolvimento”. Nesta etapa, as funcionalidades e demais características do I4.0C são definidas e especificadas por meio de documentos e modelos digitais. O “Desenvolvimento” se estende até a produção, teste e validação dos primeiros protótipos do I4.0C, descritos pelas atividades de comissionamento digital e físico na Figura 3. Tendo em vista que, ao fim da etapa “Desenvolvimento”, o I4.0C já foi validado, passa-se para o planejamento de sua produção, bem como para a idealização de outros I4.0Cs. Observa-se que um “Tipo” peça é utilizado para a produção da peça em si, ou seja, de uma “Instância”, bem como para o projeto da máquina. Analogamente, o “Tipo” da máquina é utilizado para a produção da máquina e para o projeto da fábrica. Na etapa de “Manutenção e Uso”, informações relacionadas ao I4.0C e aos processos envolvidos são atualizadas com base em dados oriundos da produção do I4.0C, dados de venda, logística, *marketing*, etc. (ADOLPHS; EPPLE, 2015; BADER *et al*., 2019).

Os I4.0Cs que são materializados se encontram na fase de “Instância”, na etapa de “Produção”. É nesta etapa que são coletados os dados reais do I4.0C que podem ser utilizados para revisão de projetos e implementação de melhorias no “Tipo” utilizado como referência para produção em versões futuras da “Instância”. Quando o I4.0C que se encontra na etapa de “Produção” é vendido, isto é, passa a ser utilizado, ele entra na etapa de “Manutenção e Uso” do ponto de vista de quem o fornece. No entanto, para o cliente que o recebe e ainda não o destinou à sua aplicação, este I4.0C se encontra ainda na fase de “Tipo”. A partir do momento que é implementado na aplicação, entra na fase de “Instância”, etapa de “Produção”. Estas relações são representadas pelas atividades de “entrega das peças” e “entrega das máquinas” na Figura 3. Só quando entra em operação, este I4.0C passa a estar, do ponto de vista do cliente, na etapa de “Manutenção e Uso”. Esta etapa se estende durante toda a vida útil desta “Instância” de I4.0C, até o momento em que é descartado.

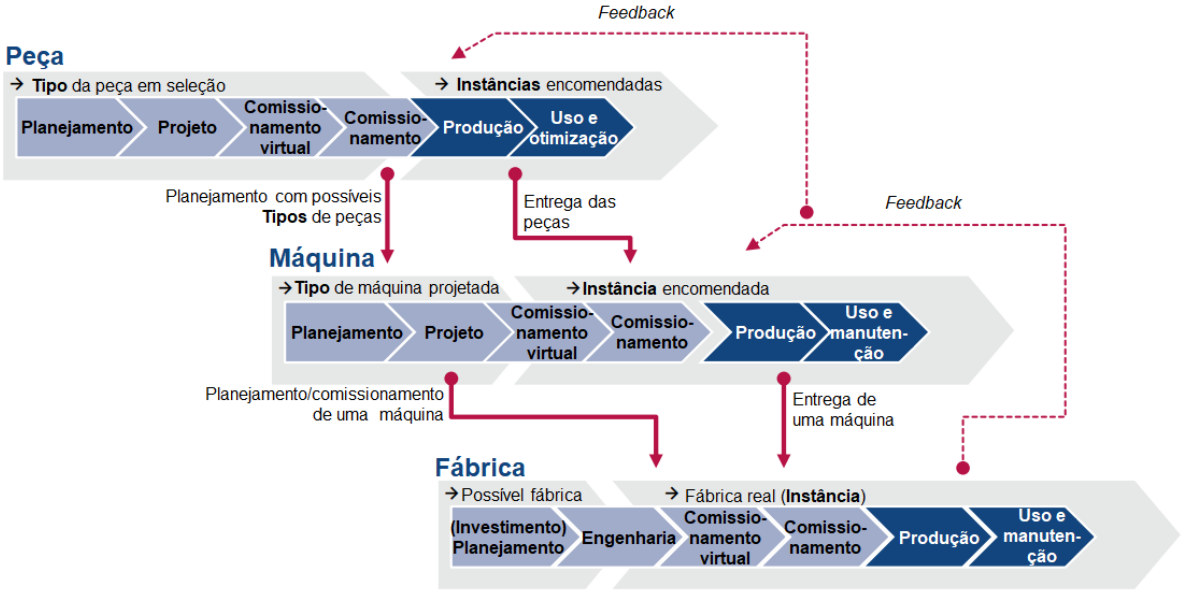


Figura 3 - Conversões entre as fases “Tipo” e “Instância” de um I4.0C.

Fonte: adaptado de Adolphs *et al*. (2016).

Em um primeiro momento, a distinção entre “Tipo” e “Instância” e as suas respectivas etapas se mostra particularmente útil para identificar o estado de um I4.0C ao longo de seu ciclo de vida. No entanto, as considerações feitas nos dois parágrafos anteriores mostram que o eixo em questão permite também representá-lo de maneiras diferentes de acordo com a sua localização na cadeia de valor. Deste modo, um I4.0C pode sofrer conversões entre “Tipo” e “Instância” várias vezes ao longo de seu ciclo de vida e da sua localização na cadeia de valor, como ilustra a Figura 3. A importância desta possibilidade de representação da cadeia de valor pode ser compreendida pela própria definição de I4.0 apresentada por Hermann, Pentek e Otto (2015): “I4.0 é um termo coletivo para tecnologias e conceitos de organização da cadeia de valor”.

O eixo vertical do RAMI 4.0 é chamado de “Eixo das Camadas”. Por meio deste eixo, aspectos estruturais do processamento de informações são contemplados em seis níveis ou camadas. Em cada um, especifica-se as propriedades e funcionalidades de I4.0Cs de acordo com a norma DIN SPEC 91345. Deste modo, é possível descrever como um I4.0C é configurado e funcionará. No RAMI 4.0, estas funcionalidades são disponibilizadas na forma de serviços, de modo que o RAMI 4.0 é então caracterizado como uma arquitetura orientada a serviços, paradigma que é explorado na Seção 3.5. A seguir, tem-se a descrição das camadas baseada em Adolphs, Ulrich e Epple (2015) e DIN - Deutsches Institut Für Normung E.V. (2016):

- Camada de Negócios: nesta camada, as visões comerciais dos I4.0Cs são representadas, de modo que é possível mapear os processos relevantes para os modelos de negócio da organização. Por meio desta camada, ocorre a orquestração dos serviços na *Camada Funcional*. Na camada de negócio, estão contempladas "condições de contorno" gerais da organização; condições monetárias, legais e regulatórias; conexões entre diferentes processos que constituem o modelo de negócios; etc;
- Camada Funcional: nesta camada, são formalmente descritas as funções lógicas necessárias de um I4.0C para o processo de negócio como todo. Além da descrição das funções, é na *Camada Funcional* que se encontram as interfaces para realização da integração horizontal entre participantes da Indústria 4.0. Uma determinada organização acessa e avalia remotamente as funcionalidades do I4.0C de outra organização por meio de interfaces disponibilizadas na *Camada Funcional*;
- Camada de Informação: descreve as informações relevantes de um I4.0C que podem ser acessadas, utilizadas e modificadas, bem como as regras que executam tais ações ao relacionarem um ou mais eventos do mundo digital a um ou mais eventos do mundo real e *vice versa*. Tendo em vista que nesta camada ocorre a gestão e análise dos dados dos I4.0Cs, o foco deste trabalho é voltado para ela;
- Camada de Comunicação: a camada padroniza a interação entre I4.0Cs (de uma mesma empresa), dando acesso às informações de um I4.0C contidas na Camada de Informação. Neste nível, cria-se um canal de comunicação entre os I4.0Cs e os dados são disponibilizados por meio de serviços e em um formato padrão (XML, JSON, AutomationML) (BADER *et al*., 2019);
- Camada de integração: representa a transição entre os mundos físico[3] e digital. A camada é responsável por fornecer dados acerca da parte real dos I4.0C contemplados na camada inferior, de tal modo a possibilitar e facilitar o processamento destes dados pela(s) camada(s) superior(es) (BOUSDEKIS *et al*., 2019), possibilitando inclusive a inserção de sistemas legados no contexto da Indústria 4.0. Não apenas os eventos gerados pelos I4.0C no mundo físico são interpretados e levados ao mundo digital por

[3] Ativos como programas de computador e documentos digitais não existem no mundo físico, isto é, se encontram no mundo digital. Neste caso, a Camada de Integração é vazia. O mesmo ocorre para ativos que foram desenvolvidos de maneira compatível com as diretrizes da I4.0, isto é, capazes de interagir com as camadas superiores sem a necessidade de uma ponte.

meio da Camada de Integração, mas é também por meio dela que o oposto ocorre, como quando se implementa o controle de processos auxiliados por computador, como exemplifica a norma DIN SPEC 91345; e

- Camada de Ativos: nela se encontra a parte real dos I4.0C e que necessariamente estão representados no mundo digital por meio das camadas superiores. Conforme foi dito anteriormente, a parte real de um I4.0C pode ser física ou não, isto é, pode existir somente no mundo digital.

2.1.2 Camada de Informação

Entre todos os aspectos do RAMI 4.0, a Camada de Informação é particularmente importante para este trabalho. De acordo com a norma DIN SPEC 91345, "A camada de 'informação' descreve os dados que são usados, gerados ou modificados pela funcionalidade técnica do componente". Com base nas atribuições designadas para Camada de Informação apresentadas pela norma, divide-se as funções desta camada em: **armazenamento**; **processamento e análise**; **integração**; e **disponibilização** de dados referentes aos componentes da I4.0. Estas quatro classes de funcionalidades são descritas na Tabela 1. As contribuições resultantes são reunidas na proposta de arquitetura para integração e análise de dados.

Tabela 1 - Classes de funcionalidades da Camada de Informação

Classe	**Funcionalidade**
Armazenamento	Descrição formal de modelos e regras;
	Persistência dos dados representados pelos modelos;
	Garantia da integridade dos dados;
Processamento e análise	Ambiente de execução para (pré-) processamento de eventos;
	Execução de regras;
	Pré-processamento de contexto;
	Aquisição de novos dados de melhor qualidade (dados, informações, conhecimento);
	Recebimento de eventos e transformação dos mesmos em uma forma adequada para os dados disponíveis para a *Camada Funcional*;
Integração	Integração consistente de diferentes dados;
Disponibilização	Fornecimento de dados estruturados por meio de interfaces de serviço.

Fonte: baseado em DIN - Deutsches Institut Für Normung E.V. (2016).

É possível identificar diferentes níveis de qualidade dos dados. Enquanto o termo "informação" é utilizado para dar nome à camada e da definição apresentada na Subseção anterior, ao observar as funções desta camada em mais detalhes, observa-se que o termo "dado"

se faz presente em detrimento do termo “informação”. Além disso, a norma DIN SPEC 91345 ainda introduz o termo “conhecimento”, como um nível de qualidade superior ao de “dado”. Ao acrescentar o termo “sabedoria”, forma-se a chamada hierarquia DIKW (*data, information, knowledge* e *wisdom* – dado, informação, conhecimento e sabedoria). Esta hierarquia serve para identificar e descrever as relações e os processos que levam a transformação de uma entidade em outra (ROWLEY, 2007). Liew (2013) inclui ainda uma quinta entidade – a inteligência – transformando a pirâmide em DIKIW (a segunda letra I corresponde à *intelligence*, do inglês inteligência) e sintetiza as relações entre estes cinco conceitos conforme ilustrado na Figura 4. Para o autor, a inteligência desempenha papel fundamental na conversão de conhecimento em sabedoria.

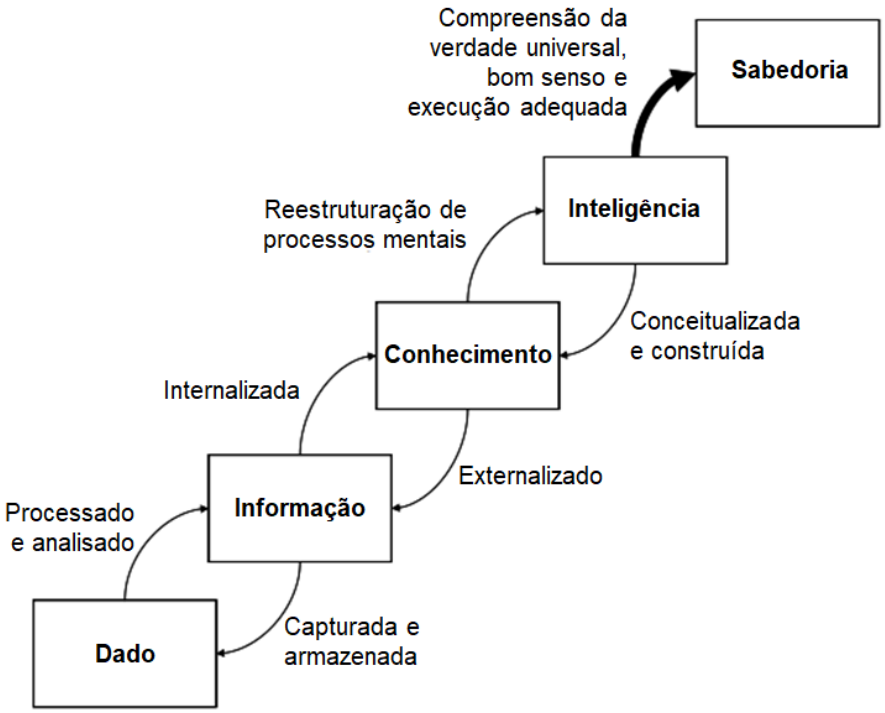


Figura 4 - Hierarquia DIKIW.

Fonte: adaptado de Liew (2013).

Em sistemas de informação (SI), os dados são também a forma com que entidades de níveis hierárquicos superiores são armazenadas. Segundo Liew (2013), o processo de conversão de “dados” em “informação” é justamente a captura e o armazenamento da primeira. De acordo com Alavi, Maryam e Leidner (2001), “conhecimento”, que corresponde à internalização de “informações”, é armazenado em SI das organizações em diferentes formatos, tais como documentos, e-mails, arquivos, etc. (FRAKES, 1992). “Inteligência” está ligada à capacidade

de processamento (LIEW, 2013) e, em SI, estas capacidades são implementadas por meio de algoritmos. Por fim, embora a literatura apresente certa limitação em relação à "sabedoria", sobretudo na área de sistemas de informação, geralmente o termo é também associado a capacidades e habilidades (LIEW, 2013). Em resumo, entidades de alto nível hierárquico da Figura 4 precisam ser levadas ao nível de dados para que possam ser armazenadas e processadas. Deste modo, os dados se tornam o recurso chave para as organizações (BATRA, 2014). "Inteligência" e "sabedoria" se tratam não de um recurso, mas de capacidades. Embora a norma DIN SPEC 91345 descreva as atribuições da Camada de Informação utilizando o termo "dado", ela também explicita a função da Camada de Informação de realizar transformações entre os níveis. Assim, entende-se que as funções da Camada de Informação se estendem aos cinco níveis da hierarquia DIKIW.

2.2 Asset Administration Shell

Inicia-se esta Seção pelo conceito de "ativo" (*asset*) que, em uma tradução não literal do inglês, pode ser entendido como ativo técnico ou ativo físico. A IEC apresenta o conceito de ativo como "objeto físico ou lógico de propriedade ou sob as funções de custódia de uma organização, tendo um valor percebido ou real para a organização" [4]. Com base nesta definição, também adotada pela Plataforma Indústria 4.0 (BADER *et al*., 2019), é possível reconhecer que um ativo pode ser algo físico (uma máquina, um equipamento, materiais, produtos) ou não (documentos eletrônicos, programas de computador). Alguns exemplos menos intuitivos de ativos são: local; tempo; estado de um ativo; seres humanos; e relacionamentos entre ativos (BEDENBENDER *et al*., 2017a). Em síntese, o ativo é tudo aquilo que possui valor e importância para uma organização.

Sabe-se que a Indústria 4.0 caracteriza um processo de transformação digital. Para que este tipo de processo ocorra, os ativos precisam ser digitalizados, isto é, seus dados devem ser levados ao mundo digital (GASTALDI *et al*., 2018; INIGO *et al*., 2020). Para a I4.0, é importante que este mapeamento seja feito de maneira padronizada, de modo a garantir a interoperabilidade entre os sistemas da I4.0 e seus elementos constituintes (INIGO *et al*., 2020). Para isso, foi

[4] International Electrotechnical Commission Glossary: https://std.iec.ch/terms/terms.nsf/9bc7f244dab1a789c12570590045fac8/98df4c86beea5ef1c1257c8500394f0b?OpenDocument

criado o *Asset Administration Shell* (AAS), que corresponde a uma representação digital padronizada do ativo da Indústria 4.0, contendo todas as suas informações e funcionalidades técnicas. O AAS fornece uma descrição mínima, única e suficiente do ativo em suas diferentes perspectivas relevantes para cada caso de uso (BEDENBENDER et al., 2017b). Ao padronizar o formato de representação e as interfaces de comunicação de ativos no mundo digital, o AAS viabiliza o intercâmbio de informações entre os participantes da I4.0, o que por sua vez garante a interoperabilidade entre componentes (BADER et al., 2019). Em resumo, o AAS corresponde à representação digital e padronizada de um ativo no contexto da I4.0.

A combinação entre ativo e AAS dá origem ao Componente da Indústria 4.0 (*I4.0C*). O I4.0C consiste na combinação entre o mundo físico e real, sendo composto pelo ativo e o seu respectivo AAS. A combinação entre estes dois elementos, com o segundo "envolvendo" o primeiro, como ilustra a Figura 5, possibilita que serviços e funcionalidades sejam oferecidos na rede de componentes da Indústria 4.0. Estas funcionalidades e serviços são possibilitadas pela identificação exclusiva e capacidade de comunicação de um I4.0C. Aqui, vale destacar que um único I4.0C pode estar associado a múltiplos ativos a depender da granularidade considerada. Deste modo, tal estrutura pode ser replicada para diferentes níveis de granularidade (por exemplo, em diferentes níveis da hierarquia). Nas subseções seguintes são abordados detalhes de estrutura, elementos, metamodelo e perspectivas de implementação do AAS.

Figura 5 - Representação do *Asset Administration Shell*

Fonte: autoria própria.

2.2.1 Estrutura e Metamodelo

Em documentações iniciais, a estrutura de um AAS era dividida em dois componentes principais (ADOLPHS *et al*., 2016). O primeiro deles era denominado Cabeçalho (*Header*), em que estão contidas as informações de identificação do ativo e do AAS em si, além de outras informações, dados e funções mais gerais do I4.0C exibidas na forma de metadados. Este conjunto de metadados era denominado "Manifesto" e se fazia presente no segundo componente do AAS, chamado de Corpo (*Body*). Além do manifesto, o Corpo do AAS continha as informações mais detalhadas sobre o ativo associado ao AAS. Para gerenciamento e acesso a serviços do AAS, o AAS também continha, no corpo, um gerenciador de componentes (*componente manager*). Documentos mais atualizados não propõem mais a divisão em Cabeçalho e Corpo, nem incluem um gerenciador de componentes. Assim, a estrutura passa a ser composta apenas pelos elementos que são utilizados para descrever os ativos. Estes elementos são organizados em uma estrutura hierárquica rígida e padronizada, mas que pode ser preenchida com dados de formatos variados, que se complementam para uma descrição completa do ativo. A Figura 6 ilustra a estrutura do *Asset Administration Shell*. Os elementos desta estrutura que caíram em desuso são representados em cinza, enquanto aqueles que continuam sendo utilizados são apresentados em azul.

Os elementos presentes no AAS utilizados para descrever um ativo se dividem em classes. Cada uma delas possui seus próprios atributos, isto é, elementos particulares que a descrevem. Além disso, existem elementos no Corpo que podem ser entendidos como subclasses, os quais possuem os mesmos atributos de uma (super)classe específica da qual são derivados, mas também contêm atributos que os diferenciam de outras desta mesma superclasse. A seguir, serão descritas as classes de elementos que constituem o AAS.

Uma primeira forma de se dividir as classes de elementos de um AAS é designando-as como Identificáveis e Referenciáveis. Elementos Identificáveis são aqueles que possuem um identificador global exclusivo. Os Referenciáveis, por sua vez, possuem um identificador que não é exclusivo globalmente, sendo único apenas dentro do contexto (definido por um Identificável) em que se encontra. As classes de elementos que derivam dos Identificáveis e Referenciáveis são chamadas de subclasses e podem ser compostas por outras subclasses. Nesta hierarquia, existe uma relação de herança dos atributos: as subclasses herdam atributos das superclasses. A Tabela 2 apresenta os Elementos Identificáveis e Referenciáveis.

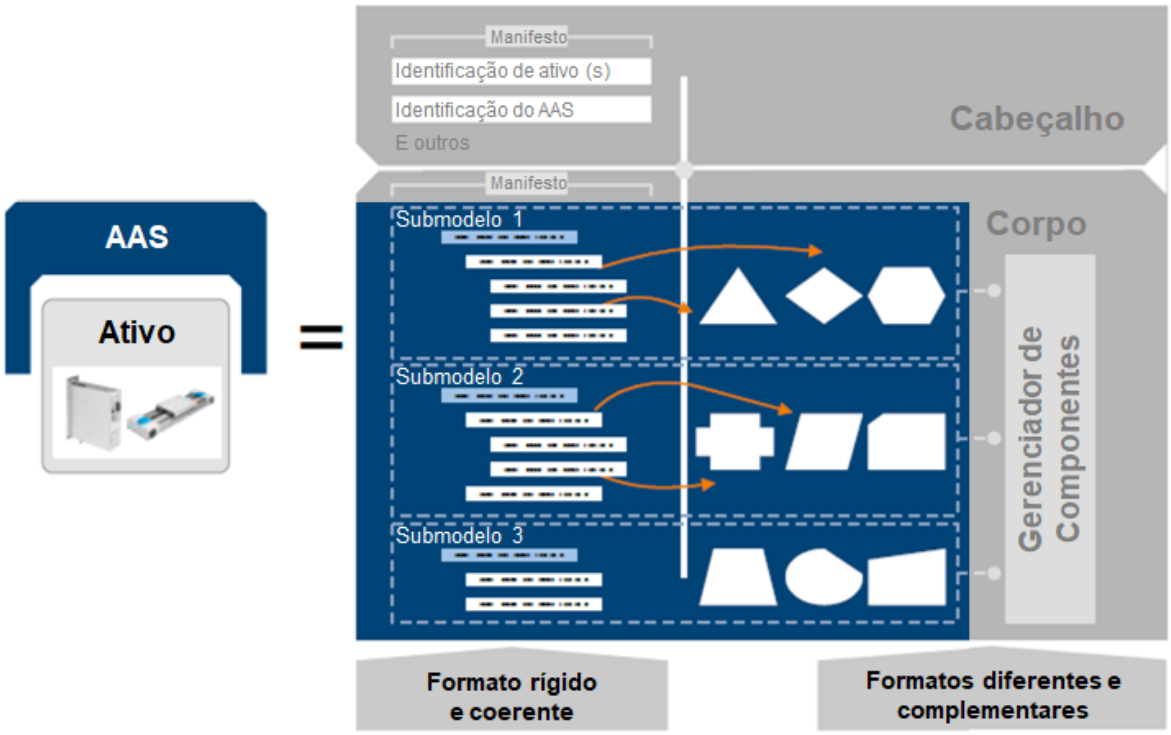


Figura 6 - Estrutura do AAS

Fonte: Adaptado de Adolphs *et al*. (2016).

Os Elementos Identificáveis podem ter identificadores adicionais específicos de domínio (proprietários). Dentre os Identificáveis, as subclasses "*Asset Administration Shell*" e "Ativo" já foram descritas anteriormente. A "Descrição Conceitual" define a descrição semântica padronizada de certos elementos (BADER et al., 2019). Por fim, tem-se o "Submodelo", já presente na Figura 6. Ela permite que um ativo seja representado em suas diferentes perspectivas. Cada "Submodelo" é composto por elementos que são utilizados para descrever o ativo em uma perspectiva específica, pertencente a um domínio em particular (BADER et al., 2019).

A classe de "Elementos Referenciáveis" possui muito mais subclasses que a de "Elementos Identificáveis". Portanto apenas algumas delas que possuem maior relevância para o trabalho serão abordadas aqui em mais detalhes. A descrição de todas as classes pode ser encontrada em Bader *et al*. (2019). Dentre os Elementos Referenciáveis, destaca-se neste trabalho a subclasse "Elemento de Submodelo". Ela é composta por elementos úteis para a descrição e diferenciação de ativos em uma perspectiva especificada pelo Submodelo. "Elementos de Submodelo" pode ser entendida como uma superclasse, em que algumas das principais subclasses são "Coleção de Elementos de Submodelo" – coleção esta que pode ser

composta por todas as outras classes de mesmo nível hierárquico – e "Elemento de Dado". Os Elementos de Dados, por sua vez, formam outra superclasse da qual deriva uma das mais importantes subclasses de elementos do AAS, a "Propriedade", descrita em maiores detalhes a seguir.

Tabela 2 - Elementos do AAS.

Classe	Elemento
Identificáveis	Ativo (*Asset*)
	Asset Administration Shell
	Descrição Conceitual *(ConceptDescription)*
	Submodelo *(Submodel)*
Referenciáveis	Regra de Permissão de Acesso *(AcessPermissionRule)*
	Elemento de Relacionamento Anotado *(AnnotatedRelationshipElement)*
	Evento Básico *(BasicEvent)*
	BLOB (*Binary Large Object*, ou coleção de dados binários)
	Competência *(Capability)*
	Dicionário de Conceito *(ConceptDictionary)*
	Elemento de Dado *(DataElement)*
	Entidade *(Entity)*
	Evento *(Event)*
	Arquivo *(File)*
	Propriedade Multi-idioma *(MultiLanguageProperty)*
	Operação *(Operation)*
	Propriedade *(Property)*
	Intervalo de Valores *(Range)*
	Elemento de Referência *(ReferenceElement)*
	Elemento de Relacionamento *(RelationshipElement)*
	Elemento de Submodelo *(SubmodelElement)*
	Coleção de Elementos de Submodelo *(SubmodelElementCollection)*
	Visão *(View)*

Fonte: baseado em Bader *et al.* (2019).

A classe de "Propriedades" contém elementos que permitem representar as características de um ativo, dada uma perspectiva definida pelo submodelo em que se encontram. Estes elementos são padronizados pela norma IEC 61360 e podem ser encontrados nos repositórios IEC *Common Data Dictionary* (CDD, dicionário de dados comuns) ou eCl@ss (ADOLPHS *et al.*, 2016; BEDENBENDER *et al.*, 2019; YE; HONG, 2019). No repositório IEC CDD, uma propriedade possui, além de seu valor propriamente dito, alguns dados adicionais como o

código, versão e revisão, identificador, definição, etc. A versão digital e gratuita do IEC CDD[5] traz exemplos de propriedades de alguns domínios específicos, como da engenharia mecânica, elétrica, telecomunicações, etc.

Existem elementos do AAS que não são enquadradas como Identificáveis ou Referenciáveis. Alguns deles são particularmente importantes para este trabalho, como "Possui Tipo" (*HasKind*), que identifica a fase do ciclo de vida em que o ativo se encontra, bem como a sua localização na cadeia de valor. Outra classe importante é a de "Referência", que permite que um elemento do AAS faça referência a outro elemento do próprio AAS em que se encontra ou a um elemento de uma entidade externa. Esta referência é feita por meio de um ou mais elementos da classe "Chave", que especifica um caminho que fornece acesso exclusivo ao elemento referenciado. A lista completa de classes de elementos do AAS, incluindo aquelas que não se enquadram como Identificáveis ou Referenciáveis é apresentada na Tabela 23 do Apêndice A, com uma descrição a seu respeito, as relações de herança entre elas, seus atributos, e a descrição destes atributos.

2.2.2 Perspectivas de Implementação

As padronizações propostas para a Indústria 4.0 precisam ser abrangentes o suficiente para não limitar a possibilidade de realização da Fábrica Inteligente nas mais diferentes empresas, indústrias, entidades e organizações. Neste sentido, ao padronizar o formato digital de representação e intercâmbio de informações, não é imposta estratégia específica para implementação do AAS. Em Bedenbender *et al*. (2017c), são exploradas algumas destas possibilidades, que levam em conta diferentes perspectivas de implementação. As diferentes possibilidades contidas nestas perspectivas proporcionam características que podem consistir em vantagens ou desvantagens para determinadas aplicações. Estas características se referem a questões como poder computacional; disponibilidade; desempenho e latência; segurança e confiabilidade; custo de manutenção, administração e gerenciamento; identificação e recuperação de falhas; entre outras. A seguir, duas perspectivas de implementação do AAS apresentadas em Bedenbender *et al*. (2017c) são descritas, juntamente com as vantagens e desvantagens que fazem parte do escopo deste trabalho.

[5] IEC 61360-4 - Common Data Dictionary (CDD - V2.0014.0017):
https://cdd.iec.ch/cdd/iec61360/iec61360.nsf/TreeFrameset?OpenFrameSet

A primeira questão a ser discutida com relação a implementação do AAS é a "proximidade" entre o mesmo e o ativo ao qual ele está associado. Três possibilidades de implementação são apresentadas, conforme ilustra a Figura 7. Na primeira delas (Figura 7 a), o AAS é embarcado no ativo, que por sua vez contém um ambiente de execução para a sua representação digital. É o caso de uma implementação baseada em borda (*edge*), que pode ser utilizada como exemplo para tal implementação. Em uma segunda possibilidade (Figura 7 b), o AAS pode estar fisicamente separado do ativo, mas residindo na infraestrutura de TI local, sendo conectado ao ativo por meio de uma rede local, por exemplo. Este caso corresponde a uma implementação baseada em névoa (*fog*). Como terceira possibilidade (Figura 7 c), a localização do AAS pode ser ainda mais distante do ativo, em uma implementação baseada em nuvem. Neste caso, AAS e ativo se conectam via rede de internet externa.

A primeira característica a ser analisada para fins de comparação entre as possibilidades de implementação apresentadas diz respeito à capacidade de processamento e armazenamento. Os dispositivos de borda são heterogêneos entre si, por exemplo: um *hardware* como uma placa de prototipagem Arduino, um controlador lógico programável e um *smartphone* possuem diferentes capacidades de armazenamento e processamento.

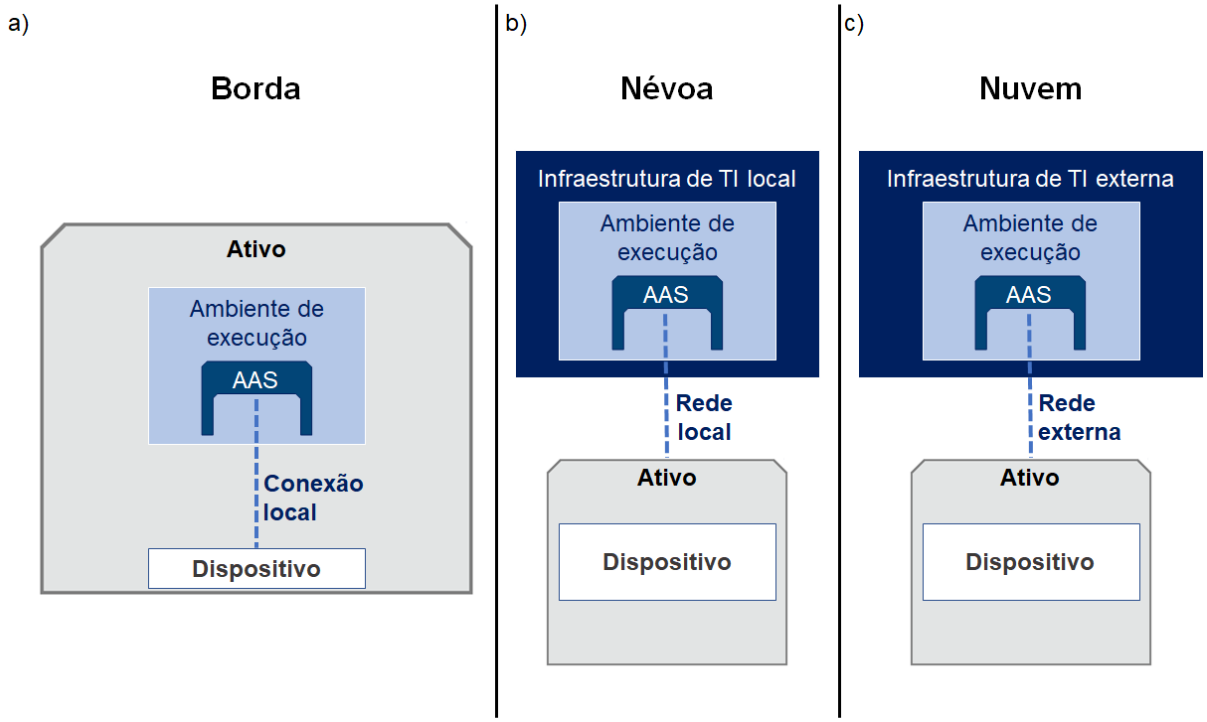


Figura 7 - Proximidade entre AAS e ativo.

Fonte: autoria própria.

A mesma heterogeneidade é observada para as plataformas de névoa e nuvem, havendo possivelmente uma interseção destas capacidades computacionais entre as plataformas. Ademais, a evolução da ciência e da tecnologia contribui ainda mais para esta heterogeneidade. A despeito deste fato, é razoável considerar que, em linhas gerais, dispositivos de borda (Figura 7 a) possuem menor capacidade de processamento e armazenamento e processamento que os dispositivos de névoa (Figura 7 b), que por sua vez são geralmente inferiores nestes mesmos quesitos em comparação aos dispositivos de nuvem (Figura 7 c).

Outra característica que permite diferenciar as possibilidades de implementação apresentadas está ligada ao custo. A implementação em borda (Figura 7 a) permite que seja possível explorar poder de computação geralmente com baixo custo em comparação às outras plataformas. A implementação baseada em névoa (Figura 7 b) exige infraestrutura e administração que podem gerar custos superiores às implementações em nuvem, sobretudo em grandes escalas de dados a serem armazenados (BEDENBENDER *et al*., 2017c).

Pode-se também comparar as plataformas em termos de latência e desempenho. Justamente pela distância física entre ativo (fonte de dados) e AAS, a implementação em borda (Figura 7 a) implica em menor latência, ao passo que a implementação em nuvem corresponderia à implementação com maior latência. A implementação em névoa apresentaria uma condição intermediária. As implementações com maior latência podem ser compensadas com maior capacidade de processamento, possibilitando aumento do desempenho, conforme argumentam Bedenbender *et al*. (2017c).

Pode-se fazer ainda uma consideração acerca das implementações híbridas, que utilizam simultaneamente plataformas de borda, névoa e nuvem. Arquiteturas que utilizam estas plataformas de maneira combinada possibilitam explorar o melhor de cada uma delas. Soluções que combinam borda, névoa e/ou nuvem são reportadas na literatura (ALLI; ALAM, 2019; BARIK *et al*., 2018; OSIA *et al*., 2018).

A segunda perspectiva de implementação considerada diz respeito à escalabilidade do AAS. Esta questão será abordada em mais detalhes na Seção 3.2, mas de maneira simplificada, a escalabilidade está relacionada à possibilidade de distribuir o armazenamento e processamento de dados em múltiplos nós de uma rede. Nesta Subseção, são apresentadas três possibilidades para distribuição do AAS: centralizado, como ilustra a Figura 8 (a), em que todas as informações e serviços são alocados em um único nó; distribuído com acoplamento fraco, conforme representado na Figura 8 (b), em que diferentes nós armazenam informações para o mesmo ativo (mesmo identificador) e podem ser acessados individualmente; e distribuído com

nó agregador, ilustrada na Figura 8 (c), que se diferencia da implementação anterior ao incluir um nó agregador, que reúne informações dos nós nos quais o AAS é distribuído, formando um ponto único de acesso aos dados.

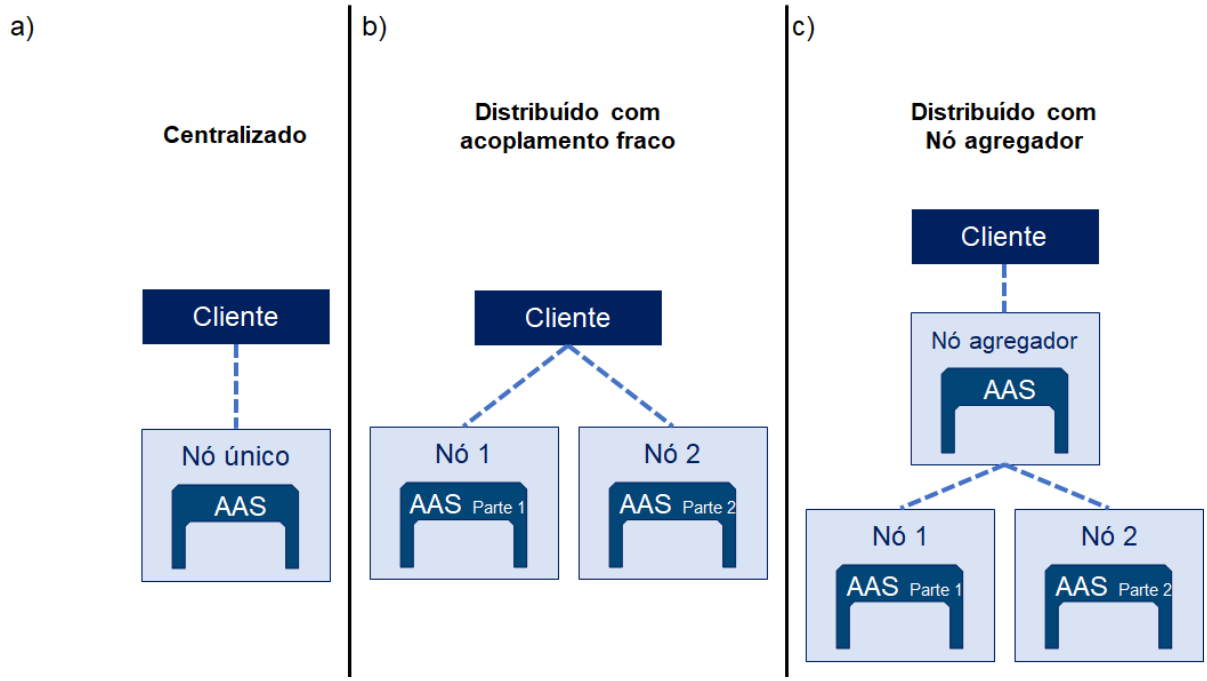


Figura 8 - Distribuição do AAS.

Fonte: Adaptado de Bedenbender *et al*. (2017c).

As características destas possibilidades de distribuição do AAS envolvem conceitos da área de banco de dados que são discutidos em detalhes na Seção 3.2. Aqui, limita-se a discutir questões de estrutura organizacional das instituições. A implementação centralizada não reflete de maneira fidedigna a realidade os sistemas atuais das organizações, que muitas vezes são intrinsecamente distribuídos (como a organização em departamentos, estações de trabalho, etc.). As implementações distribuídas representam melhor estes sistemas (como sistemas de automação distribuídos) e possibilitam que diferentes unidades da organização trabalhem concomitantemente em um mesmo ativo (como um projeto de desenvolvimento, por exemplo).

Entre as duas possibilidades de distribuição apresentadas, a implementação com acoplamento fraco possibilita ganhos de desempenho, pois os nós podem processar requisições específicas a eles, enquanto um nó agregador pode estar sujeito a sobrecargas por precisar processar requisições associadas a todos os nós que contêm fragmentos do AAS (BEDENBENDER *et al*., 2017c). Do ponto de vista de acesso às informações do AAS, as implementações centralizada e distribuída com nó agregador oferecem um único ponto de

acesso, enquanto a implementação com acoplamento fraco exige que se saiba previamente onde se localiza cada fração do AAS para então identificar o ponto de acesso de interesse.

Por fim, vale a pena ressaltar que a escalabilidade também é influenciada pela forma de virtualização do AAS. Na Figura 7, as implementações diferem quanto à distância entre o ambiente de execução do AAS e o ativo. No entanto, nenhuma consideração adicional é feita com relação ao ambiente de execução em específico, que define uma forma de virtualização do AAS. Esta forma de virtualização é uma perspectiva de implementação que implica em vantagens e desvantagens para aplicação. Três possibilidades são apresentadas e ilustradas na Figura 9: (a) a implementação baseada em sistema operacional; (b) hipervisor; e (c) contêiner. No primeiro caso, o sistema operacional é o próprio ambiente de execução do AAS, isto é, o ambiente de execução consiste em um processo de um sistema operacional dedicado ou executado dentro de outro processo. No segundo caso, o ambiente de execução do AAS é uma máquina virtual (VM, do inglês *virtual machine*). Múltiplas máquinas virtuais com seus próprios sistemas operacionais são alocadas em uma máquina *host* (hospedeira), isto é, utilizam seu *hardware e* são gerenciadas por um monitor de máquina virtual, também chamado de hipervisor. Assim como no caso anterior, o AAS ainda funcionaria como um processo do sistema operacional completo, porém neste caso, este processo compartilharia os recursos de *hardware* com outros sistemas operacionais e aplicativos. Por fim, na terceira possibilidade, o ambiente de execução do AAS é formado por contêiner, que por sua vez são executados sobre um sistema operacional. Diferentemente das máquinas virtuais, as aplicações executadas em contêineres são executadas no próprio sistema operacional da máquina *host*, exigindo apenas os mínimos recursos como aplicativos e APIs necessários para execução da aplicação, no caso, o AAS (GEREND, 2019).

Duas questões de interesse deste trabalho estão relacionadas ao ambiente de execução do AAS, isto é, à sua forma de virtualização: isolamento e desempenho. Estas duas características formam um *trade-off*, conforme apontam Xavier *et al*. (2013).

Com relação ao isolamento, no caso da implementação por meio de hipervisor, existe o isolamento entre os sistemas operacionais das máquinas virtuais, que por sua vez implica também em isolamento entre as aplicações. No caso da implementação em contêineres, só há isolamento total entre as aplicações, porém todas compartilham o mesmo sistema operacional. Os contêineres não são capazes de interagir com quaisquer outras aplicações e serviços executados no sistema operacional em que estão alocados, porém ficam susceptíveis a falhas deste. Por fim, na implementação direta em sistema operacional, não há qualquer tipo de

isolamento mencionado anteriormente, de modo que o processo correspondente ao AAS pode afetar e ser afetado por outros serviços em execução, como outro AAS (GEREND, 2019; XAVIER *et al*., 2013). Em Mavridis e Karatza (2019), propõe-se a execução de contêineres em máquinas virtuais, com o propósito de aumentar o isolamento dos primeiros, além do estudo do impacto que esta forma de virtualização traria em termos de desempenho.

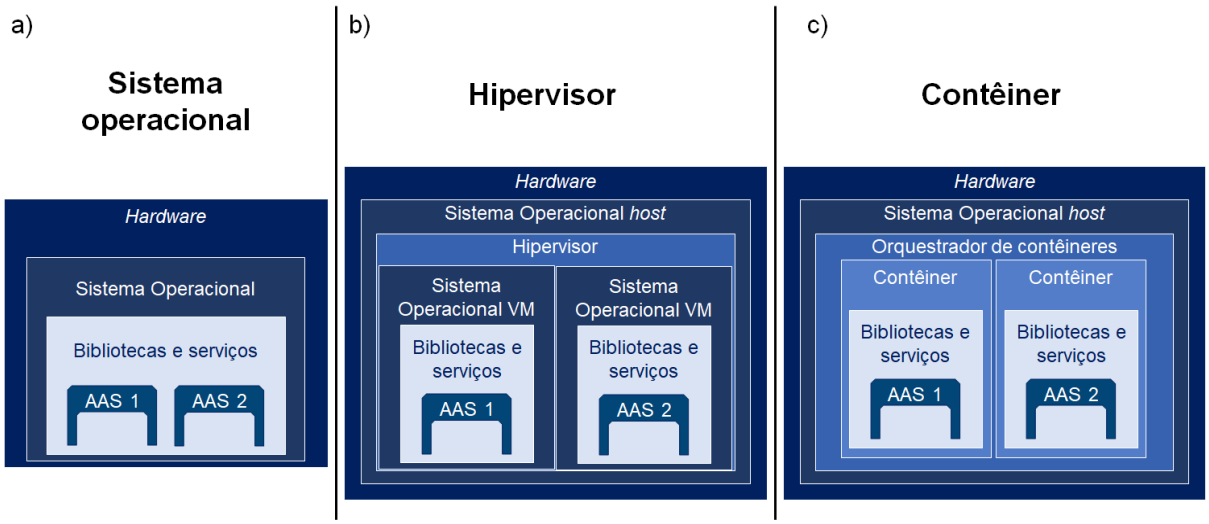


Figura 9 - Formas de virtualização do AAS.

Fonte: autoria própria.

O desempenho das formas de virtualização pode ser relacionado ao desempenho computacional, de memória e de disco. Em Xavier *et al*. (2013), os autores comparam estas três métricas de desempenho para as três formas de virtualização apresentadas aqui. Em linhas gerais, é possível afirmar que a implementação em contêiner supera a virtualização em hipervisor em termos de desempenho computacional, de memória e de disco, inclusive se aproximando da implementação direta em sistema operacional a depender do sistema de virtualização baseada em contêiner escolhido. Resultados análogos foram obtidos por Joy (2015). Em resumo, é possível afirmar que em termos de desempenho, a execução em sistema operacional é superior à virtualização em contêiner, que por sua vez supera a implementação em hipervisor.

2.3 Síntese do Capítulo

A Indústria 4.0 é apresentada neste capítulo como o recorte tecnológico da Quarta Revolução Industrial. A I4.0 é um termo utilizado para descrever o processo de transformação tecnológica caracterizado pela integração de tecnologias digitais que visam desenvolver processos, produtos e serviços inteligentes, isto é, a Fábrica Inteligente. Como guia deste processo de transformação tecnológica, foi proposto o Modelo de Arquitetura de Referência da Indústria 4.0. Este modelo é descrito na Seção 2.1, na qual o foco é voltado para a Camada de Informação, que trata dos dados que são gerados, modificados e utilizados por ativos.

O RAMI 4.0 busca estabelecer diretrizes para realização da Fábrica Inteligente de maneira padronizada. Ele permite representar um ativo sob diferentes perspectivas, tais como o seu ciclo de vida e sua cadeia de valor, seu nível hierárquico na organização, além da interação com ativos internos e externos à organização em que se encontra. Enquanto modelo de arquitetura, permite também identificar funcionalidades associadas a estes ativos, que são realizadas por meio de soluções de tecnologias da informação e comunicação. Contudo, não faz parte do escopo do RAMI 4.0 determinar quais soluções tecnológicas devem ser de fato empregadas para realizar tais funcionalidades, cabendo a cada organização participante da Indústria 4.0 definir tecnologias e arquiteturas específicas para as implementar. Mais do que identificar soluções tecnológicas adequadas, é necessário propor formas de organizá-las a fim de contribuir para a implementação do RAMI 4.0, em especial da Camada de Informação. As contribuições deste trabalho, apresentadas no Capítulo seguinte, buscam contribuir para o preenchimento destas lacunas.

Tendo em vista o foco no gerenciamento de dados no contexto da I4.0, apresentou-se o formato padronizado de representação de ativos no mundo virtual, o *Asset Administration Shell* e sua estrutura, metamodelo e perspectivas para implementação. O AAS é o principal elemento que constitui a Camada de Informação do RAMI 4.0, assim, é adotado neste trabalho como diretriz fundamental para a elaboração da proposta de arquitetura para integração e análise de dados no contexto da I4.0.

3 Fundamentos

Para atender o objetivo do presente trabalho, é necessário responder a perguntas como: qual é a natureza dos dados, isto é, como podem ser caracterizados? Com base nestas características, quais soluções de armazenamento dos dados são mais adequadas? Como os dados armazenados se fazem disponíveis para análise? O que é necessário para analisar os dados? Qual a finalidade que se pode dar para esta análise? Quais são as soluções de análise de dados mais adequadas para esta finalidade? Este capítulo apresenta a fundamentação necessária para responder essas questões.

3.1 Big Data & Analytics

O termo *Big Data & Analytics* possui dois componentes, sendo *big data* o primeiro deles. Mauro, Greco e Grimaldi (2016) definem *big data* como "o ativo de informação caracterizado por um alto volume, velocidade e variedade para exigir tecnologia e métodos[6] analíticos específicos para sua transformação em valor". Embora o termo "ativo" ainda não tenha sido definido, "ativo de informação" se refere à "base de dados". Assim, o termo *big data* é utilizado para descrever bases de dados que, por conta das características mencionadas na definição apresentada, exigem tecnologias e métodos não convencionais para sua manipulação e extração de valor.

O termo *analytics* é o segundo componente de *Big Data & Analytics*. Este termo, referido neste trabalho como *big data analytics* (análise de *big data*), diz respeito à aplicação, por meio de tecnologias específicas para o contexto de *big data*, de métodos de análise destes dados. Russom (2011) oferece uma definição de *big data analytics*: "é onde as técnicas analíticas avançadas operam em *big data*". Lichtblau *et al*. (2015) apresentam uma descrição mais detalhada para o termo: "processo de analisar *big data* de uma empresa e encontrar inter-relacionamentos úteis que apoiem as atividades da empresa". Em síntese, o termo *Big Data & Analytics* compreende o ativo de informação (*big data*), bem como as tecnologias e métodos de análise dos dados (*analytics*) para este tipo de ativo, isto é, específicos para suas características.

[6] Define-se método como o conjunto das atividades sistemáticas e racionais que, com maior segurança e economia, permite alcançar o objetivo (MARCONI; LAKATOS, 2003). Neste caso, o objetivo é a extração de valor dos dados.

Ainda com base na definição de *big data* apresentada, observa-se quatro dimensões que caracterizam este ativo de informação: o volume, velocidade, variedade e valor dos dados. Uma quinta dimensão – veracidade – ainda é considerada por alguns autores para contemplar a confiabilidade dos dados (GIL; SONG, 2016; YIN; KAYNAK, 2015). Estas cinco dimensões, também chamadas de "5V", caracterizam as bases de *big data* e possuem implicações diretas na forma com que os dados são armazenados, manipulados e analisados (NIST BIG DATA PUBLIC WORKING GROUP, 2019), ou seja, exigem soluções tecnológicas e métodos específicos para estas funções. A seguir, tem-se uma descrição destas cinco dimensões de *big data*.

- Volume: está associada à quantidade de dados envolvidos em aplicações de *big data*. Embora não exista consenso a respeito de um valor de referência em termos de volume dos dados para que uma base possa ser caracterizada como *big data*, esta dimensão geralmente é caracterizada por petabyte (10^{15}) ou exabyte (10^{18}). De acordo com Yin e Kaynak (2015), o volume de dados no setor industrial moderno tende a crescer mais de 1000 exabytes por ano;
- Velocidade: esta dimensão pode ser entendida como tendo duas componentes – a velocidade com que os dados são coletados e a velocidade com que são processados (LEE, 2017). Em aplicações mais antigas de *big data*, o processamento era comumente feito em lotes (*batches*) e, deste modo, a velocidade com que os dados são capturados é fundamental para garantir sua confiabilidade. Aplicações mais recentes possibilitam o processamento dos dados em tempo real e em fluxos de dados, de modo que, além de garantir a confiabilidade dos dados, a velocidade de processamento deve ser condizente com a velocidade de captura de dados. (GIL; SONG, 2016; LEE, 2017; YIN; KAYNAK, 2015);
- Variedade: diz respeito aos diferentes formatos possíveis dos dados. Aplicações de *big data* podem envolver dados estruturados, tais como tabelas com estrutura rígida e povoadas com valores com escopo delimitado; semiestruturados como documentos sem um *template* pré-definido, mas com autodescrição; e não estruturados, como conteúdo multimídia (imagem, áudio) (LEE, 2017);
- Veracidade: está associada à confiabilidade dos dados capturados. Rubin (2014) argumenta que a dimensão veracidade possui três componentes: objetividade/subjetividade, que está mais ligada à natureza da fonte dos dados; engano (do inglês *deception*), que se refere a erros propositais no conteúdo dos dados ou

modificações mal intencionadas dos mesmos; e implausibilidade (improbabilidade, irracionalidade) dos dados, que se refere à qualidade dos dados em termos de sua validade, isto é, de seu grau de confiança;

- Valor: está associada ao valor que se pode extrair dos dados por meio de *data analytics.* A extração de valor a partir dos dados consiste na conversão dos dados em entidades de mais alto nível hierárquico (LEE, 2017). Envolve uma série de técnicas de análise de dados, dentre os quais *aprendizado de máquina*; requer uma abordagem multidisciplinar; e, sobretudo recebe o nome "valor" por oferecer perspectivas de melhoria e redução de custos em termos de produtos e processos na indústria (OPPITZ; TOMSU, 2018; YIN; KAYNAK, 2015), de modo que é possível argumentar que há prejuízo em não se extrair valor a partir dos dados.

A extração de valor dos dados em um contexto de *big data* pressupõe a aplicação de tecnologias e métodos analíticos específicos. Esta dimensão ressalta a importância dos dados para o setor industrial, pois traz consigo a possibilidade de implementação de melhorias nas organizações a partir dos dados. Com relação às tecnologias, o foco aqui está em soluções de armazenamento, processamento, integração e disponibilização de dados. Com relação aos métodos, o foco deste trabalho é voltado para técnicas de aprendizado de máquina para fins de *big data analytics*, cujo sucesso ao longo das décadas é associado ao aumento da disponibilidade de dados, tal como vem sendo observado no setor industrial (RAPTIS; PASSARELLA; CONTI, 2019).

3.2 Armazenamento de Dados

Para alcançar o objetivo deste trabalho, é importante refletir quais soluções de armazenamento de dados no mundo digital são mais adequadas. As dimensões de *big data* impactam diretamente esta escolha. Por esta razão, são exploradas diferentes soluções de armazenamento de dados e como elas se adequam às características de *big data*. Este estudo tem início pelo conceito de dado. Uma definição útil é apresentada por Elmasri e Navathe (2005): "dados são fatos que podem ser gravados e que possuem um significado implícito". Estes fatos que são capturados e armazenados podem se apresentar na forma de símbolos (como palavras, áudios, números, diagramas e imagens) e leituras de sinais (provenientes de sensores por exemplo) (LIEW, 2013). O significado implícito está relacionado à representação de

propriedades de objetos, eventos e seu ambiente por meio dos dados (RUSSELL ACKOFF, 1989).

Uma coleção lógica e coerente de dados com um significado intrínseco forma um banco de dados (BD) (ELMASRI; NAVATHE, 2005). Um banco de dados possui um grupo de usuários interessados em seu conteúdo, além de aplicações condizentes com as necessidades destes usuários. Assim, um BD serve para armazenar e garantir a persistência e integridade dos dados que representam ativos, além de permitir que estes dados sejam disponibilizados aos usuários interessados.

Um BD é criado e mantido por meio de um sistema gerenciador de banco de dados (SGBD), um programa de computador que auxilia na manutenção e utilização de conjuntos de dados que formam os bancos de dados (RAMAKRISHNAN; GEHRKE, 2008). Estes programas contam com as seguintes vantagens: possibilitam acesso eficiente e concorrente aos dados; garantem a integridade e segurança dos dados; oferecem proteção contra falhas e acessos não autorizados; suportam múltiplas visões dos dados; e por fim, garantem independência, isto é, o isolamento entre os dados e as aplicações por meio da abstração dos dados (ELMASRI; NAVATHE, 2000; RAMAKRISHNAN; GEHRKE, 2008).

A abstração dos dados, que garante a independência entre os dados e as aplicações, ocorre por meio de modelos de dados. Um modelo de dados é um conjunto de conceitos usados para descrever a estrutura de um banco de dados (ELMASRI; NAVATHE, 2005). Esta descrição pode ocorrer em diferentes níveis entre os usuários dos dados e o *hardware* onde os dados estão armazenados, conforme listado a seguir:

- Modelo de dados conceitual: utiliza um conjunto de conceitos para descrever os dados em alto nível, isto é, descreve-os de acordo com a percepção dos usuários. Os modelos entidade-relacionamento e UML (*Unified Modeling Language*, ou Linguagem de Modelagem Unificada) são exemplos de modelos de dados conceituais;
- Modelo de dados físico: utiliza um conjunto de conceitos para descrever os dados em baixo nível, isto é, descrevê-los de acordo com a forma que estarão armazenados no *hardware*. Este conjunto de conceitos especifica o formato de um registro, como os registros devem ser ordenados e quais são os caminhos para acesso aos dados, por exemplo;
- Modelo de dados de implementação: encontra-se entre o modelo de dados conceitual e o modelo de dados físico. Os conceitos que descrevem os dados em nível de implementação ocultam alguns detalhes do armazenamento, permitindo que o usuário

final dos dados possa compreendê-los. Ao mesmo tempo, por não estarem tão distante do baixo nível, estes conceitos podem ser utilizados diretamente para implementação do BD em um sistema de computador.

3.2.1 Modelo de Dados Relacional

O modelo de dados relacional foi, por muitos anos, a escolha padrão para implementação de bancos de dados (SADALAGE; FOWLER, 2013). Ele utiliza o conceito de “relações” para representar os dados. As relações são tabelas de valores, em que cada linha (chamada formalmente de tupla) é utilizada para representar um elemento do tipo “entidade” ou “relacionamento”. Cada uma das colunas destas relações consiste em um “atributo” que é utilizado para caracterizar as entidades e relacionamentos. Os valores de uma mesma coluna, isto é, de um mesmo atributo, fazem parte de um único domínio (com nome, tipo de dado e formato bem definidos) de valores atômicos (cada valor no domínio é indivisível). Em síntese, um banco de dados relacional é composto por um conjunto de relações, compostas por tuplas e atributos, preenchidas por valores atômicos pertencentes a um domínio estabelecido (ELMASRI; NAVATHE, 2005).

A estrutura de um banco de dados relacional é definida por um esquema. A partir deste esquema, pode-se descrever as tabelas, seus atributos e relações entre elas. Este tipo de representação dos dados pode ser obtido a partir de outra representação de mais alto nível, isto é, de um modelo de dados conceitual, como o modelo entidade-relacionamento estendido. A Figura 10 apresenta um exemplo de modelo entidade relacionamento estendido utilizado para descrição de um equipamento industrial denominado aerogerador. Tal representação é baseada na norma IEC 61400-25-2 (IEC - INTERNATIONAL ELECTROTECHNICAL COMMISSION, 2015), que descreve o modelo de informação deste tipo de equipamento. A partir do modelo conceitual, deriva-se o esquema relacional apresentado na Figura 11.

3.2.2 Garantias Transacionais e Teorema CAP

SGBDs relacionais garantem quatro propriedades às transações com o objetivo de manter os dados mediante ao acesso concorrente e a falhas no sistema. Estas propriedades são atomicidade (A), consistência (C), isolamento (I) e durabilidade (D), por isso são

frequentemente referidas como propriedades ACID. Com base em Elmasri e Navathe (2000) e Ramakrishnan e Gehrke (2008), apresenta-se uma descrição de cada uma delas:

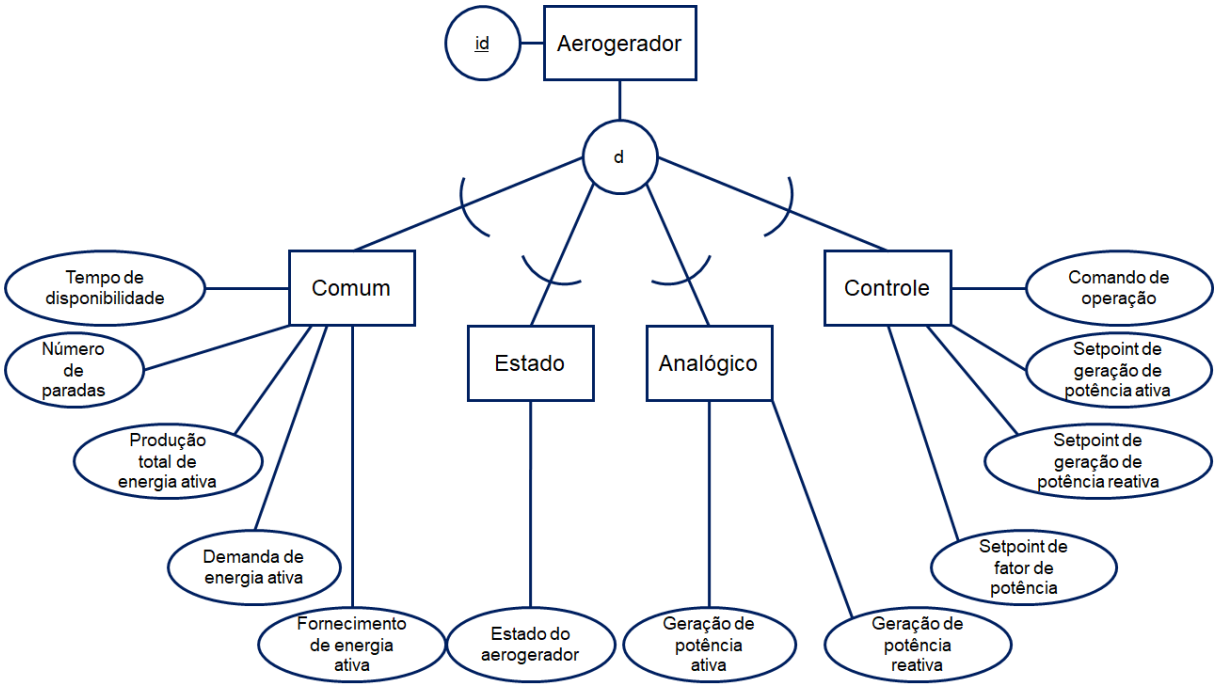


Figura 10 - Exemplo de esquema entidade-relacionamento estendido.

Fonte: autoria própria.

- Atomicidade: a transação[7] deve ser executada por completo ou não ser executada. Se, durante a transação, ocorrer alguma falha que impeça a transação de ser concluída, qualquer alteração que ela tenha executado no BD deve ser desfeita;
- Consistência: se uma transação for executada completamente do início ao fim, sem interferência de outras transações, ela deve levar o banco de dados de um estado consistente para outro. Um estado consistente do banco de dados satisfaz as restrições especificadas no esquema, bem como quaisquer outras restrições no banco de dados que devem ser mantidas;
- Isolamento: a execução de uma transação não deve ter interferência de nenhuma outra transação em execução simultânea;

[7] "Uma transação é uma unidade lógica de trabalho, que é formada por um ou mais comandos de banco de dados" (ELMASRI; NAVATHE, 2005).

- Durabilidade: as alterações aplicadas ao banco de dados por uma transação confirmada devem persistir no banco de dados. Essas mudanças não devem ser perdidas devido a qualquer falha.

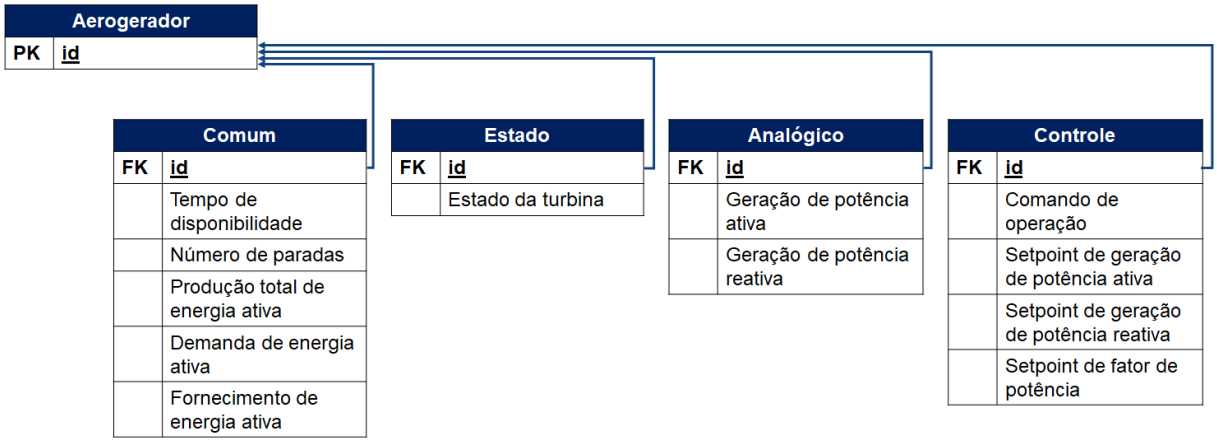


Figura 11 - Exemplo de esquema relacional.

Fonte: autoria própria.

Existem três outras propriedades desejáveis para sistemas de BD no contexto de SGBDs distribuídos: disponibilidade (A, *availability*), tolerância à partição (P, *partition tolerance*) e consistência (C, *consistency*), que neste caso possui certa diferença em relação ao conceito apresentado anteriormente. A análise destas propriedades é importante sobretudo em sistemas de bancos de dados distribuídos. Um banco de dados distribuído é definido como “uma coleção de vários bancos de dados logicamente inter-relacionados, distribuídos por uma rede de computadores” (ÖZSU; VALDURIEZ, 2004). Há três razões importantes apontadas pela literatura para a utilização de bancos de dados distribuídos: 1) o aumento no volume de dados (SADALAGE; FOWLER, 2013), que exige a capacidade de escalar horizontalmente, isto é, distribuir em diversos nós os sistemas ao invés de escalar verticalmente o *hardware* – aumentar mais recursos computacionais a uma mesma máquina, que seria mais caro e limitado; 2) a necessidade de melhor refletir a estrutura organizacional distribuída de empresas (ÖZSU; VALDURIEZ, 2011); e 3) a natureza inerentemente distribuída de uma série de aplicações, incluindo industriais (ÖZSU; VALDURIEZ, 2011). Existem três formas de se implementar um sistema de BD distribuído, descritas a seguir (ELMASRI; NAVATHE, 2000; SADALAGE; FOWLER, 2013). Vale destacar que determinados sistemas implementam versões híbridas, combinando diferentes formas de particionamento (SADALAGE; FOWLER, 2013):

- Servidor único**:** não há distribuição. O banco de dados é executado em uma única máquina que gerencia todas as operações. É o exemplo de uma implementação do AAS centralizada;
- Particionamento**:** diferentes partes dos dados em diferentes máquinas. Modelos de agregados são ideais para o particionamento, pois formam uma unidade natural de particionamento, fazendo com que certos usuários acessem, na maior parte do tempo, um mesmo servidor e não precisem reunir informações de diferentes servidores, o que aumenta o desempenho em comparação à implementação em servidor único. Além disso, o particionamento traz como vantagens o paralelismo no acesso aos dados, e a tolerância às falhas, visto que uma partição eventualmente indisponível não impede que outra continue servindo dados;
- Replicação**:** os dados são replicados no esquema primário-secundário, em que os nós primários processam as atualizações nos dados e replicam esses dados para os nós secundários ou par-a-par, em que todos os nós podem processar atualizações e as propagar.

As três propriedades mencionadas anteriormente – consistência, disponibilidade e tolerância à partição – são correlacionadas por meio do teorema CAP (BREWER, 2000). De acordo com este teorema, estas três propriedades, cujas descrições são apresentadas a seguir, não podem coexistir simultaneamente em um sistema de banco de dados:

- Consistência (C): garantia de que todos os nós tenham a mesma cópia de um dado replicado visível para as aplicações. É um pouco diferente do conceito de consistência das propriedades ACID. Naquele caso, consistência significa não violar restrições do BD. No entanto, pode-se considerar que "ter uma mesma cópia de um dado replicado em todos os nós que este dado é replicado" é uma restrição, então os conceitos passam a se assemelhar;
- Disponibilidade (A, *availability*): cada operação de escrita ou leitura será processada com sucesso (sistema disponível) ou falhará (sistema indisponível). Um nó "desligado" não é dito indisponível;
- Tolerância à partição (P, *partition tolerance*): um sistema tolerante à partição continua operante no caso de ocorrência de uma falha na rede que conecta os nós resultando em uma ou mais partições do sistema. Neste caso, os nós de uma partição só se comunicam entre si.

Segundo o teorema CAP, apenas duas das três propriedades apresentadas podem ser garantidas ao mesmo tempo. Vale destacar que tal escolha não é binária, no sentido de que é possível relaxar certas propriedades para que seja possível privilegiar outras. Por outro lado, as garantias transacionais ACID inviabilizam esta flexibilidade. Como uma alternativa, pode-se ter um BD que funciona quase o tempo todo (basicamente disponível) e não é consistente o tempo todo (estado flexível), apenas quando as escritas forem propagadas para todos os nós (eventualmente consistente) em um sistema distribuído (ver Subseção 3.2.3). Este modelo de garantias transacionais é chamado de BASE, cujas características, embora não sejam rigorosamente definidas, são apresentadas como:

- Basicamente disponível (BA, *basicaly available*): o sistema deve estar disponível mesmo que ocorram falhas parciais;
- Estado flexível (S, s*oft State*): o sistema pode não possuir dados consistentes o tempo todo;
- Eventualmente consistente (E, *eventually consistent*): a consistência será obtida assim que todas as escritas forem propagadas para todos os nós.

3.2.3 Vantagens e Desvantagens do Modelo Relacional

Desde a década de 70, o modelo relacional foi amplamente utilizado nas mais diversas aplicações de bancos de dados. Embora não tenha perdido seu espaço, existem determinadas aplicações de banco de dados cujas especificações não são plenamente satisfeitas por este modelo de dados (FOWLER, 2015; SADALAGE; FOWLER, 2013). Estas aplicações motivaram a busca por alternativas a este modelo, sem que, no entanto, o modelo relacional perdesse espaço entre as soluções de armazenamento de dados. A seguir, são apresentados exemplos de aplicações de bancos de dados e são descritas as vantagens e desvantagens do modelo relacional em se adequar às exigências destas aplicações.

Uma das formas de se classificar bancos de dados é dividi-los quanto ao número de aplicações as quais servem. São duas as possibilidades: BDs de integração e de aplicação. Os primeiros armazenam dados de múltiplas aplicações em um único BD. Este tipo de sistema tem uma estrutura muito mais complexa do que seria exigido por aplicações individuais, já que existe a necessidade de coordenar e orquestrar as aplicações, que diferem entre si sobretudo quanto à necessidade de desempenho em termos de suas operações. Os bancos de dados de aplicação, por sua vez, são acessados e atualizados por uma única aplicação. Este tipo de

implementação permite que os BDs sejam encapsulados às aplicações e a integração entre eles ocorre por meio de serviços, de modo que BDs de aplicação são fundamentais para aplicações *web* e arquiteturas orientadas à serviços em geral (SADALAGE; FOWLER, 2013).

Os bancos de dados de integração geralmente implementam o modelo relacional, já que as propriedades ACID conferem justamente o controle de concorrência desejado para coordenar as solicitações dos diferentes usuários/aplicações do BD (STONEBRAKER, 2010). No entanto, para os bancos de dados de aplicação, o modelo relacional implica certas desvantagens. A primeira delas está ligada à flexibilidade: Tendo em vista que um BD de aplicação é acessado e atualizado por uma única aplicação, sua estrutura pode ser criada de modo a beneficiá-la (SADALAGE; FOWLER, 2013). Uma estrutura de dados flexível, que poderia ser projetada com base nas necessidades de uma única aplicação, seria desejável para implementação de BDs do tipo em questão. Tal flexibilidade não é observada no modelo relacional, tanto em termos do esquema de relações, quanto das restrições impostas pelos domínios aos valores possíveis para os atributos nas relações.

A segunda característica que torna o modelo relacional desvantajoso para BDs de aplicação está ligada às funcionalidades do mesmo que se tornam desnecessárias e até mesmo indesejáveis. Um BD de aplicação geralmente necessita de uma quantidade muito menor de operações oferecidas pela linguagem SQL (SADALAGE; FOWLER, 2013).

A necessidade de distribuição do banco de dados é outra característica que impõe limitações ao uso do modelo relacional (FOWLER, 2015; HAN *et al*., 2011). O principal problema associado ao uso de BDs relacionais para implementação de sistemas de bancos de dados distribuídos está ligado ao teorema CAP e, consequentemente, às garantias transacionais ACID. Um sistema de servidor único, no qual não há distribuição dos dados, é um sistema CA: não existe tolerância à partição porque não existe partição em uma única máquina. Assim sendo, as duas outras propriedades – disponibilidade e consistência – são garantidas. A maioria dos sistemas de BDs relacionais é CA e as licenças deste tipo de SGBD são geralmente comercializadas para serem executadas em servidor único (SADALAGE; FOWLER, 2013).

Em sistemas distribuídos, existe a possibilidade de partição da rede. Neste tipo de sistema só é possível renunciar à tolerância ao particionamento se, no caso de uma falha de partição da rede, o sistema ficar completamente inoperante, o que é criticamente indesejável em certos casos. Deste modo, em sistemas distribuídos, geralmente não é desejável renunciar à tolerância à partição de rede, ou seja, não é desejável ter um sistema distribuído CA. As outras possibilidades são renunciar à consistência ou à disponibilidade. Logo, a essência do teorema

CAP pode ser entendida como: em um sistema que pode estar sujeito a particionamento, deve-se priorizar entre consistência ou disponibilidade. Este acaba sendo, na verdade, um *trade-off* entre consistência e latência: para se ter transações consistentes, é necessário um determinado tempo para que alterações nos dados sejam propagadas para todas as cópias e o sistema possa estar novamente disponível (ABADI, 2012). Em síntese, existem bancos de dados relacionais escaláveis horizontalmente, isto é, que podem ser distribuídos (CATTELL, 2010), mas o que pode tornar inviável esta distribuição é a possível alta indisponibilidade do sistema.

Devido ao fato das transações que adotam o modelo ACID serem fortemente consistentes, não é possível balancear o *trade-off* entre consistência e latência em bancos de dados distribuídos que utilizam este modelo de garantias (ABADI, 2012). Por esta razão, manter garantias ACID geralmente faz com que um BD distribuído que implemente o modelo relacional tenha latência mais alta (KHINE; WANG, 2019; SADALAGE; FOWLER, 2013). Para se ter alta disponibilidade, os dados precisam ser replicados em um ou mais nós, de modo que, se um determinado nó falha, os dados estão disponíveis em outro. A replicação pode aumentar não apenas a disponibilidade, mas também o desempenho, reduzindo a sobrecarga dos nós para operações de leitura. No entanto, para operações de escrita em que se deseja garantir que todos os nós tenham uma cópia atualizada dos dados, pode-se ter perda de desempenho (deve-se aguardar até que os dados sejam replicados em todos os nós). Por outro lado, sistemas que adotam o modelo BASE conseguem atingir menor latência, mas inconsistências podem ocorrer durante um certo intervalo de tempo (janela de inconsistência), uma vez que os diferentes nós podem apresentar versões distintas do mesmo registro.

Certas aplicações podem conter conjuntos de dados que são frequentemente acessados e manipulados juntos. Em sistemas distribuídos, estes conjuntos podem formar uma unidade natural de distribuição (SADALAGE; FOWLER, 2013), de modo que usuários interessados se direcionem sempre a um ou mais nós específicos onde eles estejam armazenados. Em bancos de dados, estes conjuntos são denominados agregados: uma estrutura rica, formada por um conjunto de dados (objetos) que podem ser armazenados como uma unidade, pois são frequentemente manipulados desta maneira.

Em termos de modelo de dados de implementação, o principal problema do modelo relacional com relação a representação de agregados é justamente a sua estrutura rígida, que impossibilita tratar conjuntos de dados como uma unidade. O modelo relacional permite representar as entidades e os relacionamentos que fazem parte de um agregado, no entanto não

possibilita representar o agregado em si, isto é, não permite identificar quais relacionamentos constituem um agregado e nem as fronteiras deste agregado.

3.2.4 Modelos de Dados NoSQL

As subseções anteriores apresentaram situações nas quais a utilização do modelo relacional não é adequada. Para lidar com estas situações, foram criados sistemas gerenciadores de bancos de dados que não utilizam o modelo relacional como modelo de dados de implementação. Estes bancos de dados passaram a ser denominados NoSQL.

O termo NoSQL não dispõe de uma definição sólida, possivelmente sendo melhor compreendido como um movimento que propõe soluções de bancos de dados não relacionais e que não utilizam a linguagem SQL (SADALAGE; FOWLER, 2013). Assim, o termo NoSQL (frequentemente interpretado como *Not Only SQL*) utilizado em seu sentido tecnológico designa uma família de SGBDs que possuem certas características em comum (ao menos para a maioria dos SGBDs), sendo a principal delas a não implementação do modelo de dados relacional (MONIRUZZAMAN; HOSSAIN, 2013). Estas características podem significar vantagens ou desvantagens para determinadas aplicações:

- Não implementam o modelo de dados relacional: a autodescrição e ausência de um esquema fixo possibilitam maior flexibilidade com relação ao conteúdo armazenado nos SGBDs, sendo adequados para gerenciamento de dados semiestruturados (ELMASRI; NAVATHE, 2000; MONIRUZZAMAN; HOSSAIN, 2013);
- Não utilizam a linguagem SQL: a ausência de uma linguagem de consulta declarativa, com uma ampla gama de “funcionalidades” às vezes desnecessárias, exige maior esforço para desenvolvedores, uma vez que as funções e operações têm que ser implementadas por meio de linguagem de programação (ELMASRI; NAVATHE, 2000);
- Ausência de transações ACID: pelo fato dos modelos de dados orientados a agregados geralmente não garantirem as propriedades transacionais ACID, os SGBDs que implementam estes modelos dispõem de maior eficiência em sistemas distribuídos (ABADI, 2012; STONEBRAKER, 2010). Como alternativa, estes SGBDs usam o modelo BASE de propriedades transacionais.
- Escaláveis horizontalmente: A capacidade de escalabilidade horizontal dos SGBDs NoSQL está atrelada a duas características principais: 1) por não ter garantias

> transacionais ACID (modelos orientados a agregados), permite relaxar a consistência e assim equilibrar o *trade-off* consistência-latência da maneira que for mais adequada para a aplicação, sem abrir mão da tolerância à partição, conforme discutido anteriormente; 2) a orientação a agregados possibilita uma unidade de particionamento "natural" ou intuitiva dos dados, pois dados de um agregado comumente são acessados juntos, podendo ser alocados em um mesmo servidor, o que faz com que o usuário destes dados recorra, na maioria das vezes, a um mesmo servidor (SADALAGE; FOWLER, 2013).

SGBDs NoSQL são comumente diferenciados com base na forma com que os dados são armazenados por eles, isto é, o modelo de dados que é empregado para o armazenamento. São quatro os principais modelos de dados implementados em SGBDs NoSQL: chave-valor, documentos, famílias de colunas e grafos.

Os SGBDs **chave-valor** são possivelmente os sistemas NoSQL mais simples. Estes SGBDs armazenam seus dados em conjunto de pares (chave-valor) sem um esquema rígido, em que cada linha corresponde a uma chave única e um conjunto de objetos auto descritos, chamados de valor, que podem assumir diferentes formatos, dos mais simples como *strings*, passando por tabelas como no modelo relacional, chegando a formatos mais elaborados como documentos do tipo *JavaScript Object Notation* (JSON) e *Extensible Markup Language* (XML). Assim, podem armazenar dados estruturados, semiestruturados e não estruturados em um formato (chave, valor).

O modelo de dados chave-valor é frequentemente representado como uma tabela contendo apenas duas colunas (uma para a chave e uma para o respectivo valor), como exemplificado na Figura 12, também derivada do modelo conceitual de descrição de aerogeradores da Figura 10. A Figura 12 contém dados para exemplificar o banco chave-valor. Este modelo de dados é orientado a agregados, o que quer dizer que cada valor associado a uma chave única pode ser entendido como um agregado de objetos que podem ser recuperados em sua totalidade por meio da chave. O conteúdo e estrutura desses agregados pode ser diferente para cada chave.

A possibilidade de armazenar qualquer tipo de dado nos agregados é garantida pela opacidade do agregado, isto é, o SGBD não interpreta o conteúdo do agregado, enxergando-o apenas como um conjunto de *bits* que deve estar sempre associado à sua chave única. Isto tem a implicação prática de geralmente não permitir recuperações parciais no conteúdo dos agregados. As operações implementadas por SGBDs do tipo chave-valor são basicamente a

inserção ou atualização de um par (chave, valor), uma recuperação de um valor a partir de sua chave ou a exclusão de uma chave.

Chave	Valor
Aerogerador_1	Tempo de disponibilidade: 7 Número de paradas: 5 Produção total de energia ativa: 5.58 MWh Demanda de energia ativa: 6 MWh Fornecimento de energia ativa: 5.58 MWh Estado do aerogerador: Em falha Geração de potência ativa: 36 kW Geração de potência reativa: 0.12 kVA Comando de operação: *False* Setpoint de geração de potência ativa: 40 kW Setpoint de geração de potência reativa: 0.01kVA Setpoint de fator de potência: 0.99
Aerogerador_2	{ "comum": { "Tempo de disponibilidade": "7 dias", "Número de paradas": "2", "Produção total de energia ativa":"6.65 MWh",, "Demanda de energia ativa":"5 MWh", "Fornecimento de energia ativa":"5 MWh“ }, "estado": { "Estado do aerogerador": "em operação"}, "analógico": { "Geração de potência ativa": "38 kW", "Geração de potência reativa": "0.12 kVA“ }, "controle": { "comando de operação": "True", "Setpoint de geração de potência ativa": "40 kW", "Setpoint de geração de potência reativa": "0.10 kW", "Setpoint de fator de potência": "0.99“ } }

Figura 12 - Exemplo de banco de dados chave-valor.

Fonte: autoria própria.

Os bancos de dados de **documentos** são aqueles em que os dados, como o próprio nome sugere, são armazenados em formato de documentos. Podem ser entendidos como um SGBD chave-valor no qual o único formato permitido para os valores são documentos como XML, ou JSON. Uma diferença fundamental entre os modelos de dados de documentos e chave-valor é que o primeiro, por armazenar apenas um formato de dados autodescritos, permite que se façam recuperações parciais do agregado. Dito de outra maneira, os agregados não são opacos, não são vistos pelo SGBD meramente como um conjunto de *bits*, sendo possível definir índices sobre os conteúdos dos agregados que permitem que operações sejam feitas em itens específicos deste conjunto de dados.

Assim como no modelo de dados chave-valor, o conteúdo de cada documento não segue um esquema fixo. Os rótulos (*labels*) do documento garantem a autodescrição dos dados, além de possibilitarem recuperações parciais, também permitem que chaves diferentes tenham documentos com conteúdo (atributos) diferentes. Assim, permite o armazenamento de dados estruturados e semiestruturados. Ainda existem SGBD que permitem o armazenamento de

dados não estruturados, como textos. Um exemplo de BD de documentos é ilustrado na Figura 13.

Chave	Documento JSON
Aerogerador_1	{ "comum": { "Tempo de disponibilidade": "7 dias", "Número de paradas": “5", "Produção total de energia ativa":"5.58 MWh", "Produção total de energia reativa":"0.013 MVArh", "Demanda de energia ativa":“6 MWh", "Fornecimento de energia ativa":"5.58 MWh“ }, "estado": { "Estado do aerogerador": “em falha"}, "analógico": { "Geração de potência ativa": "36 kW", "Geração de potência reativa": "0.12 kVA“ }, "controle": { "comando de operação": “*False*", "Setpoint de geração de potência ativa": "40 kW", "Setpoint de geração de potência reativa": "0.10 kW", "Setpoint de fator de potência": "0.99“ } }
Aerogerador_2	{ "comum": { "Tempo de disponibilidade": "7 dias", "Número de paradas": "2", "Produção total de energia ativa":"6.65 MWh", "Produção total de energia reativa":"0.02 MVArh", "Demanda de energia ativa":"5 MWh", "Fornecimento de energia ativa":"5 MWh“ }, "estado": { "Estado do aerogerador": "em operação"}, "analógico": { "Geração de potência ativa": "38 kW", "Geração de potência reativa": "0.12 kVA“ }, "controle": { "comando de operação": "*True*", "Setpoint de geração de potência ativa": "40 kW", "Setpoint de geração de potência reativa": "0.10 kW", "Setpoint de fator de potência": "0.99“ } }

Figura 13 - Exemplo de banco de dados de documento.

Fonte: autoria própria.

Nos BDs de **famílias de colunas**, os dados são armazenados em um formato de tabelas sem uma estrutura rígida, semelhante à dos bancos chave-valor. Porém, no modelo de famílias de colunas, o elemento correspondente ao valor é composto por uma série de outras tabelas que possuem seus respectivos nomes que atuam como identificadores. Nestas tabelas, as colunas são autodescritas, isto é, possuem uma chave (também chamado de qualificador) e o respectivo valor, que são os dados propriamente ditos. Cada tabela, que contém um conjunto de colunas e o respectivo valor, recebe o nome de família de colunas. Em síntese, cada registro deste tipo de banco de dados é associado a uma chave única e é composto por uma série de famílias de colunas, que são tabelas auto descritas e que efetivamente contém os dados armazenados, como ilustra a Figura 14.

Pode-se fazer algumas considerações acerca deste modelo. A primeira delas é que, embora as famílias de colunas sejam fixas e definidas na criação da tabela, para cada chave, as

famílias podem conter apenas as colunas de interesse, isto é, não precisam ser compostas pelas mesmas colunas para todas as chaves. No exemplo da Figura 14, a tabela possui duas linhas, identificadas pelas chaves “Aerogerador_1” e “Aerogerador_2”. Ambas as linhas possuem famílias de colunas com o mesmo nome: “Comum”, “Estado”, “Analógico” e “Controle”, mas com estrutura e conteúdo diferentes. A família de colunas “Comum” no segundo registro não possui o qualificador “Número de paradas” e a família de colunas “Controle” deste mesmo registro não possui o qualificador “*setpoint* de fator de potência”. Observa-se também que as famílias de colunas formam agregados de dados que são frequentemente acessados conjuntamente e que, pelo fato destas colunas conterem suas próprias chaves, os agregados não são opacos aos SGBDs, sendo possível executar recuperações parciais por meio dos agregados por meio dos índices das colunas.

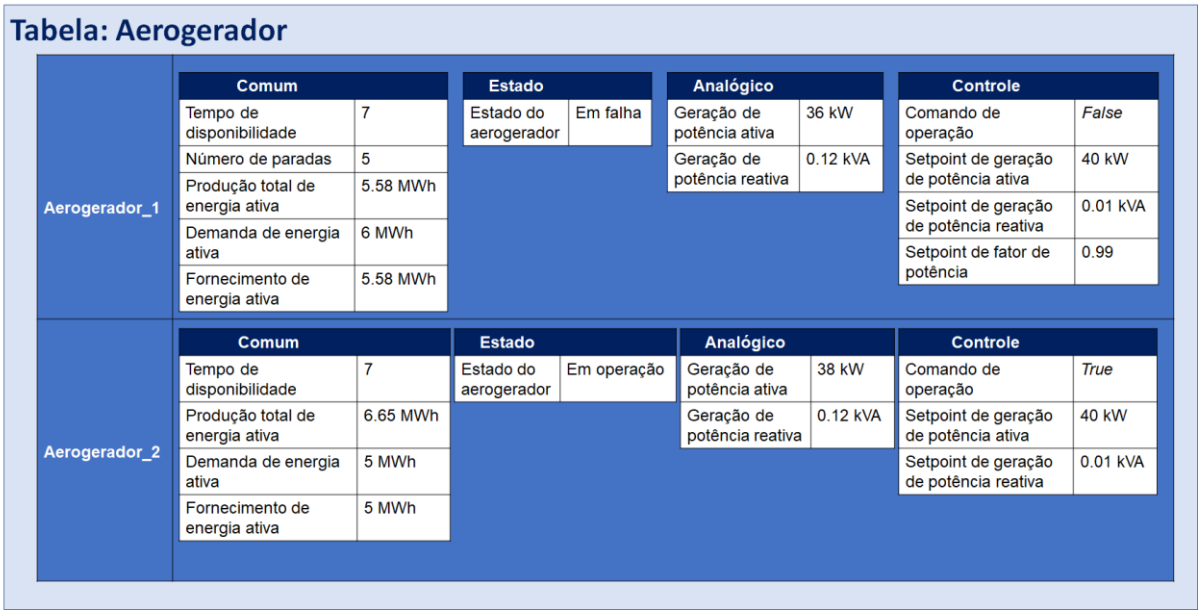


Figura 14 - Exemplo de banco de dados de famílias de colunas.

Fonte: autoria própria.

Nos SGBDs orientados a **grafos**, os dados são armazenados em uma coleção de nós, que representam entidades, e vértices direcionados, que representam relacionamentos entre estas entidades. O conjunto de nós e vértices formam grafos, nos quais os dois elementos que os compõem podem conter rótulos e atributos associados a eles, que são os dados propriamente ditos. Para exemplificar o banco de dados de grafos, a Figura 15 (a) apresenta um novo diagrama entidade relacionamento, mais simples e contendo informações de três componentes

de um aerogerador: rotor, nacele e torre[8]. Na Figura 15 (b), o BD de grafos correspondente é apresentado.

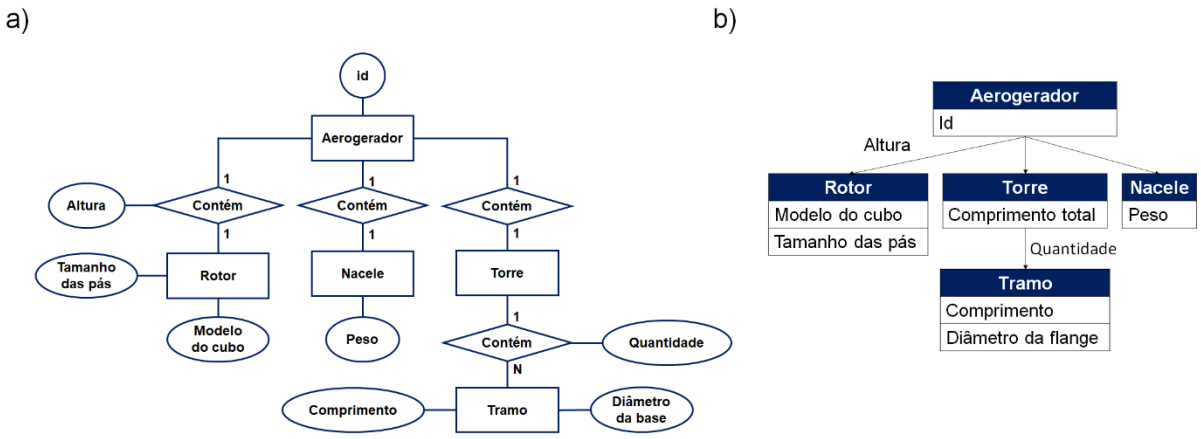


Figura 15 - Exemplo de banco de dados de grafos. a) modelo conceitual do banco de dados; e b) modelo lógico de grafos correspondente.

Fonte: autoria própria.

Com relação às características dos modelos de dados apresentadas até aqui, a flexibilidade na representação dos dados pela ausência de um esquema fixo é uma das poucas semelhanças entre os BDs de grafos e os outros modelos de dados NoSQL mencionados, pois tanto os vértices quanto os nós podem conter atributos diferentes entre si. Com relação às diferenças, o modelo de grafos não possui orientação a agregados; geralmente possui garantias transacionais ACID; é mais adequado para implementações do tipo *single server* (servidor único, isto é, sem distribuição); é capaz de representar registros relativamente pequenos com relacionamentos complexos entre si; e são mais eficientes para identificar padrões.

3.3 Integração de Dados

A criação de sistemas inteligentes baseados em dados[9] requer que se tenha uma visão mais abrangente dos ativos de uma empresa (ELMASRI; NAVATHE, 2000;

[8] Os subsistemas e componentes de um aerogerador são apresentados no Capítulo 5, Subseção 5.1.2

[9] Na terminologia de bancos de dados, estes sistemas são comumente referidos como sistemas de apoio à decisão (em inglês, *decision support systems*).

RAMAKRISHNAN; GEHRKE, 2008). Deste modo, a aplicação de métodos de *big data analytics*, bem como a própria dimensão "variedade" de *big data* pressupõem a integração de dados provenientes de diferentes fontes. Apresenta-se assim o conceito de *Data Warehouse*, bem como uma arquitetura genérica[10] e um modelo de dados conceitual que se adequam às finalidades deste tipo de sistema.

3.3.1 Data Warehouse

A integração de diferentes fontes de dados é uma necessidade recorrente quando se deseja realizar análises e extrair valor a partir de dados (ELMASRI; NAVATHE, 2000; TEOREY *et al*., 2005). Integrar dados provenientes de diferentes fontes possibilita que se tenha maior abrangência entre as visões de negócios da organização (LEE, 1999). Esta tarefa pode ser realizada por meio de sistemas específicos para a integração de dados. Dois dos mais relevantes tipos de sistemas para a integração de dados são os *data lakes* (DL) e *Data Warehouse* (DW).

- *Data lake*: trata-se de um tipo de sistema para armazenamento de *big data*, destinado a guardar dados em sua forma original (ou que tenham sofrido mínimas transformações), incluindo dados não estruturados (NAMBIAR; ATHIRA, 2022; MILOSLAVSKAYA; TOLSTOY, 2016; KHINE; WANG, 2018). Promovem um baixo custo de armazenamento, alta agilidade e podem ser facilmente reconfigurados, já que não possuem um esquema predefinido (NAMBIAR; ATHIRA, 2022).
- *Data Warehouse:* consiste em um tipo de sistema utilizado para gerenciamento de conjuntos de dados, geralmente muito grandes, não voláteis, variantes no tempo, composto por dados históricos que são extraídos de diversas fontes (isto é, diversos bancos) e integrados. Este tipo de sistema é projetado para dar apoio à extração, processamento e apresentação eficiente dos dados para fins analíticos e de tomada de decisão (ELMASRI; NAVATHE, 2000; INMON; STRAUSS; NEUSHLOSS, 2010; RAMAKRISHNAN; GEHRKE, 2008; TEOREY *et al*., 2005). São adequados para dados pós processados, em estrutura tabular, com um esquema predefinido. São amplamente utilizados em análise de dados voltada à inteligência corporativa e projetados com tal propósito (NAMBIAR; ATHIRA, 2022).

[10] Existem diferentes arquiteturas possíveis para um sistema de *data warehouse*. A variação adotada na Figura 16 é frequentemente utilizada para explicar este tipo de sistema.

A escolha pela utilização de um DW ou DL é fundamentalmente dependente das necessidades da aplicação. Tendo-se em vista que (1) no contexto da I4.0, a representação dos ativos possui uma estrutura predefinida pelo AAS e que existem interfaces e operações também padronizadas para obtenção dos dados do AAS; (2) pela estrutura predefinida, os DWs podem facilitar a integração com outras ferramentais (tais como ferramentas de ETL e análise de dados); (3) os DW apresentam alto desempenho de consultas de maior complexidade e em grandes volumes de dados; avalia-se que este tipo de ferramenta é adequada para o propósito deste trabalho em termos de integração de dados (estruturados pelo AAS) para análise a fim de se contribuir para alcançar a Fábrica Inteligente.

Pode-se identificar alguns dos aspectos que diferenciam *Data Warehouses* de BDs tradicionais:

- Transações e operações: bancos de dados convencionais, sejam eles relacionais ou NoSQL, geralmente se dedicam a aplicações de processamento de transações *online* (*OLTP, do inglês online transaction processing*, ou processamento de transações *online*), operações rotineiras de uma organização, que atuam em frações pequenas dos dados, ocorrem com grande frequência e devem ser processadas de maneira eficiente (RAMAKRISHNAN; GEHRKE, 2008). Tais operações são, por exemplo, inserções, atualizações, exclusões e consultas (ELMASRI; NAVATHE, 2000). Estes bancos também são chamados de transacionais. Os *Data Warehouses*, por sua vez, são otimizados para o processamento analítico *online* (OLAP, do inglês *online analytical processing*). As transações OLAP permitem extrair dados para análises complexas. Dedicam-se à recuperação dos dados, envolvendo operadores de agrupamento e junção, funções estatísticas e condições booleanas complexas (ELMASRI; NAVATHE, 2000), aplicadas a muitos registros. Além das transações OLAP, aplica-se aos dados de um DW algoritmos de mineração de dados para análise exploratória e descoberta (transformação) de dados em informação e conhecimento; ferramentas de visualização para as diferentes perspectivas de interesse dos dados; e geradores de relatórios (RAMAKRISHNAN; GEHRKE, 2008);
- Volume e volatilidade: análise de dados exigem não apenas uma perspectiva abrangente do sistema, mas geralmente um volume considerável de dados históricos, como séries temporais. Assim, um DW armazena volumes de dados muito maiores que suas fontes de dados – os BDs transacionais – além de garantirem a persistência destes dados por muito mais tempo, enquanto que os BDs tradicionais armazenam, geralmente, dados

atuais e não históricos (TEOREY *et al*., 2005). Deste modo, as modificações no conteúdo de um DW geralmente ocorrem de maneira incremental e em lotes (*batches*), enquanto as alterações nos BDs transacionais ocorrem de maneira contínua (ELMASRI; NAVATHE, 2000; TEOREY *et al*., 2005);

- Orientação e usuários: os DWs são sistemas dedicados a facilitar a análise exploratória dos dados (ELMASRI; NAVATHE, 2005), sendo dedicados ao auxílio à decisão e aos processos de negócio. Por esta razão, diz-se que os DWs são orientados a assunto. Geralmente se dedicam a poucos usuários. Os BDs tradicionais, por sua vez, são orientados às transações e podem ter poucos (BDs de aplicação) ou muitos usuários (TEOREY *et al*., 2005).

Apresenta-se uma arquitetura genérica de *Data Warehouse*, ilustrada na Figura 16. Em linhas gerais, neste tipo de sistema, os BDs transacionais identificados à esquerda da figura são usados para gerar um *Data Warehouse*, ilustrados à direita da Figura 16. O banco de dados da direita pode conter as mesmas informações que os da esquerda, só que em outro tipo de esquema, de modo a facilitar as operações de *data analytics* (FOWLER, 2014). Os componentes e processos deste sistema são descritos nos parágrafos seguintes.

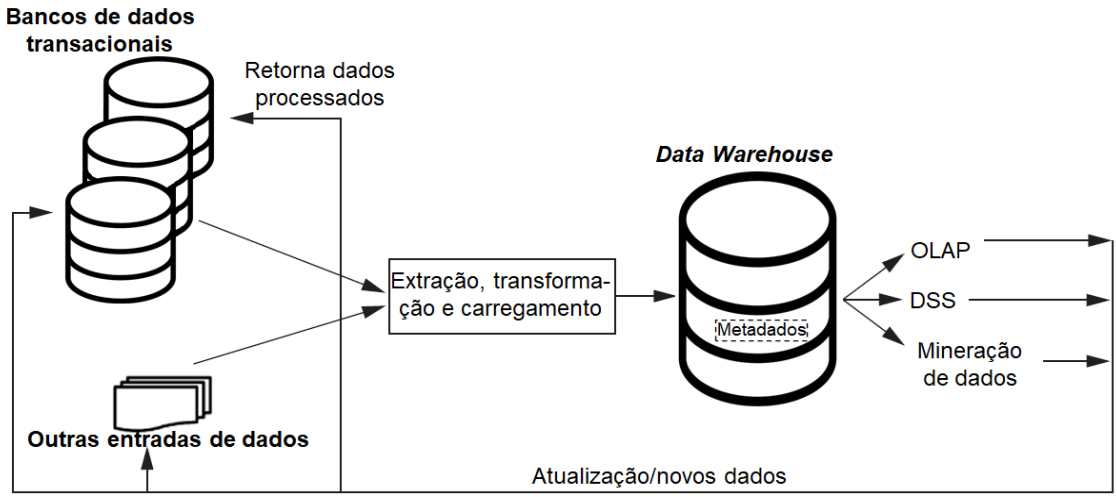


Figura 16 - Arquitetura genérica de *Data Warehouse*.

Fonte: adaptado de Elmasri e Navathe (2000).

Inicia-se pela descrição de uma etapa que envolve os procedimentos mais críticos em termos de prazo e custo para um sistema de DW. Tal etapa é composta pelos procedimentos de extração, transformação e carregamento dos dados e, por esta razão, geralmente referida como

ETL (do inglês *Extract Transform Load*) (KIMBALL; CASERTA, 2004). Esta etapa, ilustrada na Figura 17, tem início com a extração dos dados dos BDs transacionais.

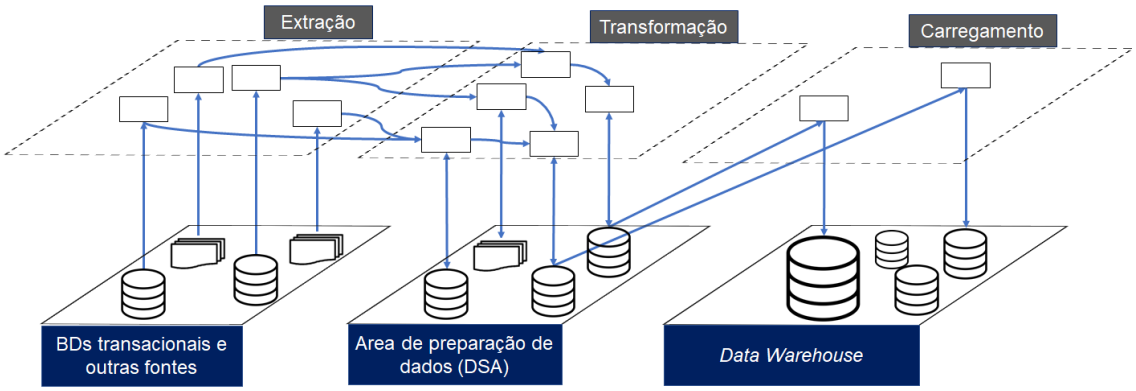


Figura 17 - Processos de extração, transformação e carga.

Fonte: adaptado de Ferreira *et al*. (2010).

Antes de carregar os dados extraídos dos BDs transacionais no DW, é necessário realizar procedimentos de pré-processamento dos dados (ELMASRI; NAVATHE, 2005), que ocorre na chamada área de preparação de dados (DSA, *data staging area*) (TEOREY *et al*., 2005). Além dos motivos mais associados à "limpeza" dos dados, deve-se considerar que cada um dos BDs transacionais pode ser construído com base em uma estrutura que favoreça suas operações. Logo, é necessário construir uma estrutura que permita integrar os dados das diferentes fontes. Assim, para que sejam integrados e armazenados no DW, os dados precisam passar por procedimento de transformação. Vale destacar que este procedimento não afeta os dados armazenados nos BDs transacionais e em outras fontes.

Por fim, os dados são armazenados no *Data Warehouse*. Devido ao grande volume dos dados a serem armazenados em um DW e tendo em vista que este tipo de sistema é orientado a assunto, isto é, é construído com base nas visões que se deseja ter dos dados, ele pode ser composto por diferentes *Data Marts*, que consolida apenas visões dedicadas a uma única área da organização.

Por esta razão, além dos dados operacionais integrados, o *Data Warehouse* também possui um componente chave que é o conjunto de metadados (ELMASRI; NAVATHE, 2000). Os metadados descrevem o banco de dados e funcionam como índices para o conteúdo do DW, sendo importantes para que os usuários finais do DW (que podem ser de diferentes setores da organização) encontrem e interpretem corretamente os dados de interesse. Inmon (2002) diferencia os metadados em técnicos e de negócios. Os primeiros fornecem uma descrição

acerca da aquisição, processamento, estruturas de armazenamento, operações e manutenção dos dados, além da descrição dos dados propriamente ditos. Os segundos contemplam as regras de negócios relevantes e decisões organizacionais.

Os dados armazenados no DW são acessados por diversas ferramentas para análise dos dados, incluindo transações OLAP, mineração de dados, geração de relatórios, entre outras. Em alguns casos, os resultados provenientes das análises dos dados são realimentados nos BDs transacionais e nas demais fontes de dados operacionais, conforme representa o fluxo de dados que parte da direita para a esquerda na Figura 16. Com isso, foram descritos os principais componentes e os fluxos de dados de um sistema de *Data Warehouse*.

3.3.2 Modelo de Dados Multidimensional

Os DWs podem possuir diferentes perfis de usuários e podem ser organizados de modo a criar unidades, isto é, subconjuntos de dados destinados a um segmento, assunto, ou problema específico da organização. Estas unidades são chamadas *Data Marts* (DM) (BONIFATI *et al*., 2001). Embora tenham um escopo mais específico e limitado do que um DW como um todo, os DMs ainda devem permitir que seus dados possam ser observados sob diferentes perspectivas (CABIBBO; TORLONE, 2004). Para isso, os DMs se baseiam no modelo de dados multidimensional (INMON, 2002)[11].

O modelo multidimensional é um modelo de dados conceitual. Este modelo é composto por fatos, que são as medidas numéricas, e dimensões, que representam as diferentes perspectivas sob as quais os dados podem ser vistos. Os valores das primeiras dependem das segundas. O modelo multidimensional é geralmente representado por meio de matrizes multidimensionais, também chamadas de cubos ou hipercubos, quando mais de três dimensões são consideradas. Em sistemas de *Data Warehouse*, cada *Data Mart* possui seu próprio cubo de dados, com as perspectivas de interesse para uma subdivisão específica da organização. Na Figura 18 se tem um cubo que representa dados tridimensionais. Este cubo contém dados de falhas mensais que ocorrem em determinados ativos. As dimensões são “Ativo”; “Número de

[11] Tendo em vista que os DMs são os blocos construtores de um DW, encontra-se com frequência na literatura que os *data warehouses* são construídos com base no modelo de dados multidimensional. No entanto, é importante destacar que, segundo Inmon (2002), o uso do modelo multidimensional para criar um DW é incorreto, pois isto beneficiaria uma parte em detrimento das demais.

falhas” e “Mês”, indicadas nas arestas do cubo. Os fatos são os dados da quantidade de falhas que ocorreram em um determinado mês para um ativo específico, contidos nos cubos menores que compõem o cubo maior. Supondo que o cubo maior faça parte do sistema de DW de uma empresa e que os ativos sejam produtos fornecidos por esta empresa, ele seria do interesse do departamento de pós-venda, por exemplo.

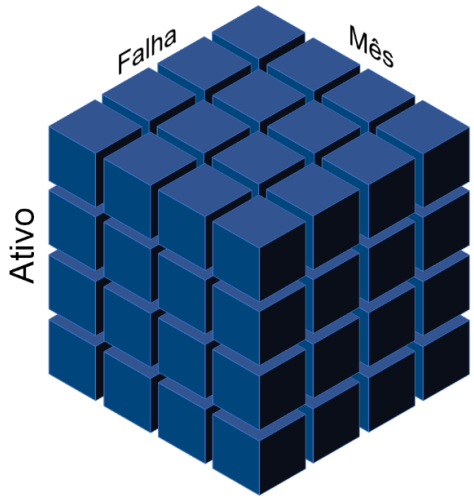


Figura 18 - Dados de falhas por ativo por mês representados em um cubo.

Fonte: autoria própria.

No nível de dados lógico, descreve-se quatro formas principais de se implementar o modelo de dados multidimensional. Assim como os sistemas NoSQL, os sistemas de gerenciamento de dados multidimensionais são geralmente identificados pela forma com que o armazenamento se dá. Os sistemas mais comuns para implementação do modelo de dados multidimensional são chamados de OLAP Multidimensionais (MOLAP), OLAP relacionais (ROLAP) e OLAP híbridos (HOLAP). Por fim, identifica-se a implementação do modelo multidimensional em sistemas NoSQL.

Inicia-se pela descrição dos sistemas MOLAP e ROLAP. Os primeiros armazenam os dados em matrizes multidimensionais. Os segundos realizam o armazenamento dos dados em forma de tabelas, utilizando o modelo relacional apresentado em 3.2.1. De acordo com (KASER; LEMIRE, 2006), sistemas MOLAP requerem um espaço muito maior para armazenamento dos dados em comparação aos sistemas ROLAP, sobretudo quando a matriz multidimensional dos dados é esparsa (isto é, possui muitos valores nulos). Além disso, nos primeiros, os dados são previamente calculados para que sejam agregados nas devidas dimensões antes de serem armazenados, o que implica frequentemente em sobrecarga do

sistema para realizar estas tarefas preliminares, porém permite que as operações OLAP possam ser realizadas mais rapidamente em comparação aos sistemas ROLAP (ZHANG *et al*., 2019b). O problema das sobrecargas em sistemas OLAP é ainda mais crítico quando as dimensões do cubo são modificadas, pois a matriz multidimensional precisa ser reconstruída (ZHANG *et al*., 2019b). Deste modo, os sistemas ROLAP facilitam a manutenção do *DM*.

Os sistemas OLAP Híbridos buscam combinar sistemas ROLAP e MOLAP de modo a extrair o melhor de cada um. Neste tipo de sistema, os bancos de dados relacionais armazenam as maiores quantidades de dados e são acessados para consultas mais detalhadas, enquanto agregações destes dados armazenados no BD relacional são armazenadas em matrizes multidimensionais sem que os dados fonte sejam de fato copiados (DUNCAN, 2018a). O resultado desta combinação é um equilíbrio para o *trade-off* entre desempenho e volume. Apesar desta vantagem, este tipo de sistema requer manutenções e atualizações mais frequentes pela complexidade da arquitetura (IBM - CLOUD EDUCATION, 2014).

Sabe-se que uma das características de *Data Warehouses* é o grande volume de dados armazenados. Conforme apresentado em 3.2, bancos de dados relacionais podem não ser a melhor opção para armazenamento de grandes volumes de dados, sobretudo quando é necessária a sua distribuição, por isso sistemas ROLAP e HOLAP podem não ser adequados em certas situações. Em sistemas MOLAP, o problema do volume seria ainda mais crítico, visto que utilizam um espaço ainda maior que sistemas ROLAP para armazenar a mesma quantidade de dados. Motivados, sobretudo, pela dificuldade de lidar com grandes volumes de dados por meio de BDs relacionais, autores propõem a implementação de *Data Warehouses* por meio de sistemas NoSQL.

É possível identificar na literatura algumas propostas para implementação do modelo de dados multidimensional em diferentes tipos de BDs NoSQL. Bicevska e Oditis (2016); Chevalier *et al*. (2015a; 2015b); Scabora *et al*. (2016); e Yangui, Nabli e Gargouri (2016) investigam o mapeamento do modelo de dados conceitual multidimensional para os modelos de dados orientado a documentos e famílias de colunas, propondo a criação de *Data Warehouses* com base nestes sistemas NoSQL. Friedrichs (2021); Liu e Vitolo (2013); e Wang *et al*. (2014) fazem o mesmo para o modelo de grafos. Com isso, constata-se que simplesmente caracterizar um sistema como um *Data Warehouse* não é suficiente para definir a escolha do modelo de dados lógico a ser utilizado.

3.4 Processamento e Análise de Dados

Na Seção 3.1 foi apresentado o conceito de *big data analytics*, que diz respeito à aplicação de métodos analíticos para transformação dos dados em entidades de maior valor. A extração de valor a partir dos dados é um objetivo amplo, com uma série de métodos associados a frações deste objetivo. Por esta razão, aqui se considera um escopo menor, concentrando-se no processamento e análise de dados por meio de uma subclasse de inteligência artificial chamada de *aprendizado de máquina*, que vem apresentando resultados relevantes em diversos campos do conhecimento. Este foco em *aprendizado de máquina* se justifica ao levar em conta que, embora exista o aumento no volume de dados disponíveis na indústria, Lieber *et al*. (2013) observam que o aproveitamento destes dados por meio de aprendizado de máquina pelo setor industrial é consideravelmente inferior a outros campos, como o setor bancário, por exemplo. A aplicação de métodos de aprendizado de máquina é precedida por uma etapa de preparação de dados, em que os dados são processados de modo a viabilizar a aplicação dos métodos e a obtenção de melhores resultados. Por esta razão, tem-se aqui as considerações acerca do processamento de dados.

3.4.1 Aprendizado de máquina

O termo "inteligência artificial" (IA) foi cunhado em 1955 por John McCarthy, como tema de um evento de dois meses que viria a ocorrer no ano seguinte no Dartmouth College, em New Hampshire (BENKO; SIK LÁNYI, 2011). O chamado "Projeto de Pesquisa de Verão de Dartmouth em Inteligência Artificial", idealizado por McCarthy e com colaborações de Marvin Minsky, Nathaniel Rochester e Claude Shannon, que propunha um curso cujo desenvolvimento se daria "com base na conjectura de que todos os aspectos da aprendizagem ou qualquer outra característica da inteligência podem, em princípio, ser descritos com tanta precisão que uma máquina pode ser fabricada para simulá-la" (MCCARTHY *et al*., 2006).

Mesmo depois de seis décadas desde a primeira utilização do termo "inteligência artificial", ainda não há um consenso em termos de uma definição precisa para IA (KOK *et al*., 2009). Um único dicionário contém uma série de definições que, embora equivalentes, diferenciam-se em algum grau e podem causar confusão com outros termos mais específicos dentro da área da IA. Sendo assim, Inteligência Artificial pode ser descrita de modo mais amplo

como qualquer técnica que permita um computador imitar o comportamento humano (AMINI, 2020)[12].

Como subárea de IA, tem-se *aprendizado de máquina* (AM). Uma primeira definição mais generalista apresenta esta subárea como o campo de estudo que lida com a questão de como construir programas de computadores que possam ser melhorados automaticamente por meio da experiência (MITCHELL, 1997). Em outras palavras, AM corresponderia, segundo esta definição, à habilidade dos programas de computador de aprenderem sem que sejam explicitamente programados. Observa-se que esta definição não faz qualquer referência à forma com que estes algoritmos podem aprender e serem melhorados automaticamente, isto é, os seus métodos de aprendizado.

Em um primeiro momento, métodos probabilísticos foram considerados inadequados para IA. Acreditava-se que aprendizado analógico e raciocínio abdutivo (*abductive reasoning*) seriam mais apropriados para tal finalidade. Com o aumento da disponibilidade de dados, o sucesso de modelos probabilísticos se tornou mais evidente e as técnicas de aprendizado de máquina baseadas em probabilidade e estatística passaram a desenvolver um papel mais expressivo em diversas áreas de aplicação (JAMES et al., 2013).

Com base no sucesso no uso das probabilidades em aprendizado de máquina, apresenta-se uma forma alternativa de definir o conceito, fazendo referência à forma com que este aprendizado se dá: "*machine learning* é o estudo de métodos orientados a dados, capazes de imitar, entender e auxiliar tarefas de processamento de informações humanas e biológicas" (BARBER, 2012). Destaca-se, nesta definição, a expressão "orientados a dados", que explicita a forma pela qual os algoritmos podem aprender e serem melhorados. A definição corresponde a aprendizado de máquina estatístico, ou simplesmente aprendizado estatístico, que reúne uma série de procedimentos que permitem aos programas aprenderem por meio de bases de dados. Caso nenhuma observação seja feita, o termo aprendizado de máquina estatístico é utilizado neste trabalho como sinônimo para aprendizado de máquina.

[12] Proveniente das notas de aula do curso "6S191 *Introduction to Deep Learning*", ministrado por Amini. MIT, Cambridge, 2020.

3.4.2 Características e Preparação de Dados

Os conceitos fundamentais acerca de aprendizado de máquina fazem referência não a modelos probabilísticos de fato, mas às características das bases de dados utilizadas por estes modelos. Em linhas gerais, uma base de dados é composta por tabelas que contém atributos (também chamados de variáveis ou covariáveis), os quais são designados por x, e rótulos (variáveis de resposta ou de classe), identificados por y. Tipicamente, cada linha da tabela é composta pelos valores observados para os atributos e, eventualmente, para o rótulo corresponde àquela observação (JAMES et al., 2013).

Como as bases de dados geralmente têm mais de um atributo, define-se o vetor bidimensional X. Para designar uma determinada observação deste vetor, utiliza-se a notação x_{ij}, em que o índice i corresponde a um atributo e o índice j corresponde a um registro de valor deste atributo. Para uma base com m atributos e N observações, $i \in [1, m]$ e $j \in [1, N]$. De maneira análoga, define-se o vetor unidimensional Y, em que o índice j é o mesmo definido anteriormente e corresponde ao valor do rótulo na observação j. Vale ressaltar que o vetor de rótulos é unidimensional, pois apenas um rótulo é definido para cada observação.

Uma vez introduzidos os conceitos de atributos e rótulos, pode-se destacar algumas de suas características: em uma dada observação, o valor de um ou mais atributos podem não ser observados. Neste caso, diz-se que há um ou mais dados faltantes. A ausência dos dados pode ser sistemática ou aleatória. Também pode ocorrer de um atributo nunca ter seu valor observado. Neste caso, diz-se que esta é uma variável escondida. A ausência de uma ou mais observações da variável de classe é um fator mais crítico, uma vez que é um dos principais fatores que determina a técnica de AM a ser aplicada.

As bases de dados geralmente precisam ser preparadas para implementação dos algoritmos de AM. Esta tarefa consiste no processamento dos dados e pode ser extremamente dispendiosa, consumindo boa parte do tempo dedicado ao projeto de um modelo de AM. O processamento de dados consiste em executar, geralmente por meio de um computador, operações em dados[13,14]. Estas operações podem ser "coleta, registro, organização, estruturação, armazenamento, adaptação ou alteração, recuperação, consulta, uso, divulgação por

[13] The Free Dictionary: www.thefreedictionary.com/data+processing

[14] Cambridge Dictionary: dictionary.cambridge.org/pt/dicionario/ingles/data-processing

transmissão, disseminação ou disponibilização, alinhamento ou combinação, restrição, apagamento ou destruição”[15].

No caso deste estudo, as propostas de bancos de dados são elaboradas já com o intuito de facilitar a aplicação de algoritmos de AM, o que se traduz na redução de esforços necessários para preparação dos dados. No entanto, tendo em vista a diversidade dos cenários de aplicação, pode-se destacar algumas das tarefas principais do processo de preparação de dados do ponto de vista de AM.

- Integração de dados: os dados associados a uma aplicação podem ser provenientes de diferentes fontes e, para fins de análise destes dados combinados, é necessário integrá-los. O propósito da etapa de integração de dados é possibilitar o acesso uniforme a este conjunto de dados provenientes das diferentes fontes (DOAN; HALEVY; IVES, 2012);
- Discretização de dados: para determinadas finalidades, é desejável realizar a conversão de dados contínuos em valores discretos. Um exemplo deste tipo de situação é a aplicação de técnicas de aprendizado supervisionado, em que a variável de classe do modelo deve pertencer a um conjunto finito de valores discretos (DUNCAN, 2018b);
- Balanceamento da base de dados: em aplicações de monitoramento, por exemplo, é razoável esperar que os ativos apresentem pleno funcionamento a maior parte do tempo em que se encontram em operação. Assim, falhas se tornam eventos raros, porém importantes para aplicações de monitoramento. Eventos raros de interesse são menos frequentes (LATECKI; LAZAREVIC; POKRAJAC, 2007) e geram bases de dados desbalanceadas. Este tipo de base dificulta o processo de aprendizagem dos modelos de AM, pois há perda de desempenho das técnicas devido à existência de dados sub representados (HE; GARCIA, 2009);
- Conversão de formato dos dados: determinados formatos de dados precisam ser convertidos para posterior aplicação de técnicas de aprendizado de máquina, seja pela limitação da plataforma de AM utilizada (ZHANG *et al*., 2019a) ou por limitações da própria técnica, que necessitam de uma forma específica de representação dos dados. Tal representação pode inclusive ser obtida automaticamente por meio do que é denominado *representation learning* (BENGIO; COURVILLE; VINCENT, 2013);

[15] European Comission: https://ec.europa.eu/info/law/law-topic/data-protection/reform/what-constitutes-data-processing_en

- Remoção de dados discrepantes: se em certos casos, os eventos raros são objeto de interesse de uma aplicação, em outras situações, eles podem ser indesejáveis para criação do modelo de AM. Estas observações raras e indesejáveis são chamadas de *outliers*, definidas formalmente como "uma observação (ou subconjunto de observações) que parece ser inconsistente com o restante desse conjunto de dados" (BARNETT; LEWIS, 1984). Por serem inconsistentes, pode ser necessário removê-las da base de dados para criação dos modelos. Certas técnicas de AM podem ser utilizadas para identificação destas observações;
- Seleção de atributos: Em alguns casos a base de dados a ser utilizada para criação de modelos de AM contém uma grande quantidade de atributos. Ao se desenvolver um modelo estatístico levando em conta muitos atributos, tem-se um problema de viés do modelo desenvolvido. Por outro lado, um número insuficiente de atributos gera problemas de variância do modelo. Tendo em conta que estes dois problemas aumentam a taxa de erro do modelo, o *trade-off* viés-variância deve ser balanceado. Além da questão de desempenho do modelo estatístico, um número de atributos muito alto eleva o custo computacional para aplicação das técnicas de AM e prejudica a interpretação do modelo. Para lidar com estes problemas, existe uma série de técnicas e métricas que podem ser aplicadas para a seleção de atributos a serem considerados em uma aplicação de AM (JAMES et al., 2013);
- Regularização: certas técnicas de AM fazem naturalmente a seleção de atributos. Nestas técnicas, todos os atributos são considerados, porém a contribuição de atributos irrelevantes é minimizada. Para esta finalidade, associa-se um coeficiente a cada um dos atributos considerados, de modo a restringir sua influência no modelo final. Este processo é conhecido como regularização (JAMES et al., 2013);
- Gerenciamento de dados faltantes: para determinadas observações, as variáveis de entrada do modelo de AM podem não estar disponíveis, configurando o que é chamado de dado faltante. Em aplicações industriais, os dados faltantes podem ser ocasionados, por exemplo, por falhas pontuais em dispositivos de detecção (as ocorrências com dados faltantes são aleatórias) ou por problemas relacionados à sincronização destes dispositivos (as ocorrências com dados faltantes são sistemáticas, têm uma frequência bem definida). García-Laencina, Sancho-Gómez e Figueiras-Vidal (2010) identificam quatro estratégias principais para se lidar com dados faltantes: exclusão de observação que contém dado faltante; inserção de dado faltante por meio de análise estatística ou

técnicas de AM; utilização de procedimentos baseados em modelos, em que a distribuição dos dados é modelada; e a utilização do dado faltante propriamente dito, como uma nova classe ou valor.

3.4.3 Métodos e Procedimentos de Aprendizado

Uma primeira forma de se classificar técnicas de aprendizado de máquina é por meio do método de aprendizado que elas adotam. Um método é formado por um conjunto de procedimentos[16] cuja implementação é descrita por meio de técnicas[17] (ANDIAPPAN; WAN, 2020). Assim, os procedimentos também podem ser utilizados para classificação das técnicas de AM, mas em um nível de granularidade menor que a classificação com base nos métodos. Por permitir a diferenciação das técnicas de AM em diferentes níveis de granularidade, conforme ilustrado na Figura 19, tanto os métodos de aprendizado quanto os seus procedimentos são utilizados aqui para definirem "classe de técnicas".

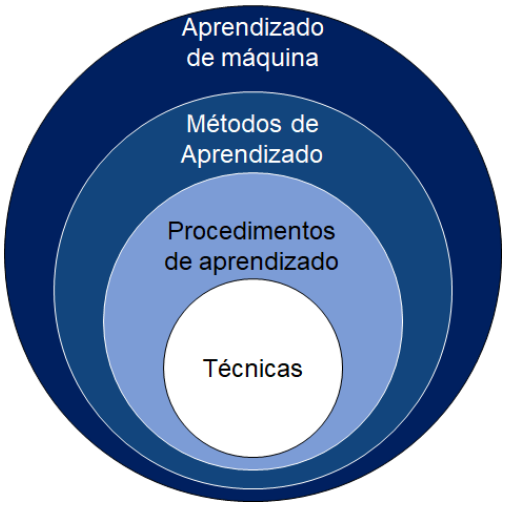


Figura 19 - Níveis de classificação de técnicas de aprendizado de máquina.

Fonte: autoria própria.

[16] Um procedimento consiste em uma atividade sistemática e racional, válida e comprovadamente verdadeira, que fornece diretrizes para aplicação de uma ou mais técnicas para realização de um trabalho (MARCONI; LAKATOS, 2003), neste caso, o aprendizado de um modelo estatístico.

[17] Uma técnica corresponde à maneira correta, hábil e segura de se utilizar ferramentas e desempenhar tarefas. Ela define o "como fazer" acerca do trabalho a ser realizado em um procedimento (ANDIAPPAN; WAN, 2020; MARCONI; LAKATOS, 2003).

Os dois primeiros métodos de aprendizado a serem apresentados diferenciam entre si no que diz respeito à presença ou ausência de *feedback* durante o processo de aprendizado. Em aprendizado de máquina, o *feedback* ocorre durante o processo de aprendizagem pela presença da variável de classe em cada observação. No caso do aprendizado supervisionado, o valor da variável de classe está presente em todas as observações. No caso do aprendizado não supervisionado, o valor da variável de classe nunca é observado (BARBER, 2012; JAMES et al., 2013). Uma forma intermediária de aprendizado em que apenas uma parcela das observações da base de dados é rotulada, chamada de semissupervisionado, também pode ser considerada (BARBER, 2012). Por fim, tem-se o aprendizado por reforço, em que o agente aprendiz interage diretamente com o ambiente e o conhecimento é criado a partir das recompensas que são oferecidas ao agente com base em suas ações em cada estado ou no próprio estado que ocupa (KAELBLING; LITTMAN; MOORE, 2020; MITCHELL, 1997).

Três métodos de aprendizado são considerados aqui: supervisionado, não supervisionado e por reforço. Seus procedimentos, conforme ilustra a Figura 20, são descritos a seguir. Em um problema prático, frequentemente várias técnicas associadas a um mesmo procedimento de aprendizado são utilizadas e os desempenhos delas para aquele problema são avaliados e comparados. Por esta razão, técnicas específicas de cada procedimento não são descritas neste trabalho. Os três métodos de aprendizado são descritos nos parágrafos seguintes.

O **aprendizado supervisionado (AS)** é um método de aprendizado de máquina amplamente utilizado em aplicações industriais (WUEST *et al*., 2016). Este tipo de aprendizado permite explorar conhecimentos de especialistas: a expertise do ser humano pode ser aproveitada na forma de rótulos para as observações. O especialista fornece um *feedback* ao algoritmo, exercendo o papel de "professor". Outra característica que leva ao amplo uso das técnicas de aprendizado supervisionado é o fato de que, apesar da natureza dos dados em aplicações industriais ser tipicamente dispersa, ela é rotulada (WUEST *et al*., 2016).

Dois tipos de procedimentos podem ser adotados para o método de aprendizado supervisionado: classificação e regressão. No primeiro caso, tem-se variáveis de resposta com valores finitos e discretos. Procedimentos de regressão são aplicados em situações em que as variáveis de classe são contínuas, pertencem ao domínio dos reais.

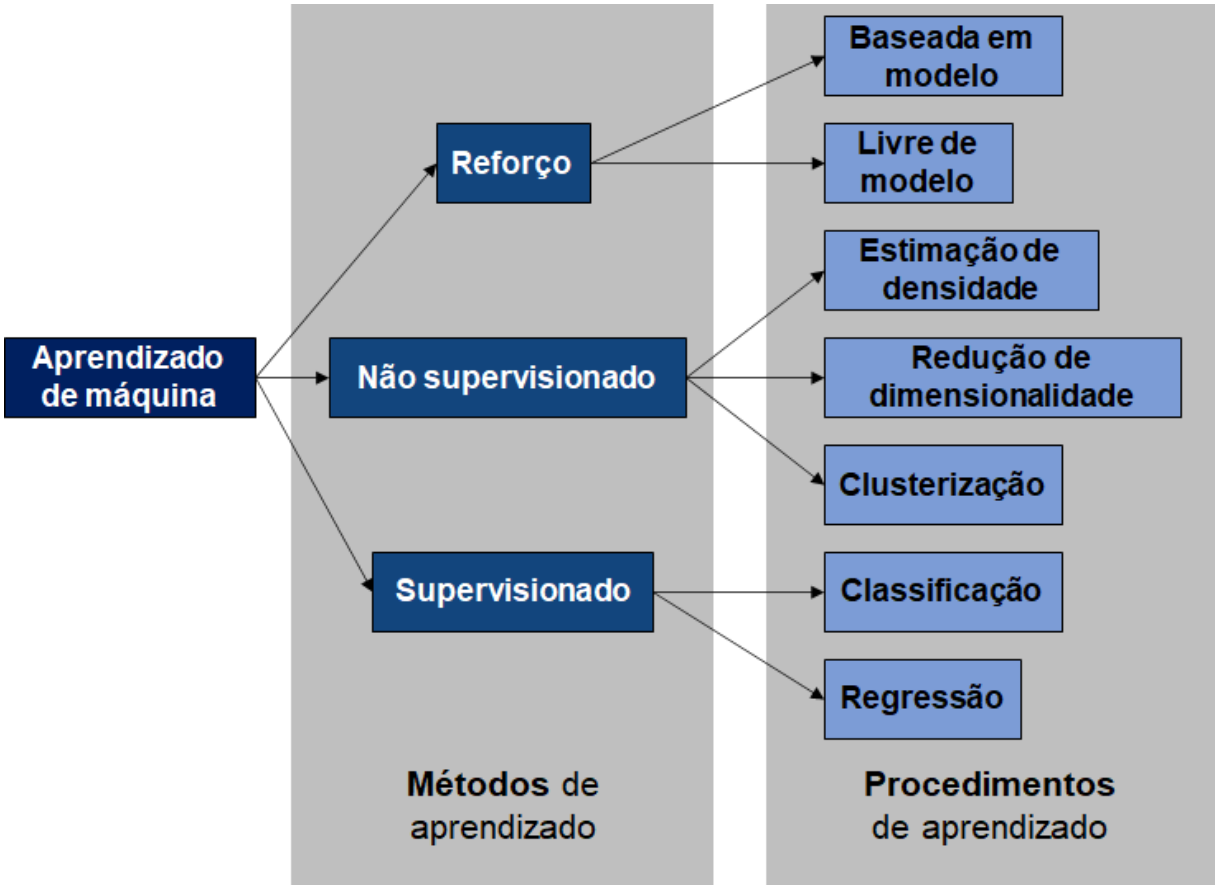


Figura 20 - Classificação de técnicas de aprendizado de máquina.

Fonte: autoria própria.

O **aprendizado não supervisionado (ANS)** se aplica em situações de ausência da variável de classe para as observações. Este tipo de aprendizado pode ser entendido como promissor, pois permite o aprendizado sem a supervisão ou intervenção humana. Ele vem sendo particularmente útil em contextos de análise de *big data*, pois facilita a identificação de padrões escondidos em um grande volume de dados. Pode-se identificar três procedimentos distintos associados ao método de aprendizado não supervisionado: agrupamento (ou clusterização, como uma tentativa de aportuguesamento do termo em inglês *clustering*), redução de dimensionalidade e estimação de densidade. "Agrupamento", de acordo com Rai e Singh (2010), consiste na divisão dos dados em grupos (chamados *clusters*) de objetos similares entre si e dissimilares entre os objetos de outros grupos. Técnicas de redução de dimensionalidade promovem a transformação de dados de muitas dimensões em uma representação com menos dimensões e mais significativa (VAN DER MAATEN; POSTMA; VAN DEN HERIK, 2009). Por fim, as técnicas de estimação de densidade são aquelas que possibilitam a estimação de uma função de densidade de probabilidade com base em dados observados.

No método de **aprendizado por reforço (AR)**, um agente aprendiz interage diretamente com o ambiente (MORALES; ZARAGOZA, 2011; WANG; USHER, 2005). O reforço que é oferecido ao agente com base no estado ocupado ou nas ações adotadas em cada estado permite que o agente explore o ambiente ao mesmo tempo em que acumula um conhecimento a respeito deste. Assim, o mecanismo de aprendizagem por reforço promove um equilíbrio entre a exploração - que promove o ganho de conhecimento por meio de ações aleatórias - e aproveitamento (*exploitation*) do conhecimento pré-existente (SUTTON; BARTO, 2020). O aprendizado por reforço é composto por:

- Um conjunto (S) de estados (s) do sistema, tal que $s \in S$;
- Um conjunto (A) de ações (a) para cada estado do sistema, tal que $a \in A$;
- Um modelo de transição de dados $T(s, a, s')$, em que uma ação a leva o sistema de um estado s a um estado s', com $s, s' \in S$; e
- Um modelo ou função de recompensa, que pode ser definida em função do estado $R(s)$ ou em função da ação adotada em um estado $R(s, a)$.

As técnicas de aprendizado por reforço se dividem em duas classes de procedimentos: baseada em modelo (*model-based*) e livre de modelo (*model-free*). No primeiro caso, tanto o modelo de transição de estados T quanto o modelo de recompensa R são aprendidos previamente por meio de experiências e posteriormente são utilizados para que as ações sejam tomadas. No segundo caso, não há a etapa de aprendizado prévio dos modelos. Tanto T quanto R são aprendidos diretamente pelo agente por meio da interação com o sistema, isto é, por tentativa e erro (DAYAN; NIV, 2008; KAELBLING; LITTMAN; MOORE, 2020; POLYDOROS; NALPANTIDIS, 2017). Graças ao uso da experiência, as técnicas baseadas em modelo são estatisticamente mais eficientes e favorecem o aproveitamento do conhecimento pré-existente. No entanto, as técnicas livres de modelo são melhor empregadas em sistemas desconhecidos e favorecem o ganho de conhecimento por meio da exploração.

3.5 Disponibilização de Dados

A interoperabilidade é um requisito fundamental para que seja possível estabelecer relacionamentos entre Componentes da I4.0 envolvidos nos serviços e processos da Fábrica Inteligente. Peter Wegner define a interoperabilidade como “a capacidade de dois ou mais componentes de *software* cooperarem apesar das diferenças de linguagem, interface e

plataforma de execução" (WEGNER, 1996). O conceito apresentado para componentes de *software* é utilizado neste trabalho para os componentes da Indústria 4.0.

A interação entre os I4.0C ocorre, inicialmente, por meio do *Asset Administration Shell*, via Camadas Funcional e de Comunicação do RAMI 4.0. As partes envolvidas no relacionamento devem ser capazes de extrair e enviar informações durante este processo, que funciona como uma negociação, antes que seja de fato estabelecido o relacionamento. Para que a troca de informações ocorra de maneira independente às tecnologias utilizadas para armazenamento dos dados na Camada de Informação, a norma DIN SPEC 91345 estabelece que a disponibilização de dados no RAMI 4.0 deve ser feita por meio de interfaces (DIN, 2016).

As interfaces são elementos presentes em arquiteturas orientadas a serviços (SOA, *service oriented architecture*). Elas permitem que os dados disponibilizados por um serviço sejam acessados (o termo "consumido" também é frequentemente empregado) por meio dos usuários deste serviço. Para que o requisito de interoperabilidade entre os I4.0C seja atendido, (BADER et al., 2020) definem interfaces padronizadas para acesso aos dados do *Asset Administration Shell*. Apresenta-se aqui os conceitos básicos de orientação à serviço, bem como as interfaces padronizadas para a I4.0.

3.5.1 Arquitetura Orientada a Serviços

Nas seções anteriores foram apresentadas as principais partes e funcionalidades que precisam ser desempenhadas para o gerenciamento de dados, isto é, o armazenamento, integração, processamento e análise de dados. Adotando uma visão de sistemas, estes componentes podem ser entendidos como componentes de um sistema que atuam em um conjunto para atingir um objetivo (OGATA, 2014). Estas partes devem ser organizadas de acordo com uma arquitetura. Uma arquitetura de sistema consiste em uma "abordagem abrangente para o desenvolvimento e instanciação das unidades funcionais que compõem um sistema" (PERREY; LYCETT, 2003).

Em sistemas industriais, o paradigma de orientação a serviços tem sido aplicado com sucesso em arquiteturas de *software* (KOMODA, 2006). Neste paradigma, as funcionalidades e capacidades das partes da arquitetura, isto é, as unidades funcionais, devem ser organizadas na forma de serviços (LASKEY; LASKEY, 2009). De uma perspectiva técnica, um serviço consiste em um componente da arquitetura capaz de realizar uma determinada tarefa (SPROTT; WILKES, 2004). Ao partir de uma visão de negócios, um serviço pode ser entendido como uma

unidade de transação por meio da qual uma demanda de um consumidor é satisfeita de acordo com um contrato negociado e preenchido pela infraestrutura de negócios (PERREY; LYCETT, 2003; SPROTT; WILKES, 2004). As perspectivas técnica e de negócios levam a discussões acerca da limitação do escopo de um serviço, a sua (in)dependência com relação ao contexto, entre outras questões. Com o intuito de alinhar as perspectivas técnica e de negócio, Perrey e Lycett (2003) propõem a seguinte abordagem:

"Os serviços técnicos são construídos em unidades lógicas de negócios sem estado que são compreensíveis, mas não têm (ou têm, de maneira limitada) conotações de processos de negócios. Eles são compostos em serviços de negócios focados no cliente externo, orquestrando-os em processos de negócios. [...]"

Assim, é possível sintetizar a função de uma arquitetura orientada a serviços: organizar a infraestrutura técnica da organização – serviços técnicos – de modo a disponibilizar serviços de negócios que possam ser criados, utilizados e atualizados de acordo com as demandas da organização (PERREY; LYCETT, 2003). A Figura 21 representa esquematicamente a criação de um serviço de negócio a partir da orquestração de unidades lógicas de negócio, que contêm os serviços técnicos e são organizadas de acordo com os processos de negócio. Os processos de negócio são representados pelas linhas que conectam as unidades lógicas de negócio. Esta abordagem contempla as perspectivas técnica e de negócios associadas ao conceito de serviço.

No paradigma de SOA, os serviços de negócios são disponibilizados e acessados por meio de interfaces (LASKEY; LASKEY, 2009; SPROTT; WILKES, 2004). As interfaces são unidades de reutilização entre os serviços e são materializadas por meio do seu mapeamento em uma ou mais interfaces de programação de aplicação (API, *application programming interface*) (BADER *et al*., 2020). Cada API é composta por uma série de operações e eventos e é implementada por meio de tecnologia e arquitetura especificadas, como por meio de uma solução que combine o protocolo HTTP (*Hypertext Transfer Protocol*) e o estilo de arquitetura de *software* REST (*Representational State Transfer*), ou o protocolo OPC UA (*Open Platform Communications - Unified Architecture*) (BADER *et al*., 2020).

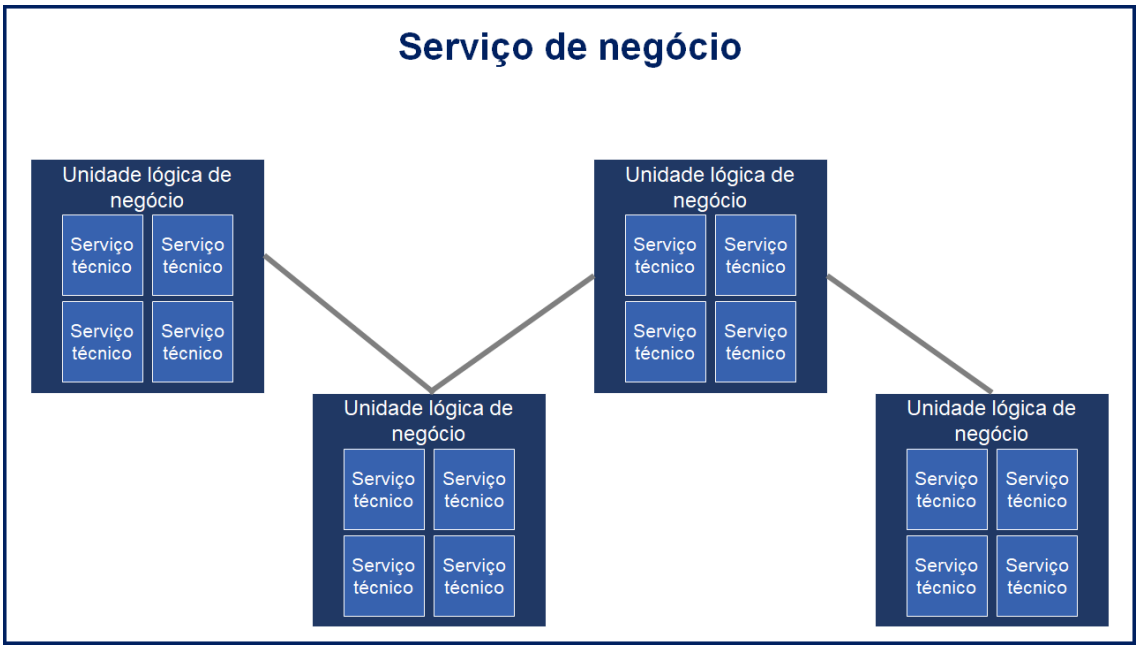


Figura 21 - Perspectivas técnica e de negócios associadas ao conceito de serviço.

Fonte: adaptado de Perrey e Lycett (2003).

3.5.2 Interfaces e Operações

No modelo de serviços da Indústria 4.0, propõe-se uma série de padronizações a fim de proporcionar a interoperabilidade entre os sistemas da I4.0 e a interação entre os ativos. A primeira delas, já apresentada, consiste na representação padronizada dos ativos no mundo digital, isto é, o AAS. Até o presente momento, foram também estabelecidas as interfaces e suas operações padrões para acesso às informações do AAS. O mapeamento destas interfaces em APIs por meio de determinadas soluções tecnológicas e estilos arquiteturais, bem como a implementação da arquitetura, encontram-se em desenvolvimento pelos grupos da Plataforma Indústria 4.0. A seguir, tem-se uma descrição a respeito das interfaces e suas respectivas operações, conforme apresentado em Bader *et al*. (2020).

A primeira interface apresentada é denominada simplesmente "*Asset Administration Shell*". Por meio desta interface, é possível acessar os elementos de um AAS, além de manipular seus submodelos como um todo, isto é, um agregado. As operações desta interface são apresentadas na Tabela 3.

A segunda interface, "Submodel", é semelhante à anterior e permite acessar os elementos de um submodelo. É importante destacar que, ainda no nível de granularidade de submodelo,

os resultados das operações ainda são agregados. A Tabela 4 apresenta as operações desta interface.

Tabela 3 - Interface "*Asset Administration Shell*".

Operação	Descrição
GetAssetAdministrationShell	Retorna o *Asset Administration Shell*
PutAssetAdministrationShell	Atualiza o *Asset Administration Shell* atual
PatchAssetAdministrationShell	Adiciona ou atualiza elementos adicionais ao *Asset Administration Shell*
PutSubmodelReference	Cria ou atualiza uma Referência de Submodelo no *Asset Administration Shell*
RemoveSubmodelReference	Remove uma Referência de Submodelo específica do *Asset Administration Shell*

Fonte: adaptado de Bader et al. (2020).

Tabela 4 - Interface "*Submodel*".

Operação	Descrição
GetSubmodel	Retorna o Submodelo
GetAllSubmodelElements	Retorna todos os Elementos de Submodelo incluindo suas hierarquias
GetAllSubmodelElementsByParentPathAndSemanticId	Retorna todos os Elementos de Submodelo do Submodelo ou do Elemento de Submodelo pai com Semantic-Id específico
GetAllSubmodelElementsBySemanticId	Retorna todos os Elementos Submodelo de um Submodelo com um Semantic-Id específico
GetSubmodelElementByPath	Retorna um Elemento de Submodelo específico do Submodelo em um caminho especificado
PutSubmodelElementByPath	Cria ou atualiza um Elemento de Submodelo existente em um caminho especificado na hierarquia de Elementos de Submodelo
SetSubmodelElementValueByPath	Define o valor do Elemento de Submodelo em um caminho especificado de acordo com o *RAW-value payload* específico do protocolo
DeleteSubmodelElementByPath	Exclui um Elemento de Submodelo em um caminho especificado dentro da hierarquia de Elementos de Submodelo
InvokeOperationSync	Invoca, de forma síncrona, uma operação em um caminho especificado com um tempo limite do cliente em ms
InvokeOperationAsync	Invoca, de forma assíncrona, uma operação em um caminho especificado com um tempo limite do cliente em ms
GetOperationAsyncResult	Retorna o OperationResult de uma operação assíncrona invocada

Fonte: adaptado de Bader et al. (2020).

A interface "*Asset Administration Shell* Serialization" possibilita a serialização do AAS, isto é, permite que o AAS seja convertido em uma estrutura ou formato que possa ser intercambiado, armazenado e reconstruído. São particularmente importantes para

implementação das Camadas de Comunicação e Funcional do RAMI 4.0. Suas operações são apresentadas na Tabela 5.

Tabela 5 - Interface "*Asset Administration Shell Serialization*"

Operação	Descrição
GetAASX	Retorna um arquivo do AASX Package apropriado com os respectivos *Asset Administration Shell*s
GetSerializationByIds	Retorna uma serialização apropriada com base no formato especificado.

Fonte: adaptado de Bader et al. (2020).

A interface "*Asset Administration Shell* Registry" permite (des)cadastrar elementos de descrição, também chamados de "descritores", de um AAS. Estes elementos contêm informações a respeito da identificação e, no caso de uma implementação distribuída, informam a localização do nó da rede onde o respectivo AAS se encontra. Esta interface possui as operações descritas na Tabela 6. Uma interface análoga, "Submodel Registry", foi proposta para os submodelos, cujas operações são apresentadas na Tabela 7.

Tabela 6 - Interface "*Asset Administration Shell* Registry".

Operação	Descrição
GetAllAssetAdministrationShellDescriptors	Retorna todos os descritores do *Asset Administration Shell*
GetAssetAdministrationShellDescriptorById	Retorna um descritor específico do *Asset Administration Shell*.
PutAssetAdministrationShellDescriptor	Cria ou atualiza um descritor de *Asset Administration Shell*.
DeleteAssetAdministrationShellDescriptorById	Deleta um descritor de *Asset Administration Shell.*

Fonte: adaptado de Bader et al. (2020).

Tabela 7 - Interface "*Submodel Registry*".

Operação	Descrição
GetAllSubmodelDescriptors	Retorna todos os descritores de Submodelo
GetSubmodelDescriptorById	Retorna um descritor específico de Submodelo
PutSubmodelDescriptor	Cria ou atualiza um descritor de Submodelo
DeleteSubmodelDescriptorById	Exclui um descritor de Submodelo

Fonte: adaptado de Bader et al. (2020).

Foram criadas também interfaces para lidar com repositórios de AAS, submodelos ou descrições conceituais. Este tipo de interface é particularmente útil quando se tem AAS ou submodelos implementados de modo distribuído entre os nós da rede e com um nó agregador que garante o acesso por parte dos clientes às entidades que se encontram distribuídas. Neste caso, os nós da rede formam repositórios que são acessados pelo nó agregador. As operações

para as interfaces “*Asset Administration Shell Repository*”, “*Submodel Repository*” e “Concept Description Repository" são apresentadas nas Tabela 8, Tabela 9 e Tabela 10, respectivamente.

Tabela 8 - Interface “*Asset Administration Shell Repository*”.

Operação	Descrição
GetAllAssetAdministrationShells	Retorna todos os *Asset Administration Shell*s
GetAllAssetAdministrationShellsById	Retorna um *Asset Administration Shell*s específico
GetAllAssetAdministrationShellsByAssetId	Retorna todos os *Asset Administration Shell*s que estão ligados a um identificador de ativos globalmente único ou ids de ativos específicos.
GetAllAssetAdministrationShellsByIdShort	Retorna todos os *Asset Administration Shell*s com um idShort específico
PutAssetAdministrationShell	Cria ou atualiza um *Asset Administration Shell*
DeleteAssetAdministrationShellById	Exclui um *Asset Administration Shell*

Fonte: adaptado de Bader et al. (2020).

Tabela 9 - Interface “*Submodel Repository*”

Operação	Descrição
GetAllSubmodels	Retorna todos os Submodelos
GetSubmodelById	Retorna um Submodelo específico
GetAllSubmodelsBySemanticId	Retorna todos os Submodelos com um SemanticId específico
GetAllSubmodelsByIdShort	Retorna todos os Submodelos com um idShort específico
PutSubmodel	Cria ou atualiza um Submodelo
DeleteSubmodelById	Exclui um Submodelo

Fonte: adaptado de Bader et al. (2020).

Tabela 10 - Interface “*Concept Description Repository*".

Operação	Descrição
GetAllConceptDescriptions	Retorna todas as Descrições Conceituais
GetConceptDescriptionById	Retorna uma Descrição Conceitual específica
GetAllConceptDescriptionsByIdShort	Retorna todas as Descrições Conceituais com um idShort específico
GetAllConceptDescriptionsByIsCaseOf	Retorna todas as Descrições Conceituais com uma Referência de IsCaseOf específica
GetAllConceptDescriptionsWithDataSpecificationReference	Retorna todas as Descrições Conceituais com uma Referência de dataSpecification específica
PutConceptDescription	Cria ou atualiza uma Descrição Conceitual
DeleteConceptDescriptionById	Exclui uma Descrição Conceitual

Fonte: adaptado de Bader et al. (2020).

Por fim, foram também criadas interfaces para publicação e descoberta de AAS e submodelos. As operações destas interfaces denominadas “*Asset Administration Shell* Basic Discovery” e “Submodel Discovery” são descritas nas Tabelas 11 e 12.

Tabela 11 - Interface "*Asset Administration Shell Basic Discovery*"

Operação	Descrição
GetAllAssetAdministrationShellIdsByAssetId	Retorna todos os Ids de *Asset Administration Shell* IDs que estão ligados a um identificador de ativo globalmente único ou ids de ativos específicos
PutAssetId	Cria ou atualiza a ligação entre o Id do *Asset Administration Shell* e o ID de ativo globalmente exclusivo

Fonte: adaptado de Bader et al. (2020).

Tabela 12 - Interface "*Submodel Discovery*".

Operação	Descrição
GetAllSubmodelIdsBySemanticId	Retorna todos os Ids de Submodelos encontrados com base em um SemanticId específico

Fonte: adaptado de Bader et al. (2020).

3.6 Síntese do Capítulo

Os tópicos apresentados neste capítulo formam os fundamentos deste estudo. Foram explorados conceitos e características do conjunto de tecnologias e métodos que são contemplados pelo termo *Big Data & Analytics*. Por conta das características de bases de dados caracterizadas como *big data*, o gerenciamento e análise destes dados exigem tecnologias e métodos específicos. Assim, foram apresentadas soluções de armazenamento, integração, processamento e análise de dados, e disponibilização de dados.

Com relação às soluções de armazenamento de dados, concentrou-se na descrição de modelos de dados relacional e não relacionais, bem como as propriedades transacionais garantidas por sistemas de bancos de dados que implementam estes modelos. Foi apresentada uma discussão sobre as vantagens e desvantagens de cada modelo, discutindo a adequação de cada um deles para i) bancos de dados de integração e de aplicação; ii) sistemas centralizados e distribuídos; e iii) conjuntos de dados que são frequentemente tratados como uma unidade, isto é, os agregados.

As soluções de armazenamento garantem uma primeira e importante etapa do processo de transformação digital, que é a digitalização de ativos. Conforme apontam Gastaldi *et al*. (2018), a etapa posterior à digitalização é, frequentemente, a integração de ativos. A análise de dados de modo a orientar as decisões geralmente pressupõe a integração de diferentes fontes de dados. Assim, tratou-se da integração de dados para fins analíticos, com foco em sistemas de

Data Warehouse. Inicialmente, foram apresentadas características que os distinguem de sistemas de banco de dados comuns. Posteriormente, foi descrita uma arquitetura genérica de *Data Warehouse*, além de seus componentes e processos. Por fim, apresentou-se o modelo de dados multidimensional, frequentemente adotado neste tipo de sistema para representar os dados em nível conceitual e realizar o mapeamento para níveis de abstração mais baixos.

Apresentou-se as técnicas de análise de dados por meio de métodos de aprendizado de máquina. O foco nesta classe de técnicas se deve ao fato de que o seu sucesso é, em grande parte, atribuído à alta disponibilidade de dados, que consiste em uma das dimensões de *big data*. Foram apresentados três diferentes métodos de aprendizado: supervisionado, não supervisionado e por reforço. Ainda, tendo em vista que uma fração significativa dos esforços despendidos em projetos de aprendizado de máquina é direcionada à preparação dos dados, algumas atividades relacionadas a esta etapa também foram apresentadas.

O último tópico foi o de disponibilização de dados. Nesse contexto, procurou-se, por meio de conceitos relacionados à arquitetura de *software*, definir a forma com que as funcionalidades técnicas do sistema são criadas e combinadas para gerar funcionalidades úteis em termos de negócio. Através do paradigma de orientação a serviços, apresentou-se o conceito de interfaces e operações, importantes para esse padrão, com exemplos de interfaces e operações padronizadas para a Indústria 4.0.

4 Resultados e Discussão

A proposta de uma infraestrutura para o tratamento de dados na I4.0 exige que seja feita a caracterização da natureza dos dados neste contexto. Tais características podem ser extraídas das dimensões de *big data*, sendo manifestadas pela estrutura do *Asset Administration Shell* e suas perspectivas de implementação. Esta caracterização é apresentada aqui e, posteriormente, discute-se como os modelos de dados de implementação abordados podem ser empregados em um contexto de I4.0. Propõe-se uma relação entre tipos de técnicas de aprendizado de máquina mais adequadas para aplicações industriais, levando em conta o ciclo de vida e a cadeia de valor de um ativo. Por fim, apresenta-se uma proposta de arquitetura para integração e análise de dados no contexto da I4.0 com base em sistemas de *Data Warehouse*.

4.1 Big Data & Analytics e Bancos de Dados na Indústria 4.0

Ainda não há consenso a respeito dos pilares tecnológicos que sustentam a Indústria 4.0. Observa-se ainda que boa parte dos autores colocam "Indústria 4.0" e "Quarta Revolução Industrial" como sinônimos, considerando que não há distinção entre os fenômenos, o que pode dificultar ainda mais a identificação dos pilares tecnológicos associados a cada um dos conceitos. A Tabela 13 apresenta a visão de diferentes autores acerca das tecnologias que habilitam a Indústria 4.0 e/ou a Quarta Revolução Industrial.

A despeito desta falta de consenso, pode-se observar que existem certas convergências entre os autores e organizações sobre quais seriam as tecnologias habilitadoras da Indústria 4.0. Observa-se, pela Tabela 13, que a única tecnologia identificada como tal em todos os trabalhos consultados foi *Big Data & Analytics*[18]. Assim, embora não haja total consenso entre os autores, pode-se reconhecer a relevância de *Big Data & Analytics* para a Indústria das próximas décadas.

Dados gerados, coletados, transmitidos e analisados possivelmente em tempo real farão parte da Fábrica Inteligente (YIN; KAYNAK, 2015). O fato de sistemas de *Big Data & Analytics* serem considerados um pilar da Indústria 4.0 advém das possibilidades de melhorias que podem ocorrer em uma empresa com base em seus dados disponíveis. Schwab (2017) e Yin, Kaynak (2015) afirmam que este tipo de sistema pode ser utilizado para o aumento de

[18] Embora *Big Data & Analytics* tenha sido colocado como uma tecnologia, o termo compreende uma série de tecnologias específicas, conforme foi apresentado na seção 3.1.

produtividade; melhor gerenciamento de riscos; redução de custos; e auxílio à tomada de decisão em geral. Por esta razão, considera-se que este conjunto de tecnologias é fundamental para a I4.0 (KAGERMANN; WAHLSTER; HELBIG, 2013).

Tabela 13 - Pilares Tecnológicos da Indústria 4.0.

Tecnologia	Rüssmann (2015)	Bechtold *et al*. (2014)	Lichtblau *et al*. (2015)	Bauer *et al*. (2015)	Petrillo *et al*. (2018)	Wan, Cai, Zhou (2015)
Big Data & Analytics	x	x	x	x	x	x
Robótica Avançada	x	x		x	x	
Integração de Sistemas	x				x	x
Internet das Coisas	x		x	x	x	x
Simulação	x				x	
Manufatura aditiva	x	x	x	x	x	
Computação em nuvem	x	x	x		x	x
Realidade virtual/ aumentada	x		x	x	x	
Cibersegurança	x				x	
Machine-to-machine		x		x		
Tecnologias móveis		x	x			
Tecnologias de localização e detecção			x			
Interfaces humano-máquina			x	x		
Autenticação e detecção de fraude			x			
Sensores inteligentes			x			
Interação com clientes			x			
Plataformas comunitárias		x				
Projeto embarcado						x
Veículos auto guiados						x
Redes sociais						x

Fonte: autoria própria.

Trabalhos atuais exploram o gerenciamento e uso dos dados no contexto da I4.0, levando em conta suas características particulares associadas ao conceito de *Big Data & Analytics*. Há

aqueles que sugerem técnicas de processamento de dados (DIEZ-OLIVAN *et al*., 2019), outros trabalhos reportam aplicações que permitiram o aperfeiçoamento de processos produtivos e logísticos e a criação de modelos de negócios a partir do uso dos dados e tecnologias digitais (ARDOLINO *et al*., 2018), há também autores que propõem arquiteturas para implementações de processamento e análise de dados (YLI-OJANPERÄ *et al*., 2019)**,** etc. Contudo, menor atenção é dada ao projeto do banco de dados a ser utilizado nas aplicações, mesmo quando se tratam de implementações do *Asset Administration Shell*, que é o formato padrão de representação e intercâmbio de informações na I4.0. Tal constatação pode ser obtida a partir de um trabalho de revisão sistemática da literatura cujos procedimentos são descritos no Apêndice B.

O foco da revisão de literatura foi nas propostas para implementações práticas do *Asset Administration Shell*. O objetivo era avaliar a quantidade de trabalhos que apresentavam uma discussão acerca das soluções de bancos de dados utilizadas. Partindo para os resultados, foi constatado que apenas 32% (8 de 25 artigos) mencionam o modelo de dados ou SGBD utilizado, o que sugere que esta questão é pouco discutida por trabalhos nesta área da I4.0. A despeito disso, considera-se que os bancos de dados não devem ser entendidos como meras ferramentas para armazenamento dos dados, mas como componentes importantes de arquiteturas, podendo impactar no seu desempenho (OLIVEIRA *et al*., 2021). Por esta razão, as escolhas referentes à solução de banco de dados não devem ser feitas arbitrariamente, mas com base em critérios, características da aplicação, dos usuários, dos dados, etc.

De acordo com o estudo conduzido aqui, deriva-se duas ponderações acerca da gestão dos dados na I4.0. A primeira é que a relevância de *Big Data & Analytics* para a I4.0 motiva o uso de suas dimensões para descrever os dados neste contexto. A segunda é que ainda há pouca discussão a respeito de soluções de bancos de dados para a I4.0. Combinando estas observações, propõe-se a caracterização qualitativa dos dados a partir das dimensões de *big data*, bem como a correlação entre estas características e as soluções de bancos de dados apresentadas na Seção 3.2. Outras características importantes para o projeto de banco de dados são fundamentalmente dependentes da aplicação e fogem ao escopo deste trabalho, que busca a abrangência de suas contribuições acerca do projeto de banco de dados para o contexto da I4.0.

4.1.1 Volume

Esta dimensão diz respeito à quantidade de dados presentes em bases de *big data*. No contexto da Indústria 4.0, a digitalização dos mais diversos ativos e a comunicação entre eles leva ao "crescimento sem precedentes" do volume de dados, segundo Tao *et al*. (2018). Wan, Cai e Zhou (2015) afirmam que bases de *big data* não podem ser manipuladas em um único computador, exigindo uma arquitetura distribuída. Assim, nota-se que a preocupação com o volume dos dados na I4.0 se reflete nas perspectivas de implementação do *Asset Administration Shell* com relação à sua distribuição. É possível argumentar que implementações centralizadas são adequadas para volumes de dados menores, enquanto as implementações distribuídas são adequadas para grandes volumes.

Conforme descrito em anteriormente, os bancos de dados relacionais apresentam limitações para lidar com grandes volumes de dados. Esta característica vai de encontro à consideração de Manyika *et al*. (2011) quanto à inadequação dos sistemas de bancos de dados tradicionais para bases de dados volumosos. Por outro lado, a necessidade de escalabilidade horizontal garantida por BDs NoSQL motivou o desenvolvimento destes sistemas e viabiliza o armazenamento e gerenciamento destas grandes massas de dados (KHINE; WANG, 2019). Assim, entende-se que sistemas NoSQL, especialmente aqueles orientados a agregados, são particularmente úteis para a I4.0 em cenários de implementação distribuída de BDs, que geralmente contém grandes volumes de dados, por facilitarem a escalabilidade horizontal.

Ainda levando em conta a questão da implementação distribuída de bancos de dados, pode-se aprofundar a discussão com relação aos modelos de dados NoSQL. Foi dito anteriormente que os agregados formam unidades naturais de distribuição. Alguns elementos do AAS podem ser entendidos como tal: o Submodelo, que corresponde a um conjunto de Elementos de Submodelo; Coleções de Elementos de Submodelo; e o próprio AAS, que forma um conjunto dos demais elementos. Assim sendo, os modelos NoSQL orientados a agregados são úteis para implementações distribuídas do AAS. Para aplicações que buscam processar conjuntos de dados como um todo, o modelo chave-valor é particularmente útil pelo fato do agregado ser opaco. Caso seja necessário acessar partes de um conjunto, os modelos de documentos e famílias de colunas são mais adequados em comparação ao modelo chave-valor, pois os agregados são transparentes no caso dos primeiros, isto é, pode-se fazer operações parciais no conjunto de dados.

4.1.2 Variedade

Esta dimensão trata dos diferentes formatos e estruturas dos dados que podem existir em um sistema de *big data*. É possível argumentar que a variedade dos dados é uma característica presente no contexto da I4.0. De acordo com Pilloni (2018), sensores são uma das principais fontes de dados de ativos industriais. Dados provenientes deste tipo de fonte podem apresentar alta heterogeneidade, tanto em termos de formato (SUN *et al*., 2020), quanto em termos de semântica (JIRKOVSKY; OBITKO; MARIK, 2017). Esta heterogeneidade pode aumentar ao se considerar o *feedback* humano (por parte de profissionais e clientes, por exemplo) como uma outra fonte de dados importante para a indústria, uma vez que a origem dos dados passa a ser outra (PILLONI, 2018). Assim, a variedade é uma característica dos dados na I4.0.

O modelo rígido dos sistemas relacionais o torna inadequado para cenários onde a heterogeneidade de dados se faz presente. Tendo em vista a necessidade de contemplar todos os ativos da I4.0, a estrutura do AAS possui uma vasta quantidade de classes (entidades) que representam cada um dos seus Elementos. A Figura 10 permite observar esta complexidade, mesmo com poucos elementos. Caso seja aplicada a normalização[19] no esquema, ele se torna ainda mais complexo. Associado à esta complexidade decorrente do mapeamento do metamodelo do AAS no esquema relacional, tem-se ainda a heterogeneidade dos ativos. Ao construir uma base composta por diferentes ativos, esta heterogeneidade pode implicar em muitos campos nulos, o que é indesejável[20].

A flexibilidade dos modelos de dados NoSQL os torna adequados para a I4.0. Tal flexibilidade possibilita o armazenamento de dados semiestruturados, que melhor caracterizam o AAS. De fato, documentos técnicos da Plataforma Indústria 4.0 (BADER *et al*., 2019) e artigos acadêmicos (CAVALIERI; SALAFIA, 2020; LANG *et al*., 2019; YE *et al*., 2020) apresentam implementações do AAS em formato XML e JSON, o que sugere que os sistemas NoSQL orientados a documentos sejam particularmente úteis, embora não sejam os únicos capazes de armazenar dados semiestruturados. Este tipo de codificação de documentos é suportado por tecnologias importantes para as Camadas de Comunicação e Funcional do RAMI

[19] Normalização é um processo que consiste em modelar os esquemas de relações para minimização da redundância e das anomalias de inserção, exclusão e modificação (ELMASRI; NAVATHE, 2005).

[20] Além de desperdiçar espaço para armazenamento, valores NULL na base de dados pode afetar o resultado de certas operações, tornando-os imprevisíveis (ELMASRI; NAVATHE, 2000).

4.0, como OPC UA (OPC FOUNDATION, 2017) e HTTP[21], por exemplo. Em síntese, os modelos de dados NoSQL se adequam à variedade característica dos dados na I4.0, possibilitando o armazenamento de registros heterogêneos em um mesmo BD. Assim, a flexibilidade dos modelos NoSQL tem a sua importância para o contexto da I4.0.

4.1.3 Velocidade

Além da alta taxa de crescimento do volume de dados, que já foi mencionada e está mais associada à velocidade de captura dos dados, pode-se também discutir a respeito da velocidade de processamento e análise. Ainda, esta dimensão é analisada a partir do conceito de disponibilidade.

Descreve-se como esta dimensão se manifesta no contexto da Indústria 4.0 e como podem orientar a escolha pelos modelos de dados. Sabe-se que a disponibilidade adquire relevância especial em sistemas distribuídos, nos quais esta propriedade deve ser analisada em conjunto com outras duas de acordo com o teorema CAP. Por isso, as perspectivas de distribuição do AAS podem ser utilizadas como ponto de partida para se analisar esta dimensão.

Considera-se três possibilidades de implementação do AAS quanto à distribuição do seu conteúdo: a implementação centralizada, distribuída com nó agregador e distribuída com acoplamento fraco. No primeiro caso, não há distribuição, se tratando de um sistema disponível e consistente. Para aplicações deste tipo, os bancos de dados relacionais são fortemente recomendados. Os outros dois casos se tratam de implementações distribuídas, em que surge o *trade-off* entre consistência e disponibilidade. No segundo caso, tem-se um único ponto de acesso aos dados, embora eles estejam distribuídos em múltiplos nós. Este nó, por ser o único a receber e processar todas requisições, pode melhor gerenciar possíveis inconsistências, mas também pode estar sujeito a sobrecarga, de modo que a consistência, possivelmente garantida pelas próprias características da implementação, pode ser relaxada em prol de maior disponibilidade. Este relaxamento é importante, pois ajuda a minimizar a sobrecarga do nó que recebe todas as requisições, respondendo-as com maior velocidade. Sistemas que implementam o modelo de garantias BASE são fortemente recomendados neste caso. Por fim, na implementação distribuída, é importante garantir a consistência dos dados, o que pode levar a

[21] JSON API - A Specification for Building APIs in JSON: www.jsonapi.org

maiores janelas de indisponibilidade, isto é, menor velocidade no processamento das operações. Neste cenário, os modelos NoSQL que implementam o modelo BASE de garantias transacionais ainda podem ser utilizados, embora os modelos que implementam as garantias ACID possam, intrinsecamente, garantir a consistência para este cenário.

4.1.4 Valor

A extração de valor dos dados envolve o emprego de técnicas de análise de dados em uma abordagem multidisciplinar. A multidisciplinaridade pode se traduzir em uma estrutura organizacional naturalmente distribuída de uma empresa ou instituição. Com relação às perspectivas de implementação do AAS, a sua distribuição em diferentes nós da rede de fato reflete melhor a estrutura das organizações atualmente (BEDENBENDER *et al*., 2017c). Nestas situações, cada uma das subdivisões de uma organização fica responsável pela fração do AAS ou pelos AAS que lhe dizem respeito. Conforme visto anteriormente, a aplicação de técnicas de análise dos dados de uma organização com o propósito de extrair valor deles exige a integração de diferentes perspectivas. Assim, a mesma abordagem multidisciplinar que pressupõe a distribuição do AAS em diferentes nós também exige que estes "fragmentos de AAS" (ou diferentes AAS) sejam integrados para extração de valor a partir de seus dados. Retomando as perspectivas de implementação do AAS quanto às suas formas de distribuição, a solução distribuída com nó agregador pode ser uma solução adequada para integração dos dados. Isto não significa que o nó agregador precisa atuar necessariamente como um nó mestre da rede, que gerencia todas as transações rotineiras, mas que este pode atuar como um nó de integração de dados e como fonte de acesso por parte daqueles membros da organização que pretendem extrair valor dos dados.

Embora a extração de valor dos dados esteja mais ligada à análise dos dados do que ao seu armazenamento, faz-se aqui um recorte desta dimensão que diz respeito aos sistemas de gerenciamento de banco de dados. Este recorte considera a interdisciplinaridade e a distribuição dos dados nas organizações como guia para o projeto de BDs. Assim, a análise da adequação dos sistemas de banco de dados para esta perspectiva "valor" é feita sobretudo levando em conta a importância da distribuição do AAS, bem como a necessidade de integração para extração de valor a partir dos dados dos ativos.

Para aprofundar esta discussão, recupera-se alguns dos conceitos apresentados sobre sistemas de *Data Warehouse* e modelo de dados multidimensional. Nós da rede, que contém apenas dados do AAS referentes à uma unidade organizacional da instituição, são aqueles que processam transações mais rotineiras, as chamadas *On-line Transaction Processing*, assim como os BD operacionais ou transacionais. Um nó da rede que promove a integração dos dados, por sua vez, processa transações com propósito analítico, *On-Line Analytical Processing*, além de fornecer dados para algoritmos e outros subsistemas, atuando de fato como um BD integrador.

Em uma instituição com estrutura organizacional distribuída, os modelos de dados que viabilizam a escalabilidade horizontal do BD, isto é, os SGBDs NoSQL orientados a agregados, são adequados para implementação dos nós "transacionais", ou seja, aqueles que processam transações rotineiras em dados específicos do AAS que dizem respeito a uma unidade da organização. Caso a distribuição do AAS não seja realizada por meio do SGBD em si, mas por meio de bancos de dados de aplicação que se comunicam por meio de interfaces de serviço, os modelos de dados NoSQL ainda são úteis. A flexibilidade proporcionada por estes modelos permite que cada departamento da organização possa adotar os modelos de dados que melhor se adequem à sua aplicação.

Um nó integrador geralmente é construído, em nível conceitual, a partir do modelo de dados multidimensional. É nele que é feita a extração de valor dos dados. O modelo multidimensional é geralmente mapeado ao nível de implementação por meio de um esquema relacional, embora existam trabalhos na literatura que buscam realizar o mapeamento do modelo multidimensional em modelos NoSQL (BICEVSKA; ODITIS, 2016; CHEVALIER *et al*., 2015b; YANGUI; NABLI; GARGOURI, 2016).

4.1.5 Veracidade

A veracidade corresponde a uma característica que deve ser preservada a fim de que a dimensão "valor" dos dados seja garantida (IVANOV; DOLGUI; SOKOLOV, 2019). Tal preocupação é observada em um contexto de I4.0, tendo em vista que alguns autores consideram como um pilar da Indústria 4.0 a segurança cibernética (ver Tabela 13), que pressupõe a proteção contra erros e modificações propositais nos dados. A preocupação com a preservação da veracidade se reflete também nas perspectivas de implementação de AAS, em que a

estratégia de virtualização afeta diretamente o isolamento entre as aplicações (MAVRIDIS; KARATZA, 2019) e, consequentemente, a confidencialidade e integridade dos dados.

Na Seção 3.1, foram apresentadas as três componentes da veracidade: i) a objetividade dos dados, que depende fundamentalmente da natureza da fonte dos dados; ii) o engano, um problema do campo da segurança cibernética; iii) e a implausibilidade. Destas dimensões, Lee (2017) aponta algumas causas para problemas de veracidade que podem ser associadas à implausibilidade, tais como inconsistência, latência e incompletude. Observa-se, portanto, que estas causas e, consequentemente, a veracidade são fundamentais para o projeto do banco de dados.

É possível relacionar as causas de implausabilidade com as propriedades do teorema CAP e, deste modo, discutir a dimensão "veracidade" para diferentes sistemas de banco de dados. A inconsistência que afeta a veracidade dos dados está diretamente ligada à consistência a qual se refere o teorema CAP. A latência está associada à propriedade da disponibilidade. A questão da incompletude, por sua vez, não é associada diretamente a uma propriedade do teorema CAP, mas à garantia transacional de atomicidade, que estabelece que uma transação deve ser realizada por completo ou não ser realizada. Assim, tem-se uma fundamentação para discutir o impacto dos modelos de dados na veracidade.

As garantias transacionais ACID contribuem para a veracidade dos dados por possibilitarem consistência e atomicidade às operações. No entanto, tais garantias implicam em alta indisponibilidade, que se traduz em *delay* nas operações. Retomando a dimensão "velocidade" de *big data*, se a velocidade de processamento dos dados obtida por meio de um sistema com garantias ACID é condizente com a velocidade de entrada dos dados no sistema, de modo que não haja processamento de dados desatualizados, então estes sistemas de bancos de dados podem ser empregados. SGBDs relacionais e orientados a grafos geralmente adotam tais garantias.

A adoção do modelo BASE de garantias transacionais promove o aumento da disponibilidade em detrimento da consistência forte. Isto implica em diminuição no *delay* a preço da possibilidade de ocorrência de inconsistências. Isso não significa que o modelo BASE é necessariamente uma escolha inadequada quando se deseja garantir a veracidade a partir da completude e consistência. Deve-se ter em mente que o modelo BASE não inviabiliza a garantia da consistência, mas permite balancear o *trade-off* entre consistência e disponibilidade de modo a melhor se adequar às necessidades da aplicação. Assim, pode-se priorizar uma das propriedades de acordo com as características da aplicação e dos problemas ligados à

implausibilidade cuja susceptibilidade de ocorrência é maior. Assim sendo, modelos de agregados são capazes de garantir a veracidade, lidando com *delay*, incompletude e inconsistência não simultaneamente, mas equilibrando os problemas de acordo com a demanda da aplicação. Em síntese, ambos os modelos ACID e BASE permitem garantir a veracidade dos dados ao preservar sua consistência. Porém, o segundo possibilita que o nível de consistência seja balanceado, de acordo com as necessidades da aplicação, de modo a não gerar longas janelas de indisponibilidade do sistema.

4.1.6 Análise Combinada

As dimensões de *big data* e outras características dos dados na I4.0 foram abordadas individualmente nas subseções anteriores para discutir a adequação dos modelos de dados a diferentes realidades da I4.0. No entanto, o projeto de banco de dados exige que interrelações entre estas características também sejam analisadas, pois é possível observar que uma dimensão pode afetar as demais no que diz respeito ao modelo de dados a ser utilizado (DE OLIVEIRA et al., 2022). As dimensões “volume” e “velocidade” (ligada a disponibilidade), por exemplo, estão correlacionadas de acordo com o teorema CAP. Ao tratar da dimensão “veracidade”, foi discutido o impacto dos modelos BASE e ACID na veracidade dos dados, porém a utilização de um ou outro modelo de garantias transacionais afeta também a distribuição dos dados que foi associada à dimensão “valor”. A dimensão variedade, que diz respeito à possibilidade de armazenar dados com estruturas mais complexas e, portanto, pressupõe a utilização de modelos de dados mais flexíveis como os orientados a agregados, por exemplo, pode afetar também na latência das requisições, sobretudo em sistemas distribuídos. Aqui, propõe-se uma análise combinada destas dimensões.

Para analisar o efeito combinado das dimensões na escolha dos modelos de dados adequados em uma determinada aplicação, é necessário fazer a distinção entre bancos de dados operacionais e analíticos. Esta classificação foi mencionada anteriormente para descrever os sistemas de *Data Warehouse*. Algumas características que distinguem estes bancos são retomadas:

- Usuários, operações e controle de concorrência: bancos de dados analíticos geralmente possuem poucos usuários que executam operações majoritariamente de leitura. Geralmente não há conflito nas operações de escrita em termos de frações dos dados. Os bancos operacionais geralmente possuem mais usuários que os analíticos. Os

usuários deste tipo de banco executam, com mais frequência, operações de escrita. Deste modo, o controle de concorrência deve ser mais rígido;

- Controle de concorrência e disponibilidade: o controle de concorrência tende a ser mais rígido nos bancos de dados operacionais para que seja garantida a consistência dos dados. Este controle de concorrência associado às transações OLTP com garantias ACID podem implicar em alta janela de indisponibilidade, conforme apontado por Stonebraker (2010). Os bancos de dados analíticos, por sua vez, por flexibilizarem o controle de concorrência, possibilitam a diminuição desta indisponibilidade;
- Disponibilidade e consistência: os bancos de dados analíticos geralmente apresentam o volume de dados alto, de modo que a distribuição do armazenamento em diferentes nós é frequentemente necessária. Neste caso, surge o *trade-off* entre disponibilidade e consistência associado ao teorema CAP. Os bancos de dados operacionais, por outro lado, podem possuir um volume muito menor de dados, possibilitando o armazenamento em um único nó. Como neste caso não há a necessidade de aguardar que todos os nós de um sistema distribuído sejam atualizados, a janela de indisponibilidade tende a ser menor.

Algumas análises qualitativas são apresentadas para sintetizar as discussões apresentadas até aqui. A primeira delas, ilustrada na Tabela 14 correlaciona as dimensões "volume", "velocidade" e "veracidade", que são associadas aos dois modelos de garantias transacionais BASE e ACID, considerando bancos de dados analíticos. Esta análise associa a velocidade à disponibilidade e a veracidade à consistência, conforme discutido nas subseções anteriores, cada um dos cenários é discutido a seguir:

Tabela 14 - Dimensões volume, velocidade e veracidade em bancos de dados analíticos.

Volume	Velocidade (disponibilidade)	Veracidade (consistência)	Modelo de garantias transacionais adequado	Cenário
Baixo	Baixa	Baixa	Ambos	1
		Alta	Ambos	2
	Alta	Baixa	BASE	3
		Alta	Ambos	4
Alto	Baixa	Baixa	Ambos	5
		Alta	Ambos	6
	Alta	Baixa	BASE	7
		Alta	Nenhum	8

Fonte: autoria própria.

- Cenário 1: nenhuma das dimensões é entendida como um requisito significativo para o projeto do banco de dados. Neste caso, considera-se que ambos os modelos de garantias transacionais podem ser adotados. Deste modo, a escolha do modelo de dados não é limitada e pode ser feita levando em conta outras exigências, como a variedade, flexibilidade de acesso e complexidade nas conexões entre os dados;
- Cenário 2: a consistência é um requisito a ser garantido. Neste cenário, não há exigência de alta disponibilidade, os dados podem ser armazenados de maneira centralizada devido ao baixo volume. Além disso, este tipo de banco de dados possui a característica de poucos usuários que executam operações de leitura, contribuindo com a consistência dos dados. Neste caso, tanto o modelo BASE quanto ACID são capazes de garantir a consistência;
- Cenário 3: há a necessidade de garantir alta disponibilidade. Para este cenário, observa-se que não há exigência de alta consistência, embora as próprias características do banco de dados analítico contribuam para tal. Sendo assim, o modelo BASE é o mais adequado, já que permite a flexibilização da consistência em prol da disponibilidade;
- Cenário 4: Com base no teorema CAP, as exigências de alta velocidade e veracidade implicam em necessidade de centralização do BD, para que não esteja sujeito à partição. Já que o volume de dados considerado é pequeno, não há problema quanto a distribuição. Para um sistema CA (*consistent and available*, consistente e disponível) como este, o modelo ACID é mais adequado e frequentemente empregado. O modelo BASE também pode ser usado, já que as características do BD analítico acabam por garantir naturalmente a consistência;
- Cenário 5: o volume é um requisito significativo no projeto do banco de dados. Considerando que não há exigência de alta disponibilidade ou consistência, não é necessário levar em conta o *trade-off* entre consistência e disponibilidade. Os modelos de dados que implementam o modelo BASE são aqueles que apresentam características que viabilizam a distribuição, sobretudo aqueles com orientação a agregados, que surgem como uma unidade natural para distribuição. No entanto, aqueles que implementam o modelo ACID podem ser igualmente empregados;
- Cenário 6: neste caso, tem-se o *trade-off* entre consistência e disponibilidade no qual a primeira propriedade deve ser priorizada. É um típico caso no qual o modelo ACID se adequa aos requisitos do banco de dados, por serem fortemente consistentes. Vale destacar que os bancos de dados deste tipo podem ser distribuídos, adequando-se ao alto

volume do cenário. No entanto, tendo em vista que este tipo de banco de dados tende a garantir a consistência por suas próprias características, o modelo BASE pode ser empregado, pois é capaz de impor a indisponibilidade até que todos os nós sejam atualizados;

- Cenário 7: neste caso, tem-se o *trade-off* entre consistência e disponibilidade no qual a segunda propriedade deve ser priorizada. É um típico caso no qual o modelo BASE melhor se adequa aos requisitos do banco de dados, por permitir a flexibilização da consistência em prol da disponibilidade;
- Cenário 8: com base no teorema CAP, este cenário não pode ter seus requisitos plenamente atendidos.

A mesma análise pode ser realizada para os bancos de dados operacionais. A síntese desta análise é apresentada na Tabela 15 e cada um dos cenários presentes é discutido na sequência.

- Cenário 1: nenhuma das dimensões é um requisito significativo para o projeto do banco de dados. Pelas mesmas razões apresentadas anteriormente, considera-se que ambos os modelos de garantias transacionais podem ser adotados. Deste modo, a escolha do modelo de dados não é limitada e pode ser feita levando em conta outros requisitos;
- Cenário 2: a consistência é um requisito a ser garantido. Neste cenário, os dados ainda podem ser armazenados de maneira centralizada devido ao baixo volume e não há exigência de alta disponibilidade. Porém, o problema da consistência para bancos de dados operacionais está também ligado ao controle de concorrência. Por esta razão, é sugerido o modelo de garantias transacionais ACID para este cenário;
- Cenário 3: há a necessidade de garantir alta disponibilidade. Para este cenário, observa-se que não há exigência de alta consistência, de modo que o modelo BASE é o mais adequado, já que permite a flexibilização da consistência em prol da disponibilidade;
- Cenário 4: com base no teorema CAP, as exigências de alta velocidade e veracidade implicam em necessidade de centralização do BD, para que não esteja sujeito à partição. Já que o volume de dados considerado é pequeno, não há problema quanto a distribuição. Para um sistema CA como este, o modelo ACID é mais adequado e frequentemente empregado. Pelo fato de que os banco de dados operacionais requerem maior controle de concorrência, opta-se por não incluir o modelo de garantias ACID neste cenário;

- Cenário 5: o volume é a dimensão que surge como um requisito significativo no projeto do banco de dados. Considerando que não há exigência de alta disponibilidade ou consistência, não é necessário levar em conta o *trade-off* entre consistência e disponibilidade. Os modelos de dados que implementam o modelo BASE são aqueles que apresentam características que viabilizam a distribuição, sobretudo aqueles com orientação a agregados, que surgem como uma unidade natural para distribuição. No entanto, aqueles que implementam o modelo ACID podem ser igualmente empregados.
- Cenário 6: neste caso, tem-se o *trade-off* entre consistência e disponibilidade no qual a primeira propriedade deve ser priorizada. É um caso no qual o modelo ACID melhor se adequa aos requisitos do banco de dados, por serem fortemente consistentes. Vale destacar que os bancos de dados deste tipo podem ser distribuídos, adequando-se ao alto volume do cenário;
- Cenário 7: neste caso, tem-se o *trade-off* entre consistência e disponibilidade no qual a segunda propriedade deve ser priorizada. É um caso no qual o modelo BASE melhor se adequa aos requisitos do banco de dados, por permitir a flexibilização da consistência em prol da disponibilidade;
- Cenário 8: com base no teorema CAP, este cenário não pode ter seus requisitos plenamente atendidos.

Tabela 15 - Dimensões volume, velocidade e veracidade em bancos de dados operacionais.

Volume	Velocidade (disponibilidade)	Veracidade (consistência)	Modelo de garantias transacionais adequado	Cenário
Baixo	Baixa	Baixa	Ambos	1
		Alta	ACID	2
	Alta	Baixa	BASE	3
		Alta	ACID	4
Alto	Baixa	Baixa	Ambos	5
		Alta	ACID	6
	Alta	Baixa	BASE	7
		Alta	Nenhum	8

Fonte: autoria própria.

As Tabelas 14 e 15 não fazem referência a um ou mais modelos de dados específicos para cada cenário. Com relação aos bancos de dados que implementam o modelo de garantias transacionais ACID, a comparação envolve os modelos de grafos e relacional. Ao analisar o

trabalho de (BATRA; TYAGI, 2012) que realizam esta comparação, observa-se que duas características que diferenciam estes bancos são a variedade – associada à flexibilidade do banco – e a complexidade das conexões (relacionamentos) entre os dados. Estas duas propriedades são também levadas em conta na comparação estabelecida por NIST Big Data Public Working Group (2019). Outros trabalhos relevantes, como o de Vicknair *et al*. (2010) levam em conta outros fatores na comparação que são mais ligados à aplicação, como o tipo de consultas e particularidades dos SGBDs. Estas e outras propriedades não são consideradas neste trabalho, no qual se busca a abrangência de suas contribuições.

Para um cenário de alta complexidade dos relacionamentos entre os dados, o modelo de grafos é superior ao relacional. A primeira razão é que a modelagem dos dados no modelo de grafos é feita de maneira mais natural e direta (ANGLES; GUTIERREZ, 2008), além de não haver limitações no número de relacionamentos entre dois nós (duas entidades). Ao mapear os relacionamentos em um esquema relacional rígido, a tarefa de incluir novos relacionamentos entre entidades no modelo relacional exige uma reestruturação significativa no esquema, o que não ocorre no modelo de grafos (SADALAGE; FOWLER, 2013).

A ausência de um esquema fixo pode, em um primeiro momento, surgir como uma vantagem do modelo de grafos em comparação ao relacional em termos de variedade. Vicknair *et al*. (2010) apontam que realizar alterações no esquema é uma tarefa muito mais significativa no modelo relacional. No entanto, os autores indicam que tanto o modelo relacional quanto o de grafos podem operar com alta eficiência para o cenário em que for projetado. Em um cenário em que a flexibilidade não esteja ligada à possibilidade de alterar conexões entre os dados, mas às entidades em si, o modelo de grafos não apresenta vantagens significativas frente ao modelo relacional. Os modelos de grafos não são recomendados, por exemplo: para cenários em que as conexões entre os dados sejam simples e as operações mais frequentes da aplicação não estejam associadas aos relacionamentos, mas às entidades em si; em situações em que múltiplas entidades sejam alteradas por uma atualização no valor de um atributo; para cenários em que a heterogeneidade, apesar de existente, é prevista desde o projeto do banco de dados, de modo que o esquema pode se manter fixo, apesar de complexo; entre outros cenários. Considerando a variedade (flexibilidade) e a complexidade das conexões (relacionamentos) entre os dados, a Tabela 16 estabelece recomendações de modelos de dados para cenários em que o modelo ACID melhor se adequa.

Tabela 16 - Veracidade, complexidade de conexão de dados e modelo ACID.

Variedade (flexibilidade)	Complexidade de conexão (relacionamentos)	Modelo de dados lógico adequado	Cenário
Baixo	Baixo	Relacional	1
	Alto	Grafos	2
Alto	Baixo	Ambos	3
	Alto	Grafos	4

Fonte: autoria própria.

- Cenário 1: tendo em vista que o cenário possibilita a existência de um esquema fixo e com relacionamentos simples, não existem razões que justifiquem o uso do modelo de grafos com base apenas nas duas características apresentadas. Diferentemente da discussão acerca do modelo de garantias transacionais, em que se optou por não restringir as escolhas, nesta análise, indica-se a opção que possivelmente seja mais otimizada;
- Cenário 2: ainda que o esquema possa se manter fixo devido à baixa variedade, o modelo de grafos se apresenta como vantajoso devido à possibilidade de representar relacionamentos complexos entre entidades;
- Cenário 3: caso a variedade não esteja necessariamente associada a alterações de esquema, o modelo relacional pode atender satisfatoriamente os requisitos da aplicação. Se, por outro lado, a flexibilidade esteja associada a alterações no esquema, sobretudo se estas modificações acontecerem nos relacionamentos (apesar da baixa complexidade de conexão entre os dados), a utilização do modelo de grafos pode ser vantajosa;
- Cenário 4: pela ausência de um esquema fixo e a possibilidade de representar relacionamentos complexos, o modelo de grafos é a opção adequada para este cenário.

A mesma análise pode ser feita para os modelos de dados com propriedades transacionais BASE. Neste caso, a flexibilidade não é um fator determinante para a comparação. As principais diferenças surgem na opacidade do agregado e na facilidade para representar relacionamentos complexos. Comparações propostas por Gessert *et al*. (2017) e Lourenço *et al*. (2015) levam em conta propriedades como consistência e disponibilidade, tratadas anteriormente neste trabalho e que também podem ser levadas em conta na escolha do modelo de dados. A comparação também pode levar em conta o tipo de consultas, como feito por Guzzi, Veltri e Cannataro (2014), que não são abordadas neste trabalho. A Tabela 17 apresenta a comparação entre os modelos de dados chave-valor, documentos e famílias de colunas com

base na flexibilidade de acesso – ligada à opacidade do agregado – e complexidade de conexão entre os dados. Cada um dos cenários é discutido na sequência:

- Cenários 1 e 2: estes cenários sugerem a ocorrência de operações mais simples, que possam recuperar o conteúdo do agregado como um todo e não frações deste agregado, mesmo no cenário 2, em que a complexidade das conexões é alta. Nestes casos, o modelo chave-valor pode executar operações com extrema eficiência e simplicidade;
- Cenário 3: a alta flexibilidade de acesso aos dados inviabiliza o uso do modelo chave-valor neste cenário. Tendo em vista a baixa complexidade na conexão entre os dados, a escolha entre famílias de colunas ou documentos pode ser feita com base em critérios específicos da aplicação;
- Cenário 4: a representação de relacionamentos complexos entre os dados pode exigir a utilização de supercolunas no modelo de famílias de colunas, que é uma funcionalidade nem sempre presente nos SGBDs. No caso do modelo de documentos, é possível criar e reproduzir estrutura semelhante sem limitações, o que o torna mais adequado para representar relacionamentos complexos.

Tabela 17 - Flexibilidade de acesso, complexidade de conexão de dados e modelos BASE.

Flexibilidade de acesso (opacidade do agregado)	Complexidade de conexão (relacionamentos)	Modelo de dados lógico adequado	Cenário
Baixo	Baixo	Chave-valor	1
	Alto	Chave-valor	2
Alto	Baixo	Famílias de colunas e Documentos	3
	Alto	Documentos	4

Fonte: autoria própria.

É possível observar que, ao se levar em conta as especificações de uma dada aplicação ao longo das dimensões, a escolha por um banco de dados gera *trade-offs* em termos das exigências que podem ser atendidas. Em determinadas aplicações, características conflitantes do ponto de vista de banco de dados podem ser igualmente importantes. Por esta razão, é comum encontrar aplicações, sobretudo em arquiteturas orientadas a serviço, em que múltiplos bancos de dados são utilizados para atender satisfatoriamente as diferentes especificações da aplicação. Trabalhos nesta área são referidos como persistência poliglota (OLIVEIRA *et al*., 2021; SADALAGE; FOWLER, 2013), em que cada banco de dados fica responsável por gerenciar dados de uma parte da aplicação.

4.2 Aprendizado de Máquina na Indústria 4.0

No Capítulo 3 foram apresentadas as classes de técnicas de aprendizado de máquina. As técnicas foram diferenciadas com base nos métodos de aprendizado (supervisionado, não supervisionado e por reforço) e em seus respectivos procedimentos. A literatura reporta uma série de aplicações industriais destas técnicas, incluindo trabalhos de revisão. Geralmente, os resultados destes trabalhos são descritos com base 1) nos métodos e procedimentos de aprendizado; 2) em exemplos de aplicações industriais propriamente ditos destas técnicas; ou 3) em ambos os paradigmas, isto é, métodos e aplicações. No entanto, apenas identificar as técnicas e aplicações industriais não é suficiente para representar o uso de AM na Indústria 4.0, nem para orientar as escolhas entre estas técnicas, contribuindo para a difusão de AM na indústria. Por esta razão, aqui é proposto um mapeamento das técnicas e aplicações de AM no RAMI 4.0.

A descrição do eixo vertical (eixos das camadas) do RAMI 4.0 permite observar que o processamento e análise dos dados de ativos acontece na Camada de Informação. Deste modo, do ponto de vista tecnológico, é na Camada de Informação que residem os algoritmos de AM. Com relação ao eixo "Ciclo de Vida e Cadeia de Valor", em cada uma das fases e etapas deste eixo, dados diferentes a respeito do ativo são gerados e podem ser aproveitados para extração de valor. Por fim, no eixo "Níveis de hierarquia", a função dos ativos na hierarquia da organização é representada. Aqui se considera que cada um dos sete níveis é composto por uma parte estática e uma parte dinâmica. A parte estática do ativo leva em conta as suas características estruturais, enquanto a parte dinâmica considera os processos associados ao ativo, bem como sua dinâmica interna. Assim, as análises a serem apresentadas, que consideram a diferenciação entre parte estática e dinâmica do ativo, estendem-se às sete categorias hierárquicas, não sendo necessária a diferenciação funcional dos ativos. Com isso, é possível identificar possibilidades de utilização de AM para promover melhorias nas partes estática e dinâmica de ativos de acordo com a sua projeção no eixo "Ciclo de Vida e Cadeia de Valor".

A estratégia adotada para mapear as técnicas e aplicações de AM no eixo "Ciclo de Vida e Cadeia de Valor" do RAMI 4.0 é representada esquematicamente na Figura 22. Ela foi adotada com base naquilo que foi descrito nos parágrafos anteriores. Por um lado, é possível identificar, para cada tipo de técnica de AM, a natureza dos dados envolvidos e as aplicações industriais destas técnicas, como indicam as setas que partem da caixa "Técnicas de aprendizado de máquina" para as caixas "Natureza dos dados" e "Aplicação". Isso pode ser feito sem nenhuma

referência ao eixo “Ciclo de Vida e Cadeia de Valor” do RAMI 4.0. Por outro lado, é possível identificar, para cada fase e etapa deste eixo, a natureza dos dados gerados e as possibilidades de implementação de melhorias na estática e dinâmica de ativos por meio deAM, conforme representado pelas setas que partem da caixa “Ciclo de Vida e Cadeia de Valor” para as caixas “Natureza dos dados” e “Aplicação”. Isto pode ser feito sem qualquer referência às técnicas de AM empregadas. A contribuição decorrente desta análise é a combinação destas duas perspectivas, conforme indicado pela seta cinza ao centro da Figura 22, resultando no mapeamento das possíveis aplicações industriais de técnicas de AM nas fases/etapas do eixo “Ciclo de Vida e Cadeia de Valor”, considerando as partes estática e dinâmica dos ativos.

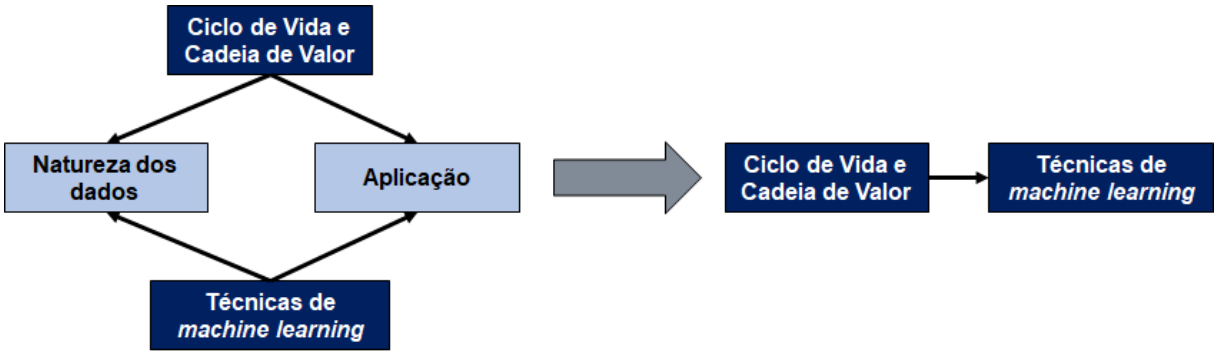


Figura 22 – Estratégia para mapeamento de técnicas de AM no RAMI 4.0.

Fonte: autoria própria.

Nesta Subseção, a forma de diferenciação das técnicas de AM com base nos métodos de aprendizado é recuperada. A seguir, tem-se uma descrição da natureza dos dados, bem como exemplos de aplicações industriais para cada tipo de técnica de AM:

Aprendizado supervisionado: ocorre quando a variável de classe é observada em todos os registros da base de dados. A diferenciação entre os procedimentos de aprendizado supervisionado se dá com base na natureza desta variável. Quando ela assume apenas valores discretos em um intervalo finito, aplica-se procedimentos de classificação e, quando esta variável admite valores reais, utiliza-se procedimentos de regressão.

Alguns exemplos de variáveis comumente encontradas em aplicações industriais cuja natureza é aquela envolvida em técnicas de classificação são: estados discretos de um sistema (operação normal, falha, tipo de falha); sinais de dispositivos de detecção discretos (chaves-limite, termostatos, pressostatos, etc.); regras e decisões (alocação de recursos, seleção de equipamentos, regras de despacho, etc.); classes de ativos, etc. Técnicas que adotam

procedimentos de classificação são comumente utilizados em aplicações de classificação de ativos baseado em atributos estáticos do mesmo (*feature-based classification*) (CHUNG; KUSIAK, 1994; KAPARTHI; SURESH, 1991; LEE; FISCHER, 1999); diagnóstico de qualidade (LEE; CHEON; KIM, 2017; RIBEIRO, 2005; YU; XI; ZHOU, 2008); escalonamento de operações (LI; OLAFSSON, 2005; SHAHZAD; MEBARKI, 2010; SHAW; PARK; RAMAN, 1992; ZHANG et al., 2017); alocação de recursos (KNAPP; WANG, 1992; ROMEO *et al*., 2020); e diagnóstico e prognóstico de condições (KOLOKAS *et al*., 2018; KRATSCH *et al*., 2020; MEIDAN *et al*., 2011; PANG *et al*., 2011; PRYTZ *et al*., 2015; SRINIVASAN *et al*., 2005; SU; HUANG, 2018; WUEST; IRGENS; THOBEN, 2014).

Entre os exemplos de variáveis pertencentes ao domínio dos reais em aplicações industriais, tem-se: sinais de dispositivos de detecção contínuos (transmissores de pressão, vazão, temperatura, velocidade); índices de desempenho (custo e tempo de execução, eficiência energética); e propriedades geométricas (dimensões, tolerâncias). Entre os exemplos de aplicações industriais de técnicas que adotam procedimentos de regressão, tem-se: projeto baseado em especificações (*specification-based design*) (CHABOT; BROWN, 1994; ROMEO *et al*., 2020; SIM; DUFFY, 1998; WHITEHALL; LU; STEPP, 1990); modelagem de processos (ANDERSEN *et al*., 1990; BHATTACHARYA; PRICE; SOLOMATINE, 2007; KANEKO; FUNATSU, 2014); escalonamento de processos (SAYADI *et al*., 2017; SCHMIDT, 1998); e prognóstico de condições (ELANGOVAN *et al*., 2015; LI *et al*., 2019; MANSOURI *et al*., 2017; MATHEW *et al*., 2018).

Aprendizado não supervisionado: método de aprendizado adotado quando não há o valor da variável de classe para os registros da base de dados. Os procedimentos pertencentes a este método são: agrupamento, redução de dimensionalidade e estimação de densidade. Diferentemente dos procedimentos envolvidos em aprendizado supervisionado, estes não se diferenciam com base na natureza dos dados envolvidos, mas no propósito da aplicação.

Como exemplo de aplicação de técnicas que implementam o procedimento de agrupamento, tem-se: o agrupamento de ativos baseado em seus atributos estáticos (*feature-based grouping*) (CHAN *et al*., 2019; DIETTERICH; MICHALSKI, 1983; LEE; FISCHER, 1999); monitoração de condições dinâmica (KHEDIRI; WEIHS; LIMAM, 2012; KU; STORER; GEORGAKIS, 1995; YU, 2013) e estáticas (AMRUTHNATH; GUPTA, 2018; DIEZ *et al*., 2016; SOUALHI; CLERC; RAZIK, 2013) de ativos; além de pré-processamento, visualização de dados e seleção de dados (ANKERST; BERCHTOLD; KEIM, 1998; CAI;

ZHANG; HE, 2010; DJENOURI; SRIVASTAVA; LIN, 2021; DY; BRODLEY, 2000; MITRA *et al*., 2002; QIAN; ZHAI, 2013; SINGHAL; JENA, 2013). Técnicas que adotam o procedimento de redução de dimensionalidade são comumente empregadas em: projeto baseado em especificações (MURDOCH; BALL, 1996) e seleção e visualização de dados (BRO; SMILDE, 2014; FAMILI *et al*., 1997; HAMEL, 2006; YAN *et al*., 2006). Por fim, técnicas que realizam a estimação de densidade são aplicadas em: monitoração de qualidade (MATUSZYK; CARDEW-HALL; ROLFE, 2010; YAO; SHAO; GE, 2019) e de condições estáticas e dinâmicas de ativos (LEE; YOO; LEE, 2004; YOO *et al*., 2004; ZHANG; QIN, 2007); além de processamento e visualização de dados (SCHUBERT; ZIMEK; KRIEGEL, 2014; TANG; HE, 2017; WAHID; ANNAVARAPU, 2021).

Aprendizado por reforço: método de aprendizado que se dá com a interação entre o agente aprendiz e o ambiente. A maioria das aplicações industriais de aprendizado por reforço são para fins de controle e escalonamento de processos. Quando apenas se conhece os estados do sistema (geralmente identificados por meio de dispositivos de detecção) e as regras (de despacho, por exemplo) ou ações (geralmente implementadas por meio de dispositivos de atuação) que podem ser adotadas para a evolução dos estados, técnicas *model-free* são comumente empregadas (ADAM; BUŞONIU; BABUŠKA, 2012; FAN; YANG, 2016; GABEL; RIEDMILLER, 2008, 2012; GASKETT, 2002; KHAN *et al*., 2012; LIU *et al*., 2019; SCHAAL, 1997; SUTTON; BARTO; WILLIAMS, 1992; WANG; USHER, 2005; WASCHNECK *et al*., 2018). Quando se conhece a dinâmica que descreve a evolução dos estados do sistema e as recompensas (geralmente expressas na forma de custo ou tempo de execução) associadas à cada ação ou a cada estado, utiliza-se técnicas *model-based* (ADAM; BUŞONIU; BABUŠKA, 2012; AISSANI; BELDJILALI; TRENTESAUX, 2009; PARK *et al*., 2020; SINGH *et al*., 2000; ZHANG *et al*., 2016, 2011; ZHU *et al*., 2020).

Apresenta-se a seguir a descrição dos dados gerados e das possíveis aplicações de AM ao longo do eixo “Ciclo de Vida” e “Cadeia de Valor” do RAMI 4.0. Esta apresentação é feita sem a referência às técnicas de AM propriamente ditas, levando-se em conta apenas as aplicações. Além disso, introduz-se também a diferenciação entre aplicações que se destinam a parte estática e dinâmica de ativos.

Fase “Tipo”, etapa “Desenvolvimento”: nesta etapa, a parte estática e a dinâmica de um ativo são idealizadas, uma primeira versão de projeto é elaborada e um protótipo é criado.

Por esta razão, não há ainda nesta etapa dados reais do ativo. Para que técnicas de AM auxiliem o desenvolvimento do projeto e protótipo, é necessária a utilização de dados de ativos análogos.

Uma aplicação de AM na etapa em questão consiste no projeto baseado em agrupamento (GUO *et al*., 2021; MONOSTORI *et al*., 1996; SIM; DUFFY, 1998): com base nas características estruturais de um ativo a ser produzido (como formas geométricas ou outros atributos, por exemplo), pode-se associá-lo a uma família de ativos correlatos. Em certos casos, é razoável esperar que o comportamento deste ativo seja semelhante ao dos ativos de mesma família. Caso este "comportamento" precise ser efetivamente levado em conta, ele pode ser especificado por meio de métricas, de modo que o projeto estático e/ou dinâmico de um ativo pode ser feito com base nestas especificações (GRECU; BROWN, 1996; SIM; DUFFY, 1998).

A estratégia descrita no parágrafo anterior é orientada à aplicação do ativo, isto é, leva em conta as características estáticas e dinâmicas desejadas do ativo expressas por métricas de desempenho (eficiência energética, capacidades, etc.). Com base nestas especificações, observa-se ativos correlatos cuja estática e/ou dinâmica são capazes de atendê-las. Para isto, utiliza-se parâmetros que caracterizam a estática e dinâmica do projeto destes ativos.

Como exemplo da aplicação mencionada no parágrafo anterior, pode-se imaginar uma peça metálica a ser submetida a um processo de usinagem. Com base nas especificações como tolerância ou rugosidade, pode-se determinar os parâmetros do processo de usinagem. Por fim, uma terceira aplicação comum nesta etapa consiste na modelagem dinâmica de sistemas de variáveis contínuas, que geralmente busca obter uma função de transferência para o processo (ANDERSEN *et al*., 1990; BHATTACHARYA; PRICE; SOLOMATINE, 2007; KANEKO; FUNATSU, 2014).

Fase "Tipo", etapa "Manutenção e Uso": consiste na etapa em que a primeira versão do projeto de ativo é de fato utilizada. Nesta etapa, dados do ativo já estão disponíveis e podem ser utilizados para atualizações no projeto, gerando novas versões. Com exceção de agrupamento em famílias de ativos, as aplicações de AM nesta etapa são iguais às da etapa anterior. A diferença é que dados reais do próprio ativo e não de ativos correlatos são utilizados para criação dos modelos de AM.

Fase "Instância", etapa "Produção": corresponde à criação de instâncias de um ativo com base em um "Tipo". Os dados envolvidos nesta etapa estão geralmente associados ao processo produtivo em si, à logística e a aspectos de qualidade. Em termos da parte estática de ativos, as aplicações de AM frequentemente se destinam à monitoração (GE *et al*., 2017; KULCSÁR *et al*., 2016) e diagnóstico (GE *et al*., 2017) de qualidade, utilizando atributos

estáticos obtidos por meio de testes e, eventualmente, rótulos que caracterizam o ativo como adequado/aprovado ou não, ou ainda identificam determinado tipo de falha.

Em termos da dinâmica de ativos, isto é, dos processos associados a ele, encontra-se com frequência na literatura aplicações de AM que promovem o escalonamento *offline* do processo produtivo ou logístico (AYTUG *et al*., 1994; PRIORE *et al*., 2014). Estas aplicações utilizam características do sistema (estados discretos, número de recursos e/ou servidores, tamanho de fila, parâmetros da dinâmica interna, regras/ações/decisões que podem ser adotadas, entre outras), características das tarefas a serem executadas e resultados do processo (tempo de processamento, tempo de completação, etc.).

Fase "Instância", etapa "Manutenção e Uso": compreende o intervalo entre o momento em que as instâncias dos ativos são atendidas pelo fornecedor e colocadas em operação por parte do cliente até o momento do descomissionamento e descarte. Durante esta etapa, é desejável que o ativo funcione da maneira para a qual foi projetado. Aplicações típicas de AM realizam a monitoração (DIEZ-OLIVAN *et al*., 2019; HANSSON *et al*., 2016; WUEST *et al*., 2016), diagnóstico (AZADEH *et al*., 2013; SRINIVASAN *et al*., 2005; WUEST; IRGENS; THOBEN, 2014) e prognóstico (CARVALHO *et al*., 2019; DIEZ-OLIVAN *et al*., 2019) das condições estáticas e dinâmicas dos ativos. No caso de monitoração, os modelos de AM são utilizados para identificar comportamentos anômalos, que resultam em *outliers* na base de dados.

Para fins de diagnóstico, é necessário não apenas diferenciar os atributos estáticos ou do comportamento de um ativo, mas saber caracterizá-los, por exemplo, como correspondendo a operação normal ou falha. Para isso, estes comportamentos são mapeados em diferentes estados do sistema. Aplicações de AM para prognóstico buscam prever comportamentos, ou seja, resultados e desempenho de processos e características da estática de ativos. As variáveis que caracterizam estas condições podem ser discretas (para prognóstico de falhas, por exemplo) ou contínuas (para prognóstico de eficiência energética ou capacidade, por exemplo).

Dados do ativo em funcionamento podem ser utilizados para garantir o cumprimento de suas funcionalidades. Assim, técnicas de AM são frequentemente aplicadas para fins de controle de processos (KAELBLING; LITTMAN; MOORE, 2020; KHAN *et al*., 2012). Além de dados provenientes de dispositivos de detecção, estes algoritmos podem se beneficiar ou não de um modelo dinâmico do sistema.

Outra aplicação de AM encontrada na literatura e que se dedica à dinâmica de ativos consiste no escalonamento em tempo real dos mesmos (KUHNLE *et al*., 2021; WANG;

USHER, 2005; WASCHNECK *et al*., 2018; ZHU *et al*., 2020). Estas aplicações fazem uso de variáveis da mesma natureza daquelas utilizadas durante o escalonamento *offline* durante a etapa de "Produção" desta mesma fase, com a diferença de que estes dados são consumidos na medida em que são gerados.

Com base no que foi apresentado, é possível correlacionar técnicas de AM com as fases e etapas do eixo "Ciclo de Vida e Cadeia de Valor" do RAMI 4.0, levando em conta as partes estática e dinâmica de ativos. Esta correlação é apresentada da seguinte maneira: para cada região do eixo em questão são apresentadas as aplicações de AM listadas, os dados associados à aplicação, a classe de técnicas de AM adequada para aquela aplicação e aquele tipo de dado, além de exemplos encontrados na literatura. Nas Tabelas 18 a 21, as siglas ANS, AS e AR são utilizadas para aprendizado não supervisionado, supervisionado e por reforço, respectivamente.

Tabela 18 - Aprendizado de máquina na fase "Tipo", etapa "Desenvolvimento".

Aplicação	**Dados**	**Classe de técnica de AM**
Agrupamento baseado em atributos (Estática)	Somente atributos-chave do projeto (formas, características geométricas, atributos da máquina)	ANS – Agrupamento
	Atributos-chave do projeto; e famílias de produtos	AS – Classificação
Projeto baseado em especificação (Estática e dinâmica)	Atributos-chave do produto; e métricas contínuas de desempenho	ANS – Redução de dimensionalidade
		AS – Regressão
	Tolerâncias de projeto; e registros de falha	AS – Classificação
	Especificações-chave de projeto; e parâmetros operacionais contínuos dos processos	AS – Regressão
	Especificações-chave do projeto; e processos, métodos e equipamentos necessários	AS – Classificação
Modelagem (Dinâmica)	Parâmetros de processo; e resultados e métricas contínuas de desempenho	AS – Regressão

Fonte: autoria própria.

Tabela 19 - Aprendizado de máquina na fase "Tipo", etapa "Manutenção e Uso".

Aplicação	**Dados**	**Classe de técnica de AM**
Projeto baseado em especificação (Estática e dinâmica)	Atributos-chave do produto; e métricas contínuas de desempenho	ANS - Redução de dimensionalidade
		AS - Regressão
	Tolerâncias de projeto; e registros de falha	AS - Regressão
	Especificações-chave de projeto; e parâmetros operacionais contínuos dos processos	AS - Regressão
	Especificações-chave do projeto; e processos, métodos e equipamentos necessários	AS - Classificação
Modelagem (Dinâmica)	Parâmetros de processo; e resultados e métricas contínuas de desempenho	AS - Regressão

Fonte: autoria própria.

Tabela 20 - Aprendizado de máquina na fase “Instância”, etapa “Produção”.

Aplicação	**Dados**	**Classe de técnica de AM**
Monitoramento de qualidade (Estática)	Somente características do produto	ANS - Agrupamento
		ANS - Estimação de densidade
Diagnóstico de qualidade (Estática)	Parâmetros do processo; e estados discretos do produto (incluindo estados de sucesso/falha e tipos de falha)	AS - Classificação
	Características do produto; e estados discretos do produto	AS - Classificação
Escalonamento offline (Dinâmica)	Características do sistema (número de recursos, tamanho do *buffer*, etc.); resultados do processo (tempo de processamento, tempo de conclusão, etc.); características da tarefa; e decisões / regras discretas (regras de despacho, alocação de recursos)	AS - Classificação
	Características do sistema; características da tarefa; e parâmetros e resultados contínuos do processo	AS - Regressão
	Estados do Sistema; regras e decisões; e modelos do sistema (transição e recompensa)	AR - Baseado em modelo

Fonte: autoria própria.

Tabela 21 - Aprendizado de máquina na fase “Instância”, etapa “Manutenção e Uso”.

Aplicação	**Dados**	**Classe de técnica de AM**
Monitoramento de condição (Estática e dinâmica)	Somente parâmetros do processo	ANS - Agrupamento
		ANS - Estimação de densidade
	Somente características do produto	ANS - Agrupamento
		ANS - Estimação de densidade
Diagnóstico de condição (Estática e dinâmica)	Parâmetros do processo; e estados discretos de processo (incluindo sucesso/falha e tipo de falha)	AS - Classificação
	Características do produto; e estados discretos do processo	AS - Classificação
Prognóstico de condição (Estática e dinâmica)	Registros de manutenção; características do produto; e parâmetros contínuos do processo, de condições do produto, métricas de desempenho e resultados do processo	AS - Regressão
	Registros de manutenção; características do produto; e parâmetros discretos do processo, de condições do produto, métricas de desempenho e resultados do processo	AS - Classificação
Escalonamento *online* (Dinâmica)	Estados do sistema; e decisões e regras somente	AR - Livre de modelo
	Estados do sistema; decisões e regras; e modelos do sistema (transição e recompensa)	AR - Baseado em modelo
Controle (Dinâmica)	Apenas sinais de dispositivos de detecção	AR - Livre de modelo
	Sinais de dispositivos de detecção; e modelo do sistema (transição e recompensa)	AR - Baseado em modelo

Fonte: autoria própria.

4.3 Proposta de arquitetura

Para elaboração de uma arquitetura de sistema, é importante considerar métodos, procedimentos, técnicas e ferramentas de projeto, de modo que a proposta de arquitetura possa ser desenvolvida de maneira sistemática. As etapas descritas por Miyagi (1996) sistematizam o desenvolvimento de um sistema de automação. Em arquitetura de *software*, padrões de projeto fornecem diretrizes mais específicas. Como exemplo, pode-se citar (DAIGNEAU, 2012; NEWMAN, 2015; ROSEN et al., 2008), que apresentam diretrizes para a elaboração de arquiteturas orientadas a serviço e microsserviço.

A proposta de arquitetura é concebida aqui a partir de uma arquitetura genérica de *Data Warehouse*. Por esta razão, o projeto da arquitetura não é iniciado pelo levantamento de requisitos, por exemplo. Além disso, tendo em vista que a ideia é que a arquitetura de DW seja concebida no contexto do RAMI 4.0, algumas definições importantes da arquitetura já estão dadas e padronizadas, como o modelo de informação, as interfaces de acesso aos dados e as camadas da arquitetura. Por fim, não está prevista a implementação desta arquitetura na prática, de modo que sua avaliação não poderá ser feita com base em métricas experimentais.

4.3.1 Infraestrutura da Camada de Informação

Idealmente, todos os elementos do RAMI 4.0 devem ser contemplados para que uma proposta de arquitetura seja condizente com as diretrizes da Indústria 4.0. Embora esta condição possa ser satisfeita partindo de uma arquitetura de *Data Warehouse*, este trabalho foca na Camada de Informação do RAMI 4.0. Por isso, apresenta-se inicialmente uma proposta para a infraestrutura desta camada antes de introduzir a arquitetura completa. Em alinhamento com os documentos oficiais mais recentes publicados acerca da implementação do RAMI 4.0, a proposta para a Camada de Informação é apresentada em um nível de abstração tal que os conceitos apresentados são independentes de tecnologias específicas (BADER et al., 2019).

Considerando as funcionalidades básicas da Camada de Informação: armazenamento; processamento e análise; integração; e disponibilização de dados, a Figura 23 apresenta uma representação esquemática para esta camada, com elementos necessários para implementação destas funcionalidades. Esta Figura ainda representa parcialmente as demais camadas, explicitando apenas os elementos que interagem diretamente com a Camada de Informação. A

seguir, descreve-se de que maneira as funcionalidades desta camada são contempladas, isto é, quais elementos são responsáveis por implementar cada uma delas:

Armazenamento: assume-se o uso de dois bancos de dados, sendo um deles para armazenamento de dados históricos e outro para armazenamento do AAS e que registra apenas o valor mais atual dos seus elementos. O banco de dados histórico registra não apenas o último valor, mas todas as ocorrências de valores assumidos por um elemento. O banco de dados do AAS é fundamental para descrição formal do modelo do AAS e para implementação das demais funcionalidades.

Nesta configuração, os valores atuais para os elementos do AAS são sempre recuperados pelo último valor registrado no BD histórico. O BD histórico é atualizado com base em eventos que ocorrem na Camada de Ativo – via Camada de Integração – ou por meio de atualizações no AAS – que ocorrem por meio das Camadas de Comunicação e Funcional. Vale ressaltar desde já que, tendo em vista as diferentes finalidades dos BDs, pode-se lançar mão da persistência poliglota para o seu projeto, a fim de melhor satisfazer seus requisitos.

Processamento e análise: inicia-se esta explanação pelas funcionalidades de recebimento e pré-processamento de eventos, com subsequente transformação dos mesmos em uma forma adequada para os dados disponíveis para os demais I4.0C. Aqui, deve-se deixar claro que se tratam de eventos provenientes das Camadas de Comunicação e Funcional (os eventos provenientes da Camada de Ativo devem ser recebidos e processados na Camada de Integração). Cada operação disponibilizada em uma interface nas Camadas de Comunicação ou Funcional deve ser mapeada em uma operação na Camada de Informação que acessa diretamente o banco de dados do AAS.

A “aquisição de dados de melhor qualidade”, que consiste na conversão de dados em instâncias como informação, conhecimento, etc. deve ser realizada por meio de algoritmos. Estes devem funcionar como ferramentas para criação de modelos estatísticos. Analogamente, algoritmos podem ser utilizados para aplicação destes modelos. Nos dois casos, os algoritmos devem ser capazes de acessar diretamente o BD do AAS, porém uma vez que algoritmo e banco de dados se encontram na mesma camada (possivelmente no mesmo ambiente de execução), o acesso aos dados não se dá pelas interfaces das Camadas de Comunicação e Funcional. Por fim, é importante destacar que esta funcionalidade pode ser realizada por ativos e não necessariamente por elementos da Camada de Informação.

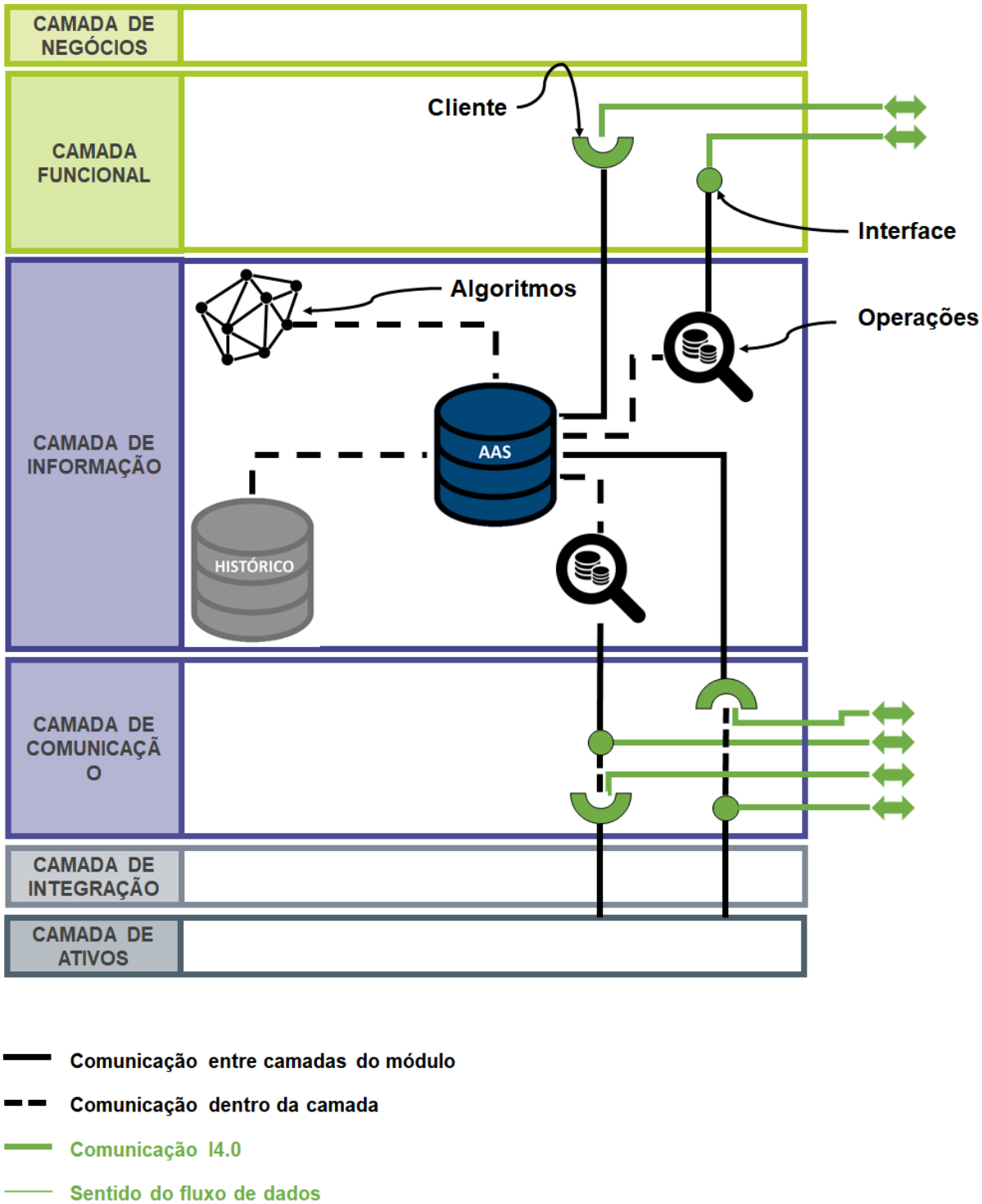


Figura 23 - Conteúdo das camadas da arquitetura, com foco na Camada de Informação do RAMI 4.0.

Fonte: autoria própria.

Integração: um mesmo ativo pode conter dados de diferentes naturezas. Para além do tipo/formato dos dados em si, os dados se diferenciam quanto à classe: dados operacionais,

históricos, manuais de instrução, descrições e etc. A integração de dados neste caso consiste em consolidar todos estes diferentes dados em uma mesma camada. Os bancos de dados da Camada de Informação devem ser capazes de lidar com tal heterogeneidade.

4.3.2 Descrição Estrutural da Arquitetura

A arquitetura genérica de DW que foi apresentada no Capítulo anterior possui uma série de variações. Blazic, Poscic e Jaksic (2017) propõem uma classificação para as possíveis arquiteturas de DW e uma comparação entre elas é proposta por Watson e Sudibyo (2006). É possível observar que, a despeito da topologia, os elementos que as constituem são em essência os mesmos. Estes elementos serão referidos como **módulos** da arquitetura e já foram apresentados anteriormente, porém não haviam sido classificados como tal. São eles:

- Módulo Fonte de Dados Operacionais (MFDO);
- Módulo de Preparação de Dados (MPD, *Data Staging Module*);
- Módulo *Data Warehouse* (MDW, *Data Warehouse Module*);
- Módulo *Data Mart* (MDM, *Data Mart Module*);
- Módulo Consumidor de Dados (MCD, *Data Consumer Module*);
- Módulo Gerenciador da Arquitetura (MGA, *Architecture Manager Module*)

O último módulo apresentado não possui uma correlação direta com arquiteturas de *Data Warehouse*, porém este tipo de Componente é observado em arquiteturas para a I4.0, como em Bedenbender *et al*. (2017a), sobretudo para orquestração dos demais I4.0Cs. Uma vez identificados os módulos da arquitetura, para os implementar de modo a seguir as diretrizes propostas para a I4.0, é importante que (i) sua estrutura e dinâmica sejam baseadas nas camadas do RAMI 4.0; e (ii) os módulos sejam reconhecidos como ativos. O segundo item traz algumas implicações imediatas:

- Cada instância destes módulos deve estar associada a um AAS e, por esta razão, os módulos necessariamente contém a sua própria Camada de Informação;
- As fontes de dados operacionais são os únicos ativos que podem existir no mundo físico. Os demais módulos têm *softwares* como seus respectivos ativos e, por esta razão, se encontram no mundo digital;

- Como consequência do item anterior, a entrada de dados no sistema se dá pelas fontes de dados operacionais. Para os demais módulos, os dados já se encontram no mundo digital.

Além destas implicações imediatas que se tem ao considerar os módulos como ativos, apresenta-se demais considerações a respeito do mapeamento da arquitetura de DW no RAMI 4.0 e da implementação do AAS para os módulos:

- Com exceção das fontes de dados operacionais, os ativos dos demais módulos exigem bancos de dados para sua operação. Além disso, estes módulos, por serem ativos, possuem os respectivos AAS, que ficam armazenados nos bancos de dados da Camada de Informação. Os BDs associados aos ativos de cada módulo armazenam dados diferentes dos armazenados na Camada de Informação;
- O fluxo de dados entre estes módulos pode ocorrer via Camada de Comunicação ou Funcional. O fluxo de entrada dos dados no sistema ocorre via Camada de Integração;
- Em linhas gerais, não é possível definir *a priori* todos os possíveis relacionamentos que um ativo estabelecerá ao longo de seu ciclo de vida e sua cadeia de valor. Em termos mais específicos para arquitetura, pode-se considerar que outros módulos poderão ser incluídos na arquitetura e estabelecer relacionamentos não previstos entre os módulos já elencados. Assim, todos os módulos devem ser capazes de atuar como cliente e servidor na arquitetura. Este tipo de arquitetura é denominado *peer-to-peer*, caracterizada por nós descentralizados e heterogêneos, que possuem a capacidade de atuar como cliente ou servidor (SCHOLLMEIER, 2001), possuindo seus próprios mecanismos de roteamento (RIPEANU, 2001). Cliente e servidor se localizam nas mesmas camadas por onde a comunicação entre os módulos ocorre;
- Os dados sempre trafegam mediante a requisição que parte de um cliente para um servidor. Eventos que ocorrem no ativo disparam a comunicação, seja por meio do AAS ou do próprio ativo, via Camada de Comunicação ou Funcional.

A seguir, os módulos são descritos em detalhes:

4.3.2.1 Módulo Fontes de Dados Operacionais

O Módulo Fontes de Dados Operacionais (MFDO) é obrigatório na arquitetura, pois sem a instância deste módulo na arquitetura, não existem os dados. Em uma arquitetura de *Data*

Warehouse, o MFDO corresponde a um banco de dados operacional ou outras fontes de dados. Na arquitetura proposta, o banco de dados em si não é o ativo deste módulo. Ao invés disso, considera-se como ativo um elemento do mundo físico ou digital cujos dados estão sendo mapeados no seu respectivo AAS e armazenados em um banco de dados. Como consequência, ao se realizar o mapeamento no RAMI 4.0, cuja camada mais inferior é a de ativos, torna-se importante contemplar, na descrição da arquitetura, o ativo que está gerando os dados a serem inseridos no sistema.

Tendo em vista que o ativo deste módulo não necessariamente dispõe de um banco de dados, os canais de interação com os demais módulos acessam o AAS e não o ativo do MFDO. Estes canais possibilitam a comunicação entre módulos e são mapeados nas Camadas de Comunicação e Funcional. O MFDO é esquematicamente representado na Figura 24. Com base no que foi apresentado, descreve-se a seguir as camadas do RAMI 4.0 neste módulo.

Os dados são originados na **Camada de Ativos**. O ativo desta camada pode existir no mundo real ou digital. Como exemplo do primeiro caso, pode-se considerar um sensor ou equipamento que está em operação e gerando dados a serem armazenados e analisados. Como exemplo do segundo, pode-se considerar um *software* executando um modelo de simulação, cujos dados serão utilizados para auxílio à tomada de decisão.

Com relação à **Camada de Informação**, sua primeira função é o armazenamento dos dados do ativo no mundo digital, contendo os dois bancos citados anteriormente, nos quais o AAS está armazenado. Após os dados percorrerem todos os módulos da arquitetura e serem analisados, algumas informações podem ser devolvidas ao MFDO. Este fluxo está previsto na arquitetura genérica de DW. No entanto, não se especifica o conteúdo presente neste fluxo de retorno. Tendo em vista que no consumidor de dados ocorre a análise dos dados e que esta proposta de arquitetura é justamente focada no seu uso para fins analíticos, considera-se que, neste fluxo de retorno, podem ser devolvidos modelos estatísticos aprendidos pelo consumidor de dados. Estes modelos podem ser utilizados para os fins listados nas Tabelas 18 a 21. A presença destes modelos no MFDO pode ser associada a algoritmos para que possam ser desempenhadas as funções como o controle do ativo, sendo estes o segundo elemento constituinte da Camada de Informação. Por fim, a Camada de Informação é também composta pelas operações que interpretam as solicitações provenientes das Camadas de Comunicação e Funcional e as convertem em consultas do tipo OLTP para o banco de dados do AAS, acessando efetivamente as suas informações.

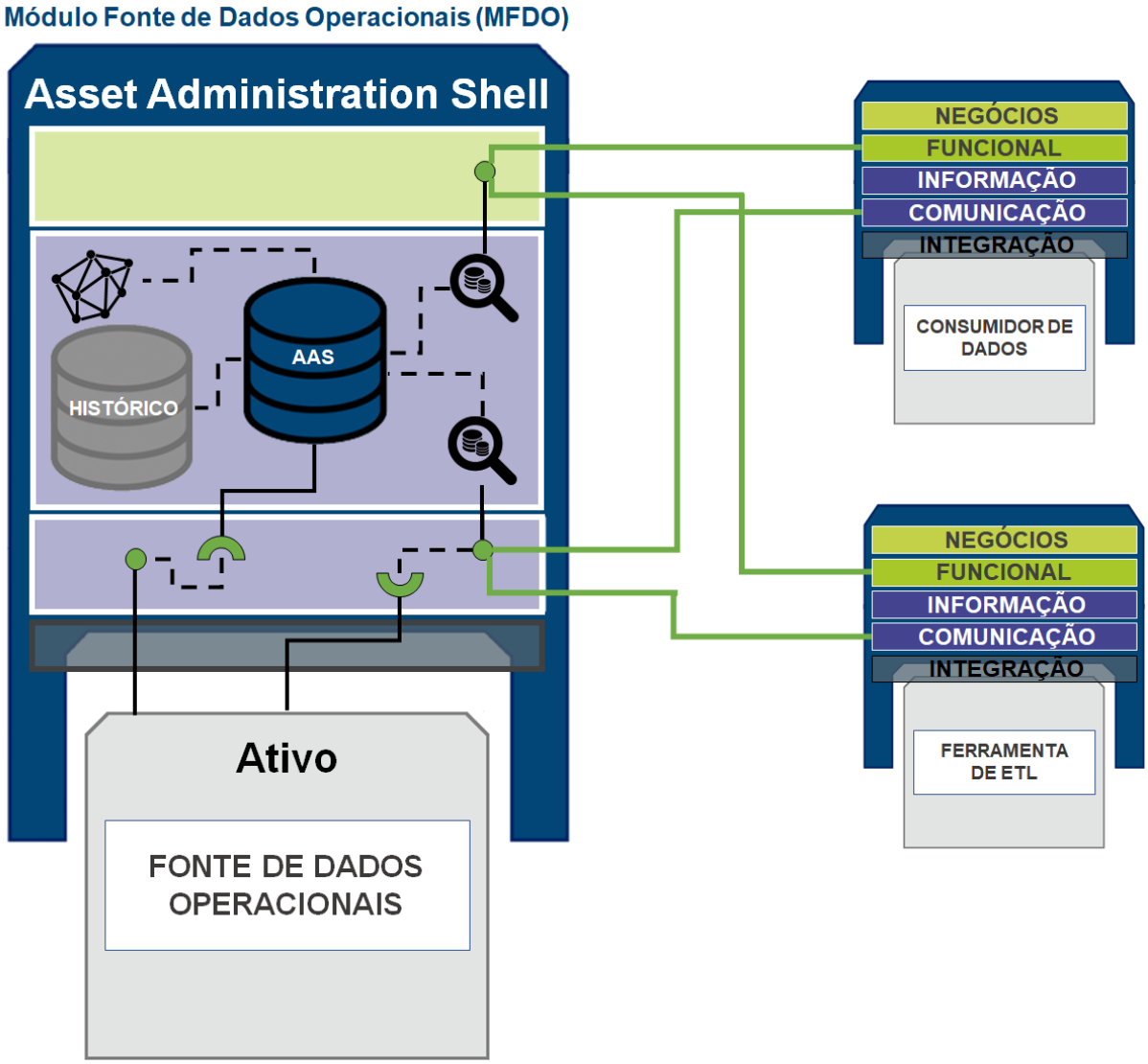


Figura 24 - Representação esquemática do MFDO.

Fonte: autoria própria.

A **Camada de Integração** é responsável por levar as informações do mundo real ao digital e vice-versa. Ela é capaz de converter os eventos do ativo físico em operações de escrita no banco de dados que contém o AAS. Tendo em vista que o fluxo de dados do usuário para o MFDO fornece informações que tornam possível atuar diretamente no ativo de modo a modificar sua operação, é necessário que a Camada de Integração disponha também de mecanismos para este tipo de interação, que pode ser entendido como um fluxo de retorno à fonte dos dados.

Embora as **Camadas de Comunicação e Funcional** possuam diferenças significativas, pode-se considerá-las como canais de interação entre o MFDO e os demais módulos, de modo que não se faz necessário evidenciar suas diferenças para a finalidade desta proposta de

arquitetura. Por esta razão, elas são descritas no mesmo tópico. No que tange à proposta da arquitetura, estas duas camadas devem disponibilizar interfaces com operações padronizadas (BADER et al., 2020) para acesso aos dados. As operações devem então ser mapeadas em operações presentes na Camada de Informação, conforme descrito anteriormente.

4.3.2.2 Módulo de Preparação de Dados

O Módulo de Preparação de Dados (MPD) é mais um módulo obrigatório na arquitetura. Mais do que simplesmente extrair os dados de uma fonte e os enviar a outra base, é neste módulo que ocorre a preparação de dados, que é uma importante etapa do processo de análise. Em alguns casos, é necessário que mais de uma instância do MPD seja utilizada na arquitetura de *Data Warehouse*, como no caso da arquitetura de três camadas ou de *Data Marts* independentes (BLAZIC; POSCIC; JAKSIC, 2017). O MPD é ilustrado na Figura 25. A seguir, são descritas as camadas do RAMI 4.0 neste módulo.

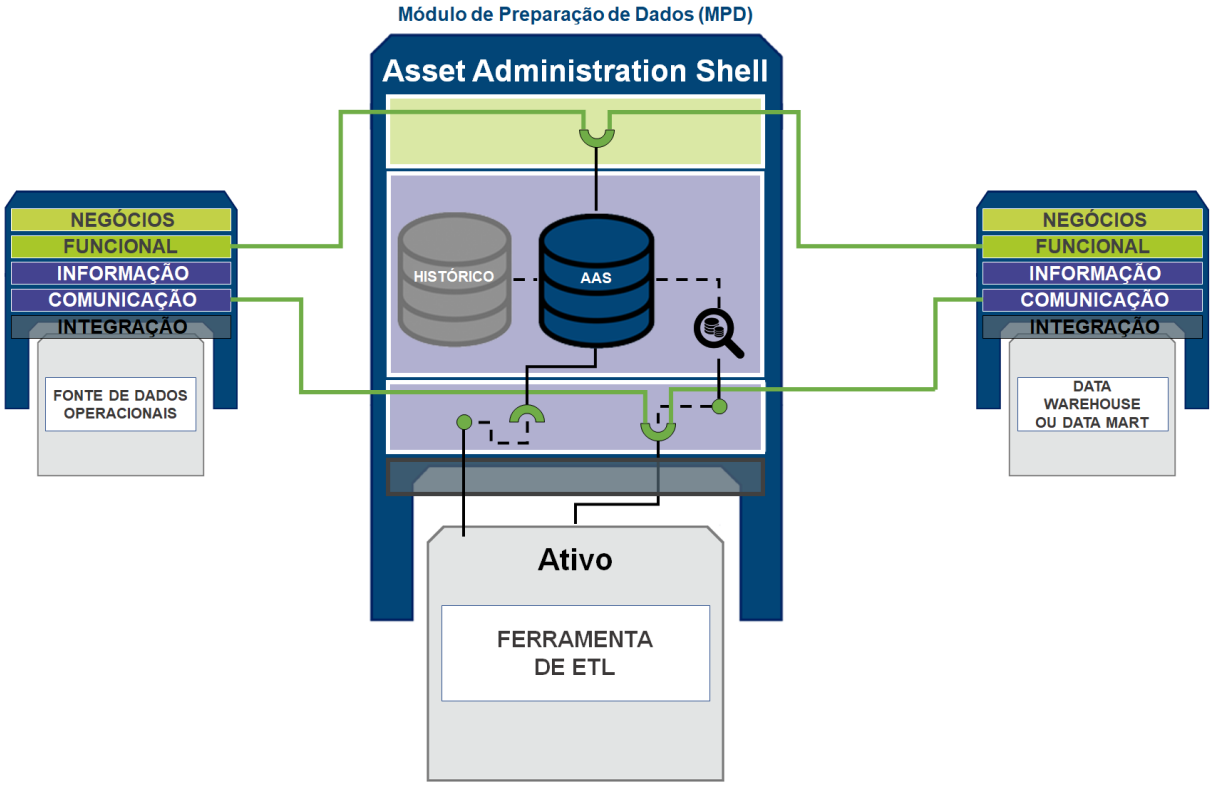


Figura 25 - Representação esquemática do MPD.

Fonte: autoria própria.

As funcionalidades deste módulo geralmente são implementadas por meio de ferramentas de *software* de ETL[23], sendo este o ativo do MPD presente na **Camada de Ativo**. O MPD possui também seu respectivo AAS, que se encontra na **Camada de Informação**. O AAS deve conter referência às fontes de dados operacionais (origem) das quais deve extrair conteúdo, bem como aos *Data Warehouse* e/ou *Data Marts* (destinos) nos quais os dados serão carregados. Em certas topologias, o MPD também pode extrair os dados do *Data Warehouse* e os carregar em *Data Marts.*

Os dados do MPD não são requisitados por nenhum outro módulo. É do próprio MPD que partem as requisições, tanto para extração de dados da origem, quanto para carregamento de dados no destino. Assim, não se faz necessário definir interfaces para este módulo. Apesar disto, as Camadas de Comunicação e Funcional se fazem presente neste módulo, para que ele possa enviar as requisições aos demais.

A interação entre o MPD e os demais módulos ocorre inicialmente via **Camada Funcional**, entre os AAS dos dois módulos. Esta interação se dá sobretudo na forma de uma negociação, conforme descrito em Bedenbender *et al.* (2017a). Durante esta negociação, pode-se estabelecer os parâmetros de comunicação entre os módulos para que o processo de extração e carregamento de dados seja efetivamente realizado.

Concluída a etapa de negociação, estabelece-se um canal na **Camada de Comunicação** entre os módulos. A comunicação entre o MPD e uma origem ocorre entre o ativo do primeiro (que efetivamente realiza a extração dos dados) e o AAS do segundo (onde estão armazenados os dados operacionais). No caso da interação entre o MPD e um destino dos dados, a comunicação ocorre entre os ativos de ambos (da ferramenta de ETL para a solução de armazenamento dos dados). O processo de transformação dos dados é próprio do ativo, isto é, trata-se de um processo do mesmo e, por esta razão, não depende de nenhuma outra camada do RAMI 4.0.

[23] Alguns dos exemplos destes *softwares* são IBM DataStage, Oracle Data Integrator, Informatica PowerCenter, Pentaho, Azure Data Factory.

4.3.2.3 Módulo Data Warehouse

Embora esteja presente na maioria das variações de arquitetura de DW, o Módulo *Data Warehouse* (MDW) não é obrigatório. Existem implementações deste tipo de sistema com *Data Marts* independentes que recebem os dados diretamente das ferramentas de ETL, sendo dispensável a existência do MDW. Nos casos em que está presente, o MDW recebe os dados provenientes das ferramentas de ETL e os armazena de tal forma a facilitar o processo de análise dos dados. Os dados do MPD podem ser acessados por múltiplos usuários finais ou mesmo *Data Marts*. A seguir, tem-se a descrição das camadas do MDW, cuja representação esquemática é apresentada na Figura 26.

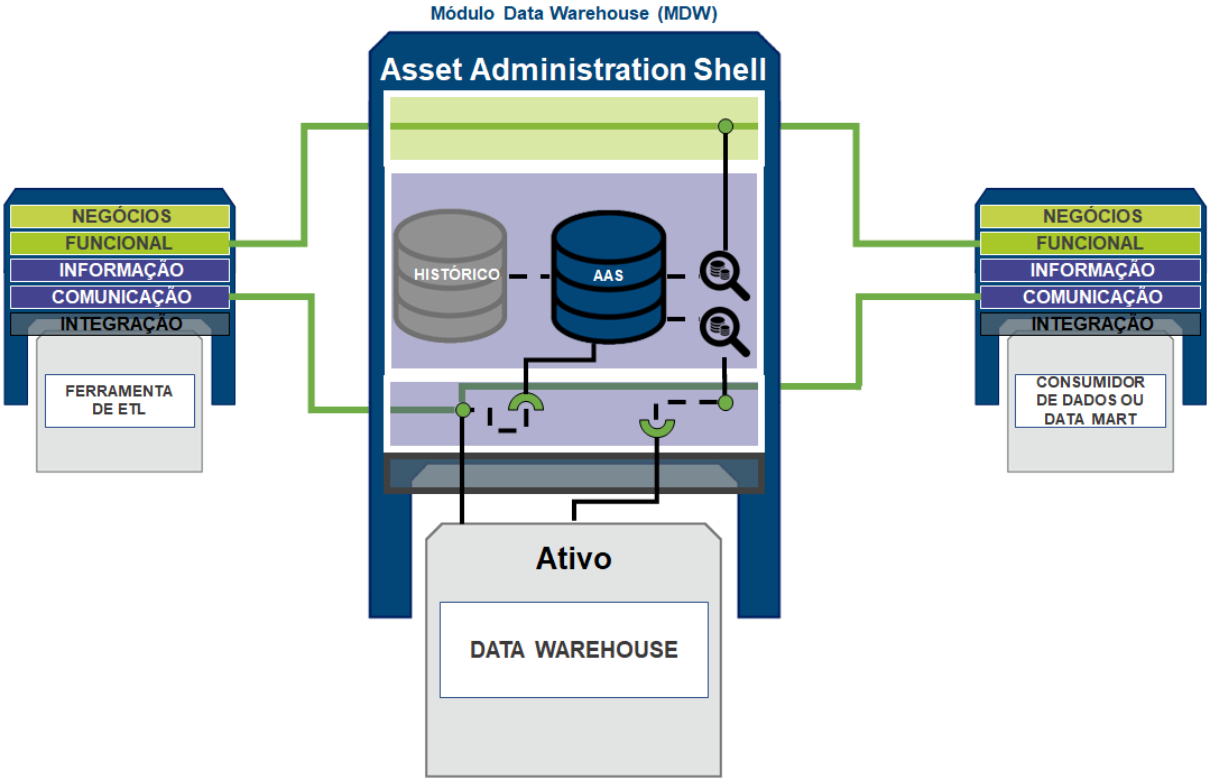


Figura 26 - Representação esquemática do MDW.

Fonte: autoria própria.

Com relação à **Camada de Ativo**, neste módulo o ativo consiste em um banco de dados analítico implementado em um SGBD. Neste banco de dados são executadas basicamente operações de escrita por parte do MPD e de leitura por parte do consumidor de dados ou de um

Data Mart[25]. A **Camada de Informação** contém o AAS do *Data Warehouse.* É importante ressaltar que este AAS não contempla dados que são provenientes das fontes, mas do *software* presente no ativo deste módulo. Sendo assim, o AAS pode ser um possível candidato para armazenamento dos metadados do DW, tais como i) a descrição dos dados propriamente ditos; ii) regras de negócios relevantes e iii) detalhes organizacionais.

Assim como no módulo anterior, a **Camada Funcional** deste módulo permite a interação inicial com os demais módulos. Esta interação se dá por meio dos AAS das partes envolvidas e na forma de uma negociação, realizando verificações e estabelecendo os parâmetros para a comunicação entre os ativos dos módulos, que se dá por meio da criação de um canal na **Camada de Comunicação** entre os ativos dos módulos envolvidos. Neste caso, o MDW exerce essencialmente a função de servidor, respondendo às requisições de escrita do MPD e de leitura do consumidor de dados ou *Data Mart*.

4.3.2.4 Módulo Data mart

O Módulo *Data Mart* (MDM) é opcional para a arquitetura e não é observado na implementação com *Data Warehouse* centralizado (BLAZIC; POSCIC; JAKSIC, 2017), por exemplo. Quando presente, o MDM integra uma fração dos dados do *Data Warehouse* ou das próprias fontes de dados, quando o primeiro não está presente. Na maioria dos casos ilustrados em Watson; Sudibyo (2006), o *Data Mart* recebe os dados de uma ferramenta de ETL, atuando assim como servidor. Porém, quando recebe dados provenientes diretamente de um *Data Warehouse*, deve atuar como cliente. A Figura 27 representa esquematicamente o módulo em questão. A seguir, descreve-se suas camadas.

A **Camada de Ativo do MDM**, assim como no caso do MDW, é composta um banco de dados mantido por um SGBD. Nele são executadas basicamente operações de escrita por parte dos mecanismos de carregamento dos dados e de leitura por parte dos usuários finais. Na **Camada de Informação** o módulo possui seu respectivo AAS, que pode variar ligeiramente nos casos em que o *Data Mart* é dependente, isto é, que seu conteúdo é extraído a partir de um *Data Warehouse*, e independente, ou seja, quando é populado por um ou mais instâncias de ETL. No primeiro caso, o MDM atua como cliente e, portanto, precisa conter referência à

[25] Exemplos de *softwares* para o ativo deste módulo são Oracle Database, Snowflake, Google BigQuery, Amazon Redshift.

origem dos dados. No segundo caso, o MDM atua como servidor e não precisa referenciar a ferramenta de ETL que fica responsável por carregar seus dados.

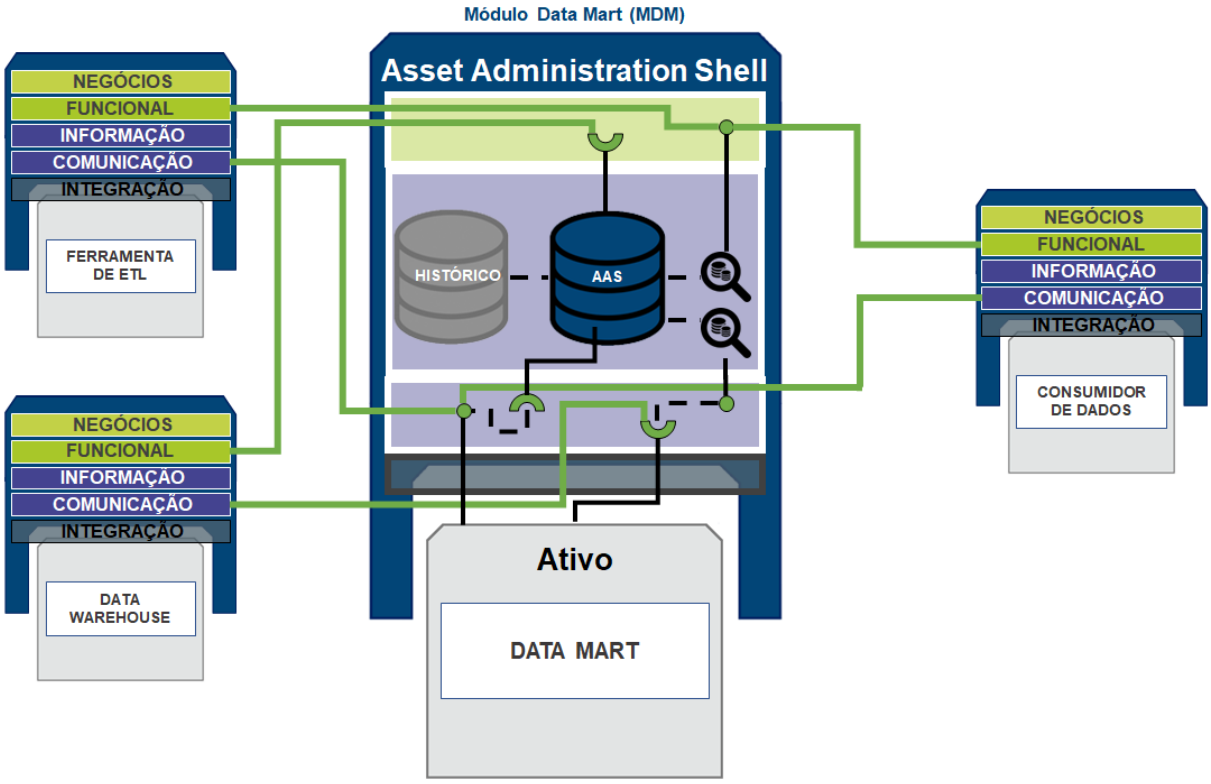


Figura 27 - Representação esquemática do MDM.

Fonte: autoria própria.

A **Camada Funcional** permite a interação inicial com os demais módulos, em todos os casos por meio do AAS. A Camada Funcional do MDM deve implementar tanto a função de cliente como de servidor. A **Camada de Comunicação** possibilita a interação entre o ativo do MDM e os ativos dos demais módulos. A interação é estabelecida após a negociação com os módulos a partir dos respectivos AAS.

4.3.2.5 Módulo Consumidor de Dados

O Módulo Consumidor de Dados (MCD) é o mais genérico da arquitetura. Ele consiste em soluções de *software* para análise dos dados de um *Data Mart* ou *Data Warehouse*. Neste módulo podem ser utilizadas as técnicas de aprendizado de máquina cujas classes e aplicações foram apresentadas anteriormente. Os modelos estatísticos elaborados por meio destas técnicas

podem ser enviados às origens dos dados para que sejam aplicados. A Figura 28 representa esquematicamente o módulo em questão e suas camadas são descritas a seguir.

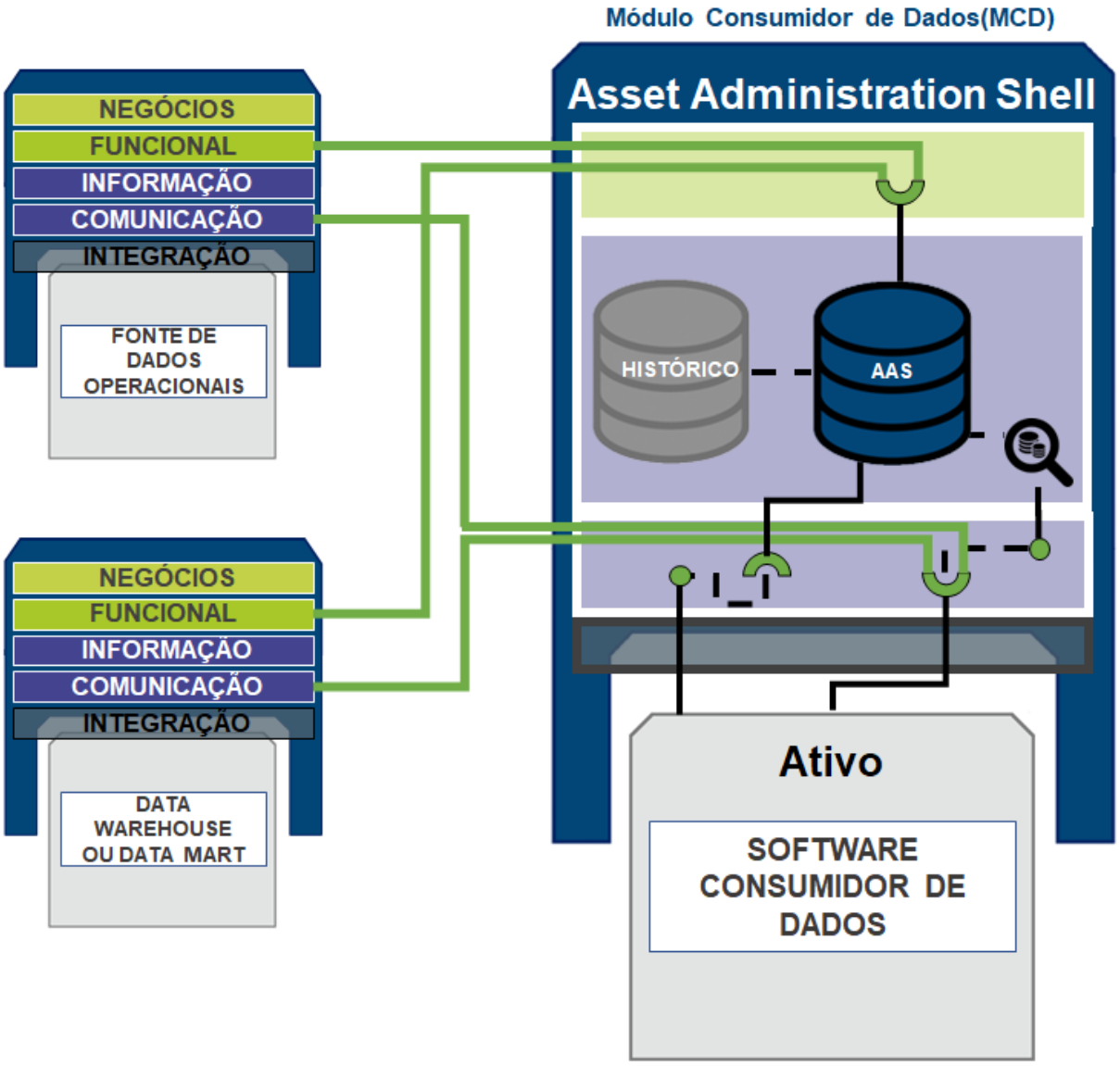


Figura 28 - Representação esquemática do MCD.

Fonte: autoria própria.

O ativo do MCD deve ser capaz de executar consultas nas interfaces dos módulos com os quais interage. Tendo em vista que, em uma arquitetura de *Data Warehouse*, estas operações geralmente são do tipo OLAP, deve-se dispor de lógicas que as mapeiem nas operações padronizadas e disponibilizadas pelas interfaces. O MCD deve dispor também de ferramentas e ambiente de execução para análise dos dados. *Frameworks* de análise e ferramentas de visualização e apresentação são exemplos destas soluções e podem compor a **Camada de Ativo**

deste módulo. Para fins de representação, descreve-se este ativo como um *software* genérico para consumo dos dados.

Tendo em vista que uma das possíveis finalidades deste módulo consiste em elaborar modelos estatísticos baseados em dados provenientes de *Data Warehouses* e *Data Marts*, é importante que o AAS deste módulo seja capaz de descrever tais modelos. Esta descrição pode incluir referências aos metadados do AAS do *Data Warehouses* e *Data Marts* que formem os parâmetros de entrada e saída dos modelos, bem como os hiperparâmetros dos modelos (como o número de camadas escondidas em uma rede neural ou o número de *clusters* em uma distribuição). Métricas de avaliação dos modelos, tais como precisão e acurácia, podem ser incluídas. Relatórios e materiais de apresentação também podem estar presentes no AAS, na **Camada de Informação**.

A **Camada Funcional** permite a interação inicial com os demais módulos, em todos os casos por meio do AAS. Em todas as interações previstas na arquitetura, o MCD atua como cliente. Uma vez concluída a etapa de negociação com os módulos a partir dos respectivos AAS via Camada Funcional, as interações via **Camada de Comunicação** são estabelecidas. No caso da interação entre MCD e os *Data Warehouses* e *Data Marts*, a comunicação ocorre por meio dos respectivos ativos dos módulos. Já com relação a interação entre MCD e as fontes de dados, a comunicação acontece entre o AAS dos dois módulos. Assim como ocorre na Camada Funcional, em todas as interações previstas na arquitetura, o MCD atua como cliente.

4.3.2.6 Módulo Gerenciador da Arquitetura

O Módulo Gerenciador de Arquitetura (MGA) não possui finalidades específicas para um sistema de *Data Warehouse* e nem é contemplado em arquiteturas para este tipo de sistema. A incorporação do MGA nesta arquitetura é baseada no trabalho de Bedenbender *et al*, (2017a) que apresenta um módulo gerenciador de produção mediando os processos realizados por unidades de produção. Suas funções são basicamente a orquestração e parametrização dos demais módulos da arquitetura. A presença de um MGA, embora não seja obrigatória, traz benefícios para a arquitetura, descritas nos parágrafos seguintes.

O MGA pode garantir o reuso dos módulos. Para descrever este benefício, pode-se tomar o Módulo de Preparação de Dados como exemplo. É possível fixar todos os parâmetros para as operações de ETL para uma instância deste módulo e ele funcionará plenamente em uma determinada aplicação. No entanto, quando não estiver sendo utilizado na aplicação para a qual

foi parametrizado, o módulo não pode ser utilizado em outra. Se esta instância puder receber estes parâmetros de um outro módulo ao invés de tê-los pré-definidos, ela se torna flexível para ser utilizada em múltiplas aplicações. O mesmo raciocínio é válido para o Módulo Consumidor de Dados, que pode realizar diferentes análises estatísticas, desde que receba os parâmetros da análise de um módulo externo, no caso, o MGA.

O MCD pode ser útil no processo de descoberta de novas instâncias dos módulos na arquitetura. Anteriormente, foi descrita a possibilidade de tornar fixos os parâmetros de operação para dois módulos. Com isso, quando uma nova instância de um dos módulos for adicionada à arquitetura, é necessário refazer a parametrização destes módulos para que possam se adequar à esta nova instância. Se, por outro lado, os parâmetros dos módulos forem flexíveis, a descoberta de uma nova instância na arquitetura pode se tornar uma responsabilidade apenas do MGA. Neste caso, cabe a este módulo informar e parametrizar os demais para que esta nova instância seja agregada às operações.

O uso do MCD pode proporcionar maior segurança à arquitetura. Quando os módulos da arquitetura têm seus parâmetros fixos, algumas informações de negócio passam a ser armazenadas nestes módulos. O MCD, por exemplo, armazenaria em sua estrutura um modelo estatístico elaborado por uma organização e que pode ser utilizado como elemento de geração de valor para seus negócios, ao transformar dados em conhecimento aplicável. Por outro lado, se este tipo de informação fica armazenado em um módulo separado – o MCD – a informação só fica disponível a este módulo em tempo de execução, isto é, enquanto este realiza alguma análise. Além das políticas de controle de acesso ao AAS, foi mencionado anteriormente que algumas perspectivas de implementação do AAS podem garantir maior segurança de suas informações. Estas perspectivas podem não necessariamente coincidir com a melhor escolha para as operações de um determinado módulo. Por isso, pode-se optar por implementar o MCD com base nestas perspectivas e com políticas de acesso mais rígidas, de modo a garantir a segurança dos dados, e os demais módulos com base nas perspectivas que ofereçam melhor desempenho para suas operações.

A estrutura do MGA é apresentada na Figura 29. O **ativo** do módulo em questão deve ser uma solução de *software* com as capacidades mencionadas. Este ativo será genericamente denominado ativo gerenciador da arquitetura. Em particular, destaca-se que esta solução de *software* deve dispor de um banco de dados contendo todas as regras de negócio que associem uma instância de MFDO aos parâmetros que devem ser inseridos para os demais módulos. Para isso, o MGA deve replicar a estrutura do AAS dos demais módulos da arquitetura,

especialmente aqueles que podem ser vistos como recursos e que podem ser reutilizados, como os Módulos de Preparação (MPD) e Consumidor de Dados (MCD). Tais regras devem ter sido elaboradas previamente e cadastradas neste ativo para que a arquitetura como um todo possa desempenhar seus processos com o mínimo de intervenção.

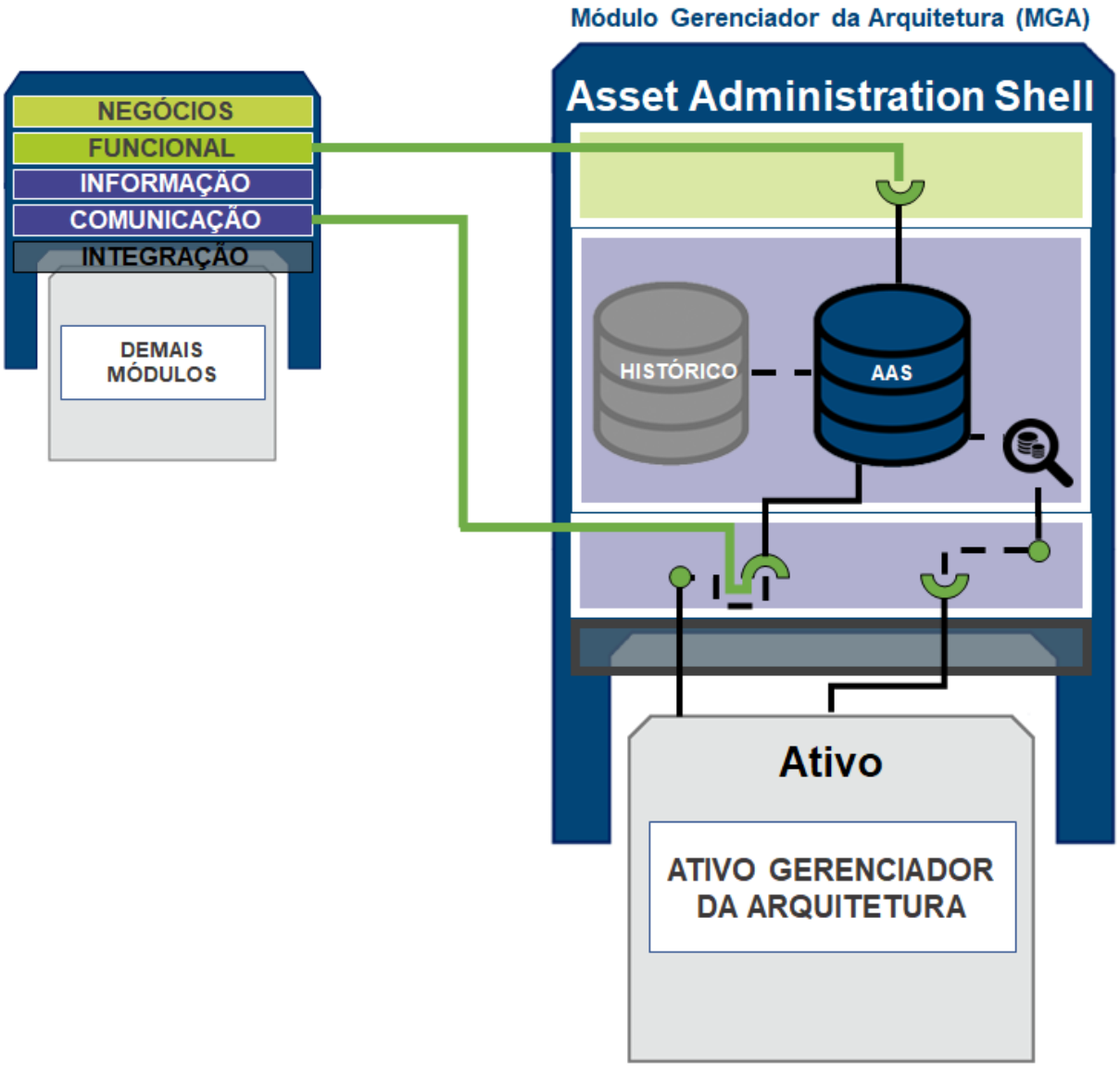


Figura 29 - Representação esquemática do MGA.

Fonte: autoria própria.

Na comunicação com os demais módulos apresentados, o MCD deve sempre exercer o papel de cliente na arquitetura[26]. Ele deve consultar os repositórios de AAS a fim de identificar a presença de uma nova instância, além de coletar e enviar parâmetros de operação para os demais módulos. Este tipo de interação ocorre sempre por meio das **Camadas de Comunicação** e **Funcional**, entre o ativo do MGA e o AAS dos demais módulos. Por fim, a **Camada de Informação** contém o AAS do MGA. Além das informações comuns para ativos de *software*, este AAS deve manter o registro dos processos que ocorrem na arquitetura como um todo.

4.3.3 Dinâmica da Arquitetura

Os módulos que foram apresentados anteriormente são agora organizados de modo a ilustrar os relacionamentos e a dinâmica da arquitetura. Inicia-se a descrição dinâmica pela etapa de descoberta das instâncias dos módulos por parte o MGA. Esta etapa é realizada por meio da operação **GetAllAssetAdministrationShellIdsByAssetId** da interface **AAS Discover**, como ilustra a Figura 30[27]. Caso o MGA esteja configurado para interagir com ativos cujos AAS estão cadastrados em um repositório específico, a descoberta pode ocorrer por meio da interface **AAS Repository**, retornando o(s) identificador(es) dos AAS cadastrados naquele repositório. Tendo em vista que AAS e Submodelo possuem identificadores únicos, esta etapa pode também ser realizada considerando diretamente os Submodelos ao invés do AAS. Em outras palavras, caso o MGA esteja configurado para interagir com instâncias de MFDO que possuam determinados Submodelos (com base em identificação semântica, por exemplo), o processo de descoberta pode ter início já partindo destes elementos do AAS. Para isso, devem ser utilizadas as interfaces **Submodel Discovery** e **Submodel Repository**.

---

[26] Este tipo de módulo pode atuar como servidor no caso da interação com outros módulos não previstos para a arquitetura. Em Bedenbender *et al*. (2017a), o módulo gerenciador da arquitetura recebe requisições e ordens de produção, as quais são respondidas por ele na forma de proposta e relatórios de produção. Neste tipo de interação, é possível interpretar o papel do DCM como de servidor.

[27] Na Figura 30 e nas demais que representam a dinâmica da arquitetura, substituiu-se, no nome das operações, AssetAdministrationShell por AAS, a fim de promover uma representação mais limpa e de fácil compreensão das Figuras.

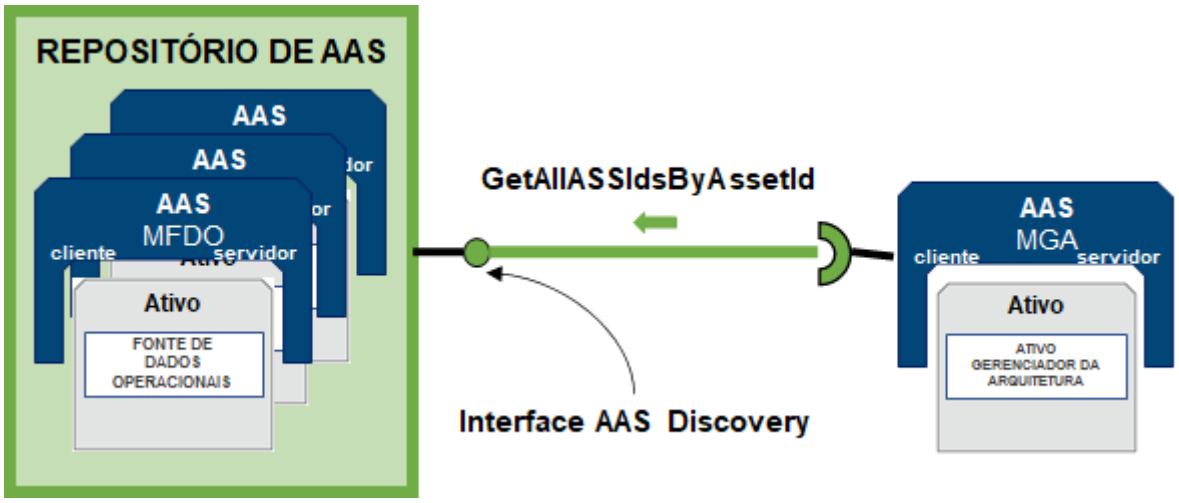


Figura 30- Descoberta de instâncias dos módulos por parte do MGA.

Fonte: autoria própria.

Após a descoberta dos AAS (ou dos Submodelos), o procedimento de negociação segue com o reconhecimento das características e funcionalidades de cada instância dos módulos envolvidos. Para esta finalidade, pode-se implementar na Camada Funcional as interfaces **AAS Registry** e **Submodel Registry**, conforme ilustrado Figura 31 (a). Feito este reconhecimento, o MGA pode acessar determinados Elementos de Submodelo destas instâncias (como Regras de Permissão de Acesso, Visões e Competências), verificando, por exemplo, se os requisitos de permissão, conexão, etc. são satisfeitos. Caso todos os requisitos para interação entre os módulos sejam atendidos, cria-se um canal na Camada de Comunicação (na forma de uma rede local virtual, por exemplo) para interação entre o ativo MGA e os AAS dos demais módulos. Esta interação deve ser registrada na forma de um Elemento de Relacionamento (Anotado ou não), a ser criado no AAS do MGA para cada instância com a qual venha a interagir. Este Elemento de Submodelo pode ser criado (**PutSubmodelElementByPath**) ou apenas ter o seu valor inserido caso tenha sido previamente definido (**SetSubmodelElementValueByPath**), de modo que ambas as operações ocorrem por meio da interface **Submodel**.

Quando as instâncias dos módulos já estiverem devidamente reconhecidas, tem-se início o processo de parametrização dos módulos por parte do MGA. Para cada instância de MFDO identificada, o ativo do MGA associa as características desta instância aos parâmetros dos demais módulos da arquitetura. O MGA identifica uma instância de MPD disponível e preenche o AAS dela com a referência da origem (MFDO) e do destino dos dados (MDW ou MDM), bem como os parâmetros para as operações de transformação que devem ser realizadas. A disponibilidade deve ser verificada pelo MGA por meio da operação

GetSubmodelElementByPath (Figura 31 b)**,** enquanto a parametrização do MPD é feita utilizando a operação **SetSubmodelElementValueByPath** (Figura 31 c), ambas da interface **Submodel**. O mesmo é feito para o MCD, informando de onde os dados para análise devem ser buscados, quais são os parâmetros da análise (do modelo estatístico, por exemplo) e para onde devem ser devolvidos os resultados.

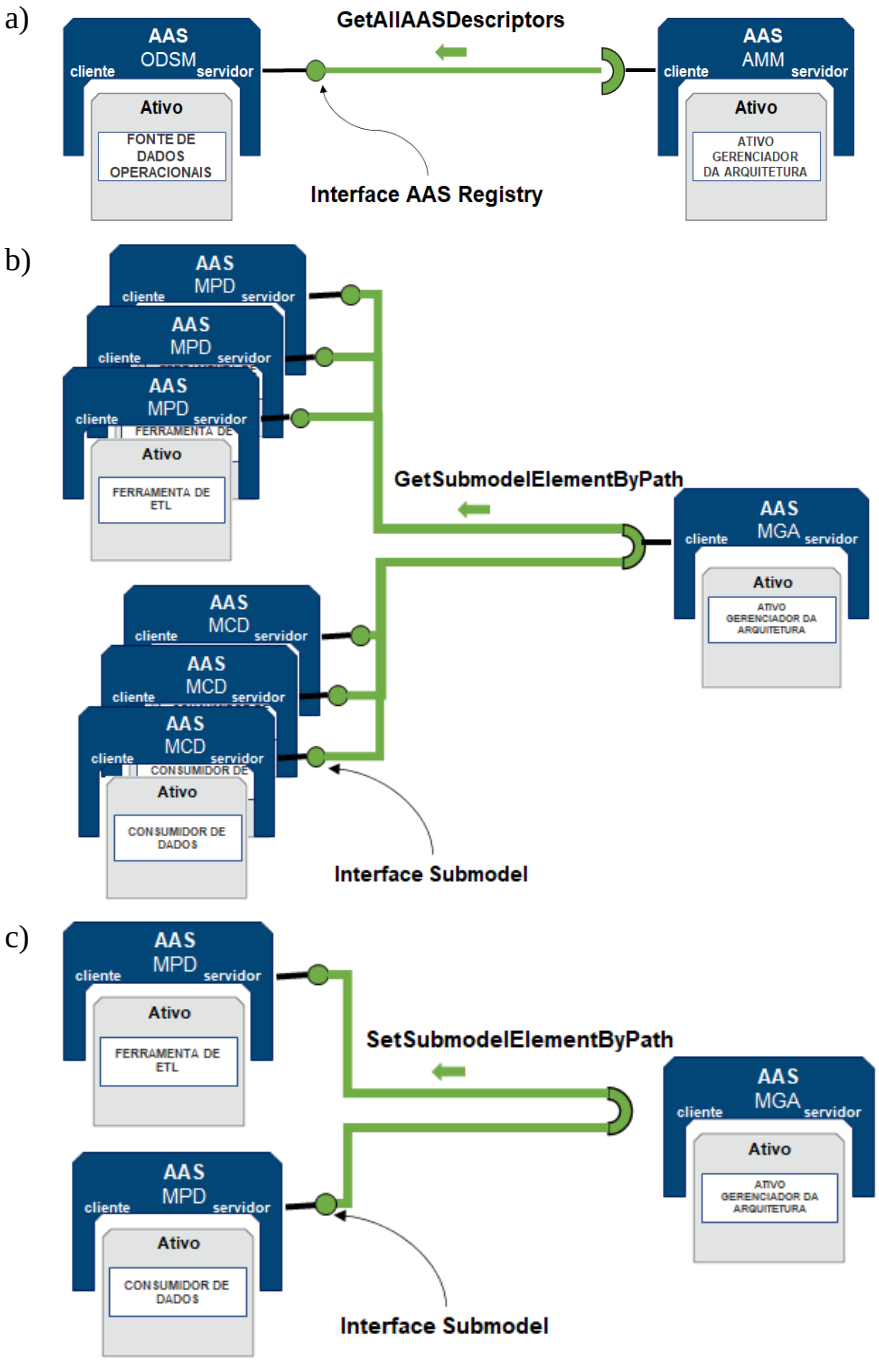


Figura 31 - Reconhecimento (a), seleção (b) e parametrização (c) dos módulos.

Fonte: autoria própria.

Feita esta parametrização, as instâncias dos módulos podem se organizar para realização dos processos para análise dos dados na arquitetura. A Figura 32 apresenta uma releitura da Figura 16 em que os módulos são apresentados, com a inclusão do MGA. É possível observar os mesmos fluxos de dados apresentados anteriormente, bem como todos os elementos necessários para compor qualquer das topologias apresentadas pelas referências citadas anteriormente. Por esta razão, a descrição da dinâmica da arquitetura é também uma adaptação daquela apresentada no Capítulo anterior, aqui contemplando os módulos e as interfaces padronizadas para a I4.0. Como o MGA não participa efetivamente do fluxo dos dados para análise, suas conexões não foram representadas na Figura 32. A fim de contemplar todos os módulos, esta Figura apresenta a arquitetura proposta na topologia *hub and spoke* (WATSON; SUDIBYO, 2006).

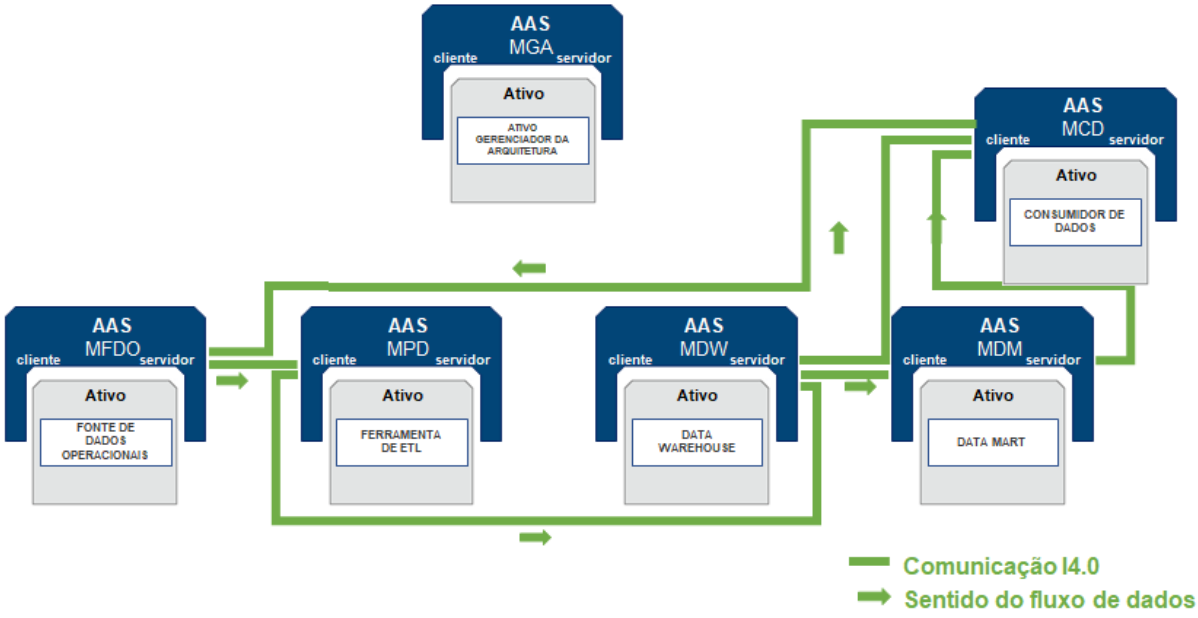


Figura 32 - Representação esquemática da arquitetura proposta.

Fonte: autoria própria.

O fluxo de dados tem início no MFDO. O ativo deste módulo contém os dados que deverão trafegar pela arquitetura. A Camada de Integração garante ao ativo deste módulo o acesso às interfaces presentes na Camada de Comunicação. Por meio das operações destas interfaces, os dados são carregados no banco de dados AAS da Camada de Informação. Considerando que os AAS e Submodelos já estão devidamente criados nos repositórios, a operação **SetSubmodelElementValueByPath** da interface **Submodel** é executada à medida

com que as variáveis do ativo físico sofrem atualizações. Esta operação afeta inicialmente o banco de dados AAS e é posteriormente propagada para o banco de dados histórico. Este processo ocorre da mesma maneira para os demais módulos, em que os dados de operação dos ativos podem ser armazenados.

Com isso, estabelece-se a interação entre o MPD e o MFDO, seguindo as etapas de negociação, criação do canal na Camada de Comunicação e registro do Elemento de Relacionamento no AAS. Uma vez estabelecido o relacionamento, pode-se dar início ao processo de extração, em que os dados do MFDO são requisitados pelo MPD. Pode-se recuperar todos os elementos de Submodelo a partir de seu identificador global ou apenas aqueles selecionados, a depender das configurações do MPD. Em ambos os casos, com base nas operações padronizadas para a interface **Submodel**, só é possível recuperar estes elementos em sua totalidade, isto é, o Elemento de Submodelo como um todo incluindo idShort, categoria, ID semântico, descrição conceitual, etc. Caso se deseje recuperar todos os Elementos do Submodelo, as operações candidatas são: **GetAllSubmodelElements**, **Get...ByParentPathAndSemanticId** e **Get...BySemanticId**. Para o segundo caso, em que apenas os elementos de interesse são extraídos, a operação **GetSubmodelElementByPath** da interface **Submodel** deve ser executada.

Quando os dados das instâncias de MFDO já foram extraídos, dá-se início a etapa de transformação, que ocorre no ativo do MPD a partir dos parâmetros definidos no seu respectivo AAS por parte do MGA. Quando a preparação dos dados é finalizada e um determinado número de registros pré-determinado foi atingido, o MPD está pronto para carregar os dados no(s) destino(s). Vale destacar que a interação entre MPD e MDW ocorre entre os ativos dos módulos, isto é, entre seus ativos. Assim sendo, as interfaces do AAS não são utilizadas no processo de carregamento.

O fluxo dos dados segue com o acesso ao Módulo de *Data Warehouse* por parte do Módulo *Data Mart* e/ou do Módulo Consumidor de Dados. Para que estes fluxos ocorram, o MGA deve ter realizado previamente a parametrização do MDM e do MCD. Vale ressaltar que, em ambos os casos, o MDW atua como servidor. No caso do fluxo entre MDW e MDM, basta que o Módulo Gerenciador da Arquitetura forneça ao segundo a referência à porção dos dados que deve ser registrada. Esta referência é inserida por meio da operação **SetSubmodelElementValueByPath**, da interface **Submodel**. Tendo sido previamente parametrizados, a dinâmica segue com o processo de negociação entre os módulos, que ocorre de forma semelhante ao descrito para a interação entre MFDO e MPD e é o único momento em

que as interfaces de AAS são utilizadas, tendo em vista que, uma vez que o canal na Camada de Comunicação é criado, a interação entre os módulos ocorre por meio dos respectivos ativos.

O último fluxo a ser descrito ocorre no retorno dos dados analisados às instâncias de MFDO. Para isso, o MGA deve ter realizado as etapas de parametrização da mesma maneira que foi descrito nas situações anteriores. Os modelos estatísticos elaborados no MCD devem ser organizados na forma de elementos de submodelo do AAS. Em princípio, estes modelos estatísticos podem não estar contemplados no AAS do Módulo Fonte de Dados Operacional. Neste caso, o MFDO deve conter ao menos a referência a este Submodelo. A criação dessa referência deve ser realizada pelo MCD, por meio da operação **PutSubmodelReference**, da interface **AAS**. Caso o Submodelo já esteja presente nas instâncias de MFDO, basta atualizar seus valores, por meio da operação **SetSubmodelElementValueByPath**, presente na interface **Submodel**. Estas requisições promovem alterações inicialmente no banco de dados AAS e são posteriormente propagadas para o banco de dados histórico.

4.3.4 AAS para Ativos de *software*

Todos os módulos da arquitetura, com exceção do MFDO, possuem *software* como ativos. Pensar o conteúdo do AAS destes ativos pode ser menos intuitivo do que quando se tratam de ativos físicos, isto é, que existem no mundo real. O dicionário eCl@ss[28], por exemplo, propõe modelos de informação compostos por propriedades para *softwares* como os que são utilizados nos módulos da arquitetura proposta. Deve-se destacar que este dicionário se encontra em aprimoramento, de modo que ainda não é possível identificar um número considerável de propriedades específicas para cada tipo de *software*, mas sim informações genéricas compartilhadas por todos, tais como:

- Propriedades do *software*: versão/revisão; licença, especificações mínimas exigidas para execução da aplicação (os ativos dos módulos);
- Propriedades do cliente e servidor: informações de acesso, especificações da comunicação, segurança;
- Propriedades de *backup* e recuperação.

[28] https://eclass.eu/

Além de informações que se destinam especificamente para ativos de *software*, existem aquelas que são comuns para todos os ativos, tais como:

- Informações do fabricante (no caso o desenvolvedor de *softwares*): nome, endereço físico e informações legais;
- Identificação: designação do produto, número de série, família de produto;
- Documentação: especificações técnicas, manuais.

4.3.5 Bancos de dados para a Camada de Informação

Foi assumida a utilização de dois bancos de dados na Camada de Informação dos módulos desta arquitetura. Estes bancos são responsáveis por armazenar dados atuais e históricos dos AAS dos módulos. Com base em Bader *et al*. (2019), o metamodelo do AAS pode ser representado por meio de um esquema entidade-relacionamento estendido[29], conforme ilustrado na Figura 33. Neste esquema, as entidades (retângulos) representam as classes de elementos do AAS. As elipses contêm atributos desses elementos. Alguns destes atributos são do tipo compostos, isto é, são conjuntos de outros atributos. Elipses com contorno tracejado indicam atributos derivados, isto é, atributos cujo valor é derivado de outros elementos do esquema. Atributos cujo nome é sublinhado consistem em identificadores ("id" para Elementos Identificáveis e "idShort" para Elementos Referenciáveis). Os losangos indicam relacionamentos entre os elementos do AAS. A maioria dos elementos é do tipo "Contém", que indica que um ou mais elementos são, na verdade, uma composição de outros. Como este tipo de elemento não corresponde a uma classe de elemento do AAS, não foi traduzido. Para as entidades e atributos, foi preservado o nome original, em inglês. Observa-se que um dos losangos corresponde a um elemento específico do AAS que não é mapeado como entidade, mas como relacionamento – trata-se da "Lista de Materiais". Os círculos com um "d" ao centro indicam uma especialização, ou seja, servem para indicar as subclasses que constituem uma superclasse. No exemplo apresentado, pode-se verificar que a superclasse "Referenciáveis" é composta pelas subclasses "AAS", "Ativo", "Descrição Conceitual" e "Submodelo".

[29] O modelo entidade-relacionamento estendido é um tipo de formalismo utilizado para representar os dados em nível conceitual, isto é, por meio elementos que podem ser facilmente interpretados pelos usuários (ver Seção 3.2).

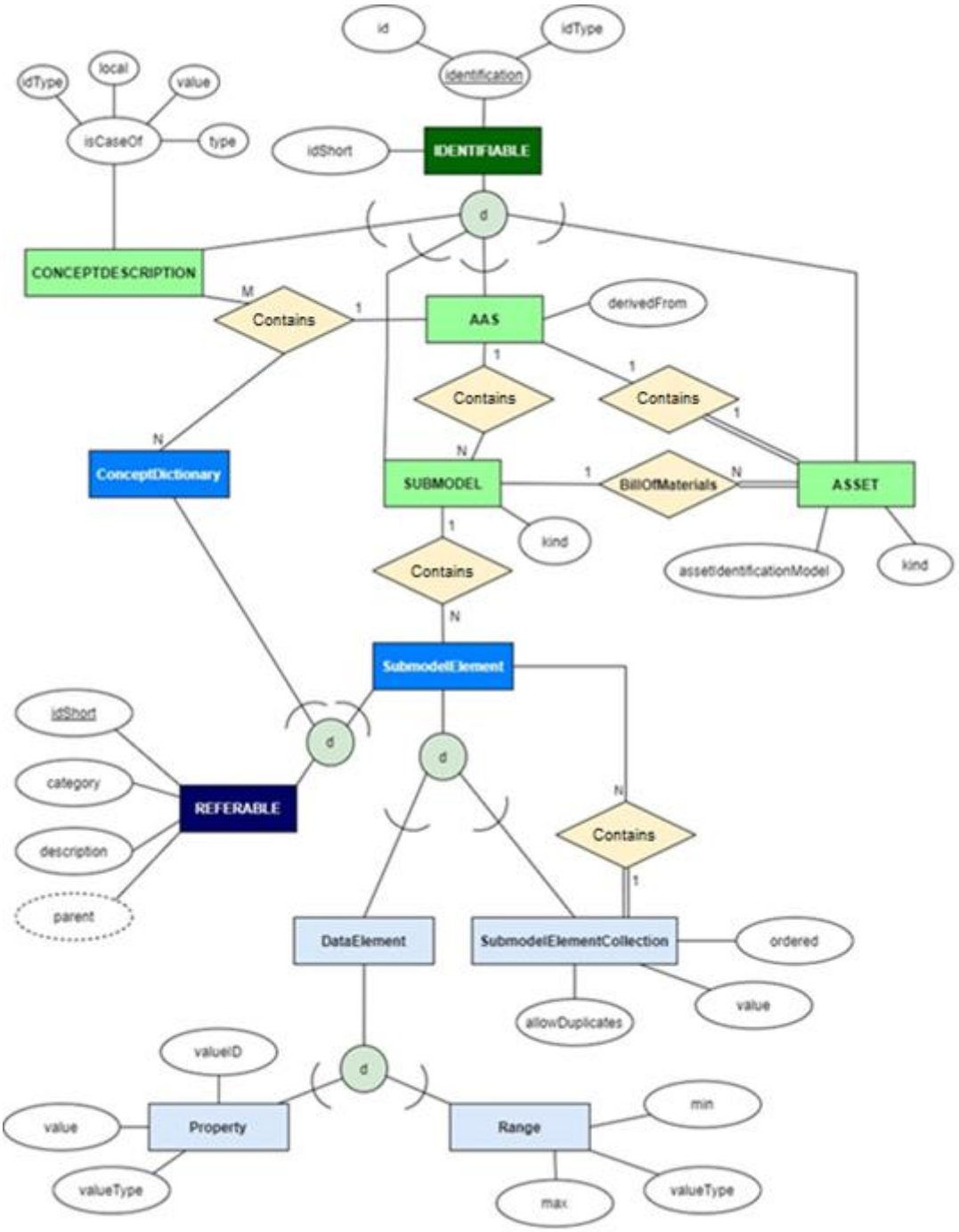


Figura 33 - Metamodelo do AAS em nível conceitual.

Fonte: autoria própria.

Conforme apresentado anteriormente, a escolha do modelo de dados lógico neste trabalho é orientada pela caracterização qualitativa do banco de dados em termos das dimensões de *big data*, na sua classificação como bancos de dados operacionais ou analíticos, e em características

como a complexidade das conexões e flexibilidade no acesso aos dados. Nos parágrafos seguintes, estas caracterizações são apresentadas.

4.3.5.1 Banco de dados “AAS”

O banco de dados “AAS”, representado em azul nas Figura 23 a Figura 28, armazena os dados mais atuais referentes aos elementos do AAS. Ele executa transações rotineiras associadas sobretudo ao processo de negociação entre os módulos via Camada Funcional, bem como aquelas ligadas à interação entre o AAS e o respectivo ativo. Este BD pode também servir para armazenar e disponibilizar dados utilizados no processo de inclusão de um módulo na arquitetura. A seguir, descreve-se a contribuição de cada dimensão de *big data* para este banco, além de aspectos de flexibilidade de acesso e complexidade das conexões entre os dados.

O **Volume** armazenado neste banco de dados depende da quantidade de elementos presentes no AAS. Considerando que apenas o registro mais atual é mantido e com base na descrição do conteúdo do AAS dos módulos da arquitetura, é possível afirmar que o volume de dados tende a ser relativamente baixo, sobretudo se comparado ao banco de dados histórico.

A **Velocidade**, neste caso associada à disponibilidade, deve ser suficientemente alta para que não cause atrasos significativos nos processos desempenhados na arquitetura. As operações destinadas a este banco de dados ocorrem nos momentos de negociação entre os módulos e interações do AAS com o respectivo ativo. É possível discorrer a respeito do volume de dados e custo computacional destas operações, com o propósito de se analisar qualitativamente a velocidade de resposta esperada para este tipo de banco.

Inicia-se esta análise pelo processo de negociação entre os módulos da arquitetura. As operações associadas a este processo dispõem de certa simplicidade, conforme ilustrado em Bedenbender *et al.* (2017a, 2017c). Durante a negociação, é verificada a compatibilidade entre os Componentes I4.0, bem como a avaliação de suas capacidades para a realização dos processos que serão executados uma vez estabelecido o relacionamento entre eles. Tratam-se, possivelmente, de transações por essência de leitura, com baixo custo computacional e com um baixo volume de dados envolvidos.

Para analisar o fluxo de dados entre AAS e ativo, faz-se uma apresentação de uma forma de classificar os elementos do AAS. Os elementos de dados, com exceção de BLOBs e arquivos, podem ser classificados como constante, parâmetro ou variável (BADER *et al*., 2019). A primeira categoria se refere a elementos cujo valor não é alterado ao longo do tempo. A segunda

se refere àqueles elementos cujo valor, uma vez definido, tipicamente não varia. Por fim, a última categoria se refere aos elementos cujo valor é calculado/aferido durante o tempo de execução/operação/funcionamento do ativo. Apenas os elementos pertencentes à última categoria sofrem atualizações que geram a interação entre ativo e AAS. Embora existam variáveis, elas não são acessadas pelos demais módulos (com exceção do MFDO), mas servem para análises/manutenções futuras que não estão previstas para esta arquitetura. Por isso, a disponibilidade deste banco não é determinante para as operações desta arquitetura. Ainda assim, é válido considerar que as variáveis devam ser atualizadas rapidamente, a fim de evitarem filas e estarem à disposição para as análises que possam incidir sobre elas. Em síntese, a alta velocidade nas operações feitas no BD AAS é uma propriedade desejável, mas que não deve gerar impactos significativos para a arquitetura;

Conforme discutido anteriormente, a **veracidade**, no recorte feito para bancos de dados, está ligada à consistência. Quanto a esta propriedade, pode-se fazer duas considerações: i) o baixo volume de dados permite que os dados possam ser armazenados em um mesmo nó, então não há riscos de inconsistência do ponto de vista de propagação dos valores para diferentes nós, como se tem em um sistema distribuído; ii) apenas os ativos dos Componentes I4.0 é que possuem a capacidade de alterar os valores dos elementos presentes no respectivo AAS. A única exceção ocorre no processo de escrita do MCD no MFDO para inserção dos modelos de aprendizado de máquina. Ainda assim, não há possibilidade de conflito entre MCD e ativo na atualização dos valores do AAS deste módulo, isto é, eles atuam em frações separadas do AAS, sem a exigência de um rígido controle de concorrência. Assim sendo, a dimensão veracidade não impõe restrições significativas para escolha do modelo de dados.

Em termos de **variedade**, o modelo de dados a ser escolhido deve apresentar alto grau de flexibilidade, dada a necessidade de poder armazenar informações dos mais diferentes tipos no AAS. Para os módulos da arquitetura proposta, os tipos de dados predominantes são propriedades e arquivos. A necessidade de representar a variedade das informações do AAS torna, por si só, os bancos de dados NoSQL candidatos mais adequados. Além disso, tendo em vista que a mesma instância de banco de dados pode armazenar o AAS de mais de um módulo, a variedade dos dados pode ser ainda maior.

Discute-se agora a **flexibilidade de acesso** aos dados. Embora o conceito de agregado esteja presente no modelo de informação do AAS nos elementos identificáveis – sobretudo o Submodelo – os módulos devem ser capazes de recuperar informações parciais destes agregados, como as propriedades, por exemplo. Sendo assim, a alta flexibilidade de acesso

principalmente nas operações de leitura é um requisito significativo para este banco de dados. Nota-se que a dinâmica da arquitetura proposta exige operações em dados presentes no interior de um agregado.

Finalmente, deve-se observar que não há, praticamente, dados compartilhados entre os módulos, com exceção dos elementos de relacionamento com anotações que podem ser mapeados nos dois AAS. Nem mesmo as interações entre os módulos refletem em aumento na complexidade dos relacionamentos, já que esta interação é mapeada em um Elemento de Relacionamento no AAS de uma das partes envolvidas. Deste modo, a **complexidade das conexões** é baixa.

A síntese do que foi apresentado, combinada com as informações das Tabelas 14 a 17, permite identificar o tipo de BD adequado para o banco de dados AAS. Em primeiro lugar, pelo tipo de operações e usuários que acessam este banco, é possível caracterizá-lo como operacional. Pelas características da arquitetura, este banco de dados não possui requisitos rígidos para as dimensões volume e veracidade, ao passo que a velocidade nas respostas às consultas tende a ser um requisito mais importante, correspondendo ao terceiro cenário para o qual o modelo **BASE** tende a ser mais adequado. Os bancos de dados que implementam essas garantias são flexíveis o suficiente para comportar a variedade necessária para o banco de dados AAS. Como a complexidade dos relacionamentos é baixa e a flexibilidade de acesso que deve ser garantida, verifica-se na Tabela 17 que os modelos de **documentos** e **famílias de colunas** se apresentam como os mais adequados (terceiro cenário). Tendo em vista que a literatura sugere a implementação do AAS em esquemas como JSON e XML, o modelo de documentos pode ser vantajoso, por apresentar uma forma de modelagem e armazenamento mais direta.

4.3.5.2 Banco de dados histórico

O banco de dados histórico, representado em cinza nas Figuras Figura 23 a Figura 29, armazena os dados históricos referentes aos elementos do AAS. Ele executa transações de grande volume para análise do funcionamento do módulo. Nos parágrafos seguintes, descreve-se a contribuição de cada dimensão de *big data* para este banco de dados, além dos aspectos de "flexibilidade de acesso" e "complexidade das conexões entre os dados".

O **volume** de dados a ser armazenado neste banco tende a ser alto e continuamente crescente à medida que o módulo se encontra em operação ou mesmo em situações de paradas,

falhas, etc. É uma dimensão que impõe exigências significativas para o projeto do banco de dados, exigindo possivelmente a sua distribuição.

Em termos de **velocidade**, as operações destinadas a este banco devem ser, em sua maioria, de escrita de registros pequenos e recuperações de grandes volumes de dados. As primeiras podem ser executadas com alta velocidade pelo baixo volume, enquanto as segundas devem exigir maior tempo para resposta. Janelas de indisponibilidade na recuperação devem ser previstas para este banco.

As análises baseadas em dados históricos devem satisfazer uma série de atributos de qualidade, de modo que garantir a **veracidade** dos dados a serem analisados é fundamental. Assim sendo, a veracidade é uma dimensão significativa para o projeto do banco de dados em questão, de modo que a consistência é uma propriedade a ser garantida. A garantia de consistência não é afetada por possível concorrência na escrita, mas pela necessidade de que os nós do sistema distribuído tenham as réplicas dos mesmos dados.

Com relação a **variedade**, o banco de dados histórico deve ser capaz de contemplar os mesmos elementos do banco de dados AAS. Por esta razão, a mesma variedade que deve ser observada no segundo pode também estar presente no primeiro. Assim sendo, a variedade deve ser garantida.

As operações de leitura dos dados históricos podem ser executadas considerando um agregado como uma unidade. Estes agregados podem ser posteriormente fragmentados e analisados em frações. No entanto, as operações de escrita, isto é, de registro dos dados históricos deve ser capaz de adentrar a estrutura do agregado, de modo que a **flexibilidade de acesso** deve ser garantida para o banco de dados em questão.

O último aspecto a ser analisado diz respeito à **complexidade das conexões** entre os dados. Tal como ocorre no banco de dados AAS, não são esperadas conexões complexas entre os dados históricos. Deste modo, a complexidade das conexões é potencialmente baixa para o banco de dados em questão.

Novamente, faz-se uma síntese dos tópicos apresentados para identificação do tipo de BD mais adequado para o banco de dados histórico, com base nas informações apresentadas nas Tabelas 14 a 17. As dimensões volume e veracidade são fundamentais para este banco de dados, de modo que, considerando o teorema CAP, a consistência deve ser priorizada no *trade-off* consistência x disponibilidade. O cenário 6 indica que tanto o modelo BASE quanto ACID podem ser utilizados. Contudo, a necessidade de representar dados de grande variedade e de

agregados, sugerem o uso de bancos de dados NoSQL, dentre os quais o modelo de grafos não exprime qualquer vantagem com relação aos demais. Assim, sugere-se o uso do modelo BASE.

Com base na exigência por alta flexibilidade de acesso e considerando a baixa complexidade nos relacionamentos, verifica-se que o cenário 3 da Tabela 17 melhor descreve este banco de dados. Diferentemente do caso anterior, em que o modelo de documentos foi sugerido, é possível argumentar que o modelo de famílias de colunas pode ser mais adequado em comparação ao de documentos, conforme apontam Chevalier *et al*. (2015b). O primeiro argumento é que, tendo em vista o grande volume de dados que caracteriza o banco de dados histórico, o modelo de famílias de coluna pode ser vantajoso devido à sua alta escalabilidade, já que possibilita o particionamento horizontal (distribuição de linhas) e vertical (famílias de colunas) dos dados. Para um banco de dados analítico como é o caso em questão, consultas de agregação são frequentemente realizadas e a possibilidade de recuperar famílias de colunas específicas permitem que este tipo de banco de dados apresente um desempenho superior ao de documentos. Para além das consultas de agregação, Scabora *et al*. (2016) destacam o bom desempenho dos BDs de famílias de colunas para transações OLAP.

4.4 Síntese do Capítulo

Este Capítulo apresentou as contribuições advindas deste trabalho. Descreveu-se de que maneira as dimensões de *big data* se manifestam em um contexto de Indústria 4.0, tomando por base a sobretudo a estrutura e as perspectivas de implementação do *Asset Administration Shell*. Com isso, complementa-se diversos trabalhos que apresentam *Big Data & Analytics* como um pilar tecnológico da I4.0 e se mostra como suas dimensões podem ser utilizadas para descrever a natureza dos dados em um contexto de I4.0.

Com base em uma série de propostas de arquitetura de sistema que buscam implementar a chamada Fábrica Inteligente, identificou-se que muitas delas tratam as soluções de armazenamento de dados como meras entidades que suportam as funcionalidades da arquitetura. No entanto, escolher qual modelo de dados lógico usar pode afetar significativamente o desempenho da arquitetura. Uma vez obtida uma caracterização qualitativa dos dados no contexto da Indústria 4.0, foi possível avaliar os modelos de dados relacional e NoSQL que melhor se adequam à natureza dos dados no contexto em questão. Por fim, identificou-se as propriedades transacionais e modelos de dados lógicos apropriados de acordo com o volume, variedade, velocidade, veracidade dos dados, complexidade das conexões e

flexibilidade de acesso aos dados, permitindo a análise da adequação de bancos de dados relacional e NoSQL para diferentes cenários da I4.0, tanto para bancos analíticos quanto transacionais.

Foi desenvolvido um mapeamento de classes de técnicas de aprendizado de máquina no Modelo de Arquitetura de Referência da Indústria 4.0. A importância de se explorar este aspecto do RAMI 4.0 está ligada ao aumento da disponibilidade de dados observado na indústria moderna, que possibilita que a orientação a dados venha a transformar a manufatura, sobretudo por meio de soluções de AM. Este mapeamento de técnicas de aprendizado de máquina no RAMI 4.0 não foi feito com base apenas nas características dos dados, mas nos possíveis uso de aprendizado de máquina em aplicações que visem implementar melhorias nas partes estática e dinâmica de ativos.

Apresenta-se, por fim, em uma proposta de arquitetura para integração e análise de dados. Esta proposta é concebida a partir de arquiteturas para sistemas de *Data Warehouse* e com seus elementos e comportamento dinâmico representados de acordo com as diretrizes da I4.0. A identificação dos módulos, sua representação como I4.0Cs e o mapeamento da dinâmica da arquitetura em operações pré-definidas garantem a adequação à padronização proposta pelo AAS e, por consequência, a interoperabilidade destes Componentes. A representação das camadas do RAMI 4.0 referentes à cada Componente, com foco na Camada de Informação, não apenas garantem a aderência desta proposta às diretrizes da I4.0, como também contribui para discussões sobre como implementar tais diretrizes.

5 Exemplo de Aplicação

Os resultados deste trabalho apresentados no Capítulo 4 são demonstrados por meio de um exemplo de aplicação. Considera-se, para isto, um complexo de geração de energia eólica denominado “Complexo Eólico Corredor do Senandes” (CECS). Neste capítulo se pretende descrever o empreendimento e de que maneira as contribuições deste trabalho poderiam ser aplicadas em um caso real. O foco desta demonstração está nas tarefas de armazenamento, integração, processamento e disponibilização dos dados que ocorrem na Camada de Informação do RAMI 4.0.

Deve-se destacar a justificativa da escolha do CECS para demonstrar as aplicações deste trabalho. Uma característica dos aerogeradores é que estes são máquinas genéricas no sentido em que não são projetadas de acordo com as condições do local em que serão instaladas, diferentemente do que ocorre nas usinas hidrelétricas de médio e grande porte, em que cada projeto de hidrogerador é único (PORTO, 2008). Com isso, é possível de se encontrar os mesmos modelos de aerogeradores em usinas eólicas no Nordeste e no Sul do Brasil, regiões que possuem diferenças significativas no que diz respeito às condições ambientais. Tendo em vista que as definições de projeto dos aerogeradores e as condições ambientais não podem ser ajustadas, a maximização da geração de energia e, consequentemente, do retorno financeiro destas máquinas pode ser obtida ao minimizar seus custos de manutenção, sua indisponibilidade e maximizar sua confiabilidade.

A manutenção preditiva aplicada a usinas eólicas é uma forma eficaz de se aumentar a disponibilidade e reduzir os custos de operação e manutenção dos equipamentos (ZHANG, 2018). Carvalho *et al*. (2019) apontam os modelos estatísticos baseados em dados como promissores para aplicações de manutenção preditiva. Andrade, Torres e Maia (2022) e Maia, Queiroz e Gomes (2022) apresentam casos de sucesso em que a predição de falhas baseada em dados obtidos a partir de sistemas de supervisão e aquisição de dados foi capaz de aumentar a geração dos aerogeradores e o retorno financeiro da usina eólica como um todo. Ao considerar estas possibilidades de melhoria aplicadas ao CECS, pode-se afirmar que este complexo eólico é um exemplo relevante para demonstrar as aplicações deste trabalho, que prevê agregar inteligência aos processos por meio da integração e análise de dados, em concordância com o propósito da Industria 4.0.

5.1 Complexo Eólico Corredor do Senandes

O CECS é um complexo eólico localizado no Município de Rio Grande – RS. O complexo é composto por sete centrais geradoras eólicas (CGE), também referidas como parques eólicos, que contam com aerogeradores do fabricante GE-ALSTOM, modelo ECO 122. Juntos, estes parques eólicos possuem potência instalada de 180.9 MW.

5.1.1 Estrutura e dados do CECS

Até o momento, quatro parques eólicos presentes no CECS já foram efetivamente construídos e são utilizados para gerar o montante de energia previsto. Estes quatro parques eólicos são denominados “Corredor do Senandes II” (8 aerogeradores); “Corredor do Senandes III” (10 aerogeradores), “Corredor do Senandes IV” (11 aerogeradores) e “Vento do Aragano I” (11 aerogeradores). Ao todo, tem-se 40 aerogeradores de 2.7 MW cada, que correspondem a 108 MW de potência instalada. Outros três parques – “Capão Grande”, “Vento do Aragano II”; e “Vento do Aragano III” – encontram-se em desenvolvimento e devem contar com outras 27 máquinas, que serão responsáveis por disponibilizar os 72.9 MW restantes para que o complexo conte com 180,9 MW de potência instalada. O CECS conta ainda com uma torre anemométrica para coleta de dados referentes às condições do vento na região. A Figura 34 apresenta a distribuição dos parques eólicos que compõem o CECS.

Além de dados do vento no local, há também a disponibilidade de uma base de dados de falhas que ocorreram nos aerogeradores nos períodos de outubro de 2017 a setembro de 2019. Esta base é composta pelo tipo e descrição da falha ocorrida em cada aerogerador, o instante em que a falha ocorreu e o instante em que a falha foi corrigida e o equipamento voltou a estar disponível. Conforme apontado por Silva (2020), é possível associar as condições de vento com as falhas ocorridas e, deste modo, valer-se dos dados para elaborar propostas de intervenção nos aerogeradores de modo a aumentar sua disponibilidade.

Neste exemplo, leva-se em conta os aerogeradores dos parques eólicos que se encontravam em operação, apresentados na Figura 34. Cada aerogerador é identificado por um conjunto de caracteres VAxyz ou Sxyz, em que VA e S correspondem aos parques eólicos “Vento do Aragano” e “Senandes”, respectivamente; xyz são números, em que x indica o parque eólico em que o aerogerador está instalado, enquanto y e z indicam o aerogerador

propriamente dito. Por exemplo, S301 corresponde ao aerogerador 1 (yz=01) do parque eólico “Senandes III” (S3).

Figura 34 - Parques eólicos e aerogeradores instalados no CECS.

Fonte: autoria própria.

5.1.2 Aerogerador

Para que seja possível correlacionar os dados de falhas e de condições ambientais, é importante que se tenha conhecimento prévio da aplicação. Assim, é necessário, primeiramente, conhecer os componentes básicos e funcionamento do aerogerador; selecionar as falhas a serem estudadas, compreender como se dão as falhas; e como o vento é capaz de influenciá-las. Por esta razão, tem-se uma descrição dos aerogeradores utilizados no CECS.

O aerogerador GE-ALSTOM ECO 122 é uma máquina de eixo horizontal. Este tipo de equipamento possui quatro subsistemas principais: rotor, *power-train*, nacele e torre (SPERA, 1979). O primeiro é composto pelas partes girantes do equipamento, sendo elas as pás, que

convertem a energia cinética do vento em trabalho mecânico no rotor; o cubo, que une as pás ao eixo de baixa velocidade (também chamado de eixo principal); e o sistema de *pitch* (ou sistema de passo), que promove a rotação das pás em torno de seu próprio eixo longitudinal (SILVA, 2020; SPERA, 1979).

O *power-train* é formado pelos componentes mecânicos e elétricos internos do aerogerador, tais como: eixos de baixa e alta velocidade, acoplados ao rotor e ao gerador, respectivamente; caixa multiplicadora, ou caixa de engrenagens, que multiplica a rotação do eixo de baixa velocidade (eixo principal) e a transfere ao eixo de alta velocidade (eixo secundário); gerador elétrico, que converte a energia mecânica em energia elétrica; freio mecânico, que reduz a rotação do rotor em caso de necessidade; além do subsistema de instrumentação e controle, que é composto por dispositivos de detecção, atuação e realização do controle (SPERA, 1979).

A nacele é uma estrutura que encapsula o subsistema de *power-train*, além de subsistema de *yaw*, que é responsável por posicionar o rotor de acordo com a direção do vento. Ainda, o tipo de aerogerador possui um anemômetro e cata-vento (ou biruta), que são dispositivos de detecção acoplados à nacele para medição da velocidade e direção do vento, respectivamente. Por fim, a torre é o componente que proporciona a sustentação e o posicionamento do rotor e da nacele, sendo dividida em segmentos chamados tramos. Três destes quatro subsistemas principais do aerogerador são ilustrados na Figura 35. O subsistema de *power-train* não é exibido, pois é encapsulado pela nacele. A Figura 36 apresenta uma ilustração dos componentes dos subsistemas mencionados.

5.1.3 Torre anemométrica

As condições ambientais de um parque eólico influenciam no desempenho dos aerogeradores (JERVELL, 2008; TAVNER *et al*., 2006, 2010). A direção e velocidade do vento são dois fatores importantes que caracterizam as condições ambientais do local e afetam diretamente a operação dos aerogeradores. A turbulência, que pode ser definida como a razão entre desvio padrão e a média da velocidade do vento horizontal, exerce influência na carga mecânica nos componentes do aerogerador. De acordo com Gasch e Twele (2012), a potência gerada por um aerogerador em condições normais é, aproximadamente, proporcional ao cubo da velocidade do vento. Por estas razões, é importante definir com precisão estas grandezas.

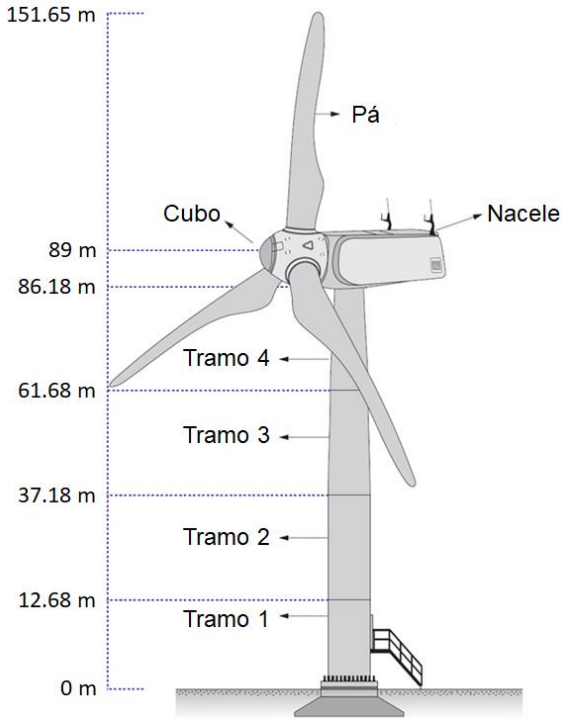


Figura 35 - Componentes básicos do aerogerador GE-ALSTOM ECO 122.

Fonte: adaptado de Silva (2020).

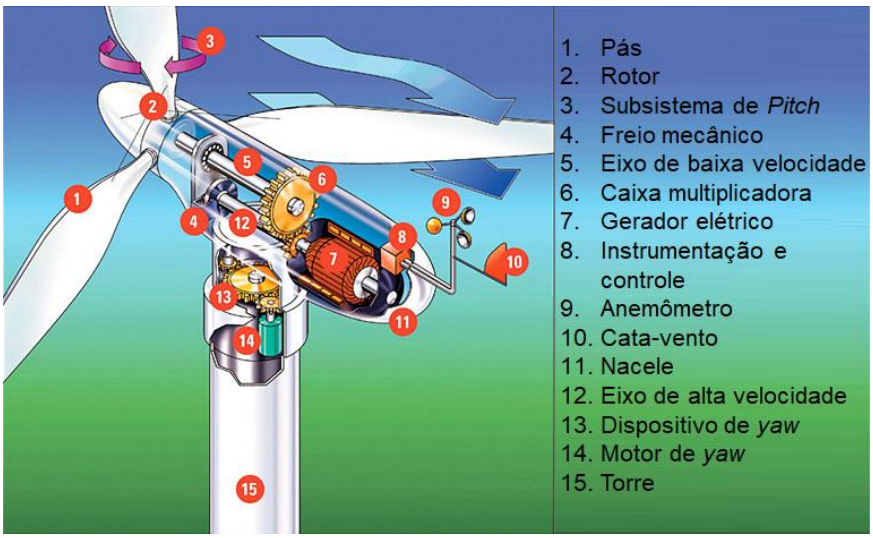


Figura 36 - Componentes detalhados de um aerogerador [30].

Em um parque ou complexo eólico, a torre anemométrica é o elemento responsável pela medição das características do vento da região, permitindo a avaliação do seu potencial eólico. Os dados de direção e velocidade do vento no CECS são obtidos a partir da Torre

[30] Kaufman Wind Energy: https://sites.google.com/site/kaufmanwindenergy/background-information-on-wind-turbines.

Anemométrica Capão Grande (TCG) (SILVA, 2020). Estas grandezas não são constantes, conforme mostram o histograma de velocidades e a rosa dos ventos elaborados a partir dos dados obtidos pela TCG. Como frequentemente é feito (GNOATTO, 2017), a distribuição da velocidade do vento é ajustada por uma distribuição de Weibull (Figura 37).

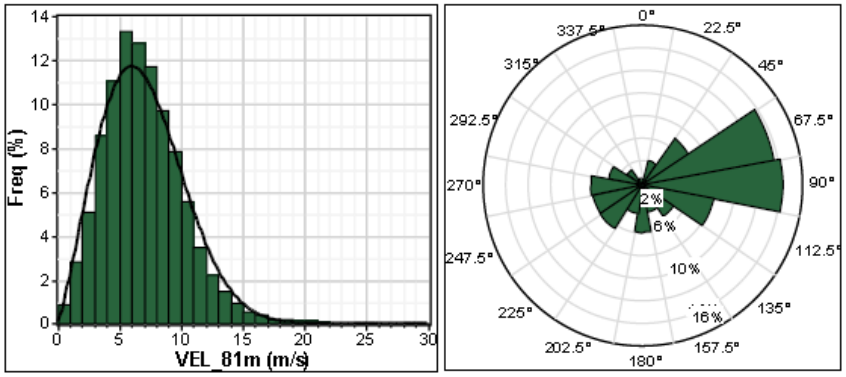


Figura 37 - Distribuição de Weibull da velocidade e Rosa dos Ventos.

Fonte: Silva (2020).

5.2 Arquitetura para o CECS

Para a aplicação da arquitetura proposta no Capítulo anterior no Complexo Eólico Corredor do Senandes, inicialmente, define-se a topologia a ser adotada. A arquitetura de *Data Marts* independentes é uma das topologias que em geral apresentam melhor desempenho, considerando quatro métricas qualitativas (WATSON; SUDIBYO, 2006), a saber:

- Qualidade da informação: diz respeito à consistência, acurácia e completude da informação;
- Qualidade do sistema: inclui flexibilidade, escalabilidade e integração;
- Impactos individuais: facilidade que os usuários encontram ao acessar e analisar os dados;
- Impactos organizacionais: aderência aos requisitos e objetivos de negócio, possibilidade de melhorias nos processos, nas estratégias de negócio, na comunicação e cooperação entre unidades organizacionais, etc.

Considerando a topologia de *Data Marts* independentes, pode-se considerar a própria composição do complexo eólico para organização dos módulos da arquitetura. Propõe-se o uso de uma instância de Módulo de Preparação de Dados (MPD) e uma de Módulo de *Data Mart*

(MDM) para cada parque eólico e para a torre anemométrica. Propõe-se também o uso de uma única instância de Módulo Consumidor de Dados (MCD) para o CECS. A Figura 38 ilustra esta arquitetura. Foram utilizadas cores diferentes além de números para representação da interação entre os módulos a fim de facilitar a compreensão da Figura e o fluxo de dados indicado pelas setas. Na Figura 38, o fluxo de dados para análise tem início pelas linhas e setas em verde (1), em que os dados partem dos MFDO em direção aos MPD. Posteriormente, os dados seguem aos MDM pelos fluxos indicados em azul (2). Em seguida, os dados seguem ao MCD, que acessa todas as instâncias de MDM. Este fluxo é representado em laranja (3) Por fim, os dados analisados por este módulo devem ser devolvidos às instâncias de MFDO na forma de modelos estatísticos, conforme indicado pelos fluxos em preto (4). Nas subseções seguintes, os módulos da arquitetura são descritos e os respectivos AAS são representados.

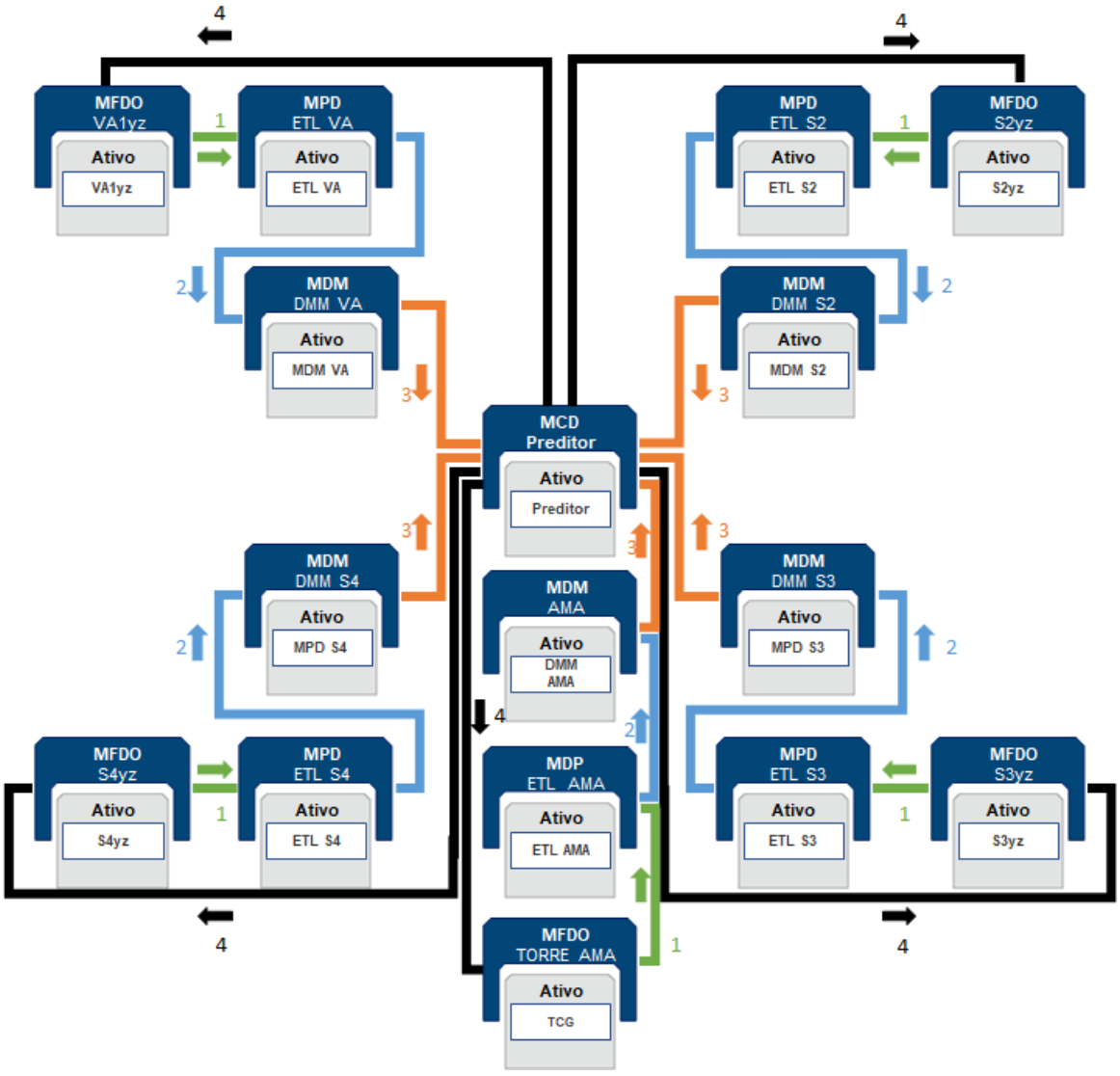


Figura 38 - Arquitetura para o CECS.

Fonte: autoria própria.

5.2.1 MFDO: aerogerador e torre anemométrica

Os ativos das fontes de dados operacionais no CECS são os aerogeradores e a torre anemométrica. Os dados das máquinas, associados às condições ambientais, fornecem o insumo para que os algoritmos de aprendizado de máquina possam realizar a predição de falhas passíveis de ocorrência. Inicialmente, são apresentados os modelos conceituais para o AAS destes dois diferentes tipos de instâncias do MFDO que têm como ativo os aerogeradores e a torre anemométrica.

A elaboração do AAS para este módulo é baseada na série de padrões IEC-61400-25 (IEC - INTERNATIONAL ELECTROTECHNICAL COMMISSION, 2015). Este conjunto de padrões define especificações para a comunicação entre os componentes de sistemas de geração de energia eólica e seus sistemas de controle e monitoramento. A fim de garantir a interoperabilidade entre estas duas partes, o padrão IEC-61400-25-2 estabelece o modelo de informação para componentes dos sistemas de energia eólica, com as definições sintáticas e semânticas para os elementos deste modelo. Optou-se por utilizar o padrão IEC em detrimento de outros como o eCl@ss por se tratar de um modelo de informação utilizado por importantes fabricantes de componentes de parques eólicos como WEG e Alstom.

O primeiro elemento que compõe o modelo de informação deste padrão é denominado "Nó Lógico" (LN, do inglês *Logical Node*). Um nó lógico reúne uma série de elementos que dizem respeito a uma perspectiva em particular do componente que está sendo descrito. Este componente pode ser um aerogerador, um parque eólico ou um "dispositivo não-aerogerador", em uma tradução da expressão do termo em inglês *non-turbine device*, utilizado no padrão. Os LNs associados ao aerogerador são utilizados para descrever seus componentes e subsistemas como o rotor, o gerador, a nacele, o subsistema de *yaw*, etc. Os parques eólicos podem ser descritos em termos de seus aerogeradores, de informações meteorológicas, de disponibilidade, etc., sendo que cada uma destas perspectivas é associada a um LN. Dispositivos não aerogeradores, como é o exemplo da torre anemométrica, possuem LNs para descrever informações meteorológicas, alarmes, de relatórios, etc. A Figura 39 apresenta um esquema para descrição de componentes de sistemas de geração de energia eólica por meio de LNs, representados por retângulos amarelos e cinzas. Vale destacar que nem todos os nós lógicos são obrigatórios para descrição do componente.

Os LNs são compostos por elementos denominados objetos de dados (DO, do inglês *data objects*). Estes podem ser entendidos como atributos, pois são utilizados para descrever o

componente dentro da perspectiva em que se encontram. Dentro dos LNs, os DOs são agrupados em sete classes, que são: descrições, informações comuns, informações de estado, valores medidos e calculados (ou valores analógicos)[31], controles, informações opcionais e configurações. Tal como os LNs, existem DOs que não são obrigatórios. Além disso, o padrão prevê que DOs que não foram previamente definidos sejam criados para descrever aspectos não contemplados.

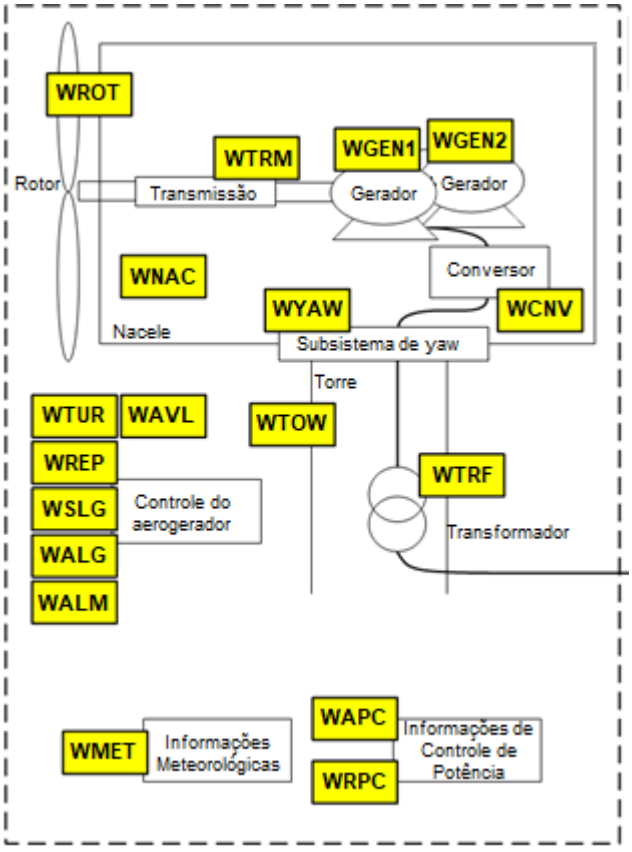


Figura 39 - Representação dos nós lógicos nos sistemas de energia eólica.

Fonte: adaptado de IEC - International Electrotechnical Commission (2015).

O padrão em questão define ainda elementos adicionais para caracterizar os DO. Para os explicar, toma-se como exemplo aqueles pertencentes à classe de valores medidos, que geralmente consistem em variáveis aferidas com certa frequência, tais como a velocidade do vento ou temperatura de algum componente. A partir destas variáveis, pode-se obter

[31] A versão de 2006 do padrão IEC 61400-25-2 se refere à esta classe como *analog information*. A versão de 2015 a denomina *measured and metered values*.

informações estatísticas como valor médio, máximo, mínimo e desvio padrão. Dito de outra maneira, estas métricas podem ser associadas ao DO.

A tarefa principal para representar os módulos do MFDO no AAS é o mapeamento entre o modelo de informação da IEC 61400-25-2 e a estrutura do AAS. Iniciando pelos LNs, pode-se associá-los aos Submodelos. Ambos são compostos por uma série de elementos utilizados para definir o ativo dentro de uma perspectiva específica. Traçando um paralelo entre as hierarquias do padrão em questão e do AAS, as classes de DO podem ser mapeadas nas Coleções de Elementos de Submodelo, já que ambas agrupam elementos com certo grau de similaridade dentro da perspectiva em que se encontram. Os DOs, por sua vez, podem ser mapeados nos Elementos de Submodelo, uma classe mais ampla do AAS que possui como subclasses os Elementos de Dados, Propriedades, Referências, Arquivos, etc. Por fim, os elementos que adicionam características aos atributos podem ser mapeados nos qualificadores, que exercem a mesma função para os Elementos de Submodelo no AAS. Este mapeamento é apresentado na Tabela 22.

Tabela 22 - Mapeamento entre IEC 61400-25-2 e AAS.

IEC 61400-25-2	**AAS**
Nós lógicos	Submodelos
Classes de objetos de dados	Coleção de Elementos de Submodelo
Objeto de dados	Elemento de Submodelo
Características dos objetos de dados	Qualificadores

Fonte: autoria própria.

A partir deste mapeamento, é possível criar o AAS para o MFDO. Com relação ao aerogerador, os Submodelos obrigatórios são:

1. WTUR: informações gerais do aerogerador;
2. WROT: informações do rotor;
3. WGEN: informações do gerador (em máquinas que possuem dois, deve-se definir WGEN1 e WGEN2);
4. WNAC: informações da nacele;
5. WYAW: informações do subsistema de yaw;
6. WALM: informações de alarme;

Além dos Submodelos correspondentes aos LNs obrigatórios de acordo com o padrão considerado, foi definido um Submodelo denominado WPDM (um acrônimo para o termo

"*Wind Turbine Prediction Model*") para contemplar as informações referentes aos modelos estatísticos que o aerogerador, enquanto MFDO, deve receber do MCD. Para a torre anemométrica, não existem LNs obrigatórios, então optou-se por incluir apenas o Submodelo WMET (informações meteorológicas) e WALM, também presente para o aerogerador.

As Figuras 40 e 41 representam esquematicamente o AAS para o aerogerador e a torre anemométrica, respectivamente. Nestas Figuras, geradas a partir do *software* AASX Package Explorer[32], os Submodelos são representados pelo símbolo SM seguidos de seu identificador local, o idshort, juntamente com o identificador global, o id. O último pode ser de vários tipos e, neste exemplo, escolheu-se o IRI (Identificador de Recurso Internacionalizado). As Coleções de Elementos de Submodelos são representadas pelo símbolo SMC seguido do idShort. Para esta implementação, os DOs foram mapeados como Propriedades (que pertencem à classe de Elementos de Submodelo) e são identificados pelo idShort seguido da unidade de medida e os qualificadores. O Submodelo "WTUR" possui, na Coleção de Elementos Submodelo *MeasuredAndMeteredValues* (Valores Medidos) as Propriedades W (geração de potência ativa) e Var (geração de potência reativa), medidas respectivamente em W e Var. Para estas duas Propriedades, foram atribuídos os qualificadores de valor médio de 10 minutos e o registro de data e hora.

Todos os elementos do AAS possuem atributos que os caracterizam e descrevem (não confundir com os DOs da IEC 61400-25-2). Aqueles de identificação já foram mencionados, como o id e idShort. Pode-se mencionar ainda o id semântico, fundamental para a garantia de interoperabilidade. Estes elementos possuem ainda uma descrição, que pode ser apresentada em vários idiomas, e uma classificação quanto à tipo ou instância. Por fim, os Elementos de Submodelos, incluindo as Coleções, possuem ainda a Descrição Conceitual, que contempla os elementos já mencionados e alguns outros, como a unidade de medida no caso das Propriedades. Na Figura 42, tem-se um exemplo dos atributos mencionados para a Coleção *MeasuredAndMeteredValues*, que é um Elemento de Submodelo.

[32] https://github.com/admin-shell-io/aasx-package-explorer

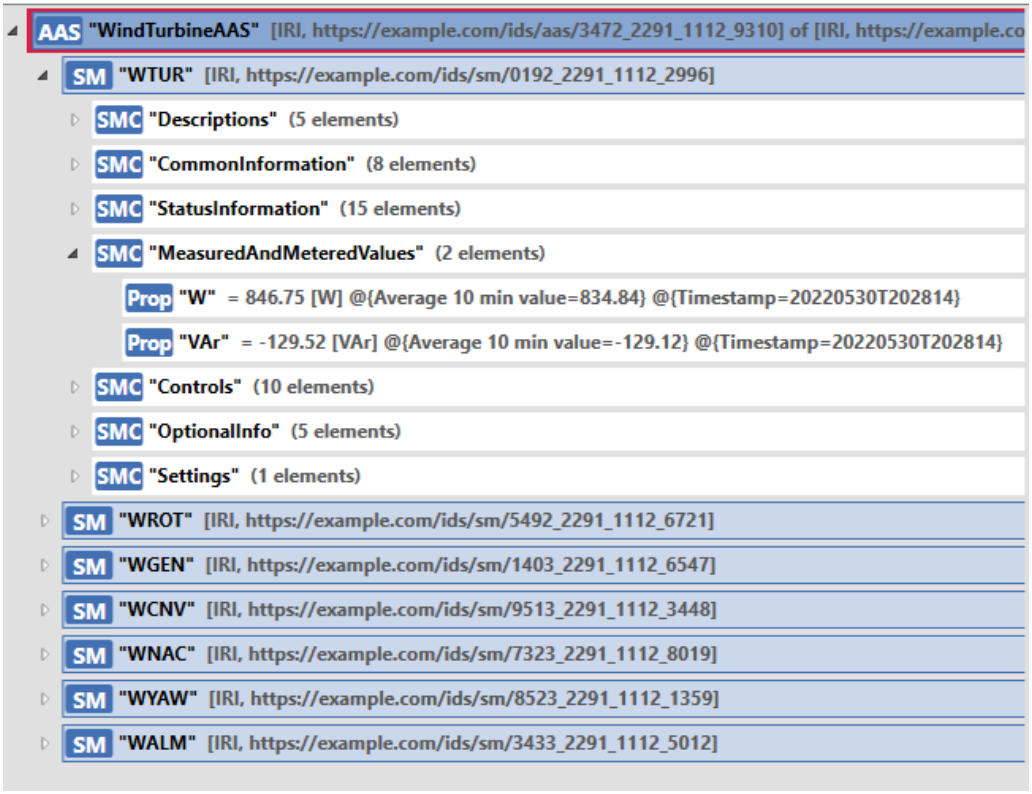


Figura 40 - Representação do AAS do aerogerador.

Fonte: autoria própria.

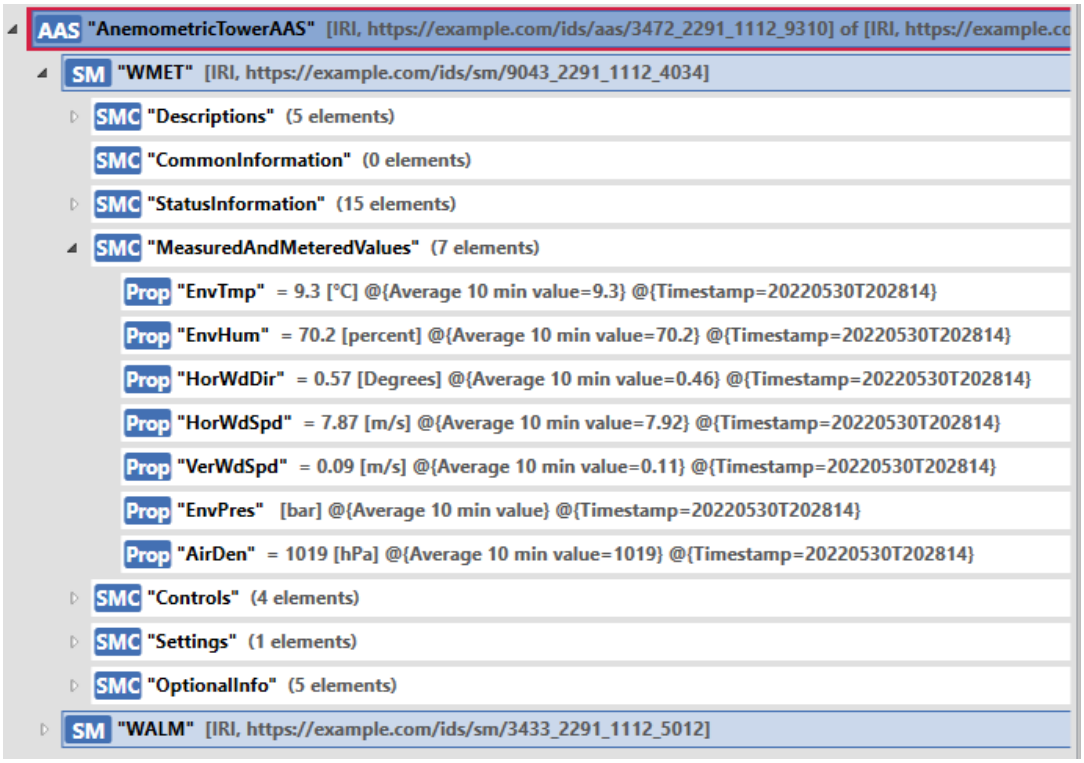


Figura 41 - Representação do AAS da torre anemométrica.

Fonte: autoria própria.

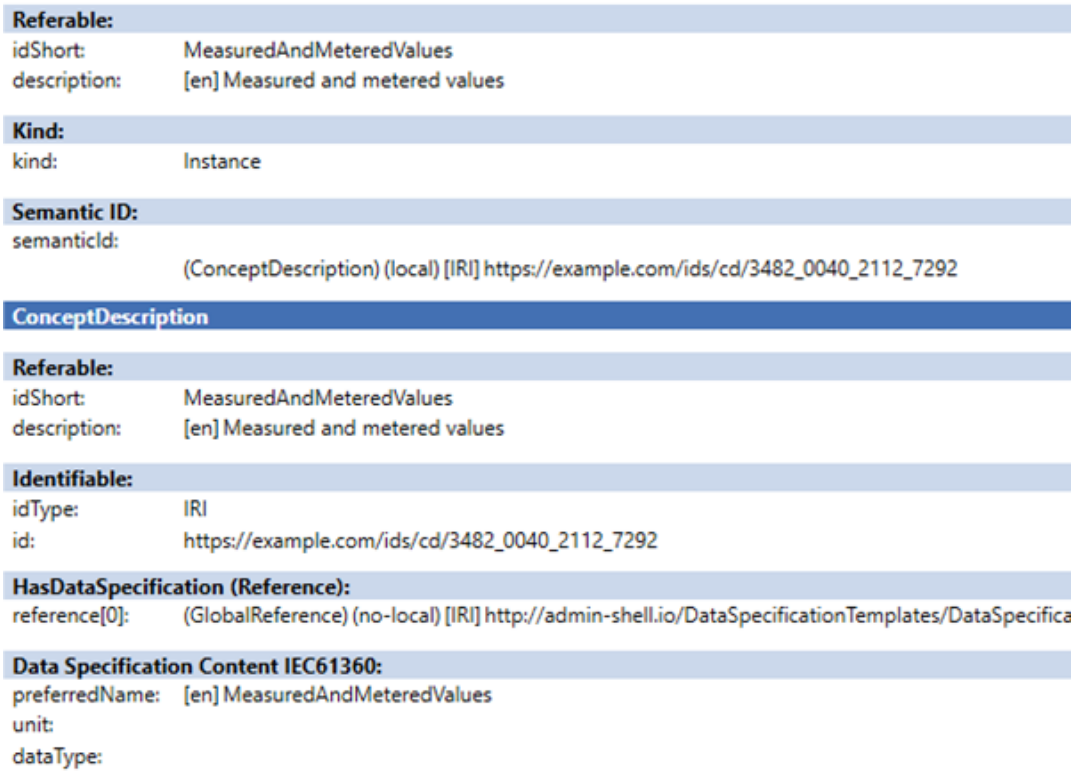


Figura 42 - Atributos de um Elemento de Submodelo.

Fonte: autoria própria.

5.2.2 MCD: Preditores

Tendo em vista que os sistemas de *Data Warehouse* são fortemente orientados às necessidades dos usuários finais, após apresentar uma descrição do MFDO, tem-se a descrição do MCD. Neste exemplo de aplicação, é atribuída ao MCD a função de realizar o aprendizado de modelos estatísticos que proporcionem melhorias na operação do complexo eólico. O aerogerador tem uma grande quantidade de componentes e subsistemas, entretanto, neste caso, o escopo está limitado à ocorrência de falhas. A base de dados de falhas dos aerogeradores permite identificar um tipo de falha em particular que ocorre no subsistema de *yaw* – denominada "sobrecarga no motor de *yaw*" – que foi responsável por causar mais de 2780 horas de indisponibilidade dos aerogeradores. Isto significa que, durante um período de 729 dias (01/10/2017 a 30/09/2019), o tempo somado de indisponibilidade dos aerogeradores equivale a um aerogerador parado por 116 dias. Dada a relevância deste tipo de falha, optou-se por estudá-la com maior profundidade.

O subsistema de *yaw* é responsável por posicionar o rotor em uma direção ótima de acordo com a direção do vento. A falha de sobrecarga no motor de *yaw* ocorre durante a rotação do rotor e, consequentemente da nacele, ao longo do eixo longitudinal da torre. Em um primeiro momento, o sistema de *yaw* faz com que o conjunto rotor e nacele estejam posicionados com uma determinada orientação. Quando o vento sofre uma variação superior a 15° na direção por ao menos 10 minutos, o dispositivo de realização do controle do subsistema de *yaw* envia um comando para que o motor de *yaw* promova a rotação do rotor e da nacele, de modo a alinhá-los à direção do vento. Se o vento mudar de direção durante o alinhamento do rotor, pode surgir um torque resistente ao movimento de rotação do motor de *yaw*, cuja intensidade é influenciada pela velocidade do vento. Neste caso, o sistema de proteção de sobrecarga de *yaw* impede o motor de operar.

Com base na descrição de como se dá a falha por sobrecarga no motor de *yaw*, é possível ter indicativos de como as condições de vento influenciam esta ocorrência. Verifica-se a necessidade de ter dados de direção e a velocidade do vento a cada 10 minutos, que são parâmetros contínuos de operação do dispositivo. É necessário ter também o registro das falhas que ocorreram nos dispositivos, um parâmetro operacional discreto. As condições de vento e os dados de falha correspondem, respectivamente, aos atributos e às variáveis de classe do modelo.

A Figura 43 ilustra o AAS para o MCD. Ele é composto por dois dos Submodelos comuns para todos os ativos de acordo com o dicionário eCl@ss: Identificação e Documentação. Em se tratando de um ativo de *software*, foi criado um Submodelo denominado Connection, em que as especificações do MCD cliente estão contempladas, bem como os relacionamentos estabelecidos com os demais módulos. O AAS contém ainda um Submodelo específico, cujo idShort é “PredictionModels”, cuja composição é descrita em detalhes a seguir.

Todos os modelos estatísticos podem estar contidos na forma de Coleções de Elementos de Submodelo. Como exemplo, considera-se apenas uma única coleção correspondente ao modelo estatístico de predição de falhas por sobrecarga no motor de *yaw*. Os elementos que caracterizam este modelo estatístico são o método, procedimento e técnica de aprendizado, as variáveis consideradas e os hiperparâmetros. Os três primeiros elementos foram mapeados em Propriedades de AAS. As variáveis de entrada e saída foram implementadas na forma de Elementos de Referência. Seu valor é composto pelo Submodelo e idShort do Elemento que está sendo referenciado, bem como um valor booleano que indica se o elemento referenciado está presente no contexto do AAS. Por fim, os hiperparâmetros do modelo estatístico foram

mapeados em Propriedades reunidas dentro de uma Coleção de Elementos de Submodelo. Para este estudo, foram considerados os hiperparâmetros utilizados pela biblioteca de *Python Sckit Learn* para o modelo de regressão logística.

Os elementos deste Submodelo devem ser referenciados pelas instâncias de MFDO que têm como ativo o aerogerador, para compor o conteúdo do Submodelo WPDM. Estas referências caracterizam o fluxo de retorno dos dados do MCD ao MFDO e permitem que o modelo estatístico seja utilizado na predição de falhas em tempo real. Este Submodelo fornece uma descrição completa dos modelos de aprendizado de máquina e permite que qualquer ferramenta possa aprender e aplicar estes modelos, reproduzindo os mesmos resultados independentemente da tecnologia adotada.

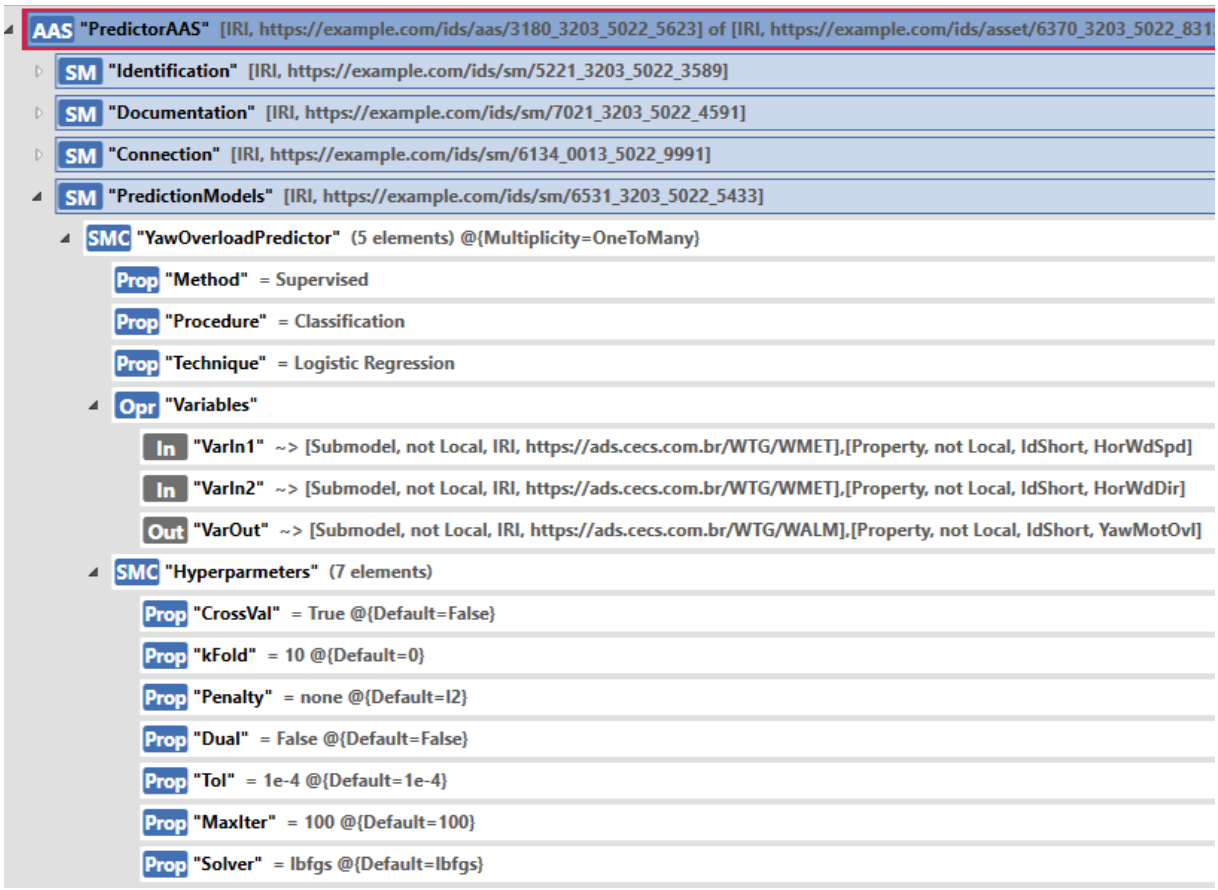


Figura 43 - Representação do AAS do MCD.

Fonte: autoria própria.

5.2.3 MDM: *Data Mart*

A descrição do MDM é feita partindo do mais genérico para o mais específico. A Figura 44, que apresenta uma representação esquemática do AAS deste módulo, possui os Submodelos organizados desta mesma maneira. Em termos genéricos, este módulo tem como ativo um *software* e, por esta razão, o seu AAS é composto por Submodelos comuns a todos os ativos desta natureza: Identificação, Documentação e Conexão. Conforme visto na Subseção anterior, os dois primeiros Submodelos têm a sua composição baseada no dicionário eCl@ss. Com relação ao terceiro, são definidos atributos do servidor de banco de dados, tais como *hostname*, porta, tipo da conexão e credenciais de acesso. Estes atributos garantem conexão ao ativo deste módulo.

O quarto Submodelo do MDM contempla seus metadados. Com base na classificação para os metadados apresentada no Capítulo 3, foram definidas três Coleções de Elementos de Submodelo:

- Metadados de negócio: esta Coleção é composta pelos elementos definidos no padrão IEC 61400-25-2 para parques eólicos. Como o ativo em questão é um *Data Warehouse* e não um parque eólico, o formalismo proposto pelo padrão não foi rigorosamente adotado. Em vez disso, optou-se por incluir diretamente os objetos de dados dos LNs definidos para um parque eólico, incluindo sua nomenclatura, tais como: a geração de potência ativa e o valor de referência para esta variável, o número de aerogeradores em operação e a quantidade de paradas e partidas destas máquinas;
- Metadados operacionais: a Coleção contempla os dados de operação do ativo deste módulo. Indica o estado da instância de *Data Mart*, a última modificação em seu conteúdo, além de registros de informação, avisos e erros ocorridos durante as operações realizadas neste *Data Mart*. Estes três últimos elementos foram representados na forma de Eventos e têm como qualificador o instante de tempo em que foram registrados;
- Metadados técnicos: é composta pelas informações associadas à estrutura do *Data Mart*. Alguns de seus atributos são comuns a este tipo de ativo, como a política de escalonamento, o tamanho, número de *clusters* em que os dados são armazenados, etc. Outros elementos desta Coleção são específicos para a instância de *Data Mart*, considerando que este é fortemente orientado às análises a serem realizadas no MCD. Estas especificidades do *Data Mart* de acordo com o consumidor de dados estão contempladas em seu esquema, representado na forma de uma Coleção de Elementos

de Submodelo. Foi mencionado no Capítulo 3 que o modelo de dados multidimensional é amplamente utilizado para este tipo de banco de dados. Sendo assim, o esquema é composto por duas outras Coleções: as dimensões (SMC DimensionTable) e os fatos (SMC FactTable). Ambos, no nível de dados lógico, são instanciados na forma de tabelas, que possuem chaves primárias e secundárias, atributos e restrições, além de outras características.

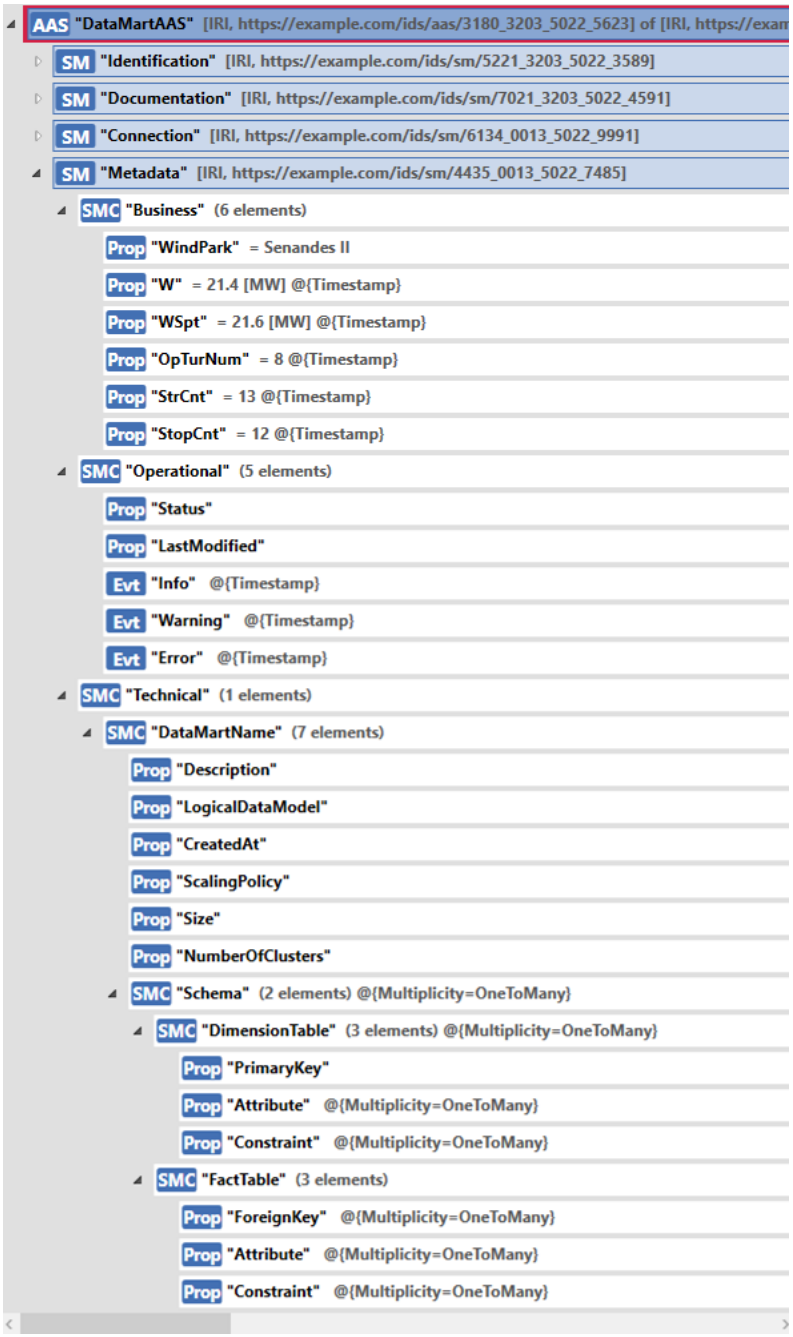


Figura 44 - Representação do AAS do MDM.

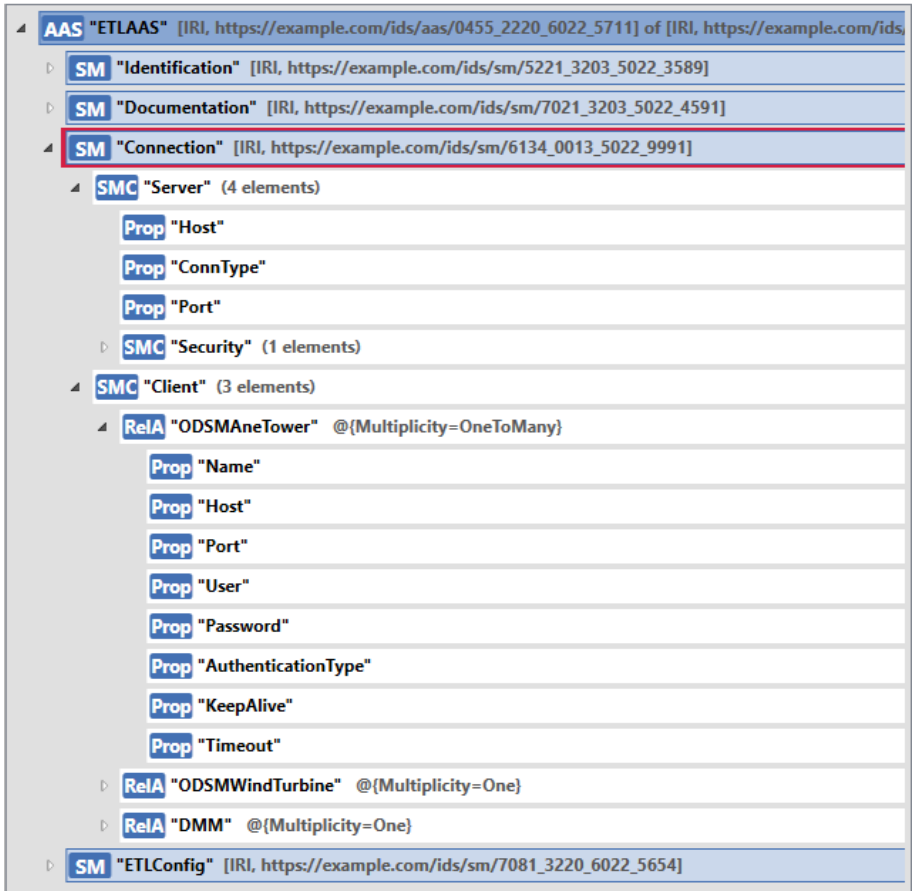
Fonte: autoria própria.

5.2.4 MPD: Ferramenta de ETL

As instâncias de MPD a serem usadas em cada parque eólico têm como ativo uma ferramenta de ETL. Além dos Submodelos padrão para identificação e documentação, este módulo tem outros dois modelos: *Connection* e *ETLConfig*. O primeiro é comum aos demais I4.0Cs da arquitetura, já que este fenômeno pressupõe efetiva interconexão entre as partes envolvidas nos processos e serviços da Fábrica Inteligente. O segundo apresenta as configurações necessárias para definição dos chamados mapas nas ferramentas de ETL. A seguir, os Submodelos são descritos.

- Submodelo *Connection*: sua estrutura é apresentada na Figura 45 (a). Este Submodelo é composto por duas Coleções de Elementos de Submodelo, servidor e cliente. A primeira contempla os atributos do servidor, como o *host* (na forma de endereço IP, por exemplo); o tipo de conexão que pode ser estabelecida (OPC UA, HTPP, MQTT); as portas disponíveis para conexão; e uma classe de elementos do AAS denominada Segurança. A Coleção cliente, por sua vez, define os relacionamentos estabelecidos com os demais módulos da arquitetura. No caso do MPD, são estabelecidas interações com as instâncias de MFDO de Aerogerador e Torre Anemométrica, além do MDM. Para cada caso, foi definido um Elemento de Relacionamento Anotado, que possui o valor e a anotação. O valor é composto pelo AAS dos dois módulos envolvidos na interação. A anotação é uma composição de Propriedades que definem características desta interação: nome, *host* e porta do módulo que atua como servidor, credenciais utilizadas na conexão, tipo da autenticação, etc;
- Submodelo *ETLConfig*: a estrutura deste Submodelo é apresentada na Figura 45 (b). Ele possui três Coleções de Elementos de Submodelo. A primeira, *Target*, é referente ao processo de extração e define as fontes de dados na forma de Referências. No caso do MPD, a referência é feita ao Elemento de Submodelo a ser extraído da instância de MFDO. A Coleção *Transformation* estabelece, na forma de Propriedades, as operações a serem realizadas nos dados provenientes do MFDO. A Coleção *Load* é análoga à *Target* e define na forma de Referência a fonte de dados – neste caso a instância de MDM – que deve receber os dados processados. Este submodelo possibilita a parametrização da ferramenta de ETL a ser utilizada, possibilitando que qualquer *software* conectado a este AAS possa realizar as mesmas tarefas desde que disponham de todas as funcionalidades inseridas na Coleção de Elementos de Submodelo em questão.

a)



b)

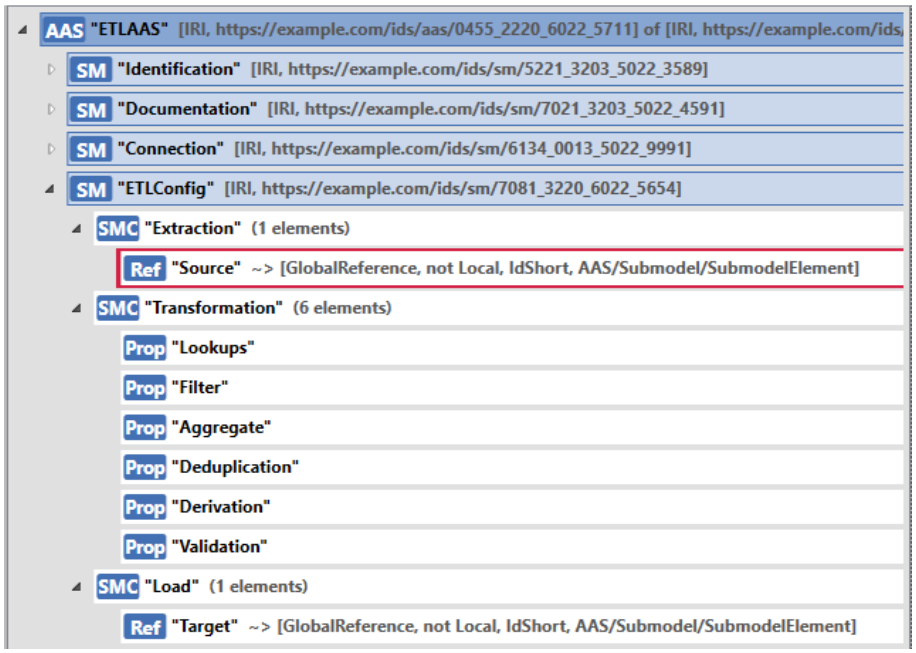


Figura 45 - Representação do AAS do MPD.

Fonte: autoria própria.

5.2.5 MGA: *software* gerenciador da arquitetura

A instância de MGA é única para todo o complexo eólico e tem como ativo um *software* para gerenciamento da arquitetura. Além dos Submodelos para identificação, documentação e conexão, este módulo tem cinco Submodelos de mesmo significado semântico, isto é, com o mesmo *template*, sendo um para cada parque. Estes submodelos contêm a lista de instâncias de cada um dos módulos referentes ao seu respectivo parque, bem como registros de eventos acerca dos processos que ocorrem na arquitetura. Por fim, no Capítulo 4, foi dito que o MGA deve ser capaz de associar as características de uma instância de um determinado módulo aos parâmetros dos demais módulos. É importante destacar que estas informações, que podem ser entendidas como regras de negócio, não são contempladas no AAS deste módulo, mas em seu ativo. O AAS do MCD é exibido na Figura 46.

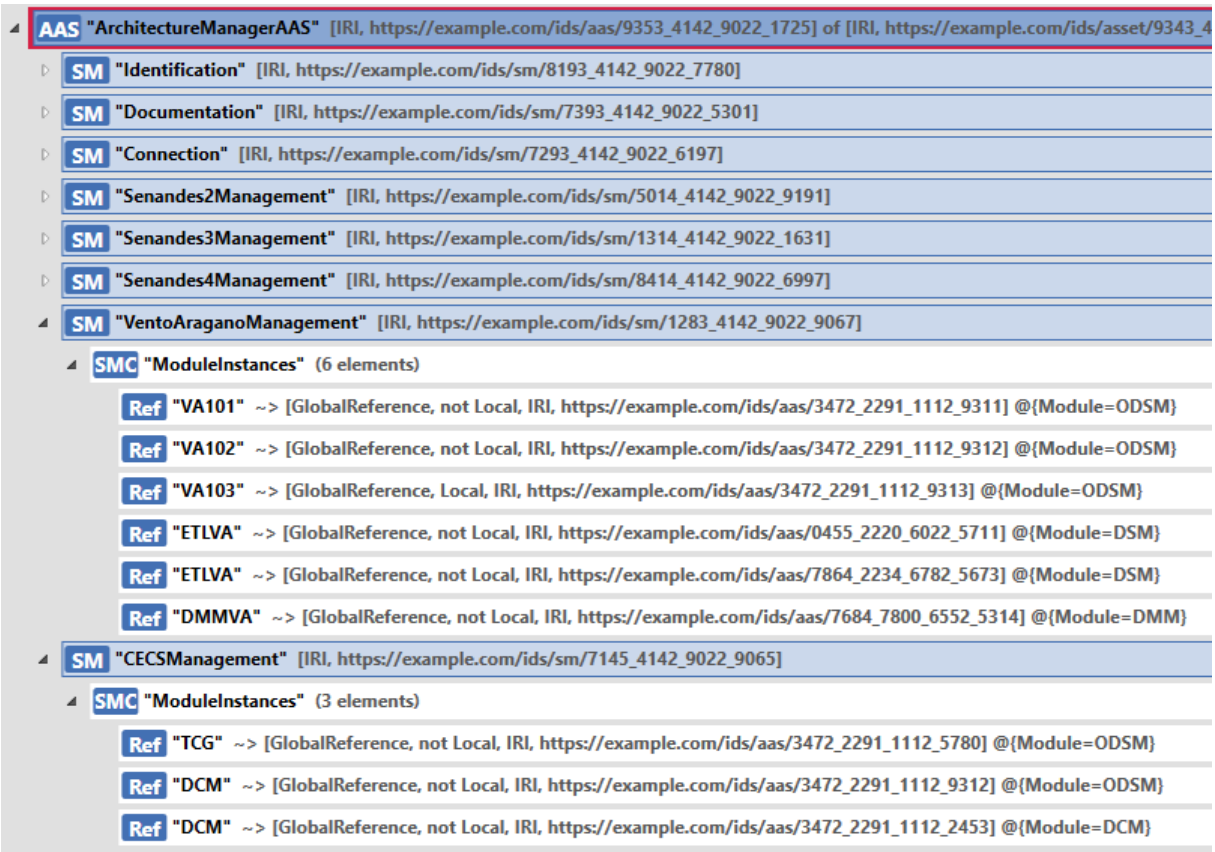


Figura 46 - Representação do AAS do MGA.

Fonte: autoria própria.

5.3 Predição de Falhas de Yaw

Aqui se apresenta um exemplo de uso de aprendizado de máquina na Indústria 4.0. Conforme descrito no Capítulo 4, existem duas reflexões a serem realizadas ao projetar este módulo: (i) qual é a natureza dos dados disponíveis para as análises; (ii) qual é o objetivo das análises a serem realizadas no MCD? A partir destas reflexões, pode-se determinar o método e procedimento de aprendizado para elaboração do modelo estatístico e comparar as técnicas de aprendizado de máquina disponíveis.

Os dados provenientes para as análises são oriundos das instâncias de MFDO, que têm como ativos o aerogerador e a torre anemométrica. A Figura 47 ilustra a fração do AAS destas duas instâncias a partir do preenchimento da estrutura da Figura 40 e Figura 41. Ao longo do eixo "Ciclo de Vida e Cadeia de Valor" do RAMI 4.0, estes ativos se encontram na fase de Instância, na etapa de Manutenção e Uso. Com base na Tabela 21, existem cinco possibilidades principais de uso dos dados para esta condição de ativo: monitoramento, diagnóstico e prognóstico de condição, escalonamento *online* e controle do ativo. Entre estas possibilidades, optou-se pelo prognóstico de condição dos aerogeradores com o intuito de realizar, por meio de aprendizado supervisionado, a predição de falhas que possam ocorrer.

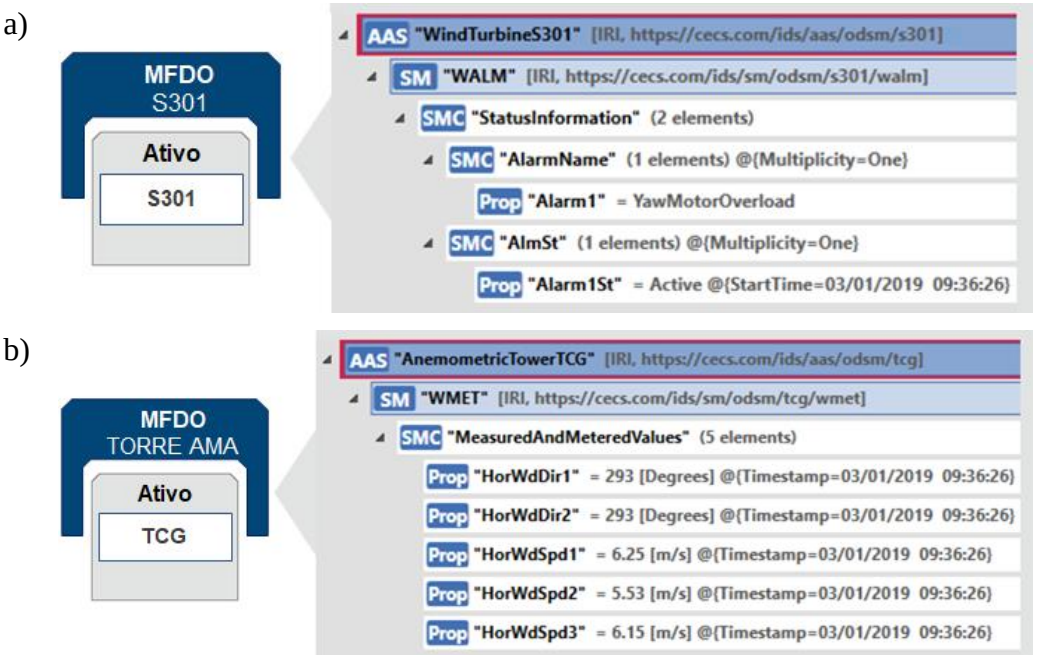


Figura 47 - Instâncias de MFDO e a fração do AAS de aerogerador (a) e torre anemométrica (b) com os dados utilizados na análise.

Fonte: autoria própria.

A primeira tarefa para realizar o aprendizado supervisionado é a preparação dos dados. A base de dados utilizada com as condições do vento contém 2160 registros (01/01/2019 a 15/01/2019) e é composta pelos seguintes atributos:

- Data e horário do registro, com frequência de 10 minutos e resolução de 1 segundo;
- Valor médio, máximo, mínimo e desvio padrão da velocidade do vento, em m/s. A velocidade foi medida a 89m, 87,5m e 60m de altura em relação ao solo. A resolução era de duas casas decimais;
- Turbulência, como o produto entre o desvio padrão e a velocidade média do vento. A turbulência foi calculada a 89m, 87,5m e 60m de altura em relação ao solo. A resolução era de duas casas decimais;
- Valor médio e desvio padrão da direção do vento, em graus. A direção foi medida a 86m e 30m de altura em relação ao solo. A resolução era de um grau.

Além dos atributos, a base de dados contém registros de falha de sobrecarga no motor de *yaw*, de 22/09/2017 a 19/10/2019. Para a construção desta base, os dados de todas as instâncias de MFDO precisam ser agregados e ter suas estatísticas extraídas (valor médio, máximo, mínimo, desvio padrão). Posteriormente, deve ser feito o cálculo de turbulência e, por fim, o carregamento dos dados na instância de MDM. A Figura 48 ilustra a instância de MPD com a fração do AAS responsável por realizar os processamentos mencionados.

A etapa seguinte da preparação de dados consiste em selecionar os registros para os quais havia informações disponíveis acerca das condições de vento. Para os registros restantes desta seleção, são determinadas as condições de vento no momento da falha. Posteriormente, selecionam-se os registros em que as condições de velocidade e direção do vento são semelhantes ao da condição das falhas. Com isso, é possível obter uma base de dados com condições de vento semelhantes e, para cada registro, o estado do alarme de sobrecarga no motor de *yaw*, indicando ou não a ocorrência de uma falha.

Espera-se que, em condições normais de funcionamento, as falhas sejam pouco frequentes. Isto se traduz em uma base de dados desbalanceada, com um número consideravelmente maior de registros sem falha. Por isso, faz-se necessário realizar o balanceamento da base de dados. Ao final, a base contém 100 registros de falha e 425 de operação normal.

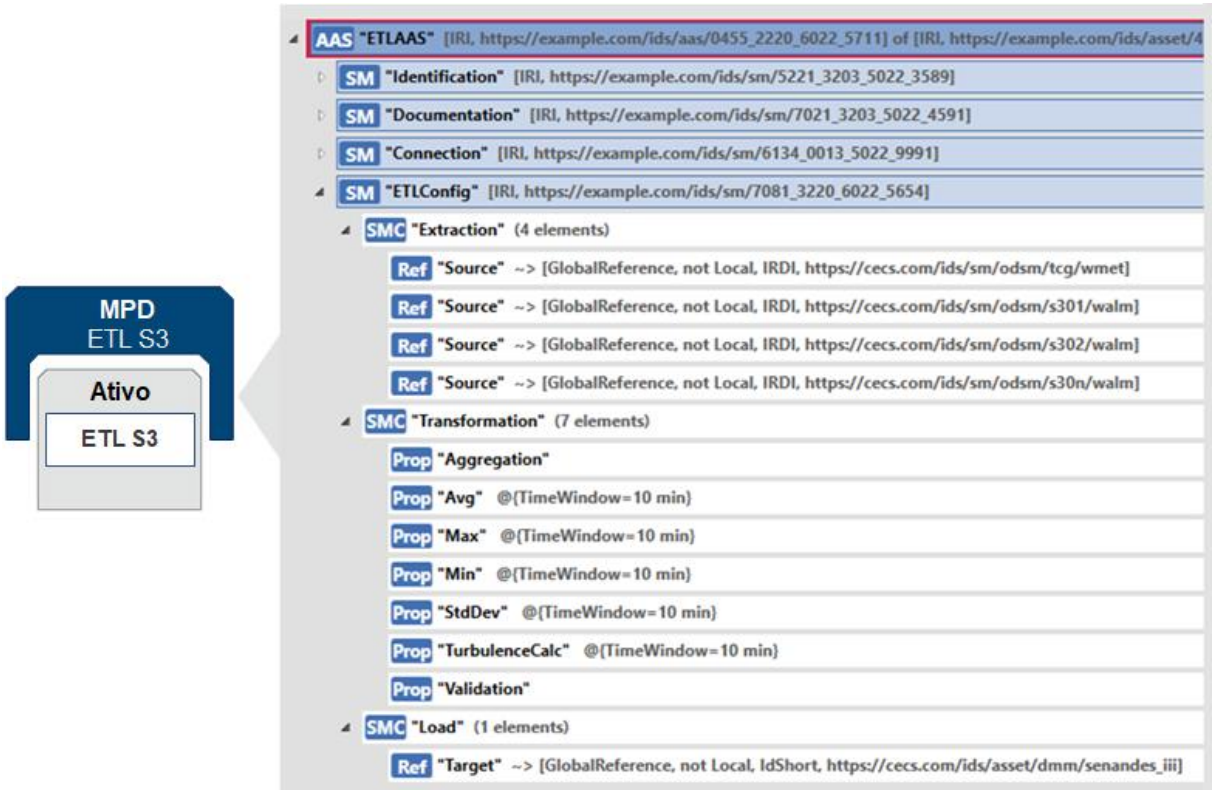


Figura 48 - Instância de MPD e a fração do AAS responsável pela parametrização das tarefas de ETL.

Fonte: autoria própria.

O passo seguinte consiste em determinar os atributos que serão utilizados. Vale ressaltar que a escolha por um subconjunto de atributos ao invés de se utilizar todos os disponíveis é um passo necessário para o desenvolvimento do modelo estatístico, proporcionando maior velocidade à execução do algoritmo, compreensibilidade do modelo e melhores métricas de qualidade aos resultados obtidos (KUMAR, 2014). Ao todo, 21 atributos estavam disponíveis (listados anteriormente). Kumar (2014) apresenta uma série de algoritmos que podem ser adotados para a seleção de atributos. Calcula-se o grau de correlação de Pearson entre os atributos e a variável de classe e os que apresentam correlação maior que 25% são selecionados. Verifica-se que não há ganho significativo nas métricas de avaliação dos modelos ao se utilizar atributos com menor correlação. Os atributos a serem adotados são: o desvio padrão da velocidade do vento para as três alturas indicadas e a velocidade máxima do vento a 60 m e a 87,5 m em relação ao solo. A Figura 49 apresenta a instância de MDM com o AAS organizado de modo a representar o esquema floco de neve, isto é, com as tabelas de dimensão (contendo os atributos utilizados na análise) e de fato (contendo o estado do alarme de sobrecarga no motor de *yaw*).

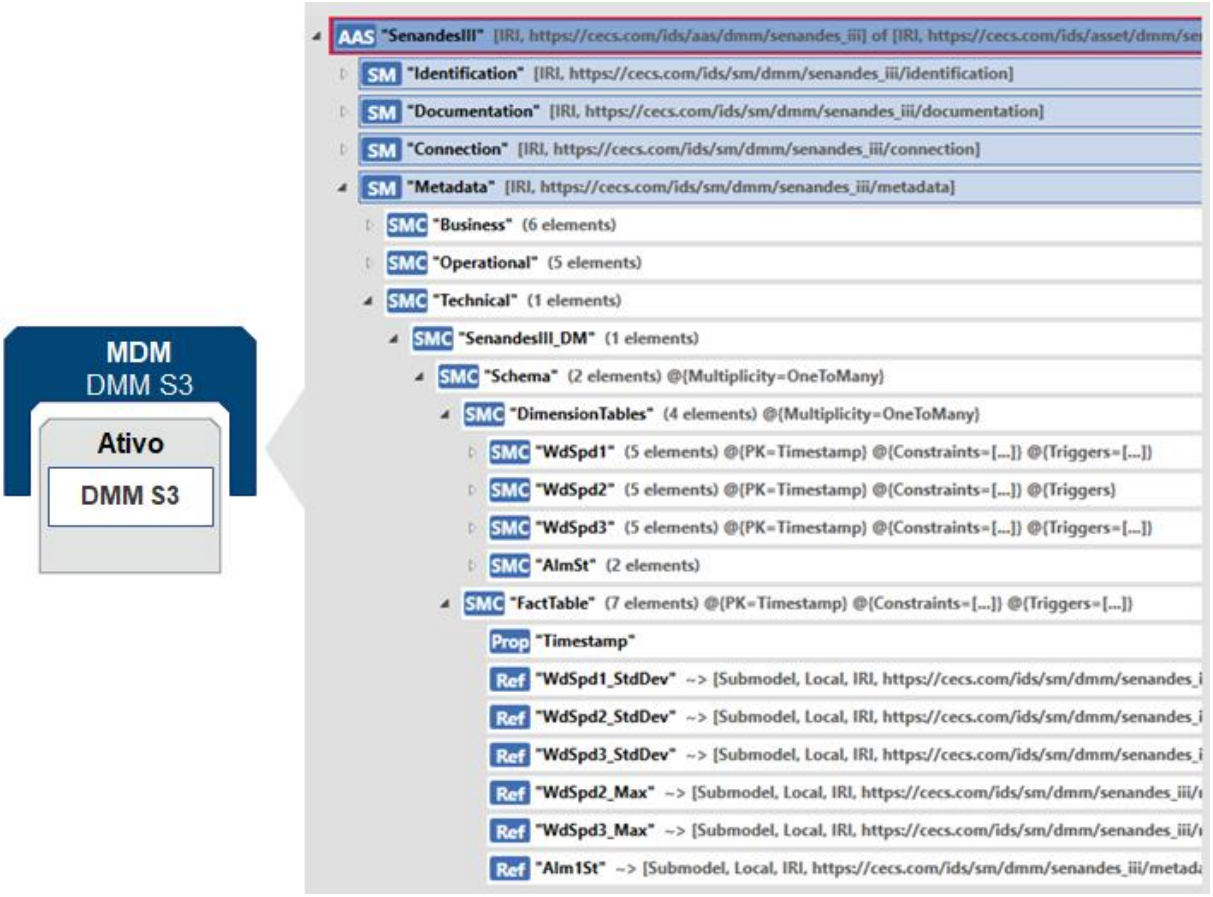


Figura 49 - Instância de MDM e a fração do AAS com as tabelas dimensão e fato.

Fonte: autoria própria.

Utilizando-se a técnica de validação cruzada, são geradas as bases de teste (aprendizado) e treino (avaliação dos resultados). Adota-se a técnica *k-Fold* de validação cruzada, com k=5 e embaralhamento dos registros. Este valor para o hiperparâmetro k é recomendado por Ríguez, Pérez e Lozano (2010) para que não se tenha viés (o que ocorre com valores mais baixos) e nem um custo computacional alto (quando se utilizam valores maiores para k). Deste modo, em cinco iterações, são geradas bases de treino com 80% dos dados originais para treinamento e 20% restante para teste.

Cinco técnicas de aprendizado de máquina de aprendizado supervisionado são escolhidas para ter o desempenho comparado: Naive Bayes gaussiano (GNB, do inglês *gaussian Naive Bayes*); K-ésimos vizinhos mais próximos (KNN, k-*nearest neighbors*) com k=11, regressão logística (LR, *logistic regression*), árvore de decisão (DT, *decision tree*) e floresta aleatória (RF, *random forest*), com 100 estimadores e entropia como critério de seleção. Para cada um dos cinco *folds*, as técnicas de aprendizado são aplicadas e avaliadas em termos de precisão,

revocação e acurácia. Os hiperparâmetros mencionados (k=11 e 100 estimadores) são obtidos após uma série de testes e se mostram escolhas adequadas para a base utilizada no que tange ao equilíbrio para o *trade-off* viés-variância. Os demais hiperparâmetros das técnicas são mantidos em seus valores padrão da biblioteca utilizada. A Figura 50 ilustra a instância de MCD e sua fração de AAS contemplando as técnicas de aprendizado de máquina a serem comparadas. Optou-se por não representar nesta Figura o conteúdo de cada Coleção de Elemento de Submodelo, uma vez que já foram ilustradas na Figura 43. Os resultados da avaliação das técnicas quanto a precisão, revocação e acurácia são apresentados respectivamente nas Figuras Figura 51, Figura 52 e Figura 53. A definição destas métricas é apresentada no Apêndice C.

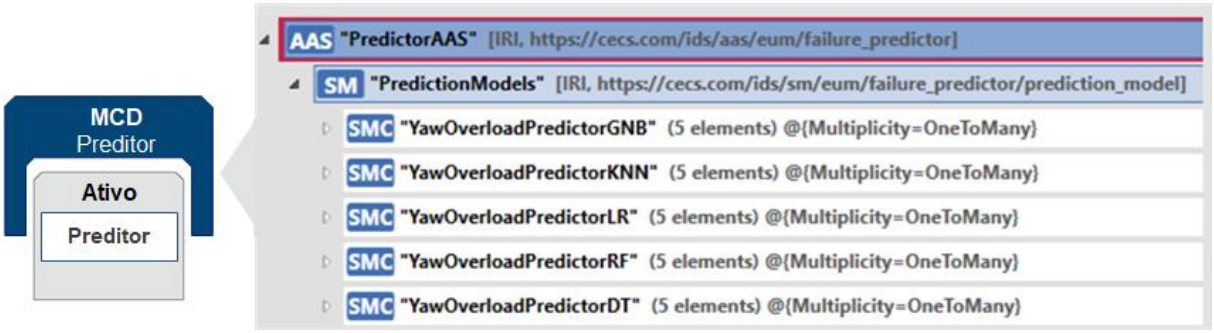


Figura 50 - Instância de MCD e a fração do AAS com técnicas de aprendizado de máquina.

Fonte: autoria própria.

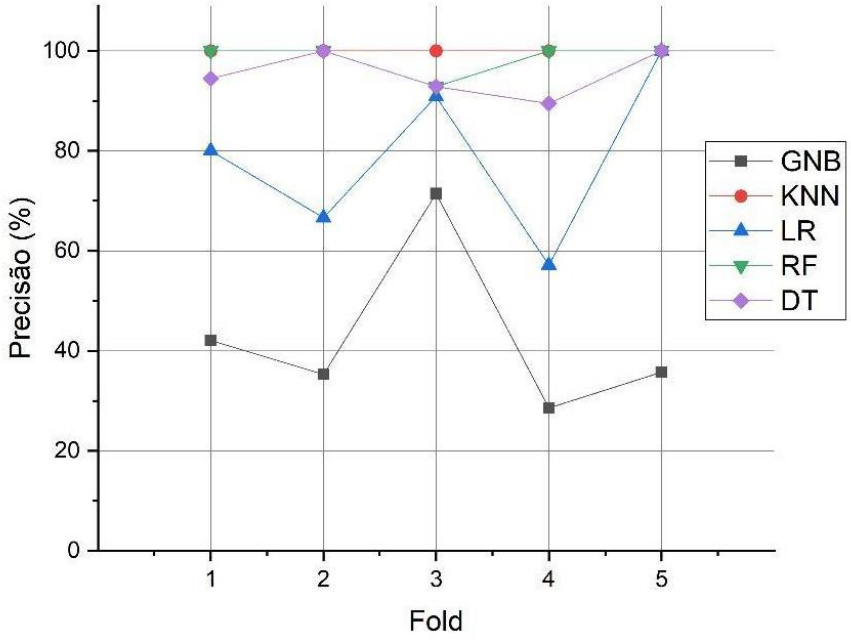


Figura 51 - Precisão dos classificadores.

Fonte: autoria própria.

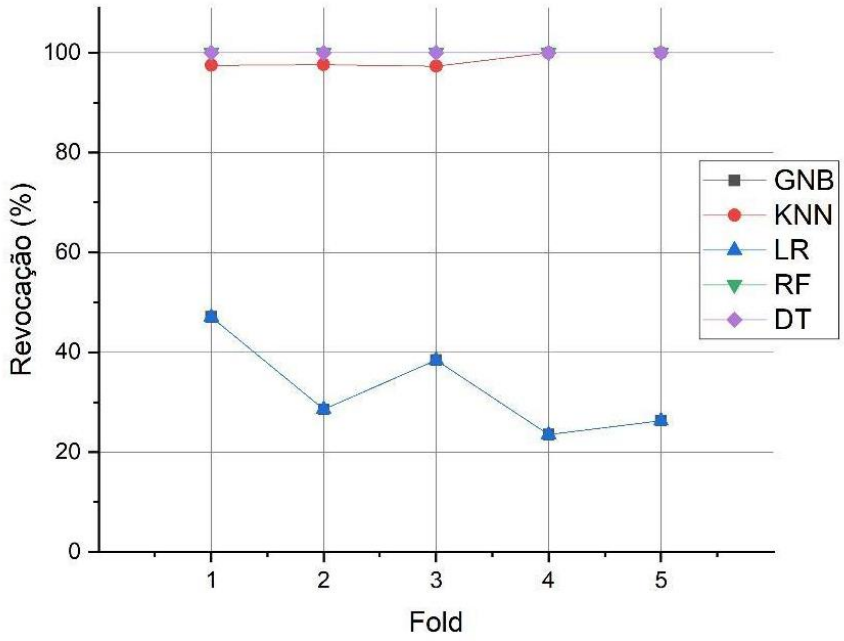


Figura 52 - Revocação dos classificadores.

Fonte: autoria própria.

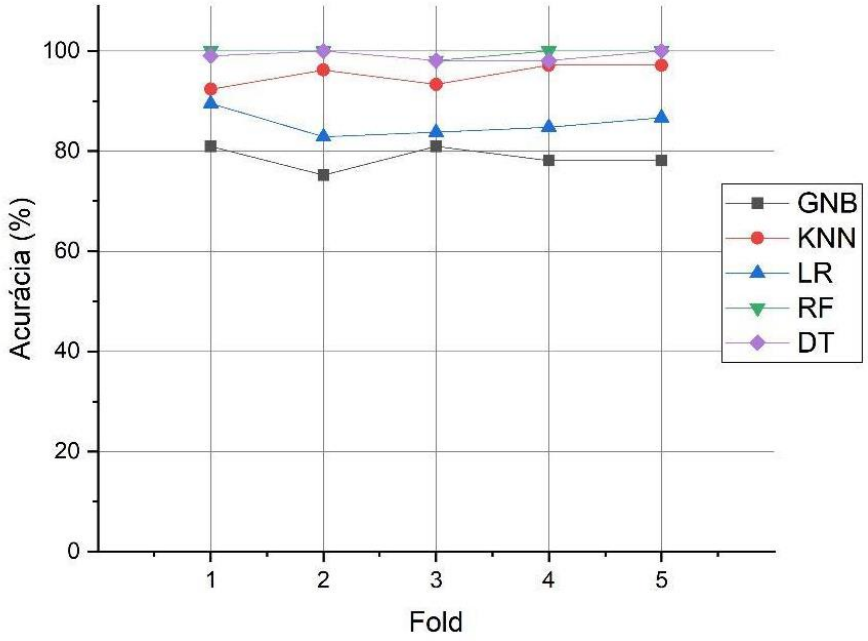


Figura 53 - Acurácia dos classificadores.

Fonte: autoria própria.

Os resultados demonstram que as técnicas que obtiveram melhor desempenho foram KNN, DT e RF. Para estas três técnicas, foram observados valores de precisão, revocação e acurácia superiores a 90%, o que sugere que é possível estimar a ocorrência de um evento de falha por sobrecarga no motor de *yaw* com alta taxa de sucesso na predição. Com isso, a lógica de operação do subsistema de *yaw* (como a atuação do sistema de frenagem, dimensionamento e acionamento dos motores, etc.) pode ser ajustada com base nas condições ambientais de modo a minimizar as chances de ocorrência da falha em questão, aumentando a disponibilidade do equipamento e, consequentemente, sua produção energética. Vale destacar que tais resultados só puderam ser obtidos ao se realizar o balanceamento da base de dados, aumentando a proporção de registros de falhas e realizando a validação cruzada.

5.4 Síntese do capítulo

Foi demonstrada a aplicação da proposta de arquitetura para integração e análise de dados em um caso real do setor de energia eólica. Foi realizada a identificação dos ativos do sistema e a associação aos módulos da arquitetura. É importante destacar que, nesta associação, o padrão de nomenclatura para os ativos do setor eólico foi mantido. Isso demonstra que o AAS não se sobrepõe aos padrões já estabelecidos, forçando a abandonar algo já consolidado, e que a estrutura e as classes do AAS permitem a compatibilidade com dicionários de dados, no caso do padrão IEC-61400-25-2. Esta é uma demonstração de como sistemas legados podem ser adequados ao contexto da I4.0.

Destaca-se aqui a elucidação do preenchimento da estrutura do AAS para torná-lo, além de um padrão de representação, um mecanismo capaz de parametrizar e viabilizar a operação dos Componentes da arquitetura. Foram apresentadas apenas as estruturas dos módulos, isto é, os Elementos que compõem o AAS. Esta estrutura garante a padronização na representação e a flexibilidade dos módulos para serem adequados às necessidades da aplicação à qual são destinados. Porém, somente com a parametrização do ASS, isto é, do preenchimento desta estrutura, é que se torna possível a operacionalização dos módulos para realização das suas tarefas.

Por fim, apresenta-se uma demonstração de como localizar um ativo no eixo “Ciclo de Vida e Cadeia de Valor” do RAMI 4.0 e identificar as possibilidades de se extrair valor a partir dos dados deste ativo por meio de aprendizado de máquina. No caso, ao desenvolver um modelo estatístico que oferece um prognóstico de falhas no aerogerador, o valor extraído aumenta sua

disponibilidade e possibilita maior geração de energia. Dada a escolha pela aplicação a qual se destina a técnica de aprendizado de máquina, foi possível identificar os tipos de dados necessários para esta aplicação e quais técnicas são mais adequadas.

6 Conclusões e Trabalhos Futuros

O objetivo da Indústria 4.0 enquanto fenômeno de transformação tecnológica é a realização da Fábrica Inteligente. Os dados são um recurso fundamental para alcançar este objetivo, pois podem ser convertidos em informação, conhecimento e inteligência. A utilização deste recurso no contexto da I4.0 pressupõe a padronização e organização dos dados em torno de uma arquitetura que possibilite a sua análise, a partir de uma visão geral da organização, para realização de processos e serviços inteligentes. Neste sentido, o trabalho apresentou uma proposta de arquitetura para integração e análise de dados com base nas diretrizes da I4.0, em especial o Modelo de arquitetura de Referência da Indústria 4.0 e o *Asset Administration Shell*.

A especificação desta arquitetura envolve tópicos relacionados a diferentes áreas do conhecimento. Além da fundamentação a respeito do fenômeno da I4.0, o desenvolvimento da arquitetura reúne tópicos de sistemas de bancos de dados e de *Data Warehouse*, aprendizado de máquina e *Big Data & Analytics*. Para cada um destes tópicos, o trabalho apresenta contribuições, com destaque para: (i) as recomendações acerca da escolha do modelo de dados mais adequado com base nas dimensões de *big data*; (ii) a orientação pela escolha da classe de técnicas de aprendizado de máquina com base nas características dos dados e no objetivo da análise; e (iii) a proposta de uma arquitetura baseada em *Data Warehouse* em conformidade com o RAMI 4.0 e o AAS.

Na literatura, é possível encontrar inúmeras comparações entre soluções de bancos de dados; trabalhos do tipo *survey* que têm como tema tópicos de aprendizado de máquina (inclusive aplicados ao setor industrial); e até mesmo propostas e implementações de arquiteturas para sistemas de manufatura inteligente. Porém, é importante reconhecer que, ao levar estas discussões para o contexto de I4.0, as diretrizes deste fenômeno nem sempre são levadas em conta. Isto acontece até mesmo em trabalhos que buscam identificar os pilares tecnológicos da I4.0. A originalidade deste trabalho consiste em adotar rigorosamente estas diretrizes ao abordar os tópicos mencionados, em particular o RAMI 4.0 e o AAS. Deste modo, "Indústria 4.0" não é adotado como um termo vago, mas embasado em definições e padrões claros.

Com relação à continuidade dos desenvolvimentos apresentados, é possível observar que, ainda que o foco deste trabalho tenha sido voltado às funcionalidades da Camada de Informação do RAMI 4.0, ainda restam tópicos em aberto referentes à esta. Como exemplo, pode-se

verificar que tópicos relacionados à segurança da informação não foram incluídos no escopo do trabalho. Este tema é abordado no padrão DIN SPEC 91345 (DIN, 2016) e o modelo de informação do AAS prevê elementos ligados à segurança da informação, porém ainda de maneira complexa e pouco desenvolvida, consistindo em uma lacuna a ser explorada. Com relação a trabalhos futuros que busquem explorar tópicos relacionados a I4.0 ou iniciativas análogas como *Industrial Internet* e *Made in China 2025* de maneira mais geral, entende-se que devem também adotar as respectivas diretrizes e padrões como forma de embasar novos estudos. Por se tratar de um fenômeno global que ainda se encontra em seus estágios iniciais, as arquiteturas de referência associadas a cada uma destas iniciativas possuem lacunas relacionadas à implementação, as quais podem ser preenchidas adotando estratégias análogas às que foram apresentadas neste trabalho.

Apêndice A – Classes de Elementos do AAS

Tabela 23 - Elementos do AAS.

Classe	Atributos (* = obrigatório)	Formato	Card
Referable <<abstract>>	idShort*	string	1
	category	string	0..1
	description	LangStringSet	0..1
	parent	Referable	0..1
Identifiable <<abstract>>	administration	AdministrativeInformation	0..1
	identification*	Identifier	1
Identifier	idType*	IdentifierType	1
	id*	Id	1
HasKind	kind	ModelingKind	0..1
AdministrativeInformation	version	string	0..1
	revision	string	0..1
HasSemantics <<abstract>>	semanticId	Reference	0..1
Qualifiable <<abstract>>	qualifier	Constraint	0..*
Constraint <<abstract>>			
Qualifier	type*	QualifierType	1
	valueType*	DataTypeDef	1
	value	ValueDataType	0..1
	valueId	Reference	0..1
Formula	dependsOn	Reference	0..*
HasDataSpecification <<abstract>>	dataSpecification	Reference	0..*
AssetAdministrationShell	derivedFrom	AssetAdministrationShell	0..1
	security	Security	0..1
	asset*	Asset	1
	submodel	Submodel	0..*
	conceptDictionary	ConceptDictionary	0..*
	view	View	0..*
Asset	kind*	AssetKind	1
	assetIdentificationModel	Submodel	0..1
	billOfMaterial	Submodel	0..1
Submodel	submodelElement	SubmodelElement	0..*
SubmodelElement <<abstract>>			
DataElement <<abstract>>			
Property	valueType*	DataTypeDef	1
	value	ValueDataType	0..1
	valueId	Reference	0..1
MultiLanguageProperty	value	LangStringSet	0..1
	valueId	Reference	0..1

Tabela 23 - Elementos do AAS (continuação).

Classe	Atributos (* = obrigatório)	Formato	Card
ReferenceElement	value	Reference	0..1
Range	valueType*	DataTypeDef	1
	min	ValueDataType	0..1
	max	ValueDataType	0..1
Blob	value	BlobType	0..1
	mimeType*	MimeType	1
File	value	PathType	0..1
	mimeType*	MimeType	1
SubmodelElementCollection	value	SubmodelElement	0...*
	ordered	boolean	0..1
	allowDuplicates	boolean	0..1
RelationshipElement	first*	Referable	1
	second*	Referable	1
AnnotatedRelationshipElement	annotation	DataElement	0..*
Operation	inputVariable	OperationVariable	0..*
	outputVariable	OperationVariable	0..*
	inoutputVariable	OperationVariable	0..*
OperationVariable	value*	SubmodelElement	1
Capability			
Entity	statement	SubmodelElement	0..*
	entityType*	EntityTypeEnum	1
	asset	Asset	0..1
Event <<abstract>>			
BasicEvent	observed*	Referable	1
View	containedElement	Referable	0..*
ConceptDictionary	conceptDescription	ConceptDescription	0..*
ConceptDescription	isCaseOf	Reference	0..*
Reference	key*	Key	1..*
Key	type*	KeyElements	1
	local*	boolean	1
	value*	string	1
	idType*	KeyType	1
LangStringSet <<DataType>>	langString	langString	1..*
ValueList <<DataType>>	valueReferencePairType	ValueReferencePair	1..*
ValueReferencePair <<DataType>>	value*	ValueDataType	1
	valueId*	Reference	1
Security	accessControlPolicyPoints*	AccessControlPolicy Points	1
	certificate	Certificate	0..*
	requiredCertificateExtension	Reference	0..*
Certificate <<abstract>>	policyAdministrationPoint*	PolicyAdministrationPoint	1

Tabela 23 - Elementos do AAS (continuação).

Classe	Atributos (* = obrigatório)	Formato	Card
BlobCertificate	blobCertificate*	Blob	1
	containedExtension	Reference	0..*
	lastCertificate*	boolean	1
AccessControlPolicyPoints	policyAdministrationPoint*	PolicyAdministrationPoint	1
	policyDecisionPoint*	PolicyDecisionPoint	1
	policyEnforcementPoint*	PolicyEnforcementPoint	1
	policyInformationPoints	PolicyInformationPoints	0..1
PolicyAdministrationPoint	localAccessControl	AccessControl	0..1
	externalAccessControl*	boolean	1
PolicyInformationPoints	internalInformationPoint	Submodel	0..*
	externalInformationPoints*	boolean	1
PolicyEnforcementPoints	externalPolicyEnforcementPoint*	boolean	1
PolicyDecisionPoint	externalPolicyDecisionPoints*		1
AccessControl	accessPermissionRule	AccessPermissionRule	0..*
	selectableSubjectAttributes	Submodel	0..1
	defaultSubjectAttributes*	Submodel	1
	selectablePermissions*	Submodel	0..1
	defaultPermissions*	Submodel	1
	selectableEnvironmentAttributes	Submodel	0..1
	defaultEnvironmentAttributes	Submodel	0..1
AccessPermissionRule	targetSubjectAttributes*	SubjectAttributes	1
	permissionsPerObject	PermissionsPerObject	0..*
PermissionsPerObject	object*	Referable	1
	targetObjectAttributes	ObjectAttributes	0..1
	permission	Permission	0..*
ObjectAttributes	objectAttribute*	DataElement	1..*
Permission	permission*	Property	1
	kindOfPermission*	PermissionKind	1
SubjectAttributes	subjectAttribute*	DataElement	1..*

Fonte: adaptado de Bader *et al*. (2019).

Apêndice B – Termos utilizados para a revisão sistemática da literatura

Este apêndice descreve os procedimentos adotados no trabalho de revisão sistemática mencionado na Seção 4.1. Este trabalho foi realizado com auxílio do *software* StArt (*State of the Art through Systematic Review*)[33].

- Estabeleceu-se em quais bases de dados seria realizada a busca por artigos: *Web of Science*, *IEEEXplore*, *Science Direct* e *Google Scholar*
- Definiu-se a seguinte *string* de busca para encontrar artigos: "*Asset Administration Shell*" AND "Database";
- Observou-se que, dentre as bases de dados selecionadas, a única delas a retornar uma quantidade considerável de artigos (139) foi o Google Scholar, que incluía artigos das demais bases e por isso foi a única utilizada;
- Foram definidas as seguintes palavras-chave para ranqueamento dos artigos: “AAS”; “*Asset Administration Shell*”; “Database”; “DBMS” (a sigla em inglês correspondente à SGBD); “Implement”; “Implementing”; “Implementation”; “Storage”;
- Para ranqueamento dos artigos, foi decidido que cada palavra-chave no título do artigo atribuiria 5 pontos ao mesmo (a plataforma *Google Scholar* não permite exportar o resumo nem palavras-chave do artigo);
- Artigos com pontuação maior ou igual a 5 foram classificados como aceitos e seu conteúdo foi analisado. Foram selecionados 25 artigos;
- Pesquisou-se quais dos artigos classificados como aceitos citavam o modelo de dados de implementação e/ou SGBD utilizado.

[33] http://lapes.dc.ufscar.br/tools/start_tool.

Apêndice C – Métricas de Avaliação do Modelo Estatístico

Na Seção 5.3, os modelos de classificação foram avaliados quanto à sua precisão, revocação e acurácia. No entanto, a definição destas métricas não foi apresentada. Este Apêndice apresenta as expressões que definem estas métricas, que podem ser derivadas a partir da matriz de confusão.

A matriz de confusão consiste em uma tabela que expõe o desempenho de modelo de aprendizado de máquina de classificação aplicado a uma base de treino, validação ou teste (na prática, o desempenho não deve ser avaliado a partir da aplicação na base de treino). Considerando uma base de dados em que a variável de classe pode assumir apenas valores booleanos falso e verdadeiro[34], como é o caso do exemplo da Seção 5.3, a aplicação de um modelo estatístico de classificação pode levar a quatro possíveis situações para cada registro da base: verdadeiro positivo (TP, do inglês *true positive*), falso positivo (FP, do inglês *false positive*), verdadeiro negativo (TN, do inglês *true negative*) e falso negativo (FN, do inglês *false negative*). A matriz de confusão é do tipo 2x2 e contém, em seus elementos, a distribuição destas quatro possibilidades, conforme ilustrado na Figura 54:

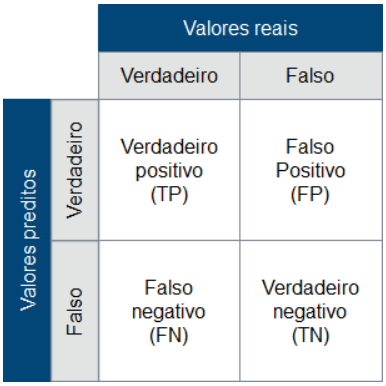


Figura 54 - Matriz de confusão.

Fonte: autoria própria.

A partir da matriz de confusão, definem-se as métricas utilizadas para a avaliação dos modelos estatísticos:

[34] A matriz de confusão pode ser derivada para classificações com múltiplas classes, isto é, para variáveis de classe que assumem um número maior de valores discretos, para além dos booleanos falso e verdadeiro.

$$Precisão = \frac{TP}{TP + FP}$$

$$Revocação = \frac{TP}{TP + FN}$$

$$Acurácia = \frac{TP + TN}{TP + TN + FP + FN}$$

Referências

ABADI, D. J. Consistency tradeoffs in modern distributed database system design: CAP is only part of the story. **Computer**, v. 45, n. 2, p. 37–42, 2012.

ADAM, S.; BUŞONIU, L.; BABUŠKA, R. Experience replay for real-time reinforcement learning control. **IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews**, v. 42, n. 2, p. 201–212, 2012.

ADOLPHS, P. et al. **Structure of the Asset Administration Shell: Continuation of the Development of the Reference Model for the Industrie 4.0 Component**. Disponível em: <https://www.plattform-i40.de/IP/Redaktion/EN/Downloads/Publikation/rami40-an-introduction.pdf?__blob=publicationFile&v=7>. Acesso em: 27 jan. 2023.

ADOLPHS, P.; EPPLE, U. **Reference Architecture Model Industrie 4.0 (RAMI4.0)**. Disponível em: <https://www.zvei.org/fileadmin/user_upload/Presse_und_Medien/Publikationen/2016/januar/GMA_Status_Report__Reference_Archtitecture_Model_Industrie_4.0__RAMI_4.0_/GMA-Status-Report-RAMI-40-July-2015.pdf>. Acesso em: 27 jan. 2023.

AISSANI, N.; BELDJILALI, B.; TRENTESAUX, D. Dynamic scheduling of maintenance tasks in the petroleum industry: A reinforcement approach. **Engineering Applications of Artificial Intelligence**, v. 22, n. 7, p. 1089–1103, 2009.

ALAVI, MARYAM, E. LEIDNER, D. Knowledge management and knowledge management systems: Conceptual foundations and research issues. **MIS quarterly**, v. 25, n. 1, p. 107–136, 2001.

ALLI, A. A.; ALAM, M. M. SecOFF-FCIoT: Machine learning based secure offloading in Fog-Cloud of things for smart city applications. **Internet of Things**, v. 7, n. 2019, p. 100070, 2019.

AMRUTHNATH, N.; GUPTA, T. **Fault class prediction in unsupervised learning using model-based clustering approach**. 2018 International Conference on Information and Computer Technologies, ICICT 2018. **Anais**...DeKalb: IEEE, 2018

ANDERSEN, K. et al. Artificial Neural Networks Applied to Arc Welding Process Modeling and Control. **IEEE Transactions on Industry Applications**, v. 26, n. 5, p. 824–830, 1990.

ANDIAPPAN, V.; WAN, Y. K. Distinguishing approach, methodology, method, procedure and technique in process systems engineering. **Clean Technologies and Environmental Policy**, v. 22, n. 3, p. 547–555, 2020.

ANDRADE, N.; TORRES, D.; MAIA, L. **Impacto da Subperformance do Sistema de Pitch**

na Produção do Aerogerador: Uma Abordagem Baseado em Dados SCADA. Brazil Wind Power. **Anais**...São Paulo: ABEEólica, 2022

ANGLES, R.; GUTIERREZ, C. Survey of graph database models. **ACM Computing Surveys**, v. 40, n. 1, p. 1–39, 2008.

ANKERST, M.; BERCHTOLD, S.; KEIM, D. A. **Similarity clustering of dimensions for an enhanced visualization of multidimensional data**. Proceedings of the IEEE Symposium on Information Visualization. **Anais**...Piemonte: IEEE, 1998

ARDOLINO, M. et al. The role of digital technologies for the service transformation of industrial companies. **International Journal of Production Research**, v. 56, n. 6, p. 2116–2132, 2018.

AYTUG, H. et al. A review of machine learning in Scheduling. **IEEE Transactions on Engineering Management**, v. 41, n. 2, p. 165–171, 1994.

AZADEH, A. et al. A flexible algorithm for fault diagnosis in a centrifugal pump with corrupted data and noise based on ANN and support vector machine with hyper-parameters optimization. **Applied Soft Computing Journal**, v. 13, n. 3, p. 1478–1485, 2013.

BADER, S. et al. **Details of the Asset Administration Shell Part 1 - The exchange of information between partners in the value chain of Industrie 4.0**. Disponível em: <https://www.plattform-i40.de/PI40/Redaktion/DE/Downloads/Publikation/Details-of-the-Asset-Administration-Shell-Part1.html>. Acesso em: 27 jan. 2023.

BADER, S. et al. **Details of the Asset Administration Shell Part 2 – Interoperability at Runtime – Exchanging Information via Application Programming Interfaces**. Disponível em: <https://www.plattform-i40.de/IP/Redaktion/EN/Downloads/Publikation/Details_of_the_Asset_Administration_Shell_Part2_V1.pdf?__blob=publicationFile&v=8>. Acesso em: 27 jan. 2023.

BARBER, D. **Bayesian Reasoning and Machine Learning**. Cambridge: Cambridge University Press, 2012.

BARIK, R. K. et al. Fog Assisted Cloud Computing in Era of Big Data and Internet-of-Things: Systems, Architectures, and Applications. In: **Cloud computing for optimization: foundations, applications, and challenges**. 1. ed. Cham: Springer, 2018. p. 367–394.

BARNETT, V.; LEWIS, T. **Outliers in statistical data**. Hoboken: John Wiley & Sons, 1984.

BATRA, S. Big Data Analytics and its Reflections on DIKW Hierarchy Surinder. **Review of Management**, v. 4, n. 1, p. 1–56, 2014.

BATRA, S.; TYAGI, C. Comparative Analysis of Relational and Graph Databases for Social

Networks. **International Journal of Soft Computing and Engineering**, v. 2, n. 2, p. 509–512, 2012.

BEDENBENDER, H. et al. **Relationships between I4.0 Components – Composite Components and Smart Production**. Disponível em: <https://www.plattform-i40.de/PI40/Redaktion/EN/Downloads/Publikation/hm-2018-relationship.html>. Acesso em: 27 jan. 2023a.

BEDENBENDER, H. et al. **Examples of the Asset Administration Shell for Industrie 4.0 Components - Basic Part**. Disponível em: <https://www.zvei.org/fileadmin/user_upload/Presse_und_Medien/Publikationen/2017/April/Asset_Administration_Shell/ZVEI_WP_Verwaltungschale_Englisch_Download_03.04.17.pdf>. Acesso em: 27 jan. 2023b.

BEDENBENDER, H. et al. **Industrie 4.0 Plug-and-Produce for Adaptable Factories: Example Use Case Definition, Models, and Implementation**. Disponível em: <https://www.zvei.org/fileadmin/user_upload/Presse_und_Medien/Publikationen/2017/Juni/Industrie_4.0_Plug_and_produce/Industrie-4.0-_Plug-and-Produce-zvei.pdf>. Acesso em: 27 jan. 2023c.

BEDENBENDER, H. et al. **Which criteria do Industrie 4.0 products need to fulfil?** Disponível em: <https://www.plattform-i40.de/IP/Redaktion/EN/Downloads/Publikation/criteria-industrie-40-products.pdf?__blob=publicationFile&v=5#:~:text=These Industrie 4.0 services must,initial implementation to full expansion.>. Acesso em: 27 jan. 2023.

BENGIO, Y.; COURVILLE, A.; VINCENT, P. Representation Learning: A Review and New Perspectives. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 35, n. 8, p. 1798–1828, 2013.

BENKO, A.; SIK LÁNYI, C. History of Artificial Intelligence. **Encyclopedia of Information Science and Technology, Second Edition**, v. 26, n. 4, p. 1759–1762, 2011.

BHATTACHARYA, B.; PRICE, R. K.; SOLOMATINE, D. P. Machine Learning Approach to Modeling Sediment Transport. **Journal of Hydraulic Engineering**, v. 133, n. 4, p. 440–450, 2007.

BICEVSKA, Z.; ODITIS, I. Towards NoSQL-based Data Warehouse Solutions. **Procedia Computer Science**, v. 104, n. December 2016, p. 104–111, 2016.

BLAZIC, G.; POSCIC, P.; JAKSIC, D. **Data warehouse architecture classification**. 2017 40th International Convention on Information and Communication Technology, Electronics

and Microelectronics, MIPRO 2017 - Proceedings. **Anais**...Opatija: IEEE, 2017

BONIFATI, A. et al. Designing data marts for data warehouses. **ACM Transactions on Software Engineering and Methodology**, v. 10, n. 4, p. 452–483, 2001.

BOUSDEKIS, A. et al. **A RAMI 4.0 view of predictive maintenance: Software architecture, platform and case study in steel industry**. International Conference on Advanced Information Systems Engineering. **Anais**...Rome: Sapienza Università di Roma, 2019

BREWER, E. Towards robust distributed systems. **Symposium on Principles of Distributed Computing**, v. 7, p. 343477-343502, 2000.

BRO, R.; SMILDE, A. K. Principal component analysis. **Analytical Methods**, v. 6, n. 9, p. 2812–2831, 2014.

CABIBBO, L.; TORLONE, R. On the integration of autonomous data marts. **Proceedings of the International Conference on Scientific and Statistical Database Management, SSDBM**, v. 16, p. 223–232, 2004.

CAI, D.; ZHANG, C.; HE, X. **Unsupervised feature selection for Multi-Cluster data**. Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. **Anais**...Washington DC: Association for Computing Machinery, 2010

CARVALHO, T. P. et al. A systematic literature review of machine learning methods applied to predictive maintenance. **Computers and Industrial Engineering**, v. 137, p. 1–10, 2019.

CATTELL, R. Scalable SQL and NoSQL data stores. **SIGMOD Record**, v. 39, n. 4, p. 12–27, 2010.

CAVALIERI, S.; SALAFIA, M. G. Insights into mapping solutions based on OPC UA information model applied to the industry 4.0 asset administration shell. **Computers**, v. 9, n. 2, p. 1–28, 2020.

CHABOT, R.; BROWN, D. C. Knowledge compilation using constraint inheritance. **Artificial Intelligence for Engineering, Design, Analysis and Manufacturing**, v. 8, n. 2, p. 125–142, 1994.

CHAN, H. et al. Machine learning coarse grained models for water. **Nature Communications**, v. 10, n. 1, p. 1–14, 2019.

CHEVALIER, M. et al. **Implementing Multidimensional Dasta Warehouse into NoSQL**. 17th International Conference on Enterprise Information Systems. **Anais**...Barcelona: Association for Computing Machinery, 2015a

CHEVALIER, M. et al. **How Can We Implement a Multidimensional Data Warehouse Using NoSQL?** 17th International Conference on Enterprise Information Systems.

Anais...Barcelona: Springer, 2015b

CHUNG, Y.; KUSIAK, A. Grouping parts with a neural network. **Journal of Manufacturing Systems**, v. 13, n. 4, p. 262–275, 1994.

DAIGNEAU, R. **Service Design Patterns: fundamental design solutions for SOAP/WSDL and restful Web Services**. Westford: Addison-Wesley, 2012.

DATHEIN, R. Inovação e Revoluções Industriais: uma apresentação das mudanças tecnológicas determinantes nos séculos XVIII e XIX. **DECON Textos didáticos**, 2003.

DAYAN, P.; NIV, Y. Reinforcement learning: The Good, The Bad and The Ugly. **Current Opinion in Neurobiology**, v. 18, n. 2, p. 185–196, 2008.

DE OLIVEIRA, V. F. et al. SQL and NoSQL databases in the context of industry 4.0. **Machines**, v. 10, n. 1, 2022.

DEUTSCHLAND.DE. **Networking the world**. Disponível em: <https://www.deutschland.de/en/topic/business/innovation-technology/networking-the-world>. Acesso em: 15 jun. 2021.

DIETTERICH, T. G.; MICHALSKI, R. S. A Comparative Review of Selected Methods for Learning From Examples. In: **Machine Learning: An Artificial Intelligence Approach**. Palo Alto: Tioga Publishing, 1983. p. 41–81.

DIEZ-OLIVAN, A. et al. Data fusion and machine learning for industrial prognosis: Trends and perspectives towards Industry 4.0. **Information Fusion**, v. 50, n. July 2018, p. 92–111, 2019.

DIEZ, A. et al. A clustering approach for structural health monitoring on bridges. **Journal of Civil Structural Health Monitoring**, v. 6, n. 3, p. 429–445, 2016.

DIN. **Reference Architecture Model Industrie 4.0 (RAMI4.0) (DIN SPEC 91345)**Berlim, 2016.

DJENOURI, Y.; SRIVASTAVA, G.; LIN, J. C. W. Fast and Accurate Convolution Neural Network for Detecting Manufacturing Data. **IEEE Transactions on Industrial Informatics**, v. 17, n. 4, p. 2947–2955, 2021.

DOAN, A.; HALEVY, A.; IVES, Z. **Principles of Data Integration**. Waltham: Elsevier, 2012.

DORST, W. et al. **Implementation Strategy Industrie 4.0**. Disponível em: <https://www.zvei.org/fileadmin/user_upload/Presse_und_Medien/Publikationen/2016/januar/Implementation_Strategy_Industrie_4.0_-_Report_on_the_results_of_Industrie_4.0_Platform/Implementation-Strategy-Industrie-40-ENG.pdf>. Acesso em: 27 jan. 2023.

DOUGLAS, E. et al. **Explorando padrões e normas associados ao RAMI 4 . 0 : Um estudo descritivo**. VIII Congresso Brasileiro de Engenharia de Produção. **Anais**...Ponta Grossa: Associação Paranaense de Engenharia de Produção, 2018

DUNCAN, O. **Partições – Modos e processamento de armazenamento de partições**. Disponível em: <https://docs.microsoft.com/pt-br/analysis-services/multidimensional-models-olap-logical-cube-objects/partitions-partition-storage-modes-and-processing?view=asallproducts-allversions>. Acesso em: 15 jun. 2021a.

DUNCAN, O. **Métodos de discretização (mineração de dados)**. Disponível em: <https://docs.microsoft.com/pt-br/analysis-services/data-mining/discretization-methods-data-mining?view=asallproducts-allversions>. Acesso em: 15 jun. 2021b.

DY, J. G.; BRODLEY, C. E. **Visualization and interactive feature selection for unsupervised data**. Proceeding of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. **Anais**...Boston: Association for Computing Machinery, 2000

ELANGOVAN, M. et al. Machine learning approach to the prediction of surface roughness using statistical features of vibration signal acquired in turning. **Procedia Computer Science**, v. 50, p. 282–288, 2015.

ELMASRI, R.; NAVATHE, S. B. **Fundamentals of Database systems**. 7. ed. Hoboken: Pearson, 2000.

ELMASRI, R.; NAVATHE, S. B. **Sistemas de Banco de Dados**. 4. ed. Londres: Pearson, 2005.

FAMILI, A. et al. Data preprocessing and intelligent data analysis. **Intelligent Data Analysis**, v. 1, n. 1, p. 3–23, 1997.

FAN, Q. Y.; YANG, G. H. Adaptive Actor-Critic Design-Based Integral Sliding-Mode Control for Partially Unknown Nonlinear Systems With Input Disturbances. **IEEE Transactions on Neural Networks and Learning Systems**, v. 27, n. 1, p. 165–177, 2016.

FERREIRA, J. et al. **O Processo ETL em Sistemas Data Warehouse**. INForum 2010 - II Simpósio de Informática. **Anais**...Braga: Universidade do Porto, 2010

FOWLER, A. **NoSQL for dummies**. Hoboken: John Wiley & Sons, 2015.

FOWLER, M. **Reporting Database**. Disponível em: <https://martinfowler.com/bliki/ReportingDatabase.html>. Acesso em: 15 jun. 2021.

FRAKES, W. B. Introduction to Information Storage and Retrieval Systems. In: **Introduction to Information Storage and Retrieval Systems**. 1. ed. Hoboken: Prentice-Hall, 1992.

FRIEDRICHS, M. BioDWH2 : an automated graph-based data. **Journal of Integrative**

Bioinformatics, v. 18, n. 2, p. 1–10, 2021.

GABEL, T.; RIEDMILLER, M. Adaptive Reactive Job-Shop Scheduling With Reinforcement Learning Agents. **International Journal of Information Technology and Intelligent Computing**, v. 24, n. 4, p. 14–18, 2008.

GABEL, T.; RIEDMILLER, M. Distributed policy search reinforcement learning for job-shop scheduling tasks. **International Journal of Production Research**, v. 50, n. 1, p. 41–61, 2012.

GARCÍA-LAENCINA, P. J.; SANCHO-GÓMEZ, J. L.; FIGUEIRAS-VIDAL, A. R. Pattern classification with missing data: A review. **Neural Computing and Applications**, v. 19, n. 2, p. 263–282, 2010.

GASCH, R.; TWELE, J. **Wind Power Plants: Fundamentals, Design, Construction and Operation**. 2. ed. Nova Iorque: Springer, 2012.

GASKETT, C. 2002. **Q-learning for robot control**. Tese (Doutorado em Engenharia de Sistemas) - Departamento de Engenharia de Sistemas, Universidade Nacional da Austrália, Camberra, 2002.

GASTALDI, L. et al. Managing the exploration-exploitation paradox in healthcare: Three complementary paths to leverage on the digital transformation. **Business Process Management Journal**, v. 24, n. 5, p. 1200–1234, 2018.

GE, Z. et al. Data Mining and Analytics in the Process Industry: The Role of Machine Learning. **IEEE Access**, v. 5, p. 20590–20616, 2017.

GEREND, J. **Contêineres vs. máquinas virtuais**. Disponível em: <https://docs.microsoft.com/pt-br/virtualization/windowscontainers/about/containers-vs-vm>. Acesso em: 15 jun. 2021.

GESSERT, F. et al. NoSQL database systems: a survey and decision guidance. **Computer Science - Research and Development**, v. 32, n. 3–4, p. 353–365, 2017.

GIL, D.; SONG, I. Y. Modeling and Management of Big Data: Challenges and opportunities. **Future Generation Computer Systems**, v. 63, p. 96–99, 2016.

GNOATTO, H. 2017. **Análise De Viabilidade Técnica E Econômica Para Implantação De Aerogerador Em Propriedades Rurais De Cascavel, Londrina E Palmas-Pr**. Dissertação (Mestrado em Engenharia de Energia na Agricultura) - Universidade Estadual do Oeste do Paraná, Cascavel, 2017.

GRECU, D. L.; BROWN, D. C. **Dimensions of Learning in Agent-based Design**. International Conference on AI in Design - Workshop on Machine Learning in Design. **Anais**...Stanford: Kluwer, 1996

GUO, K. et al. Artificial intelligence and machine learning in design of mechanical materials. **Materials Horizons**, v. 8, n. 4, p. 1153–1172, 2021.

GUZZI, P. H.; VELTRI, P.; CANNATARO, M. Experimental Evaluation of Nosql Databases. **International Journal of Database Management Systems**, v. 6, n. 3, 2014.

HAMEL, L. **Visualization of support vector machines with unsupervised learning**. Proceedings of the 2006 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology, CIBCB'06. **Anais**...Toronto: IEEE, 2006

HAN, J. et al. **Survey on NoSQL database**. Proceedings - 2011 6th International Conference on Pervasive Computing and Applications. **Anais**...Seattle: IEEE, 2011

HANSSON, K. et al. Machine Learning Algorithms in Heavy Process Manufacturing. **American Journal of Intelligent Systems**, v. 6, n. 1, p. 1–13, 2016.

HE, H.; GARCIA, E. A. Learning from imbalanced data. **IEEE Transactions on knowledge and data engineering**, v. 21, n. 9, p. 1263–1284, 2009.

HERMANN, M.; PENTEK, T.; OTTO, B. **Design Principles for Industrie 4.0 Scenarios**. International Conference on System Sciences Design. **Anais**...Koloa: IEEE, 2015Disponível em: <http://www.snom.mb.tu-dortmund.de/cms/de/forschung/Arbeitsberichte/Design-Principles-for-Industrie-4_0-Scenarios.pdf>

HOBSBAWM, E. J. **A era das revoluçoes - O mundo na década de 1780**. São Paulo: Paz e Terra, 2009.

IBM - CLOUD EDUCATION. **What is OLAP?** Disponível em: <https://www.ibm.com/cloud/learn/olap#toc-molap-vs-r-QgDGFWEu>. Acesso em: 15 jun. 2021.

IEC - INTERNATIONAL ELECTROTECHNICAL COMMISSION. **Communications for monitoring and control of wind power plants – Information models**GenevaIEC, , 2015.

INIGO, M. A. et al. **Towards an Asset Administration Shell scenario: A use case for interoperability and standardization in Industry 4.0**. Proceedings of IEEE/IFIP Network Operations and Management Symposium 2020: Management in the Age of Softwarization and Artificial Intelligence. **Anais**...Budapeste: IEEE, 2020

INMON, W. H. **Building the Data Warehouse**. 3. ed. Hoboken: John Wiley & Sons, 2002.

INMON, W. H.; STRAUSS, D.; NEUSHLOSS, G. **DW2.0: The Architecture for the Next Generation of Data Warehousing**. Burlington: Elsevier, 2010.

IVANOV, D.; DOLGUI, A.; SOKOLOV, B. The impact of digital technology and Industry 4.0 on the ripple effect and supply chain risk analytics. **International Journal of Production**

Research, v. 57, n. 3, p. 829–846, 2019.

IVI - INDUSTRIAL VALUE CHAIN INITIATIVE. **Industrial Value Chain Reference Architecture (IVRA)**. Chiyoda, 2016.

JAMES, G. et al. **An Introduction to Statistical Learning**. Cham: Springer, 2013. v. 11

JERVELL, J. T. **Estudo da Influência das Características do Vento no Desempenho de Aerogeradores**. 2008. Dissertação (Mestrado Integrado em Engenharia Mecânica) - Faculdade de Engenharia da Universidade do Porto, Porto, 2008.

JIRKOVSKY, V.; OBITKO, M.; MARIK, V. Understanding data heterogeneity in the context of cyber-physical systems integration. **IEEE Transactions on Industrial Informatics**, v. 13, n. 2, p. 660–667, 2017.

JOY, A. M. **Performance comparison between Linux containers and virtual machines**. 2015 International Conference on Advances in Computer Engineering and Applications. **Anais**...Ghaziabad: IEEE, 2015

KAELBLING, L. P.; LITTMAN, M. L.; MOORE, A. W. Reinforcement learning: a survey. **Frontiers of Information Technology and Electronic Engineering**, v. 21, n. 12, p. 1726–1744, 2020.

KAGERMANN, H. et al. **Recommendations for the Strategic Initiative Web-based Services for Businesses**. Disponível em: <https://en.acatech.de/publication/recommendations-for-the-strategic-initiative-web-based-services-for-businesses-final-report-of-the-smart-service-working-group/>. Acesso em: 27 jan. 2023.

KAGERMANN, H.; WAHLSTER, W.; HELBIG, J. **Securing the future of German manufacturing industry: Recommendations for implementing the strategic initiative INDUSTRIE 4.0**. Disponível em: <https://www.din.de/blob/76902/e8cac883f42bf28536e7e8165993f1fd/recommendations-for-implementing-industry-4-0-data.pdf>. Acesso em: 27 jan. 2023.

KANEKO, H.; FUNATSU, K. Application of Online Support Vector Regression for Soft Sensors. **AIChE Journal**, v. 60, n. 2, p. 600–612, 2014.

KAPARTHI, S.; SURESH, N. C. A neural network system for shape-based classification and coding of rotational parts. **International Journal of Production Research**, v. 29, n. 9, p. 1771–1784, 1991.

KASER, O.; LEMIRE, D. Attribute value reordering for efficient hybrid OLAP. **Information Sciences**, v. 176, n. 16, p. 2304–2336, 2006.

KHAN, S. G. et al. Reinforcement learning and optimal adaptive control: An overview and

implementation examples. **Annual Reviews in Control**, v. 36, n. 1, p. 42–59, 2012.

KHEDIRI, I. BEN; WEIHS, C.; LIMAM, M. Kernel k-means clustering based local support vector domain description fault detection of multimodal processes. **Expert Systems with Applications**, v. 39, n. 2, p. 2166–2171, 2012.

KHINE, P. P.; WANG, Z. A review of polyglot persistence in the big data world. **Information**, v. 10, n. 4, 2019.

KHINE, P. H.; WANG, Z. S. **Data Lake: A New Ideology in Big Data Era**. Wireless Communication and Sensor Network. Annual International Conferenceed. **Anais**... Wuhan: EDP Sciences, 2018.

KIMBALL, R.; CASERTA, J. **The Data Warehouse ETL Toolkit**. Nova Iorque: John Wiley & Sons, 2004.

KLINGENBERG, C. O.; BORGES, M. A. V.; ANTUNES, J. A. V. Industry 4.0 as a data-driven paradigm: a systematic literature review on technologies. **Journal of Manufacturing Technology Management**, v. 32, n. 3, p. 570–592, 2021.

KNAPP, G. M.; WANG, H. P. Acquiring, storing and utilizing process planning knowledge using neural networks. **Journal of Intelligent Manufacturing**, v. 3, n. 5, p. 333–344, 1992.

KOK, J. N. et al. Artificial Intelligence: Definition, Trends, Techniques and Cases. **Artificial Intelligence**, v. 1, p. 1096–1097, 2009.

KOLOKAS, N. et al. **Forecasting faults of industrial equipment using machine learning classifiers**. 2018 IEEE (SMC) International Conference on Innovations in Intelligent Systems and Applications. **Anais**...Thessaloniki: IEEE, 2018

KOMODA, N. **Service Oriented Architecture (SOA) in industrial systems**. 2006 IEEE International Conference on Industrial Informatics. **Anais**...Singapura: IEEE, 2006

KRATSCH, W. et al. Machine Learning in Business Process Monitoring: A Comparison of Deep Learning and Classical Approaches Used for Outcome Prediction. **Business and Information Systems Engineering**, 2020.

KU, W.; STORER, R. H.; GEORGAKIS, C. Disturbance detection and isolation by dynamic principal component analysis. **Chemometrics and Intelligent Laboratory Systems**, v. 30, n. 1, p. 179–196, 1995.

KUHNLE, A. et al. Designing an adaptive production control system using reinforcement learning. **Journal of Intelligent Manufacturing**, v. 32, n. 3, p. 855–876, 2021.

KULCSÁR, T. et al. Multivariate statistical and computational intelligence techniques for quality monitoring of production systems. **Intelligent Systems Reference Library**, v. 97, n.

December 2019, p. 237–263, 2016.

KUMAR, V. Feature Selection: A literature Review. **The Smart Computing Review**, v. 4, n. 3, 2014.

LANG, D. et al. **Utilization of the asset administration shell to support humans during the maintenance process**. International Conference on Industrial Informatics. **Anais**...Espoo: IEEE, 2019

LASKEY, K. B.; LASKEY, K. Service oriented architecture. **Wiley Interdisciplinary Reviews: Computational Statistics**, v. 1, n. 1, p. 101–105, 2009.

LATECKI, L. J.; LAZAREVIC, A.; POKRAJAC, D. **Outlier detection with kernel density functions**. International Conference on Machine Learning and Data Mining. **Anais**...Leipzig: Institute for Computer Vision and Applied Computer Science, 2007

LEE, G. Michael Stonebraker on the Importance of Data Integration. **IT Professional**, v. 1, n. 3, p. 79–80, 1999.

LEE, I. Big data: Dimensions, evolution, impacts, and challenges. **Business Horizons**, v. 60, n. 3, p. 293–303, 2017.

LEE, J. M.; YOO, C. K.; LEE, I. B. Statistical process monitoring with independent component analysis. **Journal of Process Control**, v. 14, n. 5, p. 467–485, 2004.

LEE, K. B.; CHEON, S.; KIM, C. O. A convolutional neural network for fault classification and diagnosis in semiconductor manufacturing processes. **IEEE Transactions on Semiconductor Manufacturing**, v. 30, n. 2, p. 135–142, 2017.

LEE, S. Y.; FISCHER, G. W. Grouping parts based on geometrical shapes and manufacturing attributes using a neural network. **Journal of Intelligent Manufacturing**, v. 10, n. 2, p. 199–209, 1999.

LI, X.; OLAFSSON, S. Discovering dispatching rules using data mining. **Journal of Scheduling**, v. 8, n. 6, p. 515–527, 2005.

LI, Z. et al. Prediction of surface roughness in extrusion-based additive manufacturing with machine learning. **Robotics and Computer-Integrated Manufacturing**, v. 57, n. October 2018, p. 488–495, 2019.

LICHTBLAU, K. et al. **Industry 4.0 Readiness.** AachenImpuls Stiftung - VDMA, 2015.

LIEBER, D. et al. Quality prediction in interlinked manufacturing processes based on supervised & unsupervised machine learning. **Procedia CIRP**, v. 7, p. 193–198, 2013.

LIEW, A. DIKIW: Data, information, knowledge, intelligence, wisdom and their interrelationships. **Business Management Dynamics**, v. 2, n. 10, p. 49–62, 2013.

LIN, S.-W. et al. **The Industrial Internet of Things - Volume G1: Reference Architecture**. Disponível em: <https://www.iiconsortium.org/pdf/IIRA-v1.9.pdf>. Acesso em: 27 jan. 2023.

LIU, Y. et al. **A framework for scheduling in cloud manufacturing with deep reinforcement learning**. International Conference on Industrial Informatics. **Anais**...Espoo: IEEE, 2019

LIU, Y.; VITOLO, T. M. **Graph data warehouse: Steps to integrating graph databases into the traditional conceptual structure of a data warehouse**. International Congress on Big Data. **Anais**...Santa Clara: IEEE, 2013

LOURENÇO, J. R. et al. Choosing the right NoSQL database for the job: a quality attribute evaluation. **Journal of Big Data**, v. 2, n. 1, p. 1–26, 2015.

MAIA, T.; QUEIROZ, R.; GOMES, B. **Utilização de Modelos de Machine Learning para Predição de Falhas em Rolamentos Principais de Aerogeradores**. Brazil Wind Power. **Anais**...São Paulo: ABEEólica, 2022

MANSOURI, S. S. et al. Remaining Useful Battery Life Prediction for UAVs based on Machine Learning. **IFAC-PapersOnLine**, v. 50, n. 1, p. 4727–4732, 2017.

MANYIKA, J. et al. **Big data: The next frontier for innovation, competition, and productivity**. Disponível em: <https://www.mckinsey.com/business-functions/mckinsey-digital/our-insights/big-data-the-next-frontier-for-innovation>. Acesso em: 15 jun. 2021.

MARCONI, M.; LAKATOS, E. **Fundamentos de metodologia científica**. São Paulo: Atlas, 2003.

MATHEW, V. et al. **Prediction of Remaining Useful Lifetime (RUL) of turbofan engine using machine learning**. International Conference on Circuits and Systems. **Anais**...Thiruvananthapuram: IEEE, 2018

MATUSZYK, T. I.; CARDEW-HALL, M. J.; ROLFE, B. F. The kernel density estimate/point distribution model (KDE-PDM) for statistical shape modeling of automotive stampings and assemblies. **Robotics and Computer-Integrated Manufacturing**, v. 26, n. 4, p. 370–380, 2010.

MAVRIDIS, I.; KARATZA, H. Combining containers and virtual machines to enhance isolation and extend functionality on cloud computing. **Future Generation Computer Systems**, v. 94, p. 674–696, 2019.

MCCARTHY, J. et al. A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence. **AI Magazine**, v. 27, n. 4, p. 12–14, 2006.

MEIDAN, Y. et al. Cycle-time key factor identification and prediction in semiconductor

manufacturing using machine learning and data mining. **IEEE Transactions on Semiconductor Manufacturing**, v. 24, n. 2, p. 237–248, 2011.

MILOSLAVSKAYA, N.; TOLSTOY, A. Big data, fast data and data lake concepts. **Procedia Computer Science**, v. 88, p. 300-305, 2016.

MITCHELL, T. **Machine Learning**. Nova Iorque: McGraw Hill, 1997.

MITRA, P. et al. Unsupervised Feature Selection Using Feature Similarity. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 24, n. 3, p. 301–312, 2002.

MIYAGI, P. E. **Controle programável : fundamentos do controle de sistemas a eventos discretos**. São Paulo: Blucher, 1996.

MONIRUZZAMAN, A. B. M.; HOSSAIN, S. A. NoSQL Database: New Era of Databases for Big data Analytics - Classification, Characteristics and Comparison. **International Journal of Database Theory and Application**, v. 6, n. 4, p. 1–14, 2013.

MONOSTORI, L. et al. Machine learning approaches to manufacturing. **CIRP Annals - Manufacturing Technology**, v. 45, n. 2, p. 675–712, 1996.

MORALES, E. F.; ZARAGOZA, J. H. An introduction to reinforcement learning. In: **Decision Theory Models for Applications in Artificial Intelligence: Concepts and Solutions**. 1. ed. Pensilvânia: IGI Global, 2011. p. 63–80.

MURDOCH, T.; BALL, N. Machine learning in configuration design. **Artificial Intelligence for Engineering Design, Analysis and Manufacturing: AIEDAM**, v. 10, n. 2, p. 101–113, 1996.

NAKAYAMA, R. S.; SPÍNOLA, M. DE M.; SILVA, J. R. Towards I4.0: A comprehensive analysis of evolution from I3.0. **Computers and Industrial Engineering**, v. 144, n. April, p. 106453, 2020.

NAMBIAR, A.; MUNDRA, D. An Overview of Data Warehouse and Data Lake in Modern Enterprise Data Management. **Big Data and Cognitive Computing**, v. 6, n. 4, p. 132-148, 2022.

NEWMAN, S. **Building Microservices**. Sebastopol: O'Reilly Media, 2015.

NIST BIG DATA PUBLIC WORKING GROUP. **NIST Big Data Interoperability Framework : Volume 6 - Reference Architecture**. Gaithersburg, 2019. Disponível em: <https://doi.org/10.6028/NIST.SP.1500-6r2>. Acesso em 27 Jan. 2023.

OGATA, K. **Engenharia de Controle Moderno**. 5. ed. Hoboken: Pearson, 2014.

OLIVEIRA, V. F. DE et al. **Infraestrutura de Dados para Sistemas de Manufatura Inteligente**. 14th IEEE International Conference on Industry Applications. **Anais**...São Paulo:

IEEE, 2021

OPC - OPEN PLATFORM COMMUNICATIONS. **OPC 10000-6: OPC Unified Architecture - Part 6: Mappings**. OPC Foundation, 2017.

OPPITZ, M.; TOMSU, P. **Inventing the Cloud Century: How Cloudiness Keeps Changing Our Life, Economy and Technology**. Cham: Springer, 2018.

OSIA, S. A. et al. Private and Scalable Personal Data Analytics Using Hybrid Edge-to-Cloud Deep Learning. **Computer**, v. 51, n. 5, p. 42–49, 2018.

ÖZSU, M. T.; VALDURIEZ, P. Distributed and parallel database systems. **ACM Computing Surveys**, v. 28, n. 1, p. 125–128, 2004.

ÖZSU, M. T.; VALDURIEZ, P. **Principles of Distributed Database Systems**. 3. ed. Londres: Springer, 2011.

PANG, C. K. et al. **Intelligent Energy Audit and Machine Management for Energy-Efficient Manufacturing**. International Conference on Cybernetics and Intelligent Systems. **Anais**...Qingdao: IEEE, 2011

PARK, I. B. et al. A Reinforcement Learning Approach to Robust Scheduling of Semiconductor Manufacturing Facilities. **IEEE Transactions on Automation Science and Engineering**, v. 17, n. 3, p. 1420–1431, 2020.

PERREY, R.; LYCETT, M. **Service-oriented architecture**. Symposium on Applications and the Internet Workshops. **Anais**...Orlando: IEEE, 2003

PILLONI, V. How data will transform industrial processes: Crowdsensing, crowdsourcing and big data as pillars of industry 4.0. **Future Internet**, v. 10, n. 4, 2018.

PISCHING, M. A. et al. An architecture based on RAMI 4.0 to discover equipment to process operations required by products. **Computers and Industrial Engineering**, v. 125, n. January, p. 574–591, 2018.

PISCHING, M. A. **Arquitetura para descoberta de equipamentos em processos de manufatura com foco na Indústria 4.0.** 2018. Tese (Mestrado em Engenharia Mecânica) - Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos, Universidade de São Paulo, São Paulo, 2018.

POLYDOROS, A. S.; NALPANTIDIS, L. Survey of Model-Based Reinforcement Learning: Applications on Robotics. **Journal of Intelligent and Robotic Systems: Theory and Applications**, v. 86, n. 2, p. 153–173, 2017.

PORTO, L. C. 2008. **Metodologia para prognosticar a reduçao de temperatura do enrolamento de campo de um hydrogenerator**. 2008. Dissertação (Mestrado em Engenharia

Mecânica) - Departamento de Engenharia Mecânica, Universidade Federal de Minas Gerais, Belo Horizonte, 2008.

PRIORE, P. et al. Dynamic scheduling of manufacturing systems using machine learning: An updated review. **Artificial Intelligence for Engineering Design, Analysis and Manufacturing: AIEDAM**, v. 28, n. 1, p. 83–97, 2014.

PRYTZ, R. et al. Predicting the need for vehicle compressor repairs using maintenance records and logged vehicle data. **Engineering Applications of Artificial Intelligence**, v. 41, p. 139–150, 2015.

QIAN, M.; ZHAI, C. Robust Unsupervised Feature Selection. **IJCAI International Joint Conference on Artificial Intelligence**, p. 1621–1627, 2013.

RAI, P.; SINGH, S. A Survey of Clustering Techniques. **International Journal of Computer Applications**, v. 7, n. 12, p. 1–5, 2010.

RAMAKRISHNAN, R.; GEHRKE, J. **Sistemas de gerenciamento de banco de dados**. 3. ed. Nova Iorque: McGraw Hill, 2008. v. 14

RAPTIS, T. P.; PASSARELLA, A.; CONTI, M. Data management in industry 4.0: State of the art and open challenges. **IEEE Access**, v. 7, p. 97052–97093, 2019.

RIBEIRO, B. Support vector machines for quality monitoring in a plastic injection molding process. **IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews**, v. 35, n. 3, p. 401–410, 2005.

RIPEANU, M. **Peer-to-peer architecture case study: Gnutella network**. International Conference on Peer-to-Peer Computing-to-Peer Computing. **Anais**...Linkoping: IEEE, 2001

RODRÍGUEZ, J. D.; PÉREZ, A.; LOZANO, J. A. Sensitivity Analysis of k-Fold Cross Validation in Prediction Error Estimation. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 32, n. 3, p. 569–575, 2010.

ROMEO, L. et al. Machine learning-based design support system for the prediction of heterogeneous machine parameters in industry 4.0. **Expert Systems with Applications**, v. 140, p. 1–15, 2020.

ROSEN, M. et al. **Applied SOA: Service-Oriented Archtiecture and Design Strategies**. Hoboken: Wiley Publishing, 2008.

ROWLEY, J. The wisdom hierarchy: Representations of the DIKW hierarchy. **Journal of Information Science**, v. 33, n. 2, p. 163–180, 2007.

RUBIN, V. L. Veracity roadmap: Is big data objective, truthful and credible? **Advances in Classification Research Online**, v. 24, p. 4–15, 2014.

RUSSELL ACKOFF. From Data to Wisdom. **Journal of applied systems analysis**, v. 16, n. 1, 1989.

RUSSMANN, M. et al. Industry 4.0: World Economic Forum. **The Boston Consulting Group**, p. 1–20, 2015.

RUSSOM, P. **Big Data Analytics**. Disponível em: <https://tdwi.org/research/2011/09/~/media/TDWI/TDWI/Research/BPR/2011/TDWI_BPReport_Q411_Big_Data_Analytics_Web/TDWI_BPReport_Q411_Big Data_ExecSummary.ashx>. Acesso em: 27 jan. 2023.

SADALAGE, P. J.; FOWLER, M. **NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence**. Hoboken: Pearson, 2013.

SAYADI, H. et al. Machine learning-based approaches for energy-efficiency prediction and scheduling in composite cores architectures. **Proceedings - 35th IEEE International Conference on Computer Design, ICCD 2017**, p. 129–136, 2017.

SCABORA, L. C. et al. **Physical data warehouse design on NoSQL databases: OLAP query processing over HBase**. Proceedings of the 18th International Conference on Enterprise Information Systems. **Anais**...Roma: SciTePress, 2016

SCHAAL, S. Learning from demonstration. **Advances in Neural Information Processing Systems**, p. 1040–1046, 1997.

SCHMIDT, G. Case-based reasoning for production scheduling. **International Journal of Production Economics**, v. 56–57, n. 97, p. 537–546, 1998.

SCHOLLMEIER, R. **A definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications**. International Conference on Peer-to-Peer Computing. **Anais**...Linkoping: IEEE, 2001

SCHUBERT, E.; ZIMEK, A.; KRIEGEL, H. P. **Generalized outlier detection with flexible kernel density estimates**. International Conference on Data Mining. **Anais**...Filadélfia: Society for Industrial and Applied Mathematics, 2014

SCHWAB, K. **The Fourth Industrial Revolution**. Geneva: World Economic Forum, 2017.

SHAHZAD, A.; MEBARKI, N. **Discovering dispatching rules for job shop scheduling problem through data mining**. 8th International Conference ofModeling and Simulation. **Anais**...Hammamet: Association for Computing Machinery, 2010

SHAW, M. J.; PARK, S.; RAMAN, N. Intelligent Scheduling with Machine Learning Capabilities: The Induction of Scheduling Knowledge. **IIE Transactions (Institute of Industrial Engineers)**, v. 24, n. 2, p. 156–168, 1992.

SILVA, T. C. DA. **Proposta de Planejamento de Manutenção de Aerogeradores, com Base nas Condições Ambientais Locais**. 2020. Dissertação (Mestrado em Engenharia de Energia) - Centro de Engenharia, Modelagem e Ciências Sociais Aplicadas, Universidade Federal do ABC, Santo André, 2020.

SIM, S. K.; DUFFY, A. H. B. A foundation for machine learning in design. **Artificial Intelligence for Engineering Design, Analysis and Manufacturing: AIEDAM**, v. 12, n. 2, p. 193–209, 1998.

SINGH, S. et al. Convergence Results for Single-Step On-Policy Reinforcement-Learning Algorithms. **Machine Learning**, v. 39, n. 1998, p. 287–308, 2000.

SINGHAL, S.; JENA, M. A Study on WEKA Tool for Data Preprocessing , Classification and Clustering. **International Journal of Innovative Technology and Exploring Engineering**, v. 2, n. 6, p. 250–253, 2013.

SOUALHI, A.; CLERC, G.; RAZIK, H. Detection and diagnosis of faults in induction motor using an improved artificial ant clustering technique. **IEEE Transactions on Industrial Electronics**, v. 60, n. 9, p. 4053–4062, 2013.

SPERA, D. A. **Wind turbine Technology: Fundamental Concepts of Wind Turbine Engine**. 2. ed. Nova Iorque: ASME Press, 1979.

SPROTT, D.; WILKES, L. Understanding Service-Oriented Architecture. **The Architecture Journal**, v. 1, n. 1, 2004.

SRINIVASAN, R. et al. Context-based recognition of process states using neural networks. **Chemical Engineering Science**, v. 60, n. 4, p. 935–949, 2005.

STONEBRAKER, M. SQL databases v. NoSQL databases. **Communications of the ACM**, v. 53, n. 4, p. 10–11, 2010.

SU, C. J.; HUANG, S. F. Real-time big data analytics for hard disk drive predictive maintenance. **Computers and Electrical Engineering**, v. 71, n. August, p. 93–101, 2018.

SUN, S. et al. Data handling in industry 4.0: Interoperability based on distributed ledger technology. **Sensors**, v. 20, n. 11, p. 1–22, 2020.

SUTTON, R. S.; BARTO, A. G. **Reinforcement Learning: An Introduction**. Cambridge: MIT Press, 2020.

SUTTON, R. S.; BARTO, A. G.; WILLIAMS, R. J. Reinforcement Learning is Direct Adaptive Optimal Control. **IEEE Control Systems Magazine**, v. 12, n. 2, 1992.

TANG, B.; HE, H. A local density-based approach for outlier detection. **Neurocomputing**, v. 241, p. 171–180, 2017.

TAO, F. et al. Data-driven smart manufacturing. **Journal of Manufacturing Systems**, v. 48, p. 157–169, 2018.

TAO, F. et al. Digital Twins and Cyber – Physical Systems toward Smart Manufacturing and Industry 4 . 0: Correlation and Comparison. **Engineering**, v. 5, n. 4, p. 653–661, 2019.

TAVNER, P. et al. Influence of Wind Speed on Wind Turbine Reliability. **Wind Engineering**, v. 30, n. 1, p. 55–72, 2006.

TAVNER, P. J. et al. **Study of weather and location effects on wind turbine failure rates**. European Wind Energy Conference. **Anais**...Varsóvia: The European Wind Energy Association, 2010

TEOREY, T. et al. **Database Modeling and Design**. Burlington,: Elsevier, 2005.

VAN DER MAATEN, L. J. P.; POSTMA, E. O.; VAN DEN HERIK, H. J. Dimensionality Reduction: A Comparative Review. **Journal of Machine Learning Research**, v. 10, p. 1–41, 2009.

VICKNAIR, C. et al. **A comparison of a graph database and a relational database: A data provenance perspective**. Proceedings of the Annual Southeast Conference. **Anais**...Kissimmee: Southern Region of Delta Sigma Theta Sorority, 2010

WAHID, A.; ANNAVARAPU, C. S. R. NaNOD: A natural neighbour-based outlier detection algorithm. **Neural Computing and Applications**, v. 33, n. 6, p. 2107–2123, 2021.

WAN, J.; CAI, H.; ZHOU, K. **Industrie 4.0: Enabling technologies**. International Conference on Intelligent Computing and Internet of Things. **Anais**...Harbin: IEEE, 2015

WANG, Y. C.; USHER, J. M. Application of reinforcement learning for agent-based production scheduling. **Engineering Applications of Artificial Intelligence**, v. 18, n. 1, p. 73–82, 2005.

WANG, Z. et al. **Pagrol: Parallel graph olap over large-scale attributed graphs**. International Conference on Data Engineering. **Anais**...Chicago: IEEE, 2014

WASCHNECK, B. et al. Optimization of global production scheduling with deep reinforcement learning. **Procedia CIRP**, v. 72, p. 1264–1269, 2018.

WATSON, H. J.; SUDIBYO, A. Which Data Warehouse Architecture Is Most Successful. **Business Intelligence Journal**, v. 11, n. 1, p. 4–6, 2006.

WEGNER, P. Interoperability. **ACM Computing Surveys**, v. 28, n. 1, p. 285–287, 1996.

WHITEHALL, B. L.; LU, S. C. Y.; STEPP, R. E. CAQ: A machine learning tool for engineering. **Artificial Intelligence in Engineering**, v. 5, n. 4, p. 189–198, 1990.

WUEST, T. et al. Machine learning in manufacturing: Advantages, challenges, and applications. **Production and Manufacturing Research**, v. 4, n. 1, p. 23–45, 2016.

WUEST, T.; IRGENS, C.; THOBEN, K. D. APA. **Journal of Intelligent Manufacturing**, v. 25, n. 5, p. 1167–1180, 2014.

XAVIER, M. G. et al. **Performance evaluation of container-based virtualization for high performance computing environments**. Euromicro International Conference on Parallel, Distributed, and Network-Based Processing. **Anais**...Belfast: IEEE, 2013

YAN, J. et al. Effective and efficient dimensionality reduction for large-scale and streaming data preprocessing. **IEEE Transactions on Knowledge and Data Engineering**, v. 18, n. 3, p. 320–332, 2006.

YANGUI, R.; NABLI, A.; GARGOURI, F. Automatic Transformation of Data Warehouse Schema to NoSQL Data Base: Comparative Study. **Procedia Computer Science**, v. 96, n. September, p. 255–264, 2016.

YAO, L.; SHAO, W.; GE, Z. Hierarchical Quality Monitoring for Large-Scale Industrial Plants With Big Process Data. **IEEE transactions on neural networks and learning systems**, v. 32, n. 8, p. 1–12, 2019.

YE, X. et al. Toward the Plug-and- Produce Capability for Industry 4.0. **IEEE Industrial Electronics Magazine**, v. 14, n. 4, 2020.

YE, X.; HONG, S. H. Toward Industry 4.0 Components: Insights into and Implementation of Asset Administration Shells. **IEEE Industrial Electronics Magazine**, v. 13, n. 1, p. 13–25, 2019.

YIN, S.; KAYNAK, O. Big Data for Modern Industry: Challenges and Trends. **Proceedings of the IEEE**, v. 103, n. 2, p. 143–146, 2015.

YLI-OJANPERÄ, M. et al. Adapting an agile manufacturing concept to the reference architecture model industry 4.0: A survey and case study. **Journal of Industrial Information Integration**, v. 15, n. December 2018, p. 147–160, 2019.

YOO, C. K. et al. On-line monitoring of batch processes using multiway independent component analysis. **Chemometrics and Intelligent Laboratory Systems**, v. 71, n. 2, p. 151–163, 2004.

YU, J. A support vector clustering-based probabilistic method for unsupervised fault detection and classification of complex chemical processes using unlabeled data. **AIChE Journal**, v. 59, n. 2, p. 407–419, 2013.

YU, J.; XI, L.; ZHOU, X. Intelligent monitoring and diagnosis of manufacturing processes using an integrated approach of KBANN and GA. **Computers in Industry**, v. 59, n. 5, p. 489–501, 2008.

ZHANG, H. et al. **SLAQ: Quality-Driven Scheduling for Distributed Machine Learning**. Symposium on Cloud Computing. **Anais**...Santa Clara: Association for Computing Machinery, 2017

ZHANG, T. et al. Learning deep control policies for autonomous aerial vehicles with MPC-guided policy search. **Proceedings - IEEE International Conference on Robotics and Automation**, v. 2016- June, p. 528–535, 2016.

ZHANG, X. et al. **Data Format Conversions**. Disponível em: <https://docs.microsoft.com/en-us/azure/machine-learning/studio-module-reference/data-format-conversions>. Acesso em: 15 jun. 2021a.

ZHANG, Y. et al. **Fusion OLAP: Fusing the pros of MOLAP and ROLAP together for in-memory OLAP**. International Conference on Data Engineering. **Anais**...Macau: IEEE, 2019b

ZHANG, Y.; QIN, S. J. Fault detection of nonlinear processes using multiway kernel independent component analysis. **Industrial and Engineering Chemistry Research**, v. 46, n. 23, p. 7780–7787, 2007.

ZHANG, Z. et al. Semiconductor final test scheduling with Sarsa(λ, k) algorithm. **European Journal of Operational Research**, v. 215, n. 2, p. 446–458, 2011.

ZHANG, Z. Automatic Fault Prediction of Wind Turbine Main Bearing Based on SCADA Data and Artificial Neural Network. **Open Journal of Applied Sciences**, v. 08, n. 06, p. 211–225, 2018.

ZHU, H. et al. A Deep-Reinforcement-Learning-Based Optimization Approach for Real-Time Scheduling in Cloud Manufacturing. **IEEE Access**, v. 8, p. 9987–9997, 2020.