UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO DE JOINVILLE
CURSO DE ENGENHARIA MECATRÔNICA

ERYK KOOSHIN SUGUIURA

ARQUITETURA DE COMUNICAÇÃO AUTOMOTIVA APLICADA PARA KART ELÉTRICO

Joinville
2023

ERYK KOOSHIN SUGUIURA

ARQUITETURA DE COMUNICAÇÃO AUTOMOTIVA APLICADA PARA KART ELÉTRICO

Trabalho de Conclusão de Curso apresentado como requisito parcial para obtenção do título de bacharel em Engenharia Mecatrônica no curso de Engenharia Mecatrônica, da Universidade Federal de Santa Catarina, Centro Tecnológico de Joinville.

Orientador: Dr. Gian Ricardo Berkenbrock

Joinville
2023

AGRADECIMENTOS

Gostaria de expressar minha profunda gratidão à minha família, em especial aos meus pais, cujo apoio inabalável e incentivo foram a bússola que guiou esta jornada acadêmica. Agradeço também aos meus queridos avós, tios e tias, cuja presença constante e amor foram fontes inestimáveis de inspiração.

Ao professor Dr. Gian Ricardo Berkenbrock, meu orientador, devo um agradecimento especial. Sua orientação experiente e opiniões valiosas foram fundamentais para a conclusão deste trabalho. Agradeço igualmente aos professores que gentilmente aceitaram o convite para compor a banca, o professor Ricardo Jose Pfitscher e o professor Antônio Otaviano Dourado.

Quero estender meus agradecimentos a todos os amigos que, ao longo do curso, ofereceram apoio, conselhos e compartilharam suas experiências. Suas contribuições foram essenciais para o meu crescimento acadêmico e pessoal.

RESUMO

Este trabalho contempla a análise de modelos de arquiteturas automotivas de comunicação, com intuito de aplicação em um kart elétrico. Tal análise é justificada por definir os aspectos sobre a escolha de protocolos adequados para a concepção de uma arquitetura, e na aplicação a ser desenvolvida. Assim, o objetivo deste trabalho é apresentar modelos, implementar módulos de comunicação, analisá-los, compará-los e por fim, definir o mais adequado para o sistema de um kart elétrico, auxiliando futuros desenvolvimentos sobre o veículo, ao qual este trabalho foi aplicado, provendo documentação dos motivos das escolhas para determinado modelo, facilitando a conexão com possíveis módulos adicionais futuros.

Palavras-chave: Arquitetura automotiva. Kart elétrico. Protocolos de comunicação.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Para o funcionamento de veículos modernos, são utilizados diversos componentes eletrônicos, sensores, atuadores e unidades de controle. As arquiteturas automotivas apresentam componentes para funções diversas, como sistemas de navegação, entretenimento, conforto e telemetria. A arquitetura de comunicação é a responsável pela transmissão de dados entre esses componentes.

Neste trabalho, visa-se explorar uma arquitetura automotiva adaptada para a aplicação em um kart elétrico, buscando a implementação e desenvolvimento da base do projeto. Fundamentado através do detalhamento de diferentes características de arquiteturas automotivas e protocolos de comunicação, como caracterizados no trabalho de Junior (2012).

O objetivo deste estudo é definir um modelo de arquitetura de comunicação automotiva condizente com a estrutura e funcionalidade geral de um kart elétrico. Por meio da análise de protocolos de comunicação automotivos comumente utilizados e sistemas operacionais como *Osek* ou *Autosar*, melhor detalhados na documentação sobre o tema por Lee, Kim e Jeon (2013). O projeto foi inspirado em propostas de arquiteturas de comunicação conhecidas, como os presentes em Staron (2021) e Junior (2012), e adaptadas para implementação escolhida.

Primeiramente, são introduzidos os conhecimentos teóricos sobre software, redes e topologias de arquiteturas automotivas e em seguida, da plataforma de desenvolvimento e linguagem de descrição dos modelos. Seguindo essas informações, é delimitado o escopo do projeto e apresentados os requisitos para comparação das arquiteturas, e então, obtidos os dados das análises das arquiteturas, as arquiteturas são comparadas e mostrados os resultados.

1.1 OBJETIVOS

1.1.1 Objetivo Geral

Definir um projeto específico de arquitetura de comunicação automotiva para a aplicação em um kart elétrico.

1.1.2 Objetivos Específicos

- Diferenciar as arquiteturas de rede de comunicação automotivas;
- Modelar arquiteturas automotivas para implementação;
- Implementar modelos para obter dados de análise;
- Comparar modelos pelas análises obtidas;

2 FUNDAMENTAÇÃO TEÓRICA

Para a estruturação deste capítulo, são apresentados os conceitos de arquitetura de software automotivos, abordagens de projeto de arquiteturas automotivas, redes automotivas, linguagens de descrição para arquiteturas, plataformas de desenvolvimento, métricas e trabalhos relacionados.

A função de uma arquitetura de comunicação automotiva é definir como os dados devem ser distribuídos pelo veículo, com ênfase na comunicação entre unidades de controle eletrônico (ECU), responsáveis pelo controle de funções que estarão presentes no veículo. Observando o aspecto dos sistemas embarcados necessários para as mesmas, o uso de uma arquitetura de software padronizada para acelerar desenvolvimento de projetos é comum.

2.1 ARQUITETURA DE SOFTWARE

Nesta seção serão construídas as ideias sobre arquitetura de software, as variações necessárias para o contexto automotivo e apresentadas formas de visualização reconhecidas para essas arquiteturas.

No escopo da engenharia de software, a “Arquitetura de Software” refere-se à estrutura fundamental que delineia a organização de um sistema de software, (BASS; CLEMENTS; KAZMAN, 2003). Esta estrutura abrange componentes, suas relações e os princípios que guiam seu design e evolução. A definição da arquitetura de software ocorre na fase inicial do ciclo de desenvolvimento, e busca estabelecer uma compreensão clara das interações entre os diversos elementos do sistema, viabilizando testes e manutenção do software.

A arquitetura de software é composta por diferentes perspectivas, incluindo a visão lógica que descreve a funcionalidade do sistema, a visão física representando a distribuição dos componentes em hardware, e a visão de processos que define o comportamento dinâmico do sistema. A identificação de padrões e decisões de projeto que impactam na organização do software visando garantir requisitos como, desempenho, confiabilidade e segurança ao longo do ciclo de vida do sistema, também são relevantes a essa arquitetura.

2.1.1 Arquitetura de Software Automotivo

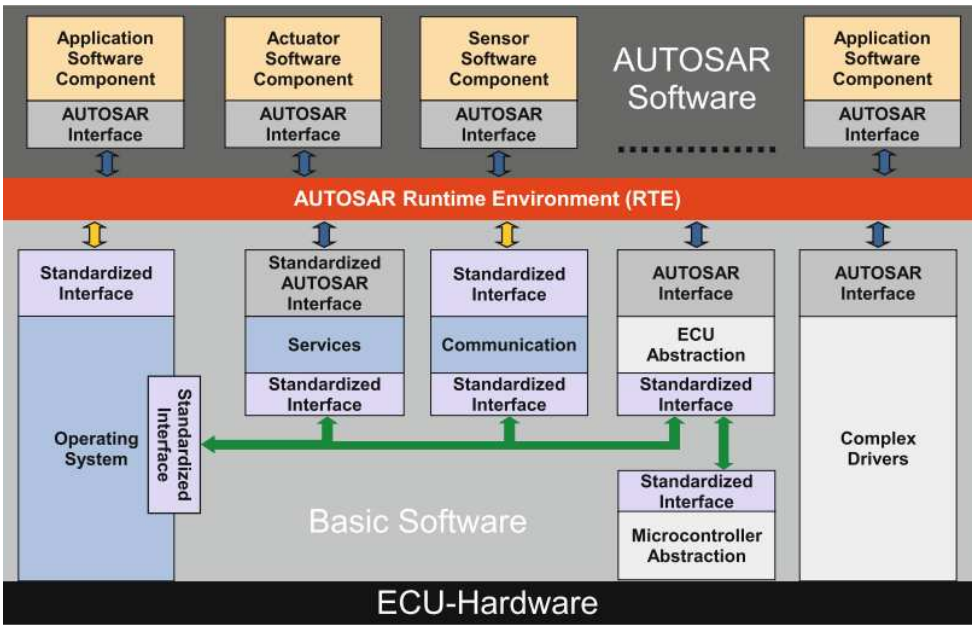
O sistema mais utilizado na área de redes de comunicação para automóveis é chamado de *Offene Systeme und deren Schnittstellen für die Elektronik in Kraftfahrzeugen* (OSEK), (SUN; WANG, 2005). Apesar de não definir características

específicas, como os protocolos de comunicação para cada barramento, possui o intuito de funcionar em microcontroladores sem gerenciamento de memória internos com implementações configuradas em tempo de compilação. Sendo favoráveis para sistemas de segurança crítica, no qual carros e automóveis terrestres estão categorizados, (MALATERRE, 1998).

Atualmente, o uso da arquitetura OSEK está em declínio devido ao desenvolvimento da *Automotive Open System Architecture* (AUTOSAR), uma vertente do OSEK atualizada para aplicações modernas. Ela faz uso das mesmas bases de implementação e organização, entretanto, adicionando configurações de modularidade e maneiras simples de adaptação para novas tecnologias relacionadas à interconectividade de veículos automotivos (LEE; KIM; JEON, 2013).

A AUTOSAR segue uma arquitetura de três camadas: a camada Application Software (ASW), responsável pelos componentes de aplicação em software; a camada Realtime Environment (RTE), um canal de comunicação entre as outras duas camadas, fazendo o uso de portas e interfaces por meio de *buffers* de sistemas de tempo real; e a camada Basic Software (BSW), que realiza o acesso de componentes de entradas e saídas, gerenciamento de memória, serviços de criptografia e *driver*s de componentes complexos.

Figura 1 – Camadas da AUTOSAR



Fonte: Fürst e Bechter (2016)

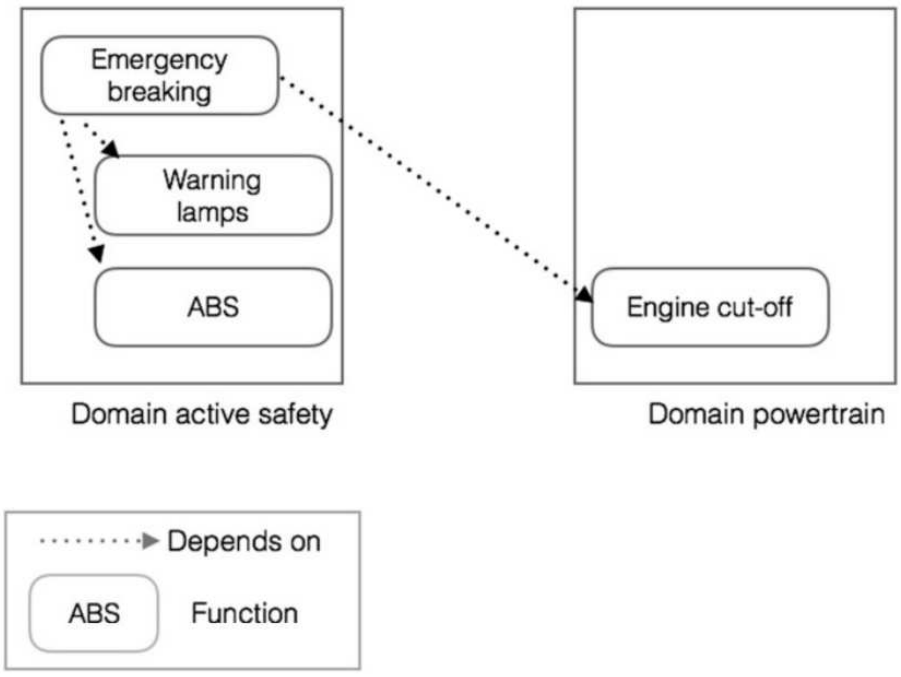
A Figura 1 apresenta uma ilustração das três camadas de aplicação da AUTOSAR. Ademais das críticas fundamentadas em testes, de Fürst e Bechter (2016), os autores consideram que o aspecto de separação entre aplicação e a infraestrutura fornecida pela AUTOSAR suplanta suas deficiências.

Considerando as linguagens de descrição de arquiteturas, foram observadas as mais comumente utilizadas no contexto de modelagem de veículos terrestres, sendo elas a linguagem Unified Modeling Language (UML), Systems Modeling Language (SySML) e a Architecture Description Language (ADL) cada uma com um uso específico no contexto de descrição de arquiteturas. Entretanto, a existência de sub linguagens diretamente derivadas da ADL, específicas para a arquitetura de carros e sistemas de tempo real, direcionaram um foco maior sobre esta linguagem.

2.1.2 Formato de visualização de arquiteturas

Existem muitas maneiras para visualização geral da arquitetura de veículos, e a definição de qual modelo seguir depende da preferência do autor e qual das visões traz uma melhor representação dos objetivos de criação dos modelos, seguindo as visões demonstradas pelo (STARON, 2021), alguns exemplos são; o modelo em “V”, o modelo “4+1”, por visão funcional, por visão de sistemas físicos e por visão lógica.

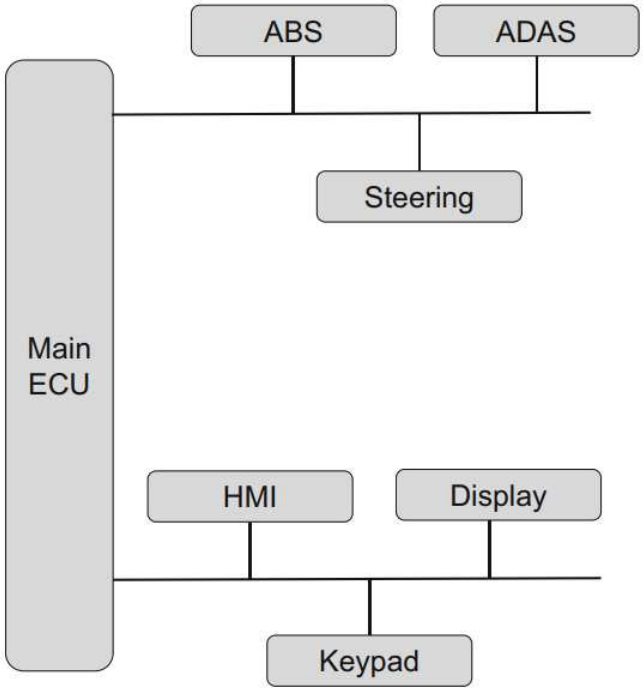
Figura 2 – Exemplo de visão funcional



Fonte: Automotive Software Architectures, Staron Miroslaw

A visão funcional, muitas vezes abreviada para arquitetura funcional, é a visão onde o foco está nas funções do veículo e suas dependências entre si. Um exemplo de tal visão é mostrado na Figura 2, nela existem três tipos de elementos, as funções, representadas como retângulos de arestas arredondadas, os domínios, simbolizados como retângulos comuns, e as relações de dependência, retratadas como setas tracejadas, como as funções podem depender uns dos outros e podem ser facilmente agrupados em “domínios”, como “powertrain” e “active safety”.

Figura 3 – Exemplo de visão do sistema físico



Fonte: Automotive Software Architectures, Staron Miroslaw

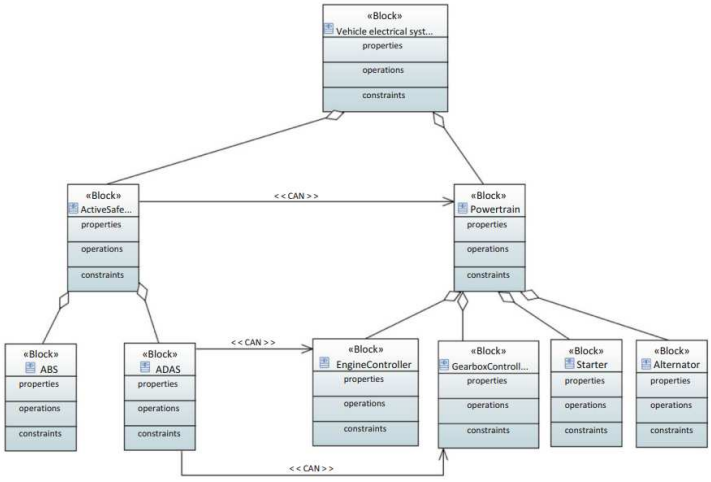
A visão do sistema físico, comumente retratada como uma visão do sistema elétrico em um nível mais alto de abstração, com diagramas de nível mais baixo que o acompanham para complementar informações (por exemplo, diagramas de classe em UML). Nesta vista, apresentado na Figura 3, podemos ver as ECUs (retângulos arredondados) de diferentes tamanhos são colocados em duas linhas físicas. Essa visão da arquitetura oferece a possibilidade de apresentar a topologia do sistema elétrico do carro e fornecer aos arquitetos uma maneira mais simples de raciocinar sobre a posição física dos computadores pelo veículo.

Este tipo de visualização acompanha costumeiramente a topologia centralizada, pois se trata de uma visão bastante simples e estática, com apenas algumas ECUs e poucos barramentos de comunicação. No design de softwares automotivos modernos, essa visão apresenta dificuldades devido ao alto número de ECUs e barramentos de comunicação.

Essa é frequentemente acompanhada pela visão por componentes lógicos, fazendo uso de linguagens de modelagem como UML ou SysML, conforme apresentado na Figura 4. A visão lógica do sistema é centralizada na parte de software do veículo, nela, mostra-se quais classes, módulos e componentes são usados no software do carro e como eles estão relacionados hierarquicamente entre si.

Para a visão lógica, arquitetos costumam usar uma variedade de diagramas de classes para apresentar vários níveis de abstração do software do carro, cada

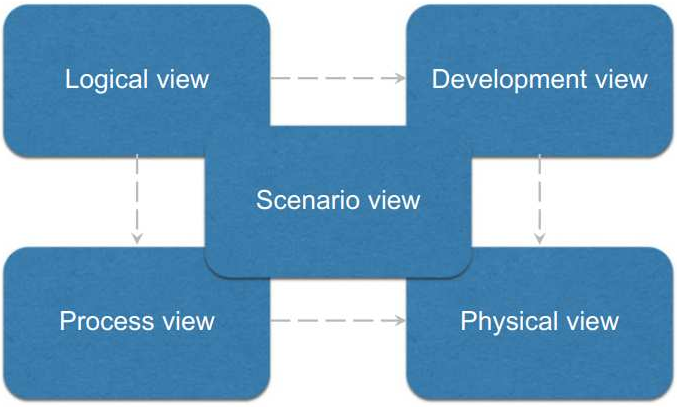
Figura 4 – Exemplo de visão por componentes lógicos



Fonte: Automotive Software Architectures, Staron Miroslaw

uma separada em seu contexto. Para o detalhamento do projeto, esses modelos de arquitetura são complementados com modelos executáveis de baixo nível, como modelos criados no software Matlab/Simulink, definindo o comportamento do software de forma matemática.

Figura 5 – Exemplo de visão por modelo 4+1



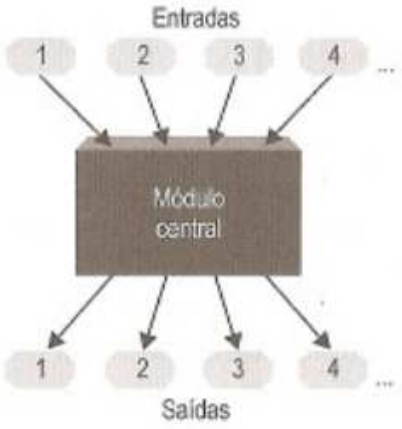
Fonte: Automotive Software Architectures, Staron Miroslaw

As visões mencionadas anteriormente, atualmente usadas na engenharia de software automotiva, acabaram sendo combinadas na visão chamada de “4+1”, exemplo representado pela Figura 5. Este modelo postula que arquiteturas de software devem usar a visão lógica, a visão de desenvolvimento, a visão de processos, a visão física, e uma visão de cenário, acima das outras.

2.2 ABORDAGENS DE PROJETO DE ARQUITETURA

De acordo com Guimarães e Saraiva (2003) as conexões dos componentes de uma arquitetura eletroeletrônica veicular podem ser divididas em arquiteturas centralizadas e arquiteturas distribuídas. A primeira, generalmente caracterizada pela facilidade de aplicação e baixas quantidades de componentes eletrônicos, e a segunda, pela flexibilidade e natureza modular proporcionada pelo uso de várias ECUs.

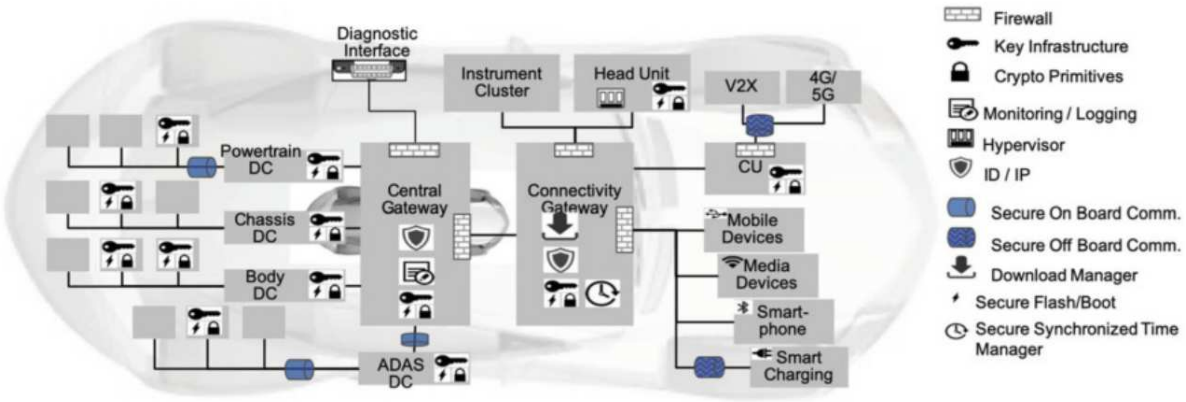
Figura 6 – Exemplo de topologia de Arquitetura centralizada



Fonte: Taliani Junior (2012)

Na arquitetura centralizada, ilustrada na Figura 6, apenas uma ECU assume a função de gerenciar a maioria, ou em alguns casos, todas as funções do veículo. Assim, esta ECU possui entradas responsáveis por receber os sinais provenientes de diversos sensores, computar os dados obtidos e agir sobre os atuadores do sistema. O uso dessa arquitetura é bastante aplicada devido ao baixo nível de complexidade comparado a sua alternativa, mas não será desenvolvido para o kart elétrico pela dificuldade de implementação de desenvolvimentos futuros, justamente pela existência do controlador não-modular único (SOUZA; CAMPOS, 2017).

Figura 7 – Exemplo de modelo de arquitetura distribuída



Fonte: Automotive Software Architectures, Staron Miroslaw

A arquitetura distribuída, ilustrada na Figura 7, em contraste, propicia uma distribuição das funções divididas em diversas ECUs, conectadas entre si por barramentos de comunicação, aumentando a dificuldade de implementação inicial, porém, possibilitando a implementação de módulos adicionais. Os barramentos aplicam protocolos de comunicação, e um veículo pode conter diversos protocolos funcionando simultaneamente, causando a necessidade de estudo sobre detalhes desses protocolos em seções seguintes, baseados no estudo de Bochmann e Sunshine (1980).

Outra vantagem da arquitetura distribuída é a menor quantidade de cabos necessária para conexão de seus elementos, pois ECUs podem ser estrategicamente posicionadas próximas aos elementos que serão conectados a elas. Propicia-se um menor tempo na montagem do veículo, menos ruídos pela quantidade e comprimento de cabos conectores, bem como uma flexibilidade para alterações, porque as funções estão distribuídas entre as várias ECUs.

De acordo com Mahmud e Alles (2005), a escolha do melhor tipo de arquitetura depende de características do projeto dos sistemas eletroeletrônicos do veículo. Quando o conteúdo eletrônico, número de variáveis de entrada e saída, e recursos são baixos para verificação e validação do sistema, ou seja, um projeto cujo custo é severamente limitado, a melhor escolha é uma arquitetura centralizada. Por outro lado, caso o objetivo seja uma maior compatibilidade com aprimoramentos pósteros, e a aplicação de diversas funcionalidades, a arquitetura distribuída é mais aconselhável.

2.3 REDES AUTOMOTIVAS

Redes Automotivas representam um campo de estudo fundamental na engenharia de sistemas veiculares, abordando as comunicações e a conectividade entre os componentes eletrônicos presentes nos veículos SANTOS (2010). Essas redes são responsáveis por viabilizar a troca de dados entre os sistemas embarcados, como sensores, unidades de controle, dispositivos de segurança e entretenimento, visando aprimorar a funcionalidade, a segurança e a eficiência dos automóveis. Essas redes podem adotar diferentes protocolos e topologias de comunicação, adaptando-se às necessidades específicas dos veículos e dos sistemas que compõem sua arquitetura.

No contexto automotivo, as redes desempenham a interconexão e a integração de sistemas complexos, permitindo a comunicação entre unidades de controle distribuídas, cada uma responsável por funções distintas do veículo. A evolução dessas redes é impulsionada pela crescente demanda por sistemas mais inteligentes e interconectados nos veículos, buscando oferecer recursos avançados de segurança, conforto e eficiência. A complexidade delas requer uma abordagem estruturada na sua concepção e implementação, visando garantir a confiabilidade e a segurança dos sistemas veiculares.

2.3.1 Classificação de redes automotivas

Para a definição de padrões dos níveis envolvidos em um processo de comunicação, a *International Standard Organization* (ISO, 1984) criou o modelo de camadas *Open System Interconnection* (OSI), que define os níveis presentes em um processo de comunicação. Por se tratar de um sistema de padrões aceito internacionalmente, será utilizada no sistema do kart elétrico, entretanto, detalhes sobre ela não foram explorados, com as camadas de aplicação e outras informações melhor definidos em Guimarães (2003).

As classificações de redes automotivas podem considerar uma multiplicidade de fatores, tais como funcionalidade, satisfação dos requisitos operacionais, facilidade de integração, capacidade de transmissão e manutenção. Em um veículo podem existir várias tecnologias de redes de comunicação automotivas interligadas, porém, considerando o porte do projeto o qual este estudo é sendo aplicado, essa interconexão é de no máximo duas categorias.

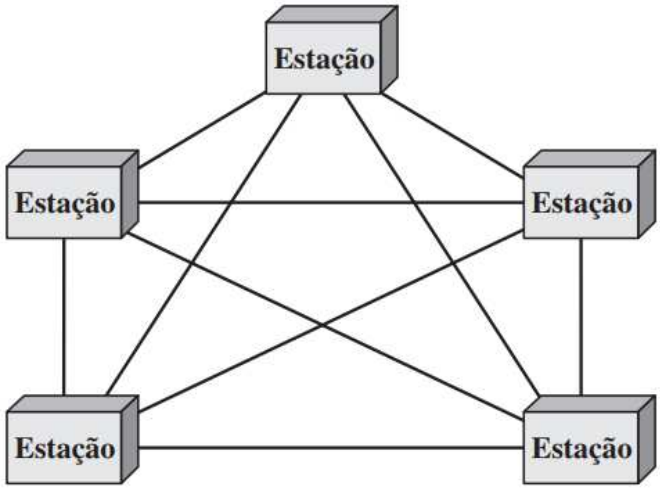
Redes podem receber denominações de classe A, classe B, classe C e classe entretenimento. Redes automotivas de Classe A consideram como aspectos relevantes funções de conforto e diagnóstico do sistema, redes automotivas de classe B, as funções relacionadas à eficiência ou dinâmica do veículo, redes automotivas de classe C, por funções de segurança crítica e redes automotivas de classe entretenimento, por funções não-funcionais e de entretenimento. De acordo com (MAHMUD; ALLES, 2005), requisitos de comunicação das aplicações como taxa de transmissão, largura de banda, atraso, *jitter* e *deadline* podem ocasionar mais subdivisões dessas categorias, mas no caso deste estudo com um kart elétrico as subdivisões não ocasionariam um grande impacto, portanto não serão detalhadas.

2.3.2 Topologias físicas de rede

De acordo com (FOROUZAN, 2012), o termo se refere à forma pela qual uma rede é organizada fisicamente. A topologia de uma rede é a representação geométrica da relação de todos os dispositivos de uma conexão. Para este estudo, serão averiguadas somente topologias básicas, dessas, existem quatro de interesse para o contexto de arquitetura automotiva: malha, estrela, anel e barramento.

Na topologia de malha, exemplificada na Figura 8, cada dispositivo possui uma conexão ponto a ponto com cada um dos demais dispositivos, garantindo a capacidade de cada conexão transportar individualmente seu próprio volume de dados, eliminando os problemas de tráfego que possam ocorrer devido a conexões compartilhadas por vários dispositivos.

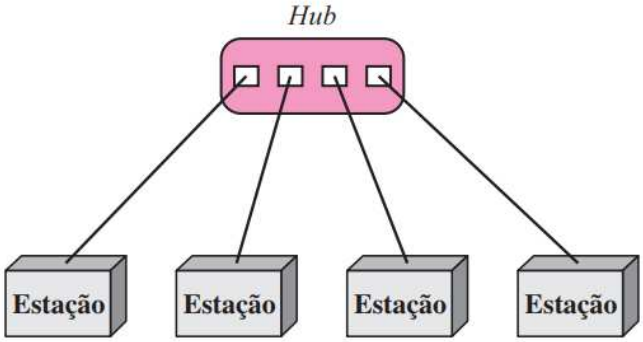
Figura 8 – Exemplo de topologia em malha



Fonte: (FOROUZAN, 2012).

Uma topologia robusta, pois caso uma conexão se tornar inutilizável, o resto do sistema ainda pode funcionar, entretanto, o custo e complexidade de implementação costuma ser alto, normalmente aplicada de forma limitada como em redes híbridas.

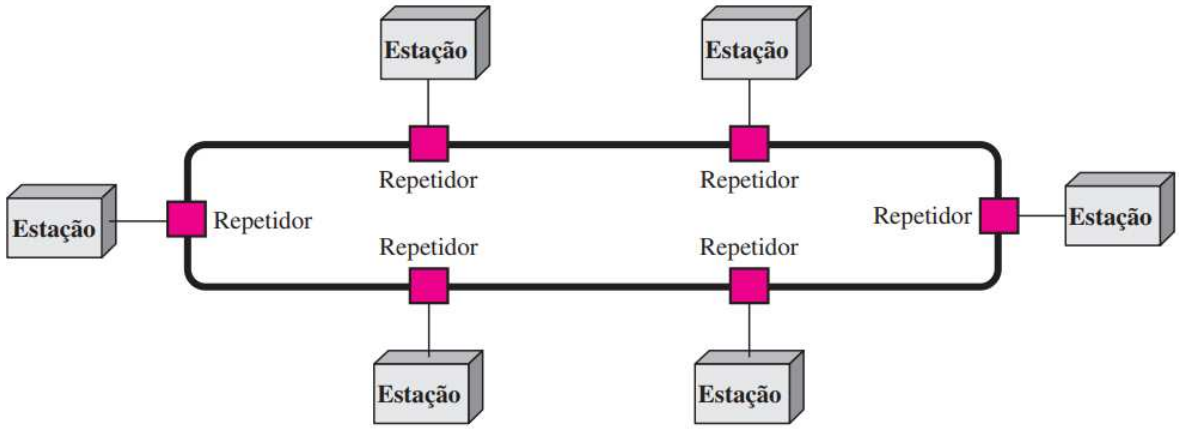
Figura 9 – Exemplo de topologia em estrela



Fonte: (FOROUZAN, 2012).

Em uma topologia estrela, ilustrada na Figura 9, cada dispositivo tem uma única conexão com o controlador central, em geral, denominado *hub*. O controlador atua como uma central de distribuição, se um dispositivo quiser enviar dados para outro dispositivo, esses dados devem passar primeiro pelo controlador, e só então, retransmitido ao outro dispositivo conectado. Essa topologia é mais barata e de mais fácil implementação se comparada a topologia de malha, entretanto, robustez menor do que a de malha, pois caso haja algum problema com o *hub*, o sistema inteiro é interrompido.

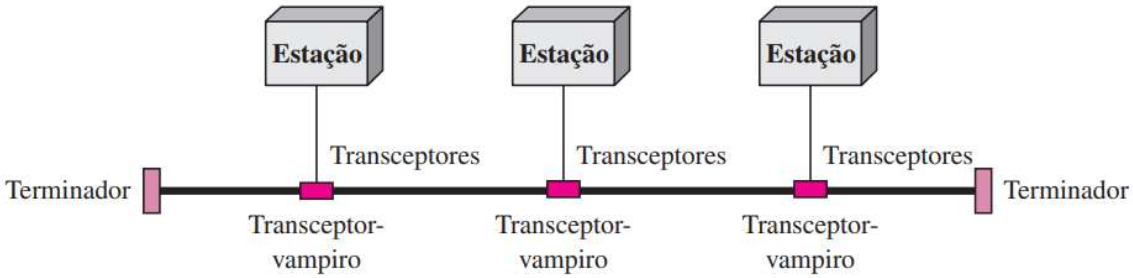
Figura 10 – Exemplo de topologia em anel



Fonte: (FOROUZAN, 2012).

Na topologia de anel, mostrada na Figura 10, cada dispositivo possui uma conexão ponto a ponto com os outros dois dispositivos. Um sinal nessa topologia é unidirecional, pois ele deve percorrer todo o anel de dispositivo para dispositivo até atingir seu destino. Ele é considerado de fácil instalação e reconfiguração, sabendo que acrescentar ou eliminar um dispositivo exige apenas a mudança de duas conexões. Os fatores limitantes dessa topologia são relacionadas ao meio de transmissão e ao tráfego, que dependem do número de dispositivos na rede, e a fragilidade, pois um problema em um único dispositivo pode tornar o sistema inteiro inoperante.

Figura 11 – Exemplo de topologia em barramento



Fonte: (FOROUZAN, 2012).

Por fim, a topologia de barramento, apresentada na Figura 11, que se difere dos exemplos anteriores por utilizar conexões multipontos, com cada conexão sendo capaz de conectar mais de dois dispositivos simultaneamente. Para sua implementação, um longo cabo atua como um caminho que interliga todos os dispositivos da rede, chamado de barramento, e as conexões que vão de um dispositivo ao cabo principal, são chamadas de transceptores. Entre as vantagens da topologia de barramento, temos a simplicidade conceitual, a facilidade de instalação, um barramento usa menos cabo que as outras topologias citadas, e uma robustez física elevada, na ideia de aplicação em ambientes ruidosos.

Entre as desvantagens, temos a dificuldade de reconfiguração e o isolamento de falhas. Normalmente, um barramento é projetado para ter a máxima eficiência na instalação, portanto, em um barramento puro, pode existir uma dificuldade ao acrescentar novos dispositivos caso não seja planejado previamente, e, caso uma falha ou a ruptura ocorra no cabo de barramento principal, toda a transmissão é interrompida. A topologia de barramento foi uma das primeiras topologias adotadas no projeto das primeiras redes locais, e mesmo com esses problemas, é a mais utilizada como base nas arquiteturas automotivas.

Essa abordagem é adotada devido à sua capacidade de facilitar a comunicação entre os diferentes módulos e sensores presentes em um veículo, simplificando a interconexão e reduzindo a necessidade de extensos cabos individuais para cada componente. A utilização de barramentos permite uma abordagem centralizada na comunicação, possibilitando o envio e recebimento de dados entre múltiplos dispositivos conectados, facilitando a interoperabilidade e o gerenciamento geral dos sistemas presentes no veículo. Essa estrutura proporciona uma configuração flexível e escalável, facilitando futuras atualizações e modificações na arquitetura.

2.3.3 Protocolos de Comunicação

Esta seção abordará os principais protocolos de comunicação utilizados em sistemas automotivos. Os protocolos desempenham a função de especificação e formatação de dados entre unidades de controle, sensores e atuadores do veículo. A análise detalhada dos protocolos, suas características e aplicabilidades são essenciais para compreender o ambiente de comunicação em sistemas automotivos, nesta seção, fundamentadas pelo estudo de (FOROUZAN, 2012).

O Controller Area Network (CAN) é um protocolo de comunicação amplamente utilizado em sistemas automotivos. Projetado especificamente para veículos automotivos, ele oferece uma forma robusta de comunicação entre os diferentes módulos eletrônicos presentes nos automóveis. O CAN é eficiente em termos de velocidade de transmissão de dados e permite a comunicação entre diversos dispositivos, como sensores, atuadores e unidades de controle, ao longo de um barramento serial bidirecional.

Esse protocolo utiliza uma arquitetura de barramento compartilhado, onde múltiplos dispositivos podem transmitir e receber mensagens no mesmo barramento. O CAN possui um mecanismo de detecção e correção de erros, tornando-o confiável para aplicações críticas em termos de segurança. Além disso, sua arquitetura descentralizada permite que diferentes dispositivos enviem mensagens simultaneamente, garantindo uma comunicação rápida e eficiente dentro do sistema automotivo.

Outro protocolo é o chamado Ethernet, bastante utilizado em redes de comunicação devido à sua confiabilidade e na capacidade de transmissão de dados. Originalmente desenvolvido para redes de computadores, foi adaptado para uma variedade de aplicações, incluindo o setor automotivo. Sua arquitetura permite a transmissão de pacotes de dados em uma variedade de velocidades, oferecendo flexibilidade na comunicação. No contexto automotivo, a Ethernet tem sido cada vez mais adotada devido à sua capacidade de suportar altas taxas de transferência de dados, proporcionando uma comunicação rápida entre os diferentes componentes do veículo.

A implementação da comunicação Ethernet em veículos automotivos oferece benefícios como, permitir o gerenciamento de informações de entretenimento e sistemas avançados de assistência ao motorista. Ela permite, também, a integração de vários dispositivos e sistemas, facilitando a comunicação entre diferentes partes do veículo. No entanto, o uso da Ethernet no contexto automotivo também apresenta desafios, como a necessidade de garantir a segurança e a confiabilidade da rede, bem como a gestão eficaz do consumo de energia para garantir o desempenho adequado dos sistemas eletrônicos do veículo.

2.4 LINGUAGEM DE MODELAGEM ARQUITETURAL

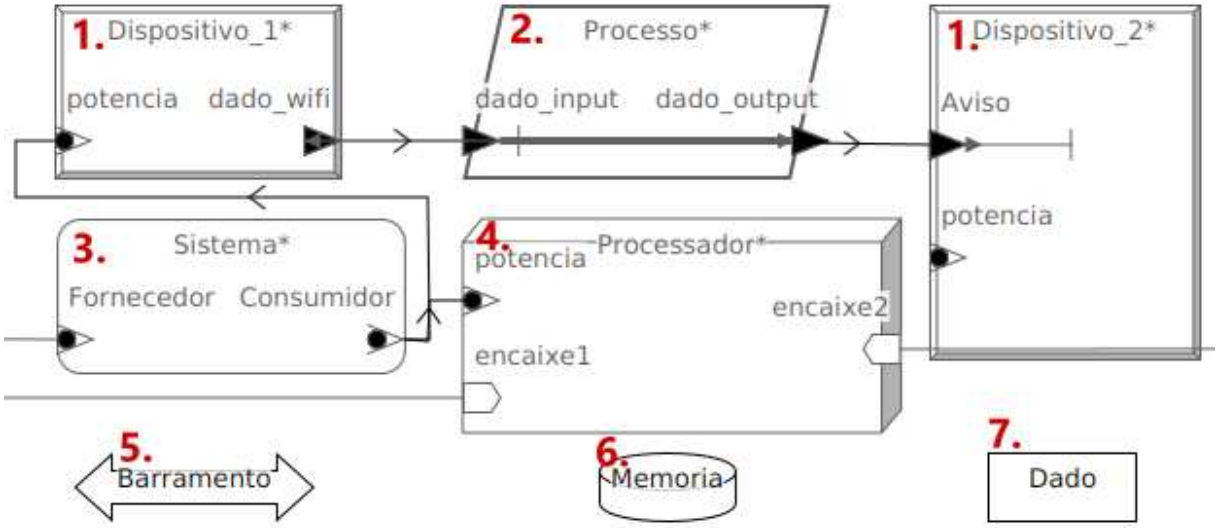
A Linguagem de Descrição de Arquitetura (AADL) é uma linguagem padronizada de modelagem arquitetural utilizada na engenharia de sistemas, especialmente em sistemas críticos de tempo-real e sistemas embarcados, como mencionado por Feiler e Gluch (2012). Desenvolvida para descrever arquiteturas de sistemas complexos, a AADL permite representar e analisar a estrutura e o comportamento de componentes, conexões e propriedades do sistema. Seu objetivo principal é fornecer uma notação para descrever a arquitetura de sistemas críticos, permitindo uma modelagem detalhada de diferentes camadas de abstração, desde componentes individuais até sistemas completos.

A AADL oferece suporte para a especificação de propriedades de sistemas críticos, incluindo a representação de requisitos temporais, espaciais e funcionais. Sua abordagem permite a análise de aspectos como a previsibilidade temporal, confiabilidade, segurança e desempenho do sistema. Além disso, a linguagem também oferece ferramentas e ambientes de suporte que possibilitam a análise estática e dinâmica da arquitetura do sistema, contribuindo para a identificação e resolução de possíveis problemas durante o desenvolvimento do projeto.

Comumente utilizada na indústria automotiva, a AADL permite descrever a arquitetura de veículos automotores, considerando a interconexão entre unidades de controle eletrônico (ECUs), redes de comunicação, sensores, atuadores e outros

componentes vitais do sistema. Sua capacidade de especificar propriedades críticas, como tempo real, confiabilidade e segurança, possibilita uma modelagem abrangente e a análise detalhada de sistemas veiculares, contribuindo para definições de modelos, sem a necessidade de implementação física.

Figura 12 – Exemplo de visualização das diferentes classes



Fonte: Desenvolvido pelo autor. (2023)

A linguagem apresenta diferentes classificações para descrever os modelos, o elemento “system”, representado como o número 3 em vermelho na Figura 12, é utilizado para definir a arquitetura de sistemas, representando o nível mais alto de abstração. Ele descreve o contexto geral do sistema e suas interações com o ambiente externo. O “system” é composto por várias instâncias de outras classes, como “process”, “device”, “processor”, “memory”, entre outros, definindo a estrutura e hierarquia do sistema.

O elemento “device”, ilustrado como o número 1 em vermelho na Figura 12, representa os dispositivos físicos do sistema, como sensores, atuadores e unidades de armazenamento. Ele descreve as características de hardware, interfaces de comunicação e comportamento desses dispositivos. O “device” pode incluir propriedades como taxa de transferência de dados, latência, consumo de energia e protocolos de comunicação.

A classe “processor” na linguagem AADL, representado pelo número vermelho 4 na Figura 12, é responsável por descrever os recursos de processamento disponíveis no sistema. Ela engloba processador de propósito geral, processador de sinais digitais, processador gráfico e outros componentes de processamento. As propriedades associadas incluem frequência, arquitetura de processador, memória cache e capacidade de processamento.

O elemento “process”, ilustrado como o número 2 em vermelho na Figura 12, representa as entidades de software executáveis ou tarefas em um sistema. Ele descreve a funcionalidade específica que será executada nos processadores disponíveis. O “process” pode incluir propriedades como prioridade de execução, consumo de memória, intervalo de tempo e recursos necessários para sua execução.

A classe “bus”, mostrado como o número 5 na Figura 12, descreve os barramentos ou canais de comunicação usados para transferência de dados entre dispositivos, processadores e outros componentes do sistema. Ela define as características físicas e lógicas da comunicação, incluindo largura de banda, protocolos suportados, latência e capacidade de conexão.

Existem também algumas outras classificações como a classe “thread”, utilizada para representar tarefas ou processos em um sistema embarcado. Essas são responsáveis pela execução de operações específicas, podendo ser associadas a processos e detalham características como prioridade, tempo de execução e restrições temporais.

A classe “data” em AADL, representa informações ou dados transmitidos entre componentes em um sistema, número 7 na Figura 12. Essa classe pode ser configurada para descrever a transferência de dados entre tarefas, processos ou dispositivos, indicando propriedades como tamanho, formato e frequência de atualização dos dados. Essas classes, quando utilizadas em conjunto e devidamente configuradas, permitem a modelagem e análise de sistemas complexos, fornecendo uma representação detalhada de seus componentes, interfaces e comportamentos.

2.4.1 Plataforma de desenvolvimento

Seguindo a padronização estipulado na indústria automotiva (RAJU; GUPTA; LOMATE, 2019), a plataforma de desenvolvimento de código aberto para AADL, é a chamada OSATE. A capacidade de modelar sistemas complexos, compostos de hardware, software e fluxo de eventos e dados, e funcionalidades de análise de modelos foram as características principais da plataforma.

O OSATE é uma ferramenta de desenvolvimento integrada (IDE) que é usada para criar, editar, verificar e analisar modelos AADL. Ela fornece recursos para modelagem de sistemas, análise de falhas, geração de código e depuração. Também possui um conjunto de extensões que permitem a integração com outras ferramentas de desenvolvimento, como ferramentas de análise de segurança e confiabilidade.

Também é possível obter diagramas estruturais diretamente das modelagens realizadas na plataforma. Esses diagramas permitem uma visualização mais compreensiva de como está estruturado o sistema, ou a arquitetura, modelada. Cada classificação apresentada pela linguagem AADL é representada por formatos diferentes nos diagramas estruturais gerados, como exemplificado na Figura 12.

2.5 PADRÕES DE MÉTRICA PARA ANÁLISES DE DESEMPENHO: ISO/IEC 15939

A engenharia de software é uma disciplina altamente complexa, e a avaliação de seus processos e produtos requer o uso de métricas objetivas e consistentes. Nesse contexto, a ISO/IEC 15939 surge como um padrão internacional que fornece diretrizes para a seleção e aplicação de métricas em projetos de engenharia de software. Este padrão estabelece princípios e processos para a medição de atributos relacionados à qualidade do software, eficiência dos processos e tomada de decisões informadas.

A ISO/IEC 15939 destaca a importância de escolher métricas relevantes, alinhadas aos objetivos do projeto e às necessidades do cliente. Ela fornece orientações sobre como definir métricas apropriadas, coletar dados de medição, analisar resultados e interpretar os dados obtidos. Ademais, a norma também enfatiza a necessidade de considerar o contexto do projeto ao selecionar métricas, reconhecendo que diferentes tipos de projetos e domínios podem exigir abordagens de medição distintas.

No contexto do desenvolvimento de uma arquitetura automotiva, a escolha e aplicação de métricas desempenham um papel crítico. As arquiteturas automotivas são altamente complexas, envolvendo uma rede de componentes eletrônicos, sistemas de controle, sensores e atuadores. A qualidade e o desempenho dessas arquiteturas afetam diretamente a segurança e a confiabilidade dos veículos.

A ISO/IEC 15939 fornece uma estrutura para orientar a seleção de métricas relevantes nesse contexto. Ao escolher métricas apropriadas, os engenheiros de software e arquitetos automotivos podem monitorar e avaliar aspectos críticos, como a eficiência da comunicação entre as ECUs, a redundância da rede de comunicação, a latência na troca de dados e a confiabilidade do sistema.

Essas métricas auxiliam na identificação precoce de problemas e na otimização da arquitetura, garantindo que ela atenda aos rigorosos requisitos de segurança e desempenho do setor automotivo. Além disso, a coleta de métricas ao longo do ciclo de vida do projeto permite a tomada de decisões informadas para melhorar continuamente a arquitetura e garantir sua conformidade com padrões regulatórios.

2.6 TRABALHOS RELACIONADOS

Para diferenciar as arquiteturas de rede de comunicação automotivas, meta inicial para a definição da arquitetura do kart estudado, pesquisou-se e analisaram-se estudos de documentos automobilísticos internacionais e monografias sobre arquiteturas de comunicação automobilísticas Staron (2021), Fürst e Bechter (2016), Mahmud e Alles (2005) e Sun e Wang (2005).

A modelagem inicial, com uma visão geral das arquiteturas, foi retirada dos roteiros apresentados por Guimarães e Saraiva (2003). Ao decorrer do desenvolvimento do projeto, outras influências geraram significantes mudanças nos

roteiros apresentados, entretanto, a metodologia inicial foi nele baseada. Um trabalho similar é o Junior (2012), um estudo e aplicação de arquitetura automotiva. A base teórica e caracterização de redes automotivas auxiliaram na formação da base teórica deste projeto e na escolha de protocolos de comunicação para os barramentos.

Trabalhos similares, como exemplificado na Figura 7, foram considerados na construção inicial das propostas de modelo, entretanto, as obras citadas expõem uma visão de nível mais alto e com projetos previamente comprovados, refletindo bem as ideias iniciais necessárias para o desenvolvimento deste trabalho. Entretanto, divergindo das implementações e organizações substanciais para o objetivo proposto, por se tratar de um projeto criado do início, ou seja, sem uma formulação prévia para basear requisitos.

O trabalho de maior influência sobre esta tese foi Staron (2021), apesar de ser escrito de forma informativa, a formulação de linhas de pensamentos, organização de modelos, métricas para análise e caracterização de ECUs foram fortemente influenciadas por este.

3 MATERIAIS E MÉTODOS

Neste capítulo será definido o escopo de trabalho, apresentados os requisitos de comunicação, escolha de topologia estrutural de rede, a ferramenta principal utilizada, os protocolos de comunicação escolhidos para implementação, escolhas para modelagem das ECUs, análises e implementações.

Figura 13 – Diagrama de atividades do trabalho

Fonte: Elaborado pelo autor. (2023)

Os procedimentos de desenvolvimento, foram realizadas seguindo o diagrama apresentado na Figura 13. O procedimento executado neste trabalho segue a linha de pensamento apresentada por Staron (2021), expandindo para testes e implementações. O autor descreve o funcionamento visando a divisão de tarefas por uma equipe completa, entretanto, neste trabalho o desenvolvimento foi realizado individualmente, portanto, o escopo foi limitado.

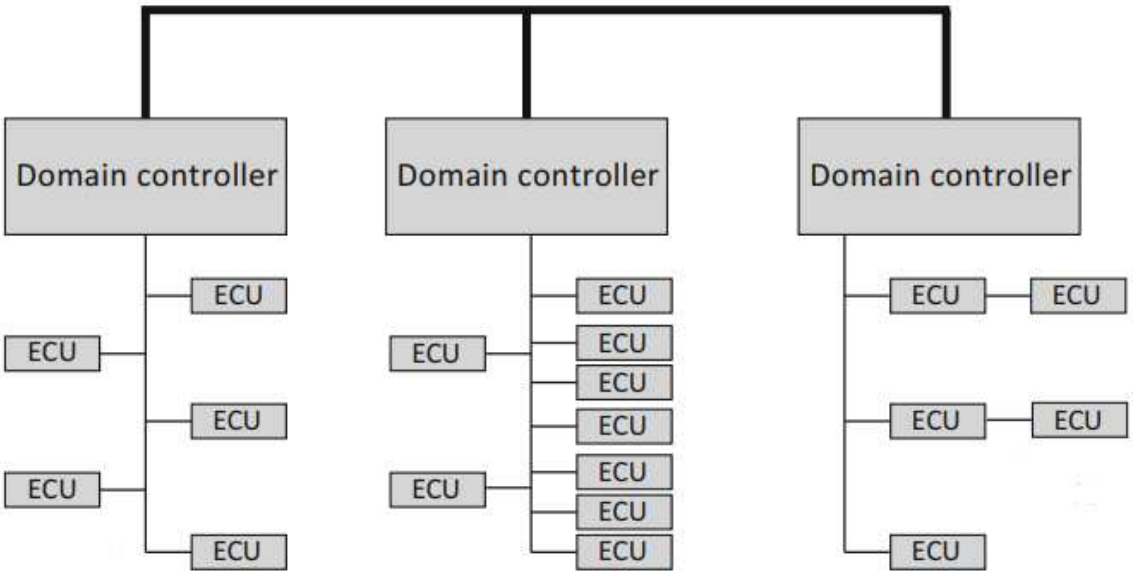
Com intuito de simplificar o uso do método escolhido como exemplo, é possível o separar em três grandes etapas, a etapa de planejamento, de modelagem e a etapa de simulação e testes. No planejamento são feitas as escolhas de visões arquiteturais e classificações gerais para organização de alto nível do projeto. Na modelagem são especificados aspectos como componentes, protocolos e estruturação do projeto. E por fim, a etapa de simulação e testes, proporcionando a implementação da parte física e de software.

3.1 ESCOPO DE TRABALHO

Idealmente a arquitetura de comunicação seria projetada em paralelo com outras ECUs fundamentais para compatibilidade e testes empíricos, mas considerações tiveram que ser aplicadas por ser um projeto individual. A implementação para este trabalho foi reduzida, portanto, para a modelagem e prototipagem de duas arquitetura de comunicação, mais um módulo de telemetria simplificado. Essas restrições permitiram a implementação de modelos menos sofisticados, mas reduzindo o escopo suficientemente para permitir o desenvolvimento do trabalho.

A definição do modelo teórico completo mais eficiente para o objetivo proposto não é possível considerando o contexto deste trabalho, mas uma implementação de uma parte da arquitetura foi apresentada. Por meio de análises obtidas através da plataforma de desenvolvimento escolhida, é possível compreender que a implementação de outras ECUs mais gerais seriam necessárias para avaliação do cenário geral.

Figura 14 – Exemplo genérico expandido



Fonte: Automotive Software Architectures, Staron Miroslaw.

Na Figura 14, também encontrada no livro (STARON, 2021), existe uma representação genérica de implementação para interconexões de controladores e ECUs, que foi considerada na implementação dos modelos gerados pela plataforma OSATE, melhor ilustrada na Seção 3.4.1. O exemplo foi utilizado somente como uma inspiração, observando que nenhuma das conexões, nem controladores foi especificada, e, apesar de aparentar ser uma visão de sistema físico, a posição específica dos componentes e ECUs também não pôde ser extraída diretamente do exemplo.

3.2 REQUISITOS DA ARQUITETURA

Os requisitos foram decididos visando o desempenho satisfatório para funções de um kart elétrico, e procurando uma alta capacidade para adições de módulos futuros. Um dos requisitos de maior relevância foi a eficiência energética, devido ao objeto de estudo selecionado. Karts elétricos são sistemas que não permitem uma conexão constante com uma fonte de energia elétrica, a menos que o objeto tenha sido especificamente projetado para essa função. Portanto, possuir uma alta eficiência energética é uma das principais características necessárias para a arquitetura de comunicação a ser modelada.

O aspecto de facilidade de implementação e compreensibilidade do projeto em sua totalidade é outro importante requisito. Como o escopo deste trabalho está limitado à implementação de somente uma parte da arquitetura automotiva, projetar um sistema de simples entendimento, reprodução e facilidade de alteração é importante. A organização utilizada no desenvolvimento deste trabalho e a implementação de protótipos seguiram uma lógica simples para auxiliar no aumento dessas características.

Seguindo a mesma linha de pensamento, outra característica de importância é a escalonabilidade do modelo, referindo-se à capacidade do sistema de tratar aumentos no número de dispositivos conectados ao sistema. A plataforma de desenvolvimento escolhida, OSATE, auxiliou nesse requisito por permitir a análise de fluxo de dados e eventos pela arquitetura modelada.

Confiabilidade e segurança são outras duas características muito importantes em projetos de modelagem de arquiteturas de comunicação. Referindo-se à capacidade do sistema de garantir que mensagens sejam entregues corretamente, com um número mínimo de erros ou perdas de dados. A modelagem foi realizada com a determinação da taxa de erros por bibliotecas padrões da OSATE.

O sistema do kart também precisa ser compatível com outros sistemas, com a ideia de intercomunicação entre o kart e dispositivos de coleta de dados, ou ferramentas de monitoramento. Para este fim, foram definidas algumas ECUs, cada uma dedicada por um determinado protocolo, e um barramento central para interface dos diferentes protocolos. Da mesma forma, foram consideradas limitações de hardware dos dispositivos conectados à rede, como o tamanho do buffer de dados ou a velocidade de processamento, certificados pelas análises obtidas pelo OSATE.

3.3 ETAPA DE PLANEJAMENTO

Discerniu-se a compatibilidade da rede de comunicação específica para os karts elétricos como um amalgamo de classificações B, C, e uma subcategoria para comunicação sem fio, as primeiras centralizadas no protocolo de comunicação CAN, e o último, por ethernet. Como parte do processo de construção desse perfil, foi necessário o levantamento e análise de microcontroladores que fazem parte das ECUs, sensores para entrada de dados e atuadores para a saída desejada, relacionando o trabalho que desenvolvem com a arquitetura definida.

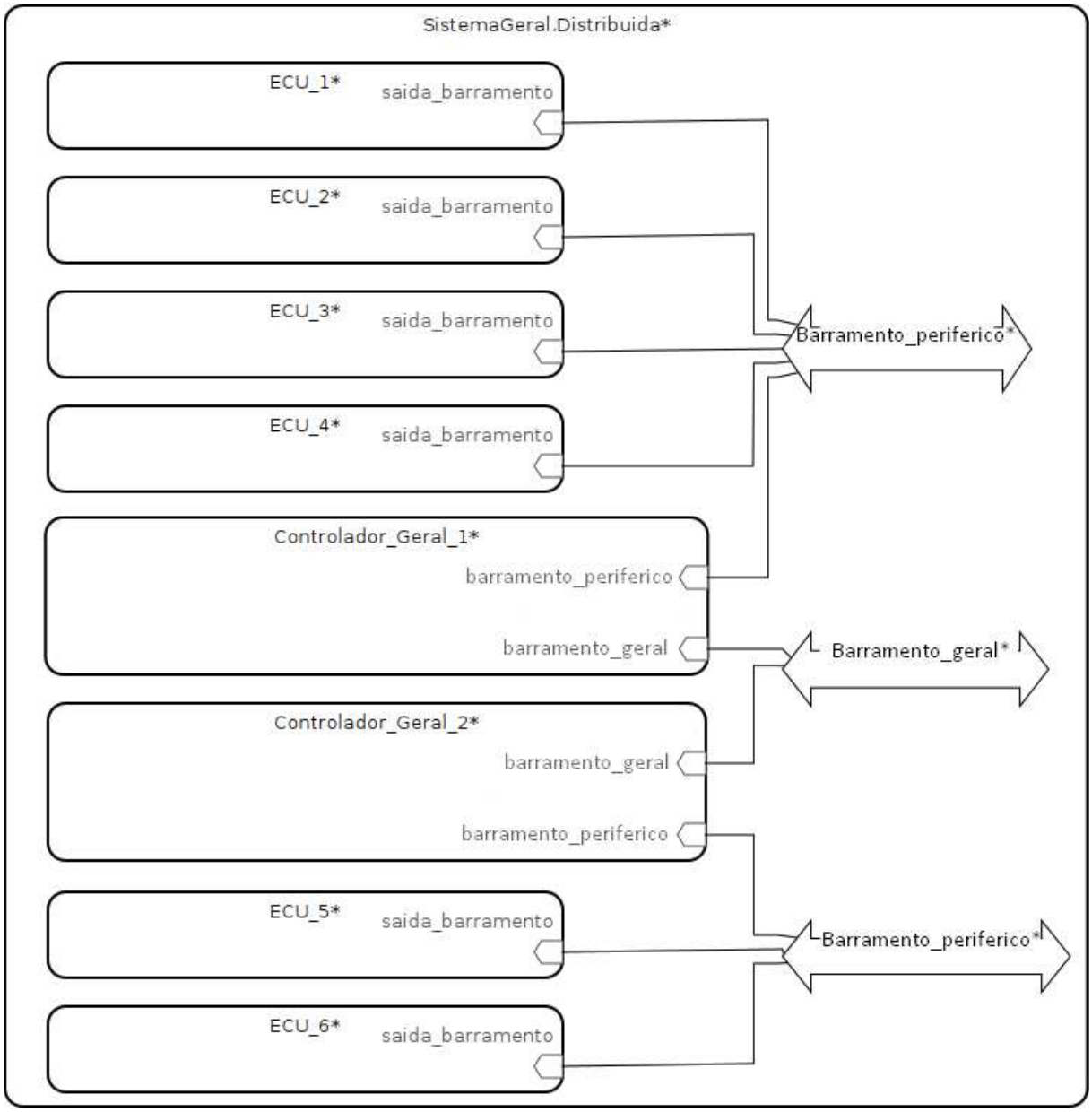
3.3.1 Topologia da rede

A topologia de rede é uma das considerações mais importantes no projeto de sistemas de comunicação, responsável pela determinação da estrutura física da rede e pela maneira como os dispositivos estão conectados. No contexto de sistemas de comunicação, existem várias topologias de rede disponíveis. Cada topologia possui

vantagens e desvantagens, e a escolha da topologia mais adequada depende da finalidade do projeto. Os de maior relevância sendo os requisitos do sistema, a quantidade de dispositivos a serem conectados e a distância física entre eles.

Por Guimarães e Saraiva (2003), existem duas abordagens de projeto de arquiteturas automotivas, a centralizada e a distribuída. Uma das características principais de diferenciação entre elas é a escalabilidade, a primeira tendo uma capacidade menor em relação à segunda. Em sistemas automotivos, a arquitetura distribuída se destaca por sua capacidade de melhorar redundância e a tolerância a falhas.

Figura 15 – Implementação de arquitetura distribuída pelo OSATE



Fonte: Desenvolvido pelo autor. (2023)

Ao distribuir as funções e recursos em diferentes módulos ou ECUs, como exemplificado na Figura 15, ela reduz a dependência de um único ponto central de falha. Proporcionando maior confiabilidade operacional, pois, caso uma ECU falhe, outras ainda podem manter o funcionamento parcial do sistema. Ela também permite a simplificação na adição e remoção de módulos, facilitando atualizações e adaptações futuras na arquitetura. Devido a esses fatores, a escolha para o kart elétrico foi a arquitetura distribuída.

3.3.2 Protocolos de comunicação

Os protocolos de comunicação escolhidos foram baseados nos padrões empregados em ambientes profissionais e industriais, nesses, a utilização do protocolo CAN é a mais comum, pois oferece vantagens significativas comparando a outros protocolos, como baixa latência, tolerância a falhas, robustez, eficiência de comunicação, flexibilidade e baixo custo. Esses benefícios tornam o CAN uma escolha preferencial em aplicações que requerem comunicação em tempo real confiável e eficiente, especialmente em setores como automotivo, automação industrial e sistemas embarcados críticos.

Portanto, a escolha do protocolo para o barramento principal da arquitetura, que realiza a comunicação entre as ECUs e controladores, será pelo CAN. Antecipando o tópico da próxima seção, a modelagem dos barramentos dentro da plataforma OSATE requer algumas propriedades para que os cálculos para geração de análises seja realizado corretamente, como pode ser observado na Figura 16.

Figura 16 – Exemplo de implementação de barramento pelo OSATE

```
-- CAN --
bus CAN --500 Kbps com mensagens de 17bytes
    properties
        Latency => 1Ms .. 1Ms;
        SEI::BandWidthCapacity => 62.5 KBytesps;
        SEI::BandWidthBudget => 62.5 KBytesps;
        Transmission_Time => [Fixed => 2ms .. 3ms;
            PerByte => 13us .. 22us;];
end CAN;
```

Fonte: Desenvolvido pelo autor. (2023)

A escolha dos materiais para este estudo foi baseada na praticidade e disponibilidade, e devido a essas características, optou-se pelo microcontrolador Arduino UNO, e Esp8266 como módulo “Wifi”. A decisão de utilizar módulos de comunicação Ethernet específicos para o Arduino UNO foi motivada pela compatibilidade direta com a plataforma escolhida.

Essa abordagem facilitou a implementação dos protótipos dos modelos, e permitiu uma comparação mais compreensível das arquiteturas propostas, focalizando o impacto da estrutura do modelo, em oposição a particularidades do hardware.

3.3.3 Critérios de análise e padrões de métrica

Os critérios de análise seguem alguns padrões de métrica básicos, demonstrados pelo livro (BOCHMANN; SUNSHINE, 1980), e procuram assimilar os aspectos de importância no desenvolvimento da arquitetura de comunicação para um kart elétrico, em oposição a um veículo automotivo generalizado, com as análises resultantes do uso da plataforma OSATE. As métricas que entram nesse critério são encontradas no Quadro 3.

Tabela 1 – Quadro com as métricas usadas

Métrica	**Descrição**
Número de componentes	A medida básica que quantifica o tamanho da arquitetura em termos de seu edifício básico (blocos ou componentes).
Número de conectores	A medida básica que quantifica a conectividade da arquitetura em termos de sua base de conectores.
Número de ECUs	A medida básica que quantifica o tamanho da arquitetura física em termos de unidades de processamento.
Número de tipos de comunicação	Quantifica a complexidade da arquitetura em termos da necessidade de implementar múltiplas comunicações mecanismos.
Número de interfaces externas	Quantifica o acoplamento entre componentes da arquitetura e sistemas externos.
Número de componentes concorrentes	A medida conta os componentes em que cálculos simultâneos fazem parte de seu comportamento.

Fonte: Capítulo 9 do livro (STARON, 2021).

No capítulo 9 do livro (STARON, 2021), são apresentadas mais métricas utilizadas para o fim de avaliação de arquiteturas automotivas. Entretanto, algumas necessitariam de informações que não foram possíveis no escopo deste trabalho. As métricas do Quadro 3 refletem quais foram possíveis de obter.

Um dos aspectos a serem considerados é o número de componentes na arquitetura. A eficiência energética de um sistema automotivo está intrinsecamente relacionada à métrica de gasto de potência. Ela reflete diretamente a eficiência operacional e o consumo de energia dos componentes eletrônicos utilizados.

Ligando outra característica as métricas padronizadas, a facilidade de implementação é dependente de diversas métricas, como o número de conexões entre os componentes, conectores, ECUs, e os tipos de comunicação presentes na arquitetura. Essas métricas representam quão complexo será a conexão física entre os

componentes, quantos fios ou conectores serão necessários, e quão complexo será a implementação dos protocolos.

Outros aspectos a serem considerados incluem o número de tipos de comunicação suportados pela arquitetura e o número de interfaces externas que permitem a interação com sistemas externos, como sistemas de diagnóstico ou comunicação com dispositivos externos. A análise desses fatores é necessária para atender aos requisitos específicos do projeto e às normas do setor.

O desenvolvimento de uma arquitetura de comunicação bem-sucedida requer a consideração dessas métricas da ISO/IEC 25000, garantindo que o sistema atenda aos padrões de qualidade e desempenho necessários para operar de forma confiável em um ambiente automotivo.

3.4 ETAPA DE MODELAGEM

Nesta seção será definida a plataforma de desenvolvimento utilizada, demonstrada a visualização obtida a partir dessa ferramenta escolhida, exemplificada como ECUs são modeladas na linguagem AADL, e por fim, modelados os cenários que serão implementados.

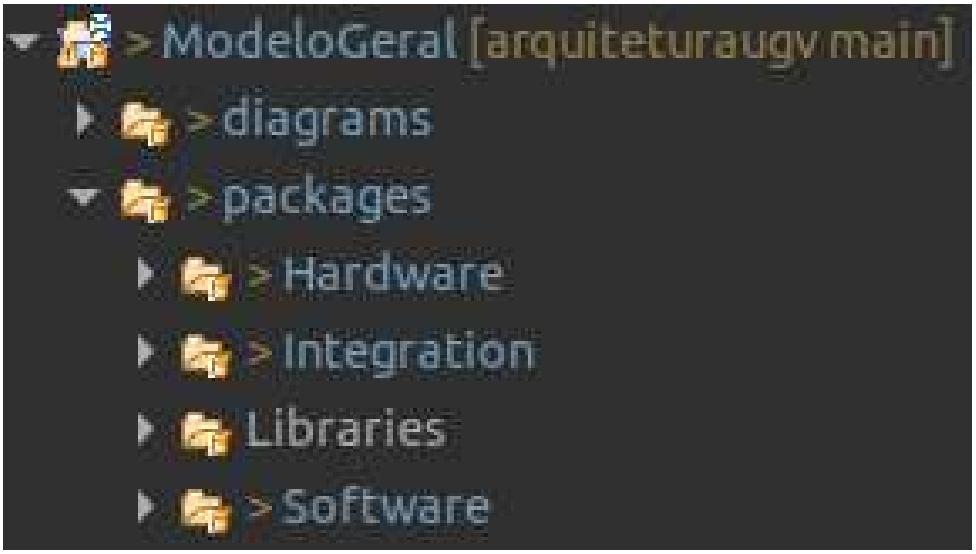
3.4.1 Plataforma de desenvolvimento

Considerando a natureza deste projeto, as análises de falhas e uso de orçamento de transmissão de dados, possibilidade de geração de códigos e facilidade de acesso a recursos informativos sobre a plataforma, neste projeto, a plataforma OSATE foi selecionada.

Dentro da plataforma OSATE, existem modelos predefinidos e exemplos de modelagens que oferecem uma estrutura organizada de arquivos e pastas. A observação desses exemplos permitiu compreender a importância de uma organização eficiente para o desenvolvimento futuro de sistemas. A estrutura bem organizada facilita não apenas o entendimento da modelagem enquanto em desenvolvimento, mas também contribui para a manutenção, modificação e expansão dos modelos, fornecendo uma base para projetos futuros.

Na organização do projeto, foi adotada uma estrutura segmentada em três grandes grupos, como demonstrado na Figura 17, refletindo as distintas áreas de atuação do sistema modelado. O primeiro grupo, denominado “Hardware”, concentra-se na definição dos componentes físicos, representados em formato de “devices” para o modelo. A seção “Software” está envolvida na modelagem dos fluxos e conexões virtuais, descrevendo a lógica do sistema. Por fim, a seção “Integração” abriga o “middleware”, responsável por promover a conexão entre o software e o hardware por

Figura 17 – Estrutura de organização do projeto



Fonte: Desenvolvido pelo autor. (2023)

meio de sistemas, viabilizando a comunicação entre essas partes do sistema de forma integrada.

Ela oferece, também, uma visão abrangente do modelo arquitetural, permitindo a extração de métricas que contribuem para a compreensão e avaliação da arquitetura de comunicação automotiva. O número de componentes pode ser obtido diretamente, fornecendo uma medida da complexidade do sistema. E a contagem de conectores fornece ideias sobre a interconexão entre os diferentes componentes, destacando pontos críticos de comunicação.

Nela também é possível realizar a análise de energia nos modelos arquiteturais, permitindo uma avaliação detalhada do consumo de energia em cada componente do sistema. Através da incorporação de propriedades específicas relacionadas ao consumo de energia nos modelos AADL, a ferramenta facilita a estimativa e verificação do “orçamento de potência” do sistema. Essas propriedades podem incluir informações sobre o consumo de energia em diferentes modos de operação, possibilitando uma análise dinâmica do orçamento de potência com base nas condições específicas de uso.

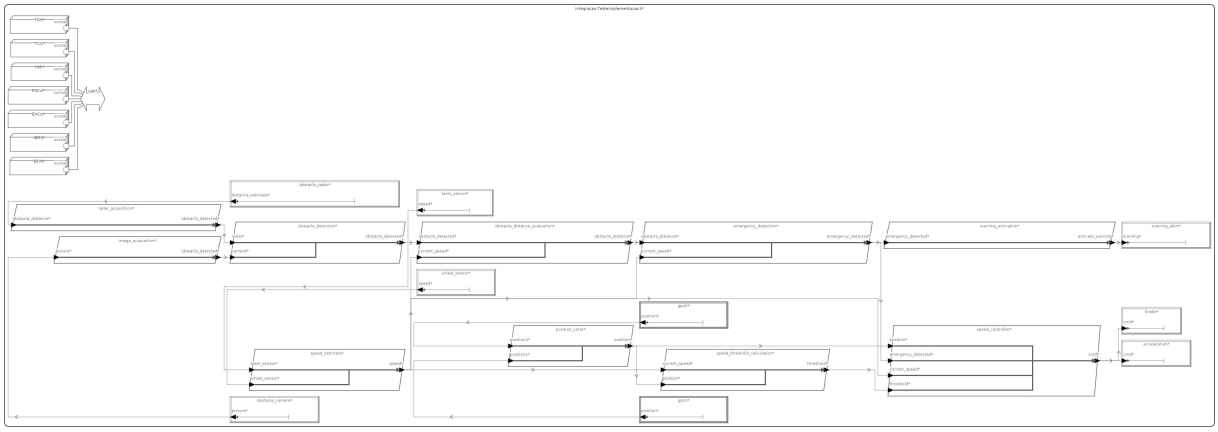
Além das análises de potência e orçamento de largura de banda, a plataforma OSATE oferece recursos avançados para realizar análises de tempo real e propagação de erro em modelos arquiteturais. A análise de tempo real permite avaliar o desempenho temporal do sistema, identificando e mitigando possíveis gargalos que poderiam comprometer a eficácia das comunicações em tempo real. Da mesma forma, a análise de propagação de erro permite modelar e entender como falhas em componentes individuais podem se propagar pelo sistema, oferecendo visões para um design mais robusto.

3.4.2 Visualização dos modelos

Sabendo que a plataforma escolhida para este trabalho é o OSATE, foram utilizadas as visualizações providas por ela, a visualização dos modelos gerados internamente pelo OSATE é feita por meio de diagramas hierárquicos e gráficos de relacionamentos, que apresentam uma visão geral da arquitetura do sistema modelado. O OSATE também permite que os usuários visualizem as propriedades e atributos associados a cada componente do modelo, como os atributos de tempo e espaço.

O OSATE também permite a visualização de modelos em diferentes níveis de abstração, entretanto, ele se limita para a visualização lógica do sistema, permitindo as outras visualizações citadas, mas, não possuindo ferramentas para auxiliar na geração de visões diferentes da lógica, como exemplificado na Figura 18. Nela, é possível fazer a visualização de um sistema completo, até a visualização de componentes individuais, permitindo que os usuários compreendam a arquitetura e o comportamento de um sistema de maneira detalhada, observação, a Figura 18 é meramente ilustrativa, melhor representada no (Anexo A).

Figura 18 – Exemplo de diagrama exportado diretamente do OSATE



Fonte: Diagrama do exemplo “bandwidth budget” da plataforma OSATE.

Outra funcionalidade importante da visualização de modelos gerados internamente pelo OSATE é a capacidade de identificar problemas e erros no modelo do sistema. Como exemplo, por meio da visualização dos modelos, é possível detectar componentes não conectados, erros de sintaxe, conflitos de atributos e outras inconsistências que podem afetar o desempenho do sistema modelado.

Em resumo, a plataforma OSATE oferece alguns recursos de visualização de modelos, permitindo que os usuários visualizem graficamente a arquitetura de um sistema modelado em AADL e identifiquem problemas e erros no modelo do sistema. Essa funcionalidade permite o desenvolvimento de sistemas críticos e complexos, possibilitando que os usuários compreendam a arquitetura e o comportamento do sistema em diferentes níveis de abstração e detectem problemas e erros antes da implementação do sistema.

3.4.3 Modelagem das ECUs

A modelagem das ECUs para análise foram gerados na plataforma OSATE, seguindo as referências da linguagem AADL, e exemplos de uso na plataforma escolhida. Aprofundando a ideia citada anteriormente de que o OSATE permite a modelagem de aspectos de hardware e software de forma conjunta, nela existem três tipos de classificações que podem ser traduzidas em partes de hardware, software, ou do próprio sistema que engloba ambas áreas.

Figura 19 – Exemplo de implementação de ECU pelo OSATE

```
processor ecu
    properties
        SEI::MIPSCapacity => 50.0 MIPS;
    annex EMV2 {**
        use types ModeloGeral::error_library;
        use behavior ModeloGeral::error_library::simple;

        error propagations
            bindings: out propagation {SoftwareFailure, HardwareFailure};
            flows
                fsoft: error source bindings {SoftwareFailure};
                fhard: error source bindings {HardwareFailure};
        end propagations;

        component error behavior
            events
                SoftwareError: error event;
                HardwareError: error event;
                Reset: recover event;
            transitions
                Operational -[SoftwareError]-> Failed;
                Operational -[HardwareError]-> Failed;
                Failed -[Reset]-> Operational;
            propagations
                normal: Operational -[]-> bindings {NoError};
                softfailed: Failed -[]-> bindings {SoftwareFailure};
                hardfailed: Failed -[]-> bindings {HardwareFailure};
        end component;

        properties
            emv2::severity => ARP4761::Major applies to SoftwareError;
            emv2::likelihood => ARP4761::Probable applies to SoftwareError;
            emv2::hazards => ([crossreference => "N/A";
                    failure => "SoftwareFault";
                    phases => ("all");
                    description => "Software failure from the platform (OS exception, etc.)";
                    comment => "Impact all components that are controlled by this software";
                ]) applies to SoftwareError;
            EMV2::OccurrenceDistribution => [ProbabilityValue => 1.35e-5; Distribution => Poisson;] applies to SoftwareError;

            emv2::severity => ARP4761::Major applies to HardwareError;
            emv2::likelihood => ARP4761::Probable applies to HardwareError;
            emv2::hazards => ([crossreference => "N/A";
                    failure => "HardwareFault";
                    phases => ("all");
                    description => "Hardware failure from the platform (OS exception, etc.)";
                    comment => "Impact all components that are controlled by this processor";
                ]) applies to HardwareError;
            EMV2::OccurrenceDistribution => [ProbabilityValue => 1.35e-5; Distribution => Poisson;] applies to HardwareError;
    **};
end ecu;
```

Fonte: Desenvolvido pelo autor. (2023)

As estruturas da AADL são separadas em “device”s, demonstrado na Figura 20, “processor”s, Figura 19, “system”s, Figura 21, e suas devidas implementações, que não serão demonstradas em figuras neste documento, mas estarão nos anexos da publicação. A subetapa de funções sendo comparadas com os processos ou projetos do OSATE, a subetapa de sistemas, componentes, e a devida implementação do AADL.

A modelagem de dispositivos, processos, processadores e sistemas na presente pesquisa foi baseada nos exemplos fornecidos pela plataforma OSATE. Esses

exemplos forneceram um ponto de partida para a compreensão e implementação da arquitetura proposta. Essa abordagem, seguindo as diretrizes fornecidas pela própria plataforma, proporcionou uma base para a modelagem e análise do sistema proposto.

A estrutura de base de todos os modelos desenvolvidos apresenta semelhanças, divergindo no classificador utilizado ao iniciar a implementação, e na seção de propriedades específicas que varia conforme o tipo do modelo. Essas propriedades devem ser ajustadas segundo as necessidades particulares de cada contexto e objetivo. A uniformidade estrutural na base dos modelos, complementada por variações específicas, oferece uma abordagem consistente na representação das diferentes configurações e funcionalidades dos sistemas veiculares desenvolvidos.

Figura 20 – Exemplo de implementação de componente pelo OSATE

```
-------------------------------------------------------------
--              Camera e Deteccao de Objetos               --
-------------------------------------------------------------
device camera
    features
        picture: out data port ModeloGeral::icd::picture;
    flows
        f0: flow source picture;
    properties
        Period => 200ms;
        compute_execution_time => 20ms .. 50ms;
    annex EMV2 {**
        use types ModeloGeral::error_library;

        error propagations
            picture: out propagation {NoValue};
            flows
                ef0: error source picture {NoValue};
        end propagations;

        properties
            emv2::severity => ARP4761::Major applies to picture;
            emv2::likelihood => ARP4761::Probable applies to picture;
            emv2::hazards => ([crossreference => "N/A";
                    failure => "Novalue";
                    phases => ("all");
                    description => "No picture from the camera";
                    comment => "Would impact the detection of obstacle if the radar is not working as well";
                ]) applies to picture.novalue;
    **};
end camera;
```

Fonte: Exemplo modificado da biblioteca padrão do OSATE.

Na Figura 19, estão definidas algumas propriedades referentes à propagação de erro, que embora não tenham sido empregadas neste trabalho, foram previamente configuradas visando facilitar desenvolvimentos futuros. A incorporação dessas configurações proporciona um suporte estrutural para análises posteriores que possam requerer esses parâmetros para avaliar a robustez e a resiliência do sistema com a implementação do Análise de Modo e Efeito de Falha (FMEA).

No âmbito da modelagem desenvolvida no ambiente OSATE, o modelo implementado baseou-se em um exemplo prévio, com adaptações para atender às demandas específicas deste estudo. O exemplo original abordava a modelagem de um sistema automotivo completo para um carro, contendo periféricos relacionados à multimídia e ao conforto dos passageiros.

Considerando o escopo restrito do presente trabalho, voltado para a implementação em um kart elétrico, foram necessárias modificações significativas. A principal adaptação envolveu a remoção dos componentes destinados aos aspectos multimídia e de conforto, dado que o kart acomoda apenas um piloto, a implementação do sistema pode ser observada na Figura 21.

Figura 21 – Implementação de do sistema pelo OSATE

```
system implementation SistemaGeral.i
    subcomponents
        Diagnostic_Interface:   system  InterfaceDiagnostico.geral;
        Power_Train:            system  PowerTrain.geral;
        Body:                   system  Corpo.geral;
        ADAS:                   system  ADAS.geral;
        Central_Gateway:        system  GatewayCentral.geral;
        Connectivity_Gateway:   system  GatewayConectividade.geral;
        Communication_Unit:     system  UnidadeComunicacao.geral;
        Sensor_Cluster:         system  ConjuntoSensorial.geral;
        Bateria:                    system  ModeloGeral::platform::FornecedorEnergia;

        BarramentoGeral:      bus ModeloGeral::platform::can;
        BarramentoPeriferico: bus ModeloGeral::platform::can;
        BarramentoInterno:    bus ModeloGeral::platform::can;

    connections
        alimentacao1: feature Bateria.Fornecedor -> Diagnostic_Interface.alimentacao;
        alimentacao2: feature Bateria.Fornecedor -> Body.alimentacao;
        alimentacao3: feature Bateria.Fornecedor -> ADAS.alimentacao;
        alimentacao4: feature Bateria.Fornecedor -> Connectivity_Gateway.alimentacao;
        alimentacao5: feature Bateria.Fornecedor -> Central_Gateway.alimentacao;
        alimentacao6: feature Bateria.Fornecedor -> Connectivity_Gateway.alimentacao;
        alimentacao7: feature Bateria.Fornecedor -> Communication_Unit.alimentacao;
        alimentacao8: feature Bateria.Fornecedor -> Sensor_Cluster.alimentacao;

        b0: bus access Connectivity_Gateway.barramento_interno      <-> BarramentoInterno;
        b1: bus access Central_Gateway.barramento_interno           <-> BarramentoInterno;

        b2: bus access Central_Gateway.barramento_geral             <-> BarramentoGeral;
        b3: bus access Connectivity_Gateway.barramento_periferico   <-> BarramentoPeriferico;

        b4: bus access Diagnostic_Interface.saida_barramento        <-> BarramentoGeral;
        b5: bus access Power_Train.saida_barramento                 <-> BarramentoGeral;
        b6: bus access Body.saida_barramento                        <-> BarramentoGeral;
        b7: bus access ADAS.saida_barramento                        <-> BarramentoGeral;

        b8: bus access Communication_Unit.saida_barramento          <-> BarramentoPeriferico;
        b9: bus access Sensor_Cluster.saida_barramento              <-> BarramentoPeriferico;

end SistemaGeral.i;
```

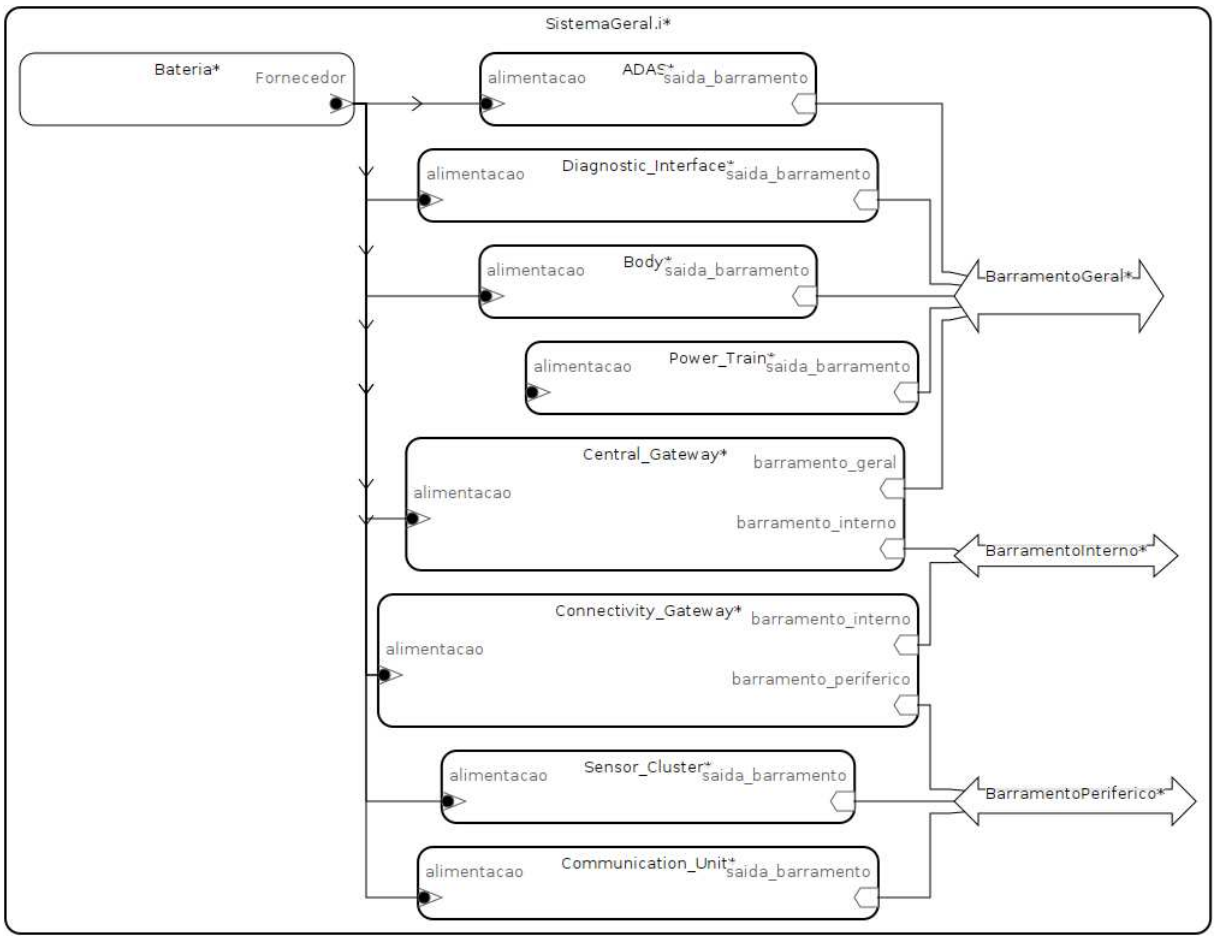
Fonte: Desenvolvido pelo autor. (2023)

Outra modificação foi a fusão de ECUs como chassi e corpo em uma única ECU, reduzindo a necessidade de uma capacidade de processamento elevada na central, responsável pelo processamento de dados provenientes de vários periféricos, diagrama mostrado na Figura 22, também representada no (Anexo B).

A função da “Central Gateway” é facilitar a comunicação e o intercâmbio de dados entre diferentes ECUs e sistemas em um veículo. Para tal, são definidas interfaces de comunicação para compatibilidade com os protocolos utilizados nos sistemas automotivos, garantindo a interoperabilidade entre as diversas ECUs e

subsistemas. Além disso, a robustez e a segurança da comunicação são elementos críticos, demandando a implementação de mecanismos eficientes para garantir a integridade dos dados e proteger contra possíveis ameaças de segurança.

Figura 22 – Modelo desenvolvido, baseado no exemplo da Figura 14



Fonte: Desenvolvido pelo autor. (2023)

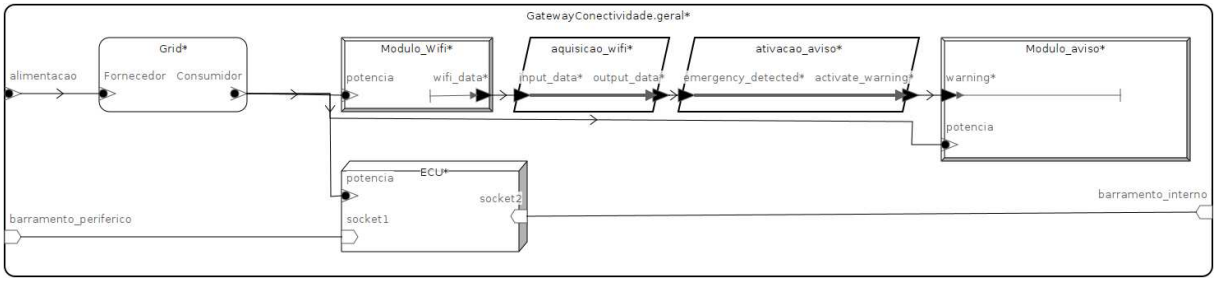
A implementação do modelo atual foi simplificada para viabilizar sua realização dentro do escopo definido. Dado que as demais ECUs ainda não foram implementadas, o modelo do “Central Gateway” foi mantido em uma forma simplificada. Isso permitiu focar na estrutura básica de comunicação e nos princípios fundamentais necessários para sua função central de gerenciamento de dados. Futuras implementações poderão expandir e aprimorar esse modelo básico para refletir mais detalhes e funcionalidades necessárias em um ambiente automotivo mais complexo.

O “Gateway de conectividade”, é responsável pela integração de diferentes tecnologias de comunicação e redes externas ao veículo, como 4G/5G, Wi-Fi, Bluetooth e outras. Assim, a ECU de “Gateway de conectividade” mostrado na Figura 23 ou (Anexo C) foi projetada considerando a capacidade de gerenciar múltiplos protocolos de comunicação, garantir a segurança da rede, oferecer suporte à conectividade de alta velocidade e manter uma interface estável com sistemas externos.

Quanto à implementação do modelo atual, foi simplificada para possibilitar sua realização física dentro do escopo deste trabalho. Dado que apenas o módulo Wi-Fi e nenhum outro módulo, ou ECU foi implementado até o momento, o modelo da ECU de “Gateway de conectividade” foi mantido em uma forma simplificada.

Embora mais complexo que o modelo da “Central Gateway” ele permanece relativamente simples devido à ausência de implementações das outras ECUs como mostrado na Figura 24 ou (Anexo D).

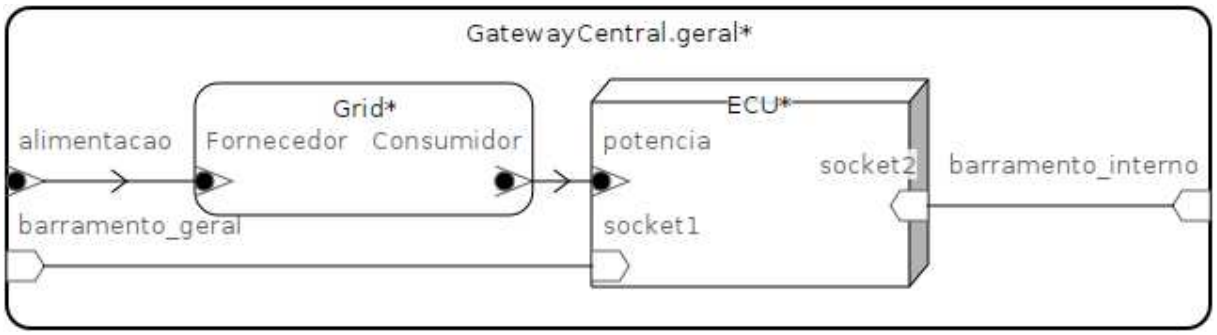
Figura 23 – Modelo do *Gateway de Conectividade*



Fonte: Desenvolvido pelo autor. (2023)

Menciona-se que, embora tenha havido consideração e até mesmo um início de aplicação do aspecto de modelagem FMEA, as análises de propagação de erro e análise de árvore de erros não puderam ser concluídas devido à simplicidade momentânea do sistema implementado. A aplicação dessas análises requer um ambiente mais complexo e detalhado, com a presença de múltiplos subsistemas e fluxo de dados ou eventos, ramificados.

Figura 24 – Modelo do *Gateway Central*



Fonte: Desenvolvido pelo autor. (2023)

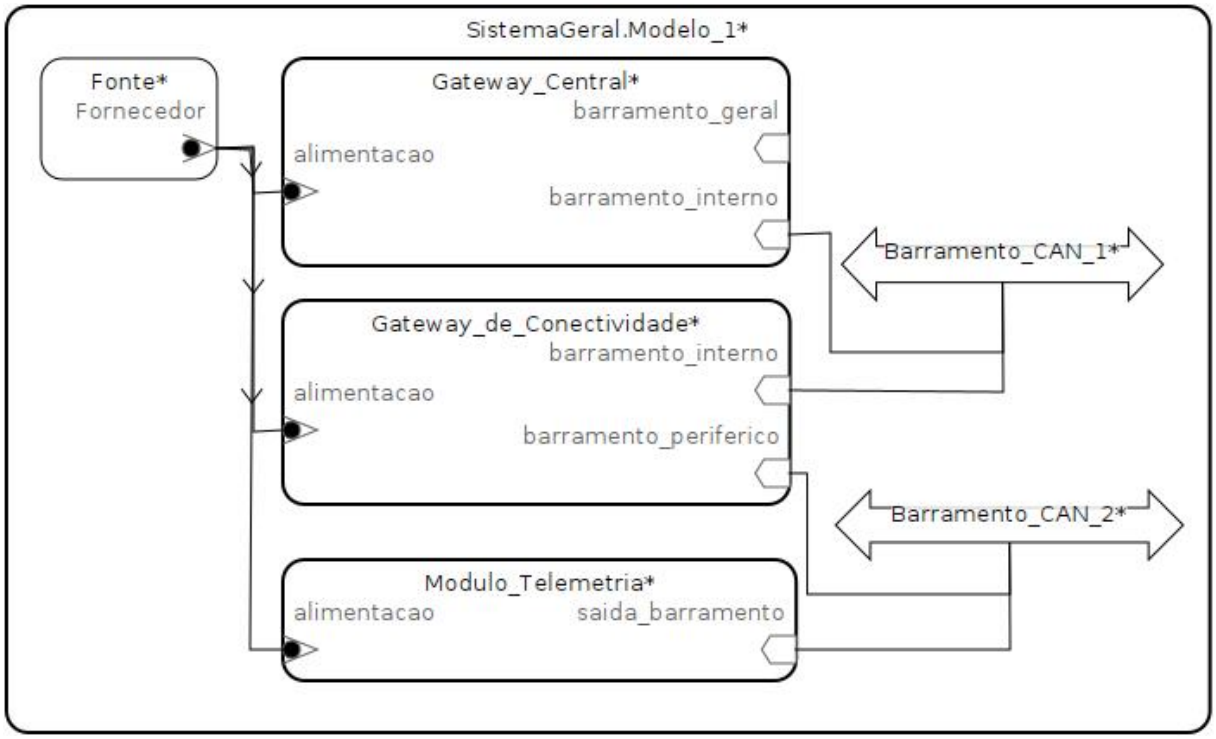
Portanto, a profundidade dessas análises ficou limitada pela natureza simplificada do modelo e pelas restrições impostas pela ausência de elementos integrados além do escopo do projeto.

3.4.4 Cenário 1

Com a modelagem das ECUs generalizadas foi possível iniciar o processo de modelagem dos dois diferentes modelos que fazem parte do objetivo principal deste

trabalho. Neste primeiro cenário são modelados, o “Gateway Central”, o “Gateway de Conectividade” e o módulo simplificado de telemetria. Os barramentos para comunicação entre os “gateways” e comunicação do “Gateway de Conectividade” com o módulo de telemetria utilizam o protocolo CAN como base.

Figura 25 – Modelo do cenário 1



Fonte: Desenvolvido pelo autor. (2023)

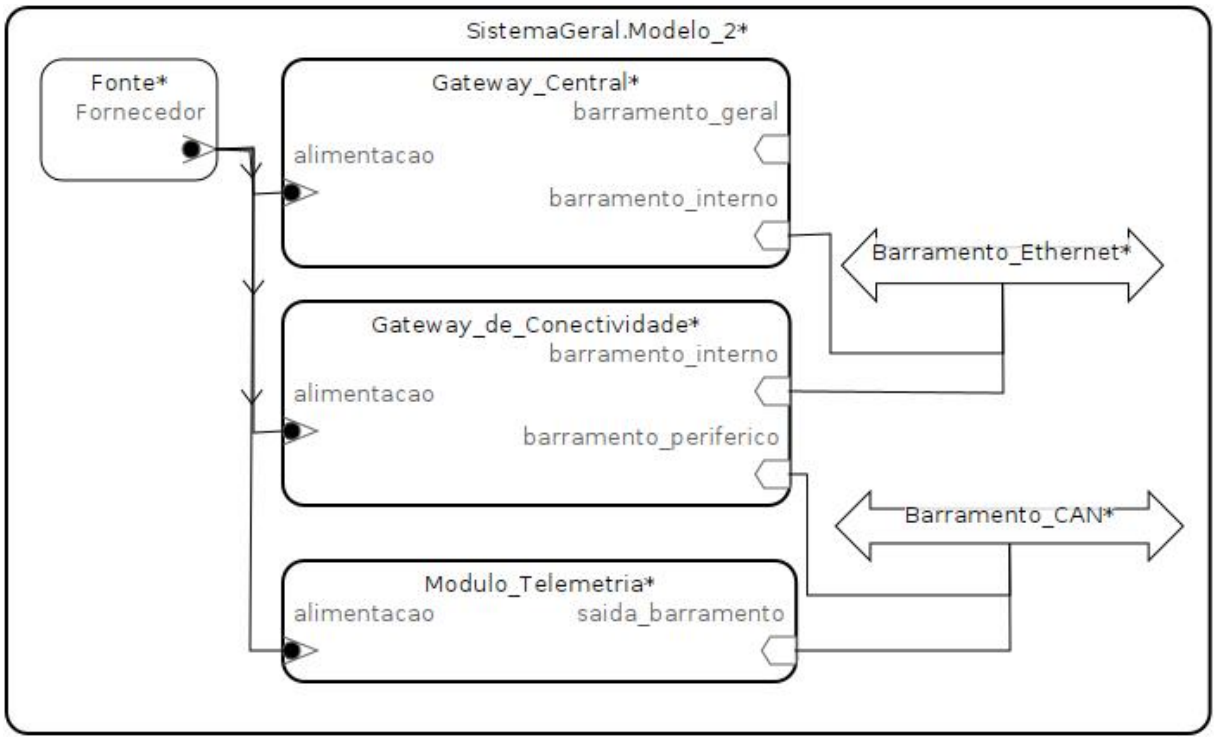
Na Figura 25 é exemplificado o diagrama de visualização estrutural obtido pela plataforma OSATE. Esse diagrama funciona essencialmente como uma visão mais detalhada da apresentada na Figura 22, na parte inferior do diagrama geral. Para implementação, foi utilizada a biblioteca “mcp_can.h”, com ela, foram criados barramentos CAN funcionando a uma frequência de 8Mhz, com uma taxa de transmissão de 500kb/s. A mensagem enviada não foi estritamente formulada, mas, para fins de comparação, o módulo de telemetria verifica a conexão “wifi” e retorna uma mensagem de 24 bytes.

O fluxo de mensagens foi implementado de modo que o “Gateway Central” espera por uma interrupção do “Gateway de Conectividade” usando o “Barramento_CAN_1*”. O “Gateway de Conectividade” envia uma mensagem de solicitação e espera uma resposta a cada 100 milissegundos para o módulo de telemetria pelo “Barramento_CAN_2*”. Esse verifica se a conexão “wifi” foi estabelecida e retorna pelo “Barramento_CAN_2*” a mensagem de 24 bytes mencionada, representando o estado da conexão. Quando essa mensagem alcança o “Gateway de Conectividade”, o dado é repassado para o “Gateway Central” via “Barramento_CAN_1*” e o fluxo é repetido.

3.4.5 Cenário 2

No segundo cenário, a comunicação entre o “Gateway Central” e o “Gateway de Conectividade” foi modelado utilizando o protocolo Ethernet. O barramento entre “Gateway de Conectividade” e módulo de telemetria foram mantidos do cenário 1, usando o protocolo CAN. Neste caso, a escolha do protocolo Ethernet, somente no barramento entre as duas ECUs implementadas, foi feita devido à propriedade de alta capacidade de transmissão de dados do protocolo Ethernet. Um aspecto que se imaginou interessante, pois nessa configuração, esse barramento seria o primeiro gargalo.

Figura 26 – Modelo do cenário 2



Fonte: Desenvolvido pelo autor. (2023)

Pela Figura 26 é possível observar que a principal mudança entre os modelos foi o barramento entre “gateways”. O que não é representado no diagrama na modelagem foram alteradas propriedades dos sistemas e dos componentes, que alteram bastante os dados para análise, entretanto, essas mudanças não são visíveis no diagrama.

Nesta implementação, também foi utilizada a biblioteca “mcp_can.h”, para o barramento CAN, entretanto, para o uso do protocolo Ethernet, foi feito o uso da biblioteca padrão “Ethernet” da IDE do Arduino. A mensagem enviada possui 24 bytes assim como no cenário anterior, para que a comparação exibisse melhor as diferenças na modelagem, e não sobre a implementação específica.

O fluxo de mensagens foi basicamente o mesmo do cenário 1. O “Gateway Central” espera por uma mensagem do “Gateway de Conectividade”, neste cenário pelo “Barramento_Ethernet*”. As mensagens entre “Gateway de Conectividade” e o

módulo de telemetria seguem o mesmo padrão do cenário 1. Quando a mensagem alcança o “Gateway de Conectividade”, o dado é repassado para o “Gateway Central” via “Barramento_Ethernet*” e o fluxo é repetido.

3.5 ETAPA DE IMPLEMENTAÇÃO

Nesta etapa, foram determinados os componentes físicos que seriam necessários para implementação dos protótipos, apresentadas as ferramentas em software com mesmo propósito e detalhada a implementação de cada cenário proposto.

A escolha dos materiais para este estudo foi baseada na praticidade e disponibilidade, e devido a essas características, optou-se pelo microcontrolador Arduino UNO, e Esp8266 como módulo “Wifi”. A decisão de utilizar módulos de comunicação Ethernet específicos para o Arduino UNO foi motivada pela compatibilidade direta com a plataforma escolhida.

3.5.1 Hardware

Nesta seção serão apresentados os componentes físicos utilizados no desenvolvimento do projeto. Serão descritos os diversos dispositivos eletrônicos, módulos, microcontroladores e outras ferramentas de hardware adotadas durante o processo de implementação da arquitetura de comunicação automotiva. Esses materiais foram essenciais para a execução do projeto, desempenhando um papel fundamental na representação prática dos modelos concebidos.

Tabela 2 – Quadro com componentes e materiais usados

Material	**Descrição**
ESP8266 12-e nodemcu kit	Componente usado como módulo de comunicação wireless.
MCP2515 TJA1050	Módulo que permite a criação de barramentos CAN, faz a “tradução” de uma entrada no protocolo SPI, para uma saída no protocolo CAN.
Arduino Uno	Controlador usado tanto na implementação da ECU Gateway Central, quanto na ECU Gateway de conectividade.
Arduino Ethernet Shield	Módulo que permite a criação de barramentos Ethernet pela arquitetura, faz a conversão de uma entrada em protocolo SPI, para uma saída no protocolo Ethernet.
Cabos jumper	Permitem a conexão entre componentes e criação física dos barramentos.
Fonte Digital Modelo PS-4000 ICEL	Utilizado como fonte de alimentação e verificar o consumo de potência para os componentes.

Fonte: Desenvolvido pelo autor. (2023)

A escolha do ESP8266 NodeMCU, representado na Figura 27, para a implementação do módulo de telemetria nesta arquitetura de comunicação apresenta vantagens significativas. Com um microcontrolador integrado de 32 bits e conectividade WiFi incorporada, o NodeMCU oferece uma solução compacta e eficiente para a transmissão de dados de telemetria. Suportando comunicação via protocolos TCP/IP e UDP, o ESP8266 proporciona uma integração com redes existentes, tornando-o propício para a transmissão remota de informações vitais do veículo.

Figura 27 – Foto do componente Esp8266

Fonte: Desenvolvido pelo autor. (2023)

O baixo consumo de energia do ESP8266 é particularmente vantajoso em aplicações automotivas, onde a eficiência energética é crucial. Além disso, a presença de GPIOs (General Purpose Input/Output) permite a conexão direta a sensores e outros dispositivos, simplificando a expansão do módulo de telemetria para atender às necessidades específicas de um sistema de monitoramento veicular.

O módulo standalone MCP2515 TJA1050, mostrado na Figura 28, emergiu como uma escolha para implementações de barramento CAN, pois é equipado com um controlador CAN de alta velocidade e um transceptor TJA1050 integrado, oferecendo uma solução compacta para sistemas de comunicação automotiva. Compatível com velocidades de transmissão de até 1 Mbps, o MCP2515 demonstra uma capacidade robusta para processar dados em tempo real, fundamental para aplicações em ambientes automotivos exigentes.

No contexto da implementação para o barramento CAN, destacam-se as características que otimizam a eficiência do MCP2515 TJA1050. Este componente apresenta o recurso bit de retorno por hardware, essa funcionalidade permite que o controlador CAN do MCP2515 TJA1050 detecte automaticamente a transmissão bem-sucedida da mensagem, simplificando a lógica de controle no nível de software e otimizando o desempenho geral do barramento.

A versatilidade do MCP2515 TJA1050 é apresentada em sua compatibilidade com uma variedade de controladores, incluindo Arduinos, módulos ESP e até mesmo o

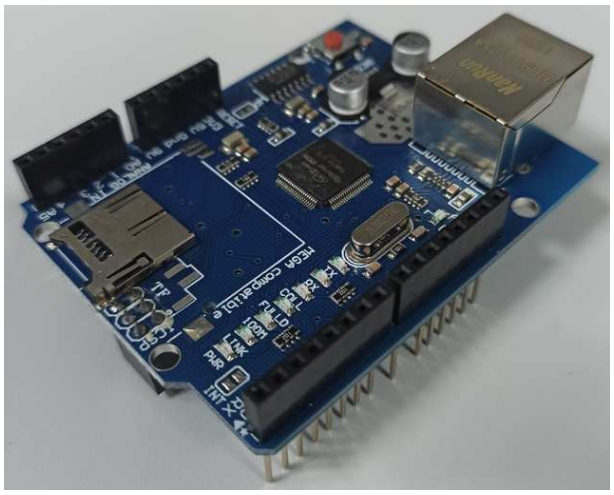
Figura 28 – Foto do módulo MCP2515

Fonte: Desenvolvido pelo autor. (2023)

Raspberry Pi. Essa interoperabilidade simplifica a integração do módulo em diferentes plataformas de desenvolvimento, proporcionando flexibilidade na implementação de sistemas de comunicação automotiva.

O Arduino Ethernet Shield, representado na Figura 29, se destaca como uma solução para integração de conectividade Ethernet em projetos de sistemas embarcados. Este módulo se acopla diretamente ao Arduino UNO, adicionando uma porta Ethernet que suporta velocidades de até 100 Mbps. Além disso, o Arduino Ethernet Shield oferece suporte a diversos protocolos de rede, tornando-o compatível com várias aplicações.

Figura 29 – Módulo de Ethernet para Arduino UNO



Fonte: Desenvolvido pelo autor. (2023)

A eficácia do Arduino Ethernet Shield provém de sua capacidade de simplificar a implementação de comunicação Ethernet em comparação com outros módulos

disponíveis. A integração direta com a plataforma Arduino e documentação detalhada contribuem para uma experiência de desenvolvimento intuitiva e eficiente, tornando este módulo uma escolha destacada para projetos que exigem conectividade Ethernet e um tempo de implementação curto.

A escolha do Arduino UNO, exemplificado na Figura 30 para a implementação de protótipos de ECUs na área de comunicações é fundamentada em sua acessibilidade e suporte abrangente, a prevalência desse microcontrolador facilita a localização de materiais de apoio e documentação. Além disso, a disponibilidade de módulos otimizados para o Arduino UNO simplifica o processo de prototipagem, permitindo uma abordagem eficiente na construção de ECUs experimentais para sistemas como o deste projeto.

Figura 30 – Imagem do Arduino UNO

Fonte: Desenvolvido pelo autor. (2023)

A consideração inicial de implementar um microcontrolador de baixo consumo, que opera a 3.3V, como o ESP32, foi ponderada para esta aplicação. Contudo, a escolha recaiu sobre o Arduino UNO, principalmente devido à sua imediata disponibilidade, compatibilidade com módulos Ethernet, e materiais de apoio acessíveis. Esses fatores foram decisivos na seleção do Arduino UNO em detrimento de outros microcontroladores, demonstrando a importância de considerações práticas durante a escolha dos componentes para a arquitetura de comunicação automotiva proposta.

A utilização do ESP8266 NodeMCU para o módulo de telemetria já ofereceu uma eficiência energética substancial, compensando parcialmente as considerações de consumo. A preferência pelo Arduino UNO foi reforçada pela necessidade de

manter uma flexibilidade prática na observação do impacto das mudanças propostas na arquitetura, sem estar estritamente atrelado ao hardware. Essa abordagem facilita a experimentação e comparação de diferentes modelos arquiteturais, concentrando-se na arquitetura em si, sem a complexidade adicional de variações no hardware.

A Fonte Digital Modelo PS-4000 da ICEL foi utilizada nos testes de potência realizados durante a implementação dos módulos neste projeto. A capacidade de ajustar variáveis como tensão e corrente com precisão permitiu a análise do consumo de energia, necessário para complementar os modelos na OSATE. O computador utilizado para as implementações foi um Notebook Samsung Book i5, com o sistema operacional Ubuntu 20.04.6 LTS.

3.5.2 Software

Para o controle dos microcontroladores, a plataforma “Visual Studio Code”, foi utilizada. Essa IDE, amplamente utilizada por desenvolvedores de software para criação, edição e depuração de códigos em uma variedade de linguagens de programação, oferece uma interface que permite personalização por meio de uma variedade de extensões. Essas extensões expandem a funcionalidade básica do Visual Studio Code, adaptando-o para atender às necessidades específicas de diferentes linguagens, “frameworks” e propósitos de desenvolvimento.

A extensão “PlatformIO” complementa o Visual Studio Code como um ambiente de desenvolvimento para sistemas embarcados e IoT (Internet das Coisas). Especificamente, a PlatformIO oferece ferramentas para desenvolvimento de firmware, facilitando a criação e o gerenciamento de projetos para microcontroladores, placas de desenvolvimento e diferentes plataformas de hardware. Essa extensão é conhecida por sua interoperabilidade com uma variedade de plataformas de hardware, sua integração com compiladores e suas capacidades de depuração, facilitando o desenvolvimento de aplicativos embarcados de maneira eficiente e flexível. Sua presença dentro do ambiente do Visual Studio Code amplia as possibilidades de desenvolvimento de sistemas embarcados para uma comunidade diversificada de desenvolvedores, no caso deste projeto, permitindo o desenvolvimento de programas para o Arduino Uno e Esp8266 em um único ambiente de trabalho.

Também foram utilizadas bibliotecas para facilitar a implementação dos protótipos. Uma delas sendo a “mcp_can”, desenvolvida por coryjfowler, sendo projetada para oferecer suporte e facilitar a implementação de funcionalidades relacionadas ao protocolo CAN em dispositivos baseados em Arduino. Essa biblioteca consiste em um conjunto de funções e métodos que permitem o acesso e controle eficiente de módulos e chips MCP2515, utilizados para traduzir mensagens em SPI para o protocolo CAN.

A biblioteca padrão do Arduino, “SPI”, oferece suporte a placas baseadas em Arduino, permitindo a interconexão com diversos dispositivos periféricos, como sensores e módulos de comunicação por meio de um barramento serial de dados e controle. Essa biblioteca apresenta um conjunto de funções que facilitam a configuração e a comunicação com dispositivos periféricos compatíveis com a interface SPI. Com essa capacidade integrada, os desenvolvedores podem interligar múltiplos dispositivos, viabilizando a troca de dados entre o Arduino e os periféricos conectados de forma eficiente e assíncrona, contribuindo significativamente para a flexibilidade e expansão das capacidades de hardware das placas Arduino.

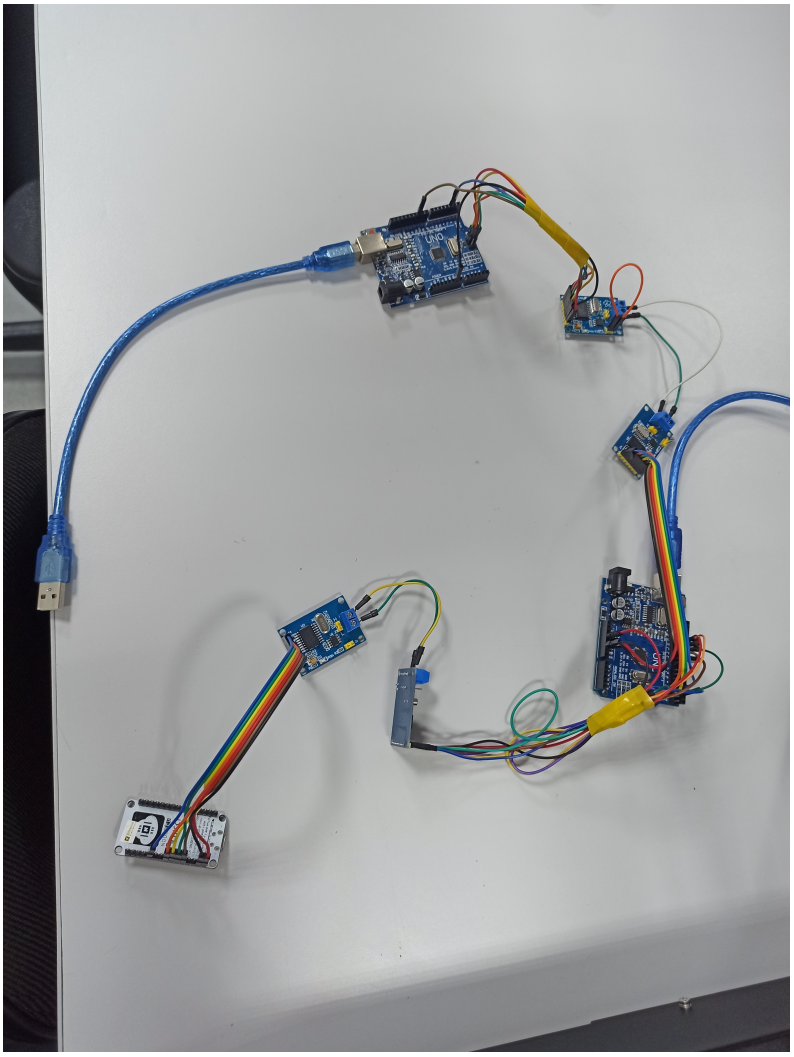
Para o funcionamento do protótipo com o barramento Ethernet, a biblioteca “Ethernet”, padrão para Arduino, foi a ferramenta que permitiu a conexão de placas Arduino a redes locais. Ela permite que dispositivos baseados em Arduino se conectem a servidores, façam requisições HTTP, enviem e recebam dados pela rede, entre outras funcionalidades de comunicação.

4 RESULTADOS E DISCUSSÕES

Neste capítulo serão apresentadas como as análises dos modelos foram realizadas, comparados os modelos propostos e as limitações das análises pelo escopo definido.

As implementações físicas não eram necessárias, mas, pela especificação de propriedades na plataforma OSATE, possibilitadas por ela, as análises de custo de banda larga e uso de energia foram aprimoradas. Nesta seção de implementações, optou-se por focar em dois modelos específicos para a arquitetura de comunicação automotiva do Kart elétrico. O primeiro modelo, demonstrado na Figura 31, adota uma abordagem comum, na qual as ECUs se comunicam exclusivamente por meio de barramentos CAN.

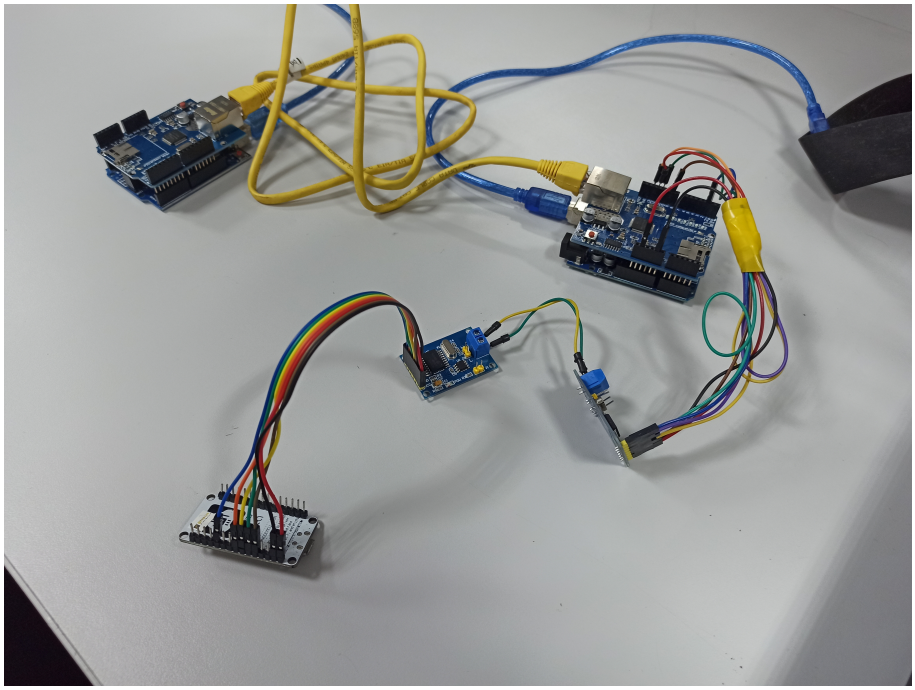
Figura 31 – Protótipo da modelagem com o protocolo CAN em todos os barramentos.



Fonte: Desenvolvido pelo autor. (2023)

O segundo modelo, representado na Figura 32 incorpora um barramento CAN conectado a módulos periféricos, enquanto as ECUs principais responsáveis pela telemetria se comunicam mediante um barramento Ethernet. Ambas as implementações foram conduzidas usando o microcontrolador Arduino UNO como base, com o módulo WiFi Esp8266 atuando como controlador. A escolha desses componentes foi motivada pela prontidão e disponibilidade ao autor. Além da disponibilidade, esses microcontroladores foram selecionados devido à compatibilidade direta dos módulos de comunicação Ethernet com o Arduino UNO.

Figura 32 – Protótipo da modelagem com o protocolo Ethernet entre “Gateways”.



Fonte: Desenvolvido pelo autor. (2023)

Essa escolha simplificou as comparações, reduzindo as variáveis ao mínimo, permitindo uma observação do impacto da arquitetura selecionada, em detrimento das características específicas do hardware. Isso se alinha à flexibilidade da plataforma de desenvolvimento OSATE, que facilita a adaptação das propriedades da plataforma sem a necessidade de alterações físicas em diversos hardwares.

4.1 ANÁLISE DOS MODELOS

As análises foram feitas usando a ferramenta interna do OSATE, que gera automaticamente planilhas com informações relevantes a análise escolhida pelo usuário. Existem muitos tipos de análise que podem ser gerados, entretanto, o foco para este trabalho foi em duas delas, a análise de orçamento, que verifica os aspectos de gasto, ou uso, dos barramentos de comunicação, retornando porcentagem em uso, possibilidade de erro, entre outras características, conforme observável na Figura 33.

Figura 33 – Quadro saído diretamente do OSATE

Bus load analysis of GatewayConectividade_CAN_Instance				
Physical Bus	Capacity (KB/s)	Budget (KB/s)	Required Budget (KB/s)	Actual (KB/s)
can	62.5	62.5	44.6	0.47500000000000003
Bus can has data overhead of 0 bytes				
Bound Virtual Bus/Connection	Capacity (KB/s)	Budget (KB/s)	Required Budget (KB/s)	Actual (KB/s)
barramento_periferico -> ECU_CAN.socket1		1.0		0.0
Modulo_Wifi.wifi_data -> aquisicao_wifi.thr.input_data		20.8		0.4
aquisicao_wifi.thr.output_data -> ativacao_aviso.thr.emergency_detected		20.8		0.0125
ativacao_aviso.thr.activate_warning -> Modulo_aviso.warning		1.0		0.0625
ECU_CAN.socket1 -> barramento_periferico		1.0		0.0

Fonte: Desenvolvido pelo autor. (2023)

O segundo tipo de análise é o de potência elétrica, representado na Figura 34, que retorna informações relevantes a alimentação e gasto de energia dos componentes, mostrando individualmente o uso de cada componente do sistema, permitindo verificar se o sistema de alimentação elétrica do projeto permite a adição de novos módulos, ou se é possível rearranjar componentes já existentes, para aprimorar o desempenho do projeto na totalidade.

Figura 34 – Quadro saída diretamente do OSATE

Computing Electrical Power for Grid			
Capacity: 40.0 W			
Supply: 0.0 mW			
Budget: 980.0 mW	420.0 mW for ECU	540.0 mW for Modulo_Wifi	20.0 mW for Modulo_aviso
Grid power budget total 980.0 mW within capacity 40.0 W			
budget total 980.0 mW within capacity 40.0 W			

Fonte: Desenvolvido pelo autor. (2023)

Seguindo padrões demonstrados por Dhakal et al. (2021), são apresentadas categorias de classificação de arquiteturas automotivas, divididas de A até D, de acordo com seus propósitos e funcionalidades. No entanto, ao considerar que essas classificações foram desenvolvidas para diferenciar categorias de veículos automotivos mais convencionais, a aplicação direta dessas categorias em um contexto específico de um kart elétrico não proporcionaria uma diferenciação significativa na análise dos modelos testados. Dessa forma, tal aspecto não foi aprofundado neste trabalho, já que as nuances de categorização propostas não seriam pertinentes ou impactantes na análise dos modelos de arquitetura de comunicação específicos para o kart elétrico.

A análise de capacidade de largura de banda no OSATE é realizada mediante a definição de propriedades que descrevem as características de comunicação dos componentes do sistema. Ao integrar informações sobre a largura de banda necessária para cada comunicação, a plataforma permite uma estimativa do limite de largura de banda global do sistema.

Essa abordagem garante que o sistema de comunicação automotiva proposto seja capaz de atender aos requisitos de largura de banda em diferentes cenários operacionais. A integração dessas propriedades no modelo AADL e a execução de

análises específicas no OSATE fornecem uma visão do desempenho de comunicação do sistema, auxiliando na otimização do limite de largura de banda durante o design e implementação.

Embora as análises de tempo real e propagação de erro sejam essenciais para uma implementação completa da FMEA, ressalta-se que o escopo deste trabalho não abrange esses aspectos. Dada a complexidade e a extensão que tais análises podem alcançar, a concentração primária está nas análises de potência e limite de largura de banda. Proporcionando uma abordagem focalizada, alinhada às metas específicas do projeto.

As análises também possibilitam a identificação do número de ECUs, fornecendo uma visão quantitativa da distribuição de controladores no sistema. Além disso, métricas como o número de tipos de comunicação e o número de interfaces externas oferecem uma compreensão detalhada da diversidade e abertura do sistema para interações externas. Essas métricas, quando derivadas das análises no OSATE, permitem avaliações quantitativas, essenciais para o desenvolvimento e aprimoramento contínuo da arquitetura de comunicação automotiva proposta.

O Quadro 3 apresenta uma síntese quantitativa da arquitetura de comunicação automotiva modelada, fornecendo percepções valiosas derivadas das análises realizadas na plataforma OSATE. Os valores expressos neste quadro refletem métricas apresentadas anteriormente neste documento, tais como o número de componentes, conectores, ECUs, tipos de comunicação, interfaces externas e componentes concorrentes. Essas métricas são indicadores-chave que delineiam a complexidade e a eficácia da arquitetura proposta. A análise detalhada desses dados contribui para uma compreensão mais aprofundada do sistema, fornecendo uma base para a avaliação e o refinamento contínuo do modelo arquitetural.

Tabela 3 – Quadro com as métricas usadas

Métrica	**Implementação CAN**	**Implementação Ethernet**
Número de componentes	11	12
Número de conectores	15	13
Número de ECUs	2	2
Número de tipos de comunicação	1	2
Número de interfaces externas	2	2
Número de componentes concorrentes	1	1

Fonte: Desenvolvido pelo autor. (2023)

A escolha entre uma implementação exclusiva com barramentos CAN e uma abordagem mista com barramentos CAN para periféricos e Ethernet para comunicação entre ECUs está enraizada em uma série de considerações críticas. A ideia inicial era utilizar unicamente as métricas retiradas do livro, entretanto, a determinação do modelo mais adequado para a implementação no contexto específico do kart elétrico se mostrou um tanto ambígua dessa maneira, pois a complexidade da implementação não gerou diferenças significativas.

Dessa forma, para fornecer uma análise mais robusta e criteriosa, foram introduzidos critérios adicionais e métricas obtidas a partir das análises realizadas na plataforma OSATE. A inclusão destas métricas complementares, tais como análise de carga de barramento e a potência utilizada por cada Unidade de Controle Eletrônico (ECU), oferece mais dados comparativos, melhorando a avaliação e tornando-a mais criteriosa na escolha do modelo ideal para implementação no contexto do kart elétrico.

A implementação exclusiva com barramentos CAN destaca-se pela sua simplicidade e robustez. Com menos complexidade de fiação, essa abordagem tende a ser eficiente em termos de custo e implementação. No entanto, essa simplicidade pode resultar em limitações, particularmente em relação à largura de banda e à velocidade de comunicação.

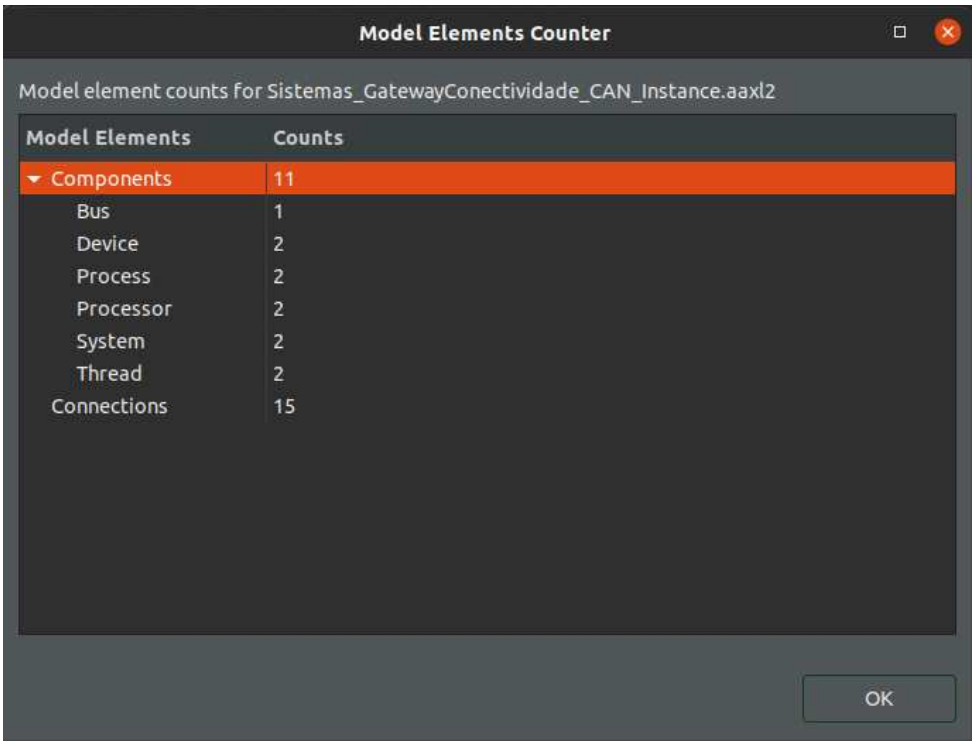
Por outro lado, a implementação mista busca tirar proveito das vantagens específicas de cada barramento. O uso do barramento CAN para periféricos mantém a eficiência e confiabilidade, enquanto a integração do barramento Ethernet para comunicação entre ECUs oferece uma solução de alta largura de banda e velocidade. Contudo, essa abordagem naturalmente adiciona uma complexidade de implementação pelo maior número de tipos de barramento, e pode acarretar custos adicionais.

A decisão entre essas abordagens deve ser orientada pelos requisitos específicos do sistema, equilibrando as necessidades de eficiência, largura de banda e complexidade. Este problema ilustra a importância da escolha de quais modelos devem seguir para implementação, considerando as métricas previamente discutidas.

4.2 COMPARAÇÃO DOS MODELOS PROPOSTOS

As métricas selecionadas para avaliação da arquitetura de comunicação automotiva foram criteriosamente escolhidas com base na bibliografia adotada como fundamento do projeto. O enfoque principal recai sobre métricas estruturais que oferecem uma visão detalhada da organização e interconexão dos componentes, como o número de ECUs, conectores e tipos de comunicação. Essas métricas, provenientes de uma perspectiva mais estrutural, fornecem uma compreensão da complexidade e eficácia intrínseca da arquitetura. Um exemplo é mostrado na Figura 35.

Figura 35 – Dados saídos diretamente do OSATE



Fonte: Desenvolvido pelo autor, via OSATE. (2023)

No entanto, reconhece-se a importância de uma visão mais ampla ao considerar métricas, as dimensões de custo, consumo e limite de largura de banda emergem como elementos importantes a serem avaliados. O consumo de recursos, um indicador-chave para implementações práticas, é uma das métricas que destaca a eficiência da arquitetura. A gestão do limite de largura de banda, por sua vez, é essencial para garantir uma comunicação eficiente e livre de congestionamentos.

Ao incorporar essas métricas no Quadro 3, busca-se proporcionar uma visão holística da arquitetura, considerando não apenas a estrutura de software, mas também os aspectos econômicos e de desempenho, aspectos importantes que não aparecem nas métricas anteriores, pelo foco na parte de arquitetura de software do exemplo.

O processo de escolha do modelo arquitetural na presente pesquisa foi orientado pela busca por valores ótimos em todas as métricas apresentadas. Ressaltando que, nas métricas discutidas, um resultado bom é representado por valores menores, com exceção da capacidade de banda larga. Sendo assim, a seleção do modelo final foi conduzida levando em consideração a minimização de todas as métricas em questão. Este critério de escolha assegura que o modelo escolhido seja aquele que, de maneira geral, apresenta a menor quantidade em todas as dimensões avaliadas, consolidando assim uma abordagem eficiente e equilibrada para a arquitetura de comunicação automotiva proposta.

Tabela 4 – Quadro com as métricas “extras”

Métrica	**Implementação CAN**	**Implementação Ethernet**
Potencia máxima fornecida	40	40
Gasto em Potência (mW)	980	1694
Capacidade de Banda Larga (KB/s)	62,5	100000
Custo de Banda Larga (KB/s)	44,6	42,6
Custo no Barramento virtual (KB/s)	0,475	1,150

Fonte: Fonte: Desenvolvido pelo autor. (2023)

As métricas selecionadas implicam em diferentes fatores, que providenciariam os pesos de cada uma das métricas em análises, esses fatores não aparecem diretamente na literatura por serem valores específicos de cada projeto. Descrevendo as mais importantes para este trabalho, o número de componentes indica a quantidade de componentes físicos presentes na implementação do modelo, demonstrando uma quantificação do custo monetário necessário para implementação. O número de conectores permite uma visão da complexidade de implementação, pois quanto maior a quantidade de conexões, mais complexo o sistema se torna. E o número de tipos de comunicação tem uma ligação direta com a complexidade de implementação do software.

Das métricas incorporadas pelo Quadro 4, todas permitem uma análise direta com valores mostrados. Com esse entendimento e resumindo a seção "Requisitos de comunicação"em especificações baseadas nas métricas, para este projeto, o custo baixo, a facilidade de implementação, eficiência energética e confiabilidade são as características de maior relevância. Neste caso, o modelo com todos os barramentos usando o protocolo CAN se mostrou a melhor escolha entre os modelos comparados, possuindo menos componentes, número de tipos de comunicação, gasto em potência, e custo total de banda larga.

4.3 LIMITAÇÕES

Considera-se que a implementação de protótipos, especialmente quando adaptada de uma estrutura para outro propósito, como um kart elétrico, apresenta certas limitações. Elas impactam diretamente a representação confiável do modelo, restringindo a profundidade e a abrangência das análises. Ademais, pela implementação de protótipos, a complexidade e os desafios encontrados em sistemas

automotivos não é capturada holisticamente, reduzindo a precisão das conclusões a serem derivadas.

Também é relevante reforçar que as mensagens comunicadas entre os módulos eram pequenas, e possuíam o uso exclusivo do barramento, pois a implementação do protótipo consiste em cada barramento com somente dois módulos. A literatura na seção de revisão teórica sugere que uma melhoria nas métricas da implementação do modelo com Ethernet poderia ser alcançada caso as mensagens transmitidas fossem maiores ou se houvesse um aumento no número de dispositivos conectados ao barramento. Entretanto, por motivos anteriormente explicitados, não existiam outros módulos para conexão nos barramentos, e as mensagens finais também não seriam muito maiores que as usadas para estes testes.

No contexto da análise de métricas, a relativa baixa quantidade para avaliação dos modelos de arquitetura, e a inexistência de outros módulos ou projeto anterior, acaba restringindo a compreensão completa do desempenho do sistema. Essa escolha limitada de métricas deixou algumas lacunas na análise de aspectos como escalabilidade, confiabilidade e eficiência, que seriam validados com a aplicação da FMEA, como exemplo. Essa limitação impactou na capacidade de realizar comparações abrangentes entre diferentes modelos de arquitetura, dificultando a identificação da solução mais otimizada para o contexto de aplicação específica, tal como um kart elétrico.

5 CONCLUSÕES

Neste trabalho, buscou-se alcançar um conjunto de objetivos centrados na compreensão e aplicação de arquiteturas de rede de comunicação automotivas em um kart elétrico. A meta principal consistiu na definição de um projeto específico de arquitetura de comunicação automotiva para essa aplicação. Esse objetivo foi cumprido mediante o desenvolvimento de modelos, implementação de protótipos, comparação de métricas e análises dos dados obtidos.

Porém, observa-se que a escolha desses modelos não implica necessariamente que são os mais efetivos em qualquer cenário, foram os modelos mais viáveis dentro do escopo selecionado. Limitações de tempo, recursos e complexidade técnica, foram fatores considerados ao determinar a viabilidade da implementação. Assim, embora os modelos escolhidos sejam válidos, reconhece-se que existem abordagens mais completas, entretanto, que extrapolam o âmbito específico deste projeto.

Observando os objetivos específicos, foi possível atingir a diferenciação das arquiteturas de comunicação automotivas, apresentando suas classificações e topologias normalmente aplicadas, para embasamento no desenvolvimento dos modelos e cenários propostos. A modelagem das arquiteturas foi possibilitada principalmente pelo estudo e uso da linguagem AADL. A capacidade de representar a estrutura e o comportamento de componentes, conexões e propriedades do sistema, junto a possibilidade de modelagem de diferentes camadas de abstração auxiliou bastante esse processo.

A implementação dos modelos foi possível a partir de protótipos. Idealmente a implementação seria completa, com o desenvolvimento das partes estruturais, eletrônicas e de software de forma definitiva. Mas, no escopo deste trabalho o objetivo foi cumprido, finalizada uma implementação que permite a comunicação entre controladores e um módulo simplificado de telemetria conforme os modelos criados. A plataforma OSATE permitiu a obtenção de métricas dos modelos descritos em AADL, que possibilitaram a comparação dos modelos de arquitetura propostos nos cenários 1 e 2. Essa comparação foi ponderada com os requisitos e características definidas na etapa de planejamento, possibilitando a avaliação numérica dos modelos.

Devido à escolha de implementar dois modelos, ambos foram projetados para expressarem mais as diferenças na arquitetura do que os componentes físicos, por exemplo. Incorporando parâmetros e componentes que refletem a realidade de uma arquitetura automotiva. Esta abordagem teve como objetivo representar uma arquitetura no contexto automotivo, mas, sendo possível no tempo estipulado para este trabalho.

A implementação centrada no módulo de comunicação e telemetria, embora tenha proporcionado uma avaliação das métricas predeterminadas, representa apenas uma parte do funcionamento geral da arquitetura automotiva. Sugerindo que seria necessário incorporar diversos módulos e ECUs para uma compreensão do desempenho da arquitetura em sua totalidade.

Ademais, apesar da identificação de um modelo implementado como a escolha mais apropriada, a abordagem simplificada adotada neste trabalho não permitiu a inclusão da FMEA. O caráter simplificado das operações e fluxo de dados na implementação dos protótipos não ofereceu uma complexidade ou ramificação suficiente para justificar a aplicação dessa análise. A FMEA, com sua ênfase na identificação e avaliação de potenciais falhas no sistema, ofereceria uma visão do aspecto de modularidade da arquitetura, destacando áreas específicas que poderiam demandar maior atenção ou aprimoramento.

Apesar das limitações, foi possível estabelecer uma base para futuros projetos de pesquisa em arquiteturas de comunicação automotiva. As análises e métricas selecionadas fornecem uma compreensão inicial, enquanto a implementação dos protótipos oferece um ponto de partida para desenvolvimentos futuros. O escopo deste trabalho oferece um ponto de partida para futuros desenvolvimentos e um material de apoio específico para auxiliar o progresso em projetos subsequentes.

Ao decorrer do desenvolvimento deste trabalho foram observados algumas dificuldades. A maior delas foi o estudo da linguagem AADL, e a implementação na plataforma OSATE, principalmente por ser uma linguagem não conhecida previamente pelo autor, mas também pela escassez de exemplos de implementação. Um tempo considerável foi usado para experimentação na plataforma até que conceitos fossem bem compreendidos. O projeto deste trabalho foi bastante comentado com o intuito de auxiliar autores futuros com o aprendizado dessa linguagem. A esperança é que projetos subsequentes poderão se concentrar mais na implementação de módulos específicos, necessitando de menos tempo procurando e experimentando com a ferramenta nesa parte de modelagem.

Considerando a natureza de protótipo das implementações deste trabalho e o escopo limitado, uma sugestão direta seria o desenvolvimento da parte estrutural das ECUs propostas, ou, aprofundamento dos modelos propostos adicionando a análise FMEA, por exemplo. O desenvolvimento de outros módulos e ECUs, como o Sistema de Gerenciamento de Bateria (BMS), no caso da Figura 22 o chamado de “Bateria”, “Power_Train”, e Sistemas Avançados de Assistência ao Motorista (ADAS), também seriam sugestões de futuros trabalhos que complementariam este trabalho.

REFERÊNCIAS

BASS, L.; CLEMENTS, P.; KAZMAN, R. **Software architecture in practice**. [S.l.]: Addison-Wesley Professional, 2003.

BOCHMANN, G.; SUNSHINE, C. Formal methods in communication protocol design. **IEEE transactions on Communications**, v. 28, n. 4, p. 624–631, 1980.

DHAKAL, S. et al. Oasd: An open approach to self-driving vehicle. In: **2021 Fourth International Conference on Connected and Autonomous Driving (MetroCAD)**. [S.l.: s.n.], 2021. p. 54–61.

FEILER, P.; GLUCH, D. **Model-Based Engineering with AADL: An Introduction to the SAE Architecture Analysis Design Language**. [s.n.], 2012. Carnegie Mellon University, Software Engineering Institute's Digital Library. Accessed: 2023-Nov-23. Disponível em: https://shorturl.at/xFG58.

FOROUZAN, B. A. **Data Communications and Networking Global Edition 5e**. [S.l.]: McGraw Hill, 2012.

FÜRST, S.; BECHTER, M. Autosar for connected and autonomous vehicles: the autosar adaptive platform. in: IEEE. **Proceedings** of the 2016 46th annual ieee/ifip international conference on dependable systems and networks workshop (dsn-w). p. 215–217–, 2016. Disponível em: https://ieeexplore.ieee.org/document/7575379. Acesso em: 4 jul. 2022.

GUIMARÃES, A. D. A.; SARAIVA, A. M. Um roteiro de implementação de uma rede can (controller area network). In: CONFERÊNCIA INTERNACIONAL DE ENGENHARIA AUTOMOTIVA-SIMEA. **Anais da Revista SABER Eletrônica – N°363 Conferência Internacional de Engenharia Automotiva-SIMEA**. p. 40–43, 2003. Disponível em: http://www.alexag.com.br/Artigos/SIMEA2003.pdf. Acesso em: 4 jul. 2022.

TALIANI JUNIOR, H. T. **Estudo dos protocolos de comunicação das arquiteturas eletroeletrônicas automotivas, com foco nas suas características e respectivas aplicações, visando o direcionamento para uso adequado e customizado em cada categoria de veículo**. Monografia (Especialização em Engenharia de Processos Industriais) Escola de Engenharia Mauá, Centro Universitário do Instituto Mauá de Tecnologia, São Caetano do Sul, São Caetano do Sul, 2012.

LEE, Y. H.; KIM, J. H.; JEON, J. W. Applying autosar network management in osek/vdx for compatibility of autosar and osek/vdx. In: **Proceedings** of the FISITA 2012 WORLD AUTOMOTIVE CONGRESS. p. 693–704. Springer, Berlin, Heidelberg, 2013. Disponível em: https://link.springer.com/chapter/10.1007/978-3-642-33829-8_65. Acesso em: 4 jul. 2022.

MAHMUD, S. M.; ALLES, S. In-vehicle network architecture for the next-generation vehicles. **SAE transactions**, v. 114, n. 7, p. 466–475, 2005.

MALATERRE, P. Automotive trends: rationale and motivations. In: **IEE Seminar on OSEK/VDX Open Systems in Automotive Networks (Ref. No. 1998/523)**). p. 1/1–1/8.

London, United Kingdom, 1998. Disponível em: https://ieeexplore.ieee.org/document/744161. Acesso em: 4 jul. 2022.

RAJU, V. M.; GUPTA, V.; LOMATE, S. Performance of open autonomous vehicle platforms: Autoware and apollo. In: **2019 IEEE 5th International Conference for Convergence in Technology (I2CT)**. [S.l.: s.n.], 2019. p. 1–5.

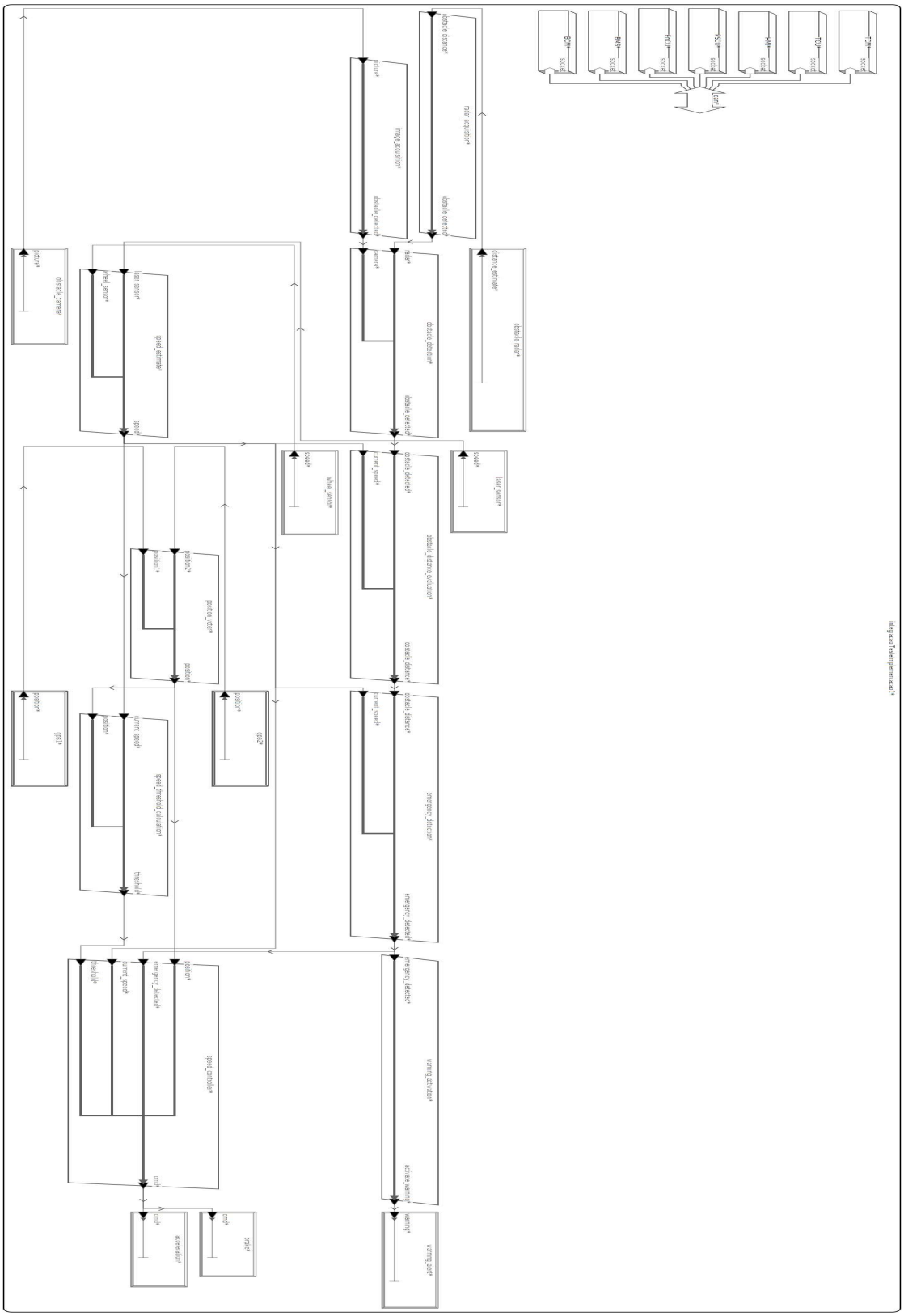
SANTOS, M. M. D. **Redes de Comunicação Automotiva: Características, Tecnologias e Aplicações**. [S.l.]: São Paulo: Érica, 2010.

SOUZA, A. G. de; CAMPOS, G. L. Rede can veicular: levantamento bibliográfico e apresentação de conceitos iniciais. **ForScience**, v. 5, n. 1, p. 122–153, 2017.

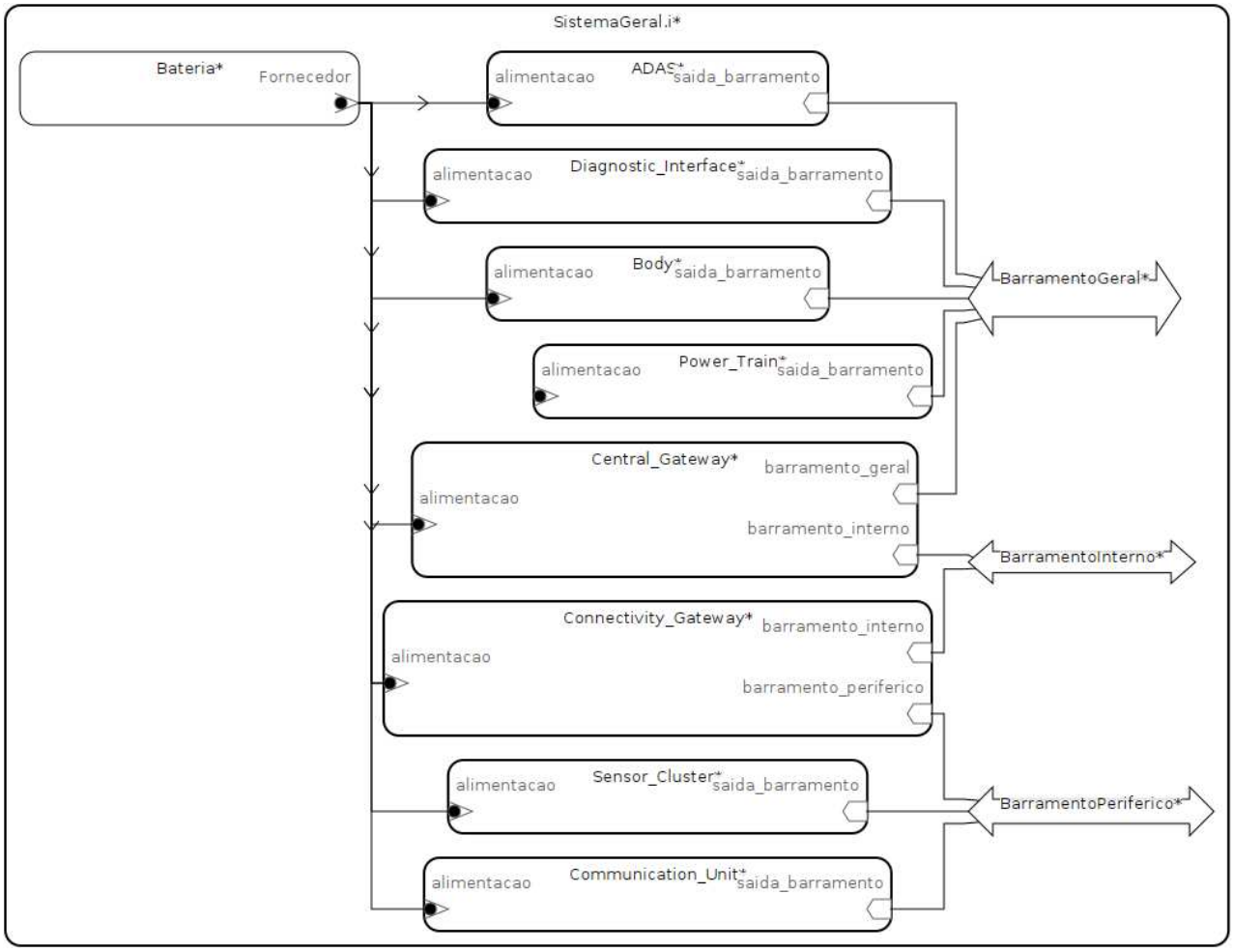
STARON, M. **Automotive software architectures**. [S.l.]: Springer, 2021.

SUN, Y.; WANG, F.-Y. A design architecture for osek/vdx-based vehicular application specific embedded operating systems. In: IEEE. **Proceedings** of the intelligent vehicles symposium. p. 882–887. 2005. Disponível em: https://ieeexplore.ieee.org/abstract/document/1505217. Acesso em: 4 jul. 2022.

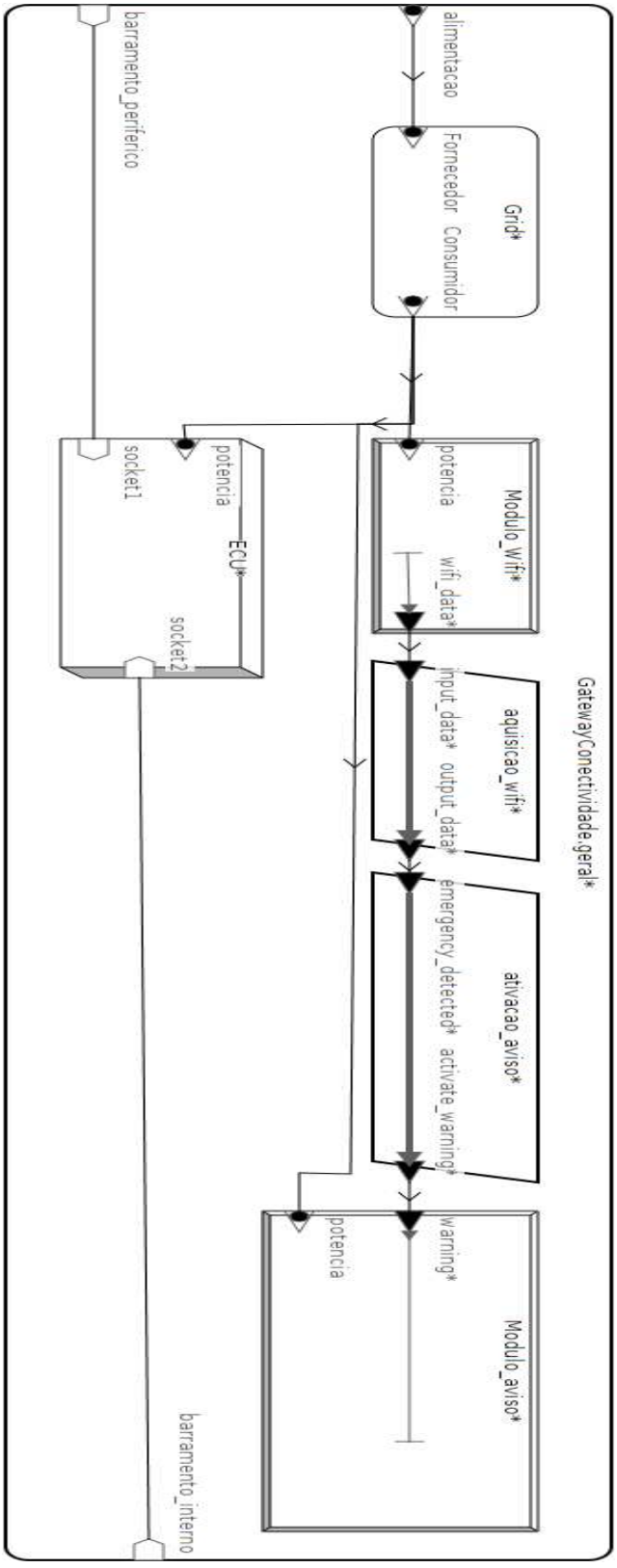
ANEXO A - FIGURA 18 AMPLIADA



ANEXO B - FIGURA 21 AMPLIADA



ANEXO C - FIGURA 23 AMPLIADA



ANEXO D - FIGURA 24 AMPLIADA