UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

ALEXANDRE DOS SANTOS ROQUE

METODOLOGIA PARA TESTE E ANÁLISE DE DEGRADAÇÃO DE DESEMPENHO EM PROTOCOLOS DE COMUNICAÇÃO INTRA-VEICULARES

Porto Alegre
2020

ALEXANDRE DOS SANTOS ROQUE

METODOLOGIA PARA TESTE E ANÁLISE DE DEGRADAÇÃO DE DESEMPENHO EM PROTOCOLOS DE COMUNICAÇÃO INTRA-VEICULARES

Tese de doutorado apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Rio Grande do Sul como parte dos requisitos para a obtenção do título de Doutor em Engenharia Elétrica.

Área de concentração: Controle e Automação

ORIENTADOR: Prof. Dr. Carlos Eduardo Pereira

Porto Alegre
2020

ALEXANDRE DOS SANTOS ROQUE

METODOLOGIA PARA TESTE E ANÁLISE DE DEGRADAÇÃO DE DESEMPENHO EM PROTOCOLOS DE COMUNICAÇÃO INTRA-VEICULARES

Esta tese foi julgada adequada para a obtenção do título de Doutor em Engenharia Elétrica e aprovada em sua forma final pelo Orientador e pela Banca Examinadora.

Orientador: _____
Prof. Dr. Carlos Eduardo Pereira, UFRGS
Doutor pela Universidade de Stuttgart, Stuttgart, Alemanha

Banca Examinadora:

Prof. Dr. Antônio Alfredo Ferreira Loureiro, UFMG
Doutor pela Universidade da Colúmbia Britânica – Vancouver, Canadá

Prof. Dr. João César Netto, UFRGS
Doutor pela Universidade Católica de Louvain – Louvain, Bélgica

Prof. Dr. Ivan Müller, UFRGS
Doutor pela Universidade Federal do Rio Grande do Sul – Porto Alegre, Brasil

Prof. Dr. Edison Pignaton De Freitas, UFRGS
Doutor pela Universidade de Halmstad, Halmstad, Suécia

Prof. Dr. Renato Ventura Bayan Henriques, UFRGS
Doutor pela Universidade Federal de Minas Gerais – Belo Horizonte, Brasil

Coordenador do PPGEE: _____
Prof. Dr. João Manoel Gomes da Silva Jr.

Porto Alegre, Janeiro de 2020.

DEDICATÓRIA

Dedico este trabalho, primeiramente, a Deus, que me deu forças para vencer todas as dificuldades. A minha mãe Josete (*in memoriam*), que dedicou parte de sua vida dando condições para que eu estudasse e infelizmente não pode estar presente neste momento da minha vida. Em especial, dedico este trabalho a todos aqueles que com dificuldades conciliam trabalho, vida pessoal e estudos, valorizando ainda mais as conquistas alcançadas.

AGRADECIMENTOS

Agradeço ao Programa de Pós-Graduação em Engenharia Elétrica, PPGEE, pela oportunidade de realização desta pesquisa. Aos colegas e amigos do PPGEE e do GCAR (Grupo de Pesquisa em Controle, Automação e Robótica), pelo auxílio durante as tarefas desenvolvidas nas disciplinas do curso, pelas contribuições e publicações realizadas em conjunto. Em especial um agradecimento ao meu colega e amigo Daniel Pohren, pelas contribuições e evoluções que em conjunto fizemos em nossas pesquisas.

Agradeço o apoio e confiança do Prof. Dr. Eng. Carlos Eduardo Pereira, pelas orientações e oportunidades dadas a mim durante a realização do doutorado e estágio no exterior. Também agradeço ao prof. Dr. Eng. Edison Pignaton Freitas pelas coorientações e contribuições que foram importantes para a realização desta pesquisa.

Devo destacar aqui também, a gratidão pelas oportunidades e orientações recebidas do Prof. Dr. Eng. Nasser Jazdi, do Instituto de Tecnologia em Automação e Engenharia de Software (IAS) da Universidade de Stuttgart, Alemanha, onde realizei meu estágio de doutorado.

Agradeço a empresa alemã Vector Informatik GmbH, pelas licenças de software disponibilizadas e pela oportunidade de estágio na Alemanha, oportunizando o contato com engenheiros experientes e o aprendizado de suas importantes ferramentas, amplamente usadas pela indústria automotiva.

Tenho que destacar e agradecer a minha família, especialmente a minha esposa Lidiane, minhas filhas Isabelli e Amanda, pelo carinho, apoio e compreensão durante os momentos difíceis, no período de estágio no exterior, sendo o suporte para a realização deste período de estudos.

Por fim, e mais importante, agradeço a Deus por todas as oportunidades e bençãos durante a caminhada de minha vida, para com saúde e serenidade superar os desafios que até aqui passei.

RESUMO

Considerar os efeitos de falhas e interferências que afetam as redes intra-veiculares desde o projeto dos seus sistemas de controle tornou-se fundamental, pois, a complexidade da eletrônica embarcada, o aumento do fluxo de informação e também as possibilidades de ataques maliciosos, tornaram o projeto destes sistemas uma tarefa cada vez mais complexa. Neste contexto, a presente tese visa explorar formas de integrar e modelar os efeitos de degradação causados por diferentes tipos de falhas que afetam os protocolos de comunicação, na interconexão das unidades de controle eletrônicas (ECUs). Dentre estas falhas, a pesquisa destaca o estudo aprofundado dos transientes elétricos rápidos – EFT, que degradam o desempenho e geram efeitos como perda de pacotes e atrasos de comunicação. Desta forma, contribui-se com uma metodologia para o tratamento de falhas em sistemas críticos de tempo real, desde as fases iniciais do projeto, utilizando a modelagem orientada a aspectos para modelar e especificar requisitos do sistema, de acordo com características transversais dos requisitos não funcionais relacionados a falhas. Para a definição dos requisitos não funcionais, esta pesquisa usa como base o framework RT-FRIDA (*Real-Time From Requirements to Design using Aspects*), o qual foi estendido para agregar com mais detalhes a modelagem de falhas. Para fins de validação da metodologia foi desenvolvido um mecanismo de diagnóstico de degradação de desempenho, o qual foi integrado a um sistema de controle de suspensão ativa. O estudo foi avaliado em diferentes cenários de carga da rede e com injeções de falhas usando dois tipos de hardwares que seguem normas de teste usadas na indústria. Os resultados evidenciaram a aplicabilidade da metodologia, com a modelagem de um mecanismo de diagnóstico que detectou e registrou os distúrbios de desempenho nos cenários estudados. As análises enfatizam a degradação de desempenho acentuada registrada com as injeções EFT de maior amplitude de tensão e menor tempo de rajada, com carga de ocupação da rede acima de 30%. Os experimentos avaliaram o desempenho dos atuais protocolos de comunicação, com melhores resultados obtidos em FlexRay e CAN-FD, o que confirma a evolução dos protocolos para atender as recentes demandas de desempenho da indústria automotiva.

Palavras-chave: Protocolos de comunicação intra-veiculares, Transientes elétricos rápidos, Modelagem baseada em testes, Modelagem Orientada a Apectos.

ABSTRACT

Embedded computing applications are increasingly demanding performance and reliability because these factors are critical to the safety of real-time systems. Reliability aspects in design phases is a fundamental point of many researches because with the increase of embedded electronics, network data transmission and also possibilities of attacks on them, make the design of these systems an increasingly complex task. The present thesis aims to explore and correlate different fault types that degrade vehicular communication protocols performance used to interconnect embedded control units (ECUs). Among these faults, the electrical fast transients - EFT are highlighted, since they generate effects such as packet loss and communication delays. Thus, a methodology based on aspect-oriented modeling concepts, in real-time critical systems is proposed, to model and specify system requirements according to cross-cutting concerns of non-functional requirements related to faults. For non-functional requirements specification, this work is based on RT-FRIDA (Real-Time From Requirements to Design using Aspects) framework, which was be extended for fault modeling. Thus, the novel methodology allows fault modeling following the aspect-oriented principles from the early design phases. For the methodology validation purposes, a performance degradation diagnostic mechanism was developed, which was integrated into an active suspension control system. The study was evaluated in different network busload scenarios and with fault injections using two hardware types, certified by standards used in the automotive industry. The results present that the developed mechanism detected performance disturbances, recording occurrence data in the studied scenarios. The analyzes emphasize the best performance degradation observed with EFT injection of higher voltage amplitude, shorter burst time, and busload above 30%. The experiments evaluated the performance of current communication protocols, with better results obtained in FlexRay and CAN-FD, which confirms the protocol's evolution to meet the recent performance demands of the automotive industry.

Keywords: In-Vehicle communication protocols, Electrical fast transients, Modeling-based testing, Aspect-oriented modeling.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS

ABS	Anti-lock Braking System
ACOD	Aspect Crosscutting Overview Diagram
AOD	Aspect-Oriented Development
AOM	Aspect-Oriented Modeling
AUTOSAR	AUTomotive Open System ARchitecture
BER	Bit-Error Rate
BBW	Brake-by-wire
BRS	Bit Rate Switch
CAN	Controller Area Network
CAN-FD	CAN with Flexible Data Rate
CAPL	Communication Access Programming Language
CLP	Controlador Lógico Programável
CPS	Cyber Physical Systems
CRC	Cyclic Redundancy Check
CSMA/AMP	Carrier Sense Multiple Access with Arbitration on Message Priority
CSMA/BA	Carrier Sense Multiple Access with Bitwise Arbitration
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
DERTS	Distributed Embedded Real-Time System
DBW	Drive-by-wire
DCS	Distributed Control Systems
DECS	Distributed Embedded Control Systems
DERTS	Distributed Embedded Real-Time Systems
ECU	Electronic Control Unit
EC	Elementary Cycle
EDL	Extended Data Lenght
EFT	Electrical Fast Transients

EMI	Electromagnetic Interference
EMC	Electromagnetic Compatibility
EOF	End of Frame
ESI	Error State Indicator
FBW	Fly-by-wire
FDMA	Frequency Division Multiple Access
FDI	Fault Detection Isolation
FMEA	Failures Modes and Effects Analysis
FTCS	Fault Tolerant Control Systems
FTCAN	Fault Tolerant CAN
FTT-CAN	Flexible Time-Triggered CAN
RT-FRIDA	Real-Time From Requirements using Aspects
GCAR	Grupo de Controle, Automação e Robótica
GSN	Goal Structuring Notation
IEC	International Electrotechnical Commission
ISO	International Organization for Standardization
LIN	Local Interconnect Network
MAC	Media Access Control
MDE	Model-Driven Engineering
MOST	Media Oriented Systems Transport
NFR	Non-Functional Requirements
NRZ	Non-Return-To-Zero
OSI	Open Systems Interconnection
PPGEE	Programa de Pós-Graduação em Engenharia Elétrica
PLR	Packet-Loss Rate
QoS	Quality of Service
RTA	Real-Time analysis
SAE	Society of Automotive Engineers
SBW	Steer-By-wire
SIG	Softgoal Interdependence Graph
TDMA	Time Division Multiple Access
TT-CAN	Time-Triggered CAN
UML	Unified Modeling Language
XBW	X-By-wire

SUMÁRIO

1 INTRODUÇÃO

Sistemas computacionais embarcados estão presentes em diferentes atividades humanas necessitando de evolução constante para se adaptar a diferentes situações de criticidade. De acordo com Wolf (2008) o projeto de sistemas embarcados é extremamente complexo, por envolver questões como portabilidade, compromisso entre consumo de energia e perda de desempenho, baixa disponibilidade de memória, necessidade de segurança, confiabilidade e escalabilidade. Além destas questões destacadas, outro aspecto importante é a adaptabilidade de um sistema embarcado perante uma falha, característica que provê um requisito cada vez mais explorado, a capacidade de sobrevivência ou manutenção segura do funcionamento do sistema. A adaptabilidade é uma característica chave para os dispositivos embarcados, permitindo a execução de aplicações com baixo consumo de energia e a operação mesmo durante a ocorrência de falhas, devido a fatores como o alto poder de processamento disponível, ao baixo custo de dispositivos de memórias e a possibilidade de programação destes (PEREIRA; CARRO, 2007).

Neste contexto, o projeto de sistemas embarcados influencia o desenvolvimento de pesquisas ligadas à confiabilidade em protocolos de comunicação industriais, onde todos estes fatores destacados também são relevantes e carecem de estudo e evolução constante. Com o aumento da complexidade dos sistemas de controle que operam e dependem de redes de comunicação, o número de falhas nos processos de comunicação entre estes componentes tende a crescer significativamente. Desta forma, muitas falhas têm impacto não somente no desempenho, mas também na confiabilidade do sistema e na segurança dos usuários.

Um exemplo desta complexidade crescente inerente a interconexão de unidades de controle são as redes intra-veiculares, pois, para efetivar o processo de controle os sistemas embarcados compõem complexos sistemas ciber físicos (*Cyber Physical Systems* – CPS). Estes sistemas trabalham em rede com protocolos de comunicação específicos, como por exemplo, os protocolos CAN (*Controller Area Network*), TTCAN (*Time-triggered* CAN), FlexRay e LIN (*Local Interconnect Network*) (TUOHY *et al.*, 2015). Redes de CPS ainda devem atender requisitos de tempo real e prover garantias de qualidade de serviço (QoS – *quality of service*) que são críticas para muitas aplicações. Desta

forma, sistemas de controle embarcados e distribuídos devem prever comunicação com suporte à detecção, monitoramento e prevenção a falhas críticas, de forma a agregar maior confiabilidade ao processo de controle.

Devido ao grande número de componentes interconectados em redes de comunicação embarcadas, são adotadas diferentes estratégias de diagnóstico de falhas e de redundância de hardware, que normalmente são reativas e demandam um número excessivo de retransmissões de mensagens de controle. Assim, modelar falhas nas fases de projeto pode contribuir para o desenvolvimento de mecanismos de diagnóstico de falhas e melhorar o desempenho de aplicações de tempo real. Falhas afetam o processo de comunicação gerando efeitos transversais, atingindo de forma global o processo de controle, que por sua vez pode ser responsável por tarefas críticas. Além disso, mesmo se falhas críticas não ocorrerem estas podem degradar silenciosamente o desempenho e a vida útil de componentes, por exemplo, devido a transientes elétricos e interferências eletromagnéticas, (TUOHY *et al.*, 2015), (VYATKIN, 2013).

Modelar e especificar estes tipos de falhas como requisitos não funcionais se torna importante para contribuir com o aumento da confiabilidade dos sistemas de controle em redes intra-veiculares, e a orientação a aspectos se enquadra neste contexto devido às características de modelagem de requisitos que afetam de forma transversal o sistema (WEHRMEISTER; PEREIRA; RAMMIG, 2013). A abordagem central desta tese é a combinação de técnicas de modelagem orientada a aspectos com a especificação de requisitos não funcionais, permitindo assim o desenvolvimento de mecanismos de diagnóstico e detecção de falhas em protocolos de comunicação usados em redes intra-veiculares. A pesquisa estuda e correlaciona tipos de falhas que degradam o desempenho dos protocolos de comunicação, como transientes elétricos e interferências eletromagnéticas, seus efeitos na perda de pacotes e geração de atrasos de comunicação. Para tal o framework RT-FRIDA (*Real Time From Requirements using Aspects*) (FREITAS, 2007), é usado e estendido, compondo a base para o desenvolvimento de uma nova metodologia que contemple a modelagem de falhas, de acordo com os princípios da modelagem orientada a aspectos – AOM (*Aspect-oriented modeling*). Essa combinação contribui para o diagnóstico proativo de falhas, o que por consequência também contribui para a segurança funcional dos processos que dependem das redes intra-veiculares, possibilitando também a redução da manutenção das unidades de controle eletrônicas.

1.1 Motivação e Definição do Problema

Os fatores motivadores desta pesquisa estão relacionados ao crescente aumento de falhas que afetam as redes intra-veiculares, tendo como suporte a adoção de padrões e normas que guiam os procedimentos de teste. Transientes elétricos rápidos (EFT), interferências eletromagnéticas (EMI) e também ataques maliciosos, tendem a afetar e degra-

dar cada vez mais os protocolos de comunicação usados nestas redes. Tais falhas, em seu contexto físico, geram subsequentemente um impacto negativo no desempenho e na confiabilidade dos sistemas de controle críticos e de tempo real, os quais possuem restrições temporais rígidas.

O problema de pesquisa é centrado na carência de processos e métodos de modelagem que abordem diferentes tipos de falhas em fases iniciais de projeto dos sistemas embarcados, considerando o contexto da indústria automotiva. Tais abordagens se limitam a testes de susceptibilidade a falhas e carecem de um link permanente entre fases de teste, modelagem e especificação de requisitos de projeto. Desta forma, o presente estudo foca em falhas que afetam e degradam o desempenho dos principais protocolos usados em redes intra-veiculares (CAN, FlexRay e CAN-FD).

1.2 Objetivos e Contribuições

Como hipótese a problemática da pesquisa têm-se os seguintes objetivos: estudo e desenvolvimento de métodos de teste e injeção de falhas em redes intra-veiculares; análise aprofundada de susceptibilidade às falhas; e o desenvolvimento de uma metodologia que possibilite, de forma sistemática, estabelecer um link permanente entre fases de teste, fases de modelagem e especificação de requisitos, no contexto dos sistemas embarcados críticos e de tempo real. Assim, o tratamento das falhas é realizado considerando os preceitos da modelagem orientada a aspectos, tendo como base a extensão do framework RT-FRIDA, o qual fornece suporte à especificação de requisitos não funcionais relacionados às falhas.

As principais contribuições da pesquisa são:

- Análises de susceptibilidade a falhas transientes nos principais protocolos de comunicação.
- Um método de teste e injeção de falhas seguindo padrões usados pela indústria automotiva.
- Modelagem de falhas apoiada por preceitos de orientação a aspectos, com a extensão do framework RT-FRIDA.
- Um mecanismo de diagnóstico, que em tempo de execução, gera dados relevantes para a análise de degradação de desempenho em redes intra-veiculares.

1.3 Organização

A presente Tese é organizada da seguinte forma: no Capítulo 2 é apresentada a base teórica que serve para a compreensão do contexto da pesquisa, com a especificação dos

principais protocolos de comunicação veiculares, aspectos de confiabilidade e susceptibilidade a falhas, bem como a relação com o projeto de sistemas de controle distribuídos e de tempo real; no Capítulo 3 é apresentado o estado da arte, destacando os principais trabalhos relacionados e os problemas encontrados, dando ênfase assim, ao problema de pesquisa abordado na tese; no Capítulo 4 é apresentada a proposta da tese, destacando a susceptibilidade a falhas dos três principais protocolos usados em redes automotivas, e a metodologia orientada a aspectos desenvolvida e adotada; o Capítulo 5, descreve um estudo de caso com a aplicação prática de um mecanismo de diagnóstico modelado e desenvolvido para validação da metodologia, avaliando por meio de dois tipos de hardwares de injeção de falhas, as possibilidades de diagnóstico das degradações de desempenho nos protocolos de comunicação; o Capítulo 6 apresenta os resultados e contribuições da metodologia proposta; por fim, o Capítulo 7 apresenta conclusões e perspectivas futuras da pesquisa que foi desenvolvida.

2 BASE TEÓRICA

Este capítulo apresenta a base teórica que serve de subsídio para a pesquisa no contexto de modelagem de falhas e protocolos de comunicação. Primeiramente são apresentados os principais protocolos de comunicação usados na indústria, dando mais ênfase aos protocolos usados em redes automotivas, pois, estes compõem o foco central do estudo realizado. Em seguida são apresentados os tipos de falhas que afetam os protocolos de comunicação e as redes onde estes são aplicados. Como estas redes interconectam diversos sistemas de controle embarcados, são apresentados também os principais conceitos de sistemas embarcados distribuídos e de tempo real. Na última parte deste capítulo são abordados conceitos relacionados à modelagem orientada a aspectos e também à engenharia de requisitos para os sistemas embarcados de tempo real.

2.1 Redes e Protocolos de Comunicação Industriais

Em diferentes campos da indústria as redes de comunicação são responsáveis por interligar diversos sistemas de controle, que compõem sistemas de controle distribuídos. Uma rede constituída e construída com vários microprocessadores e dispositivos de entrada e saída interconectados, responsável por tarefas de controle, é chamada de rede de controle embarcada distribuída (*Distributed Embedded Control System* – DECS) (COLNARIC; VERBER, 2007) (KOPETZ, 2011)]. De acordo com Wolf (2008) o automóvel é um excelente exemplo de um sistema embarcado distribuído: os microprocessadores são distribuídos por todo o automóvel, executando cálculos de forma cooperativa e coordenando a operação do veículo por meio de redes de comunicação. Estas redes fazem uso de protocolos de comunicação que estão em constante evolução, devido à complexidade crescente das redes e suas respectivas topologias. Pode-se destacar como principais características destas redes a distribuição do processamento, balanceamento de recursos, redução de custos, possibilidades de diagnóstico e manutenção do sistema, além da flexibilidade e escalabilidade.

Na indústria as redes de comunicação fizeram com que ao longo do tempo, diversos protocolos fossem criados, com características que permitissem a agilidade e produtivi-

dade nos variados processos. Um termo bastante conhecido são os chamados “Fieldbuses”, que compõem um grupo importante de protocolos especificados e padronizados para realizar a comunicação de aplicações comerciais de controle (PEREIRA; NEUMANN, 2009).

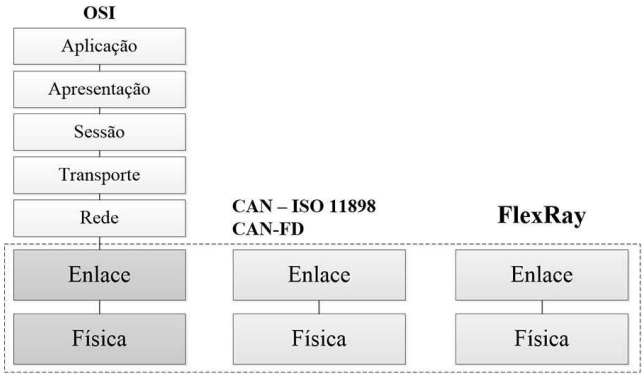
As redes de comunicação são organizadas em camadas, normalmente hierarquizadas e que são provedoras de serviços uma para as outras. O número de camadas depende do tipo de sistema de controle, do meio físico de comunicação e das possíveis abstrações inerentes à aplicação. O objetivo geral é que essas camadas forneçam serviços de forma transparente entre os níveis, visando abstrair aspectos de complexidade de implementação que devem ser específicos em cada camada.

Nestas redes os protocolos definem a forma de comunicação entre os diferentes “nós” (estações de trabalho, sensores, atuadores, etc.) que devem ser padronizados para garantir a comunicação efetiva entre os mesmos. O protocolo é um acordo entre as partes que se comunicam, onde estas partes se comunicam seguindo as regras das diferentes camadas e são normalmente chamadas de pares (*peers*). É possível verificar que com este tipo de comunicação em camadas e com a especificação de diferentes protocolos, a troca de informações entre os processos é afetada por atrasos temporais, os quais devem ser tratados e considerados para melhoria de desempenho das aplicações industriais.

Em sua base, para esta comunicação ocorrer de forma padronizada, a organização internacional para a normalização (*International Standards Organization - ISO*) foi a primeira a estabelecer um padrão de arquitetura para a interconexão de sistemas computacionais, assim, em 1984 foi formalizado o padrão de interconexão de sistemas abertos (*Open Systems Interconection* - OSI) (TANENBAUM; VAN STEEN, 2007). Porém o modelo OSI é um modelo de referência e não pode ser considerado uma arquitetura de rede, pois, não especifica os serviços e os protocolos que serão usados em cada camada, apenas informa o que cada camada deve fazer. Desta forma, neste trabalho serão enfatizadas as camadas iniciais (física e enlace), que são implementadas pelos protocolos de comunicação veiculares. A Figura 1 ilustra os protocolos utilizados no estudo e sua relação com o modelo de referência especificado.

Dentro do contexto de padrões de comunicação em redes, também podem ser associadas as redes industriais, que ao longo do tempo também destacaram a necessidade de interligação de computadores para formar sistemas distribuídos. Este modelo permite observar que o processo de comunicação é complexo, pois o canal de comunicação pode não estar disponível por ser dedicado a uma comunicação em um tempo específico. Diferentes protocolos são desenvolvidos e dedicados a camadas especificas de acordo com o modelo de referência OSI. Por exemplo, alguns protocolos procuram resolver os problemas de comunicação inerentes ao acesso ao canal, os principais estão relacionados à subcamada de acesso ao meio MAC (*media access control*), que é classificada como uma sub-camada da camada de enlace de dados. Com relação ao acesso à camada MAC, os protocolos são

Figura 1 – Protocolos de comunicação veiculares e suas camadas.



Fonte: Autor.

divididos em *time-triggered* e *event-triggered*.

Em sistemas *time-triggered*, a competição entre componentes de rede pelo uso do meio de transmissão é evitada através de políticas de escalonamento baseadas no método de divisão do tempo, mais conhecidos como TDMA (*Time Division Multiple Access*) que permite garantias temporais, baixo *jitter* e comportamento previsível. Já em comunicações *event-triggered*, a requisição de acesso ao meio ocorre sob demanda, característica que provê maior flexibilidade ao sistema, permitindo adicionar ou remover componentes da rede. Exemplos de métodos que trabalham com este tipo de política são o CSMA/BA (*Carrier Sense Multiple Access with Bitwise Arbitration*), CSMA-CD (*Carrier Sense Multiple Access with Collision Detection*), CSMA/AMP (*Carrier Sense Multiple Access with Arbitration on Message Priority*), FDMA (*Frequency Division Multiple Access*) (GE; YANG; HAN, 2017).

De acordo com (GROOVER, 2007) as redes industriais propiciam o sincronismo entre os dispositivos e viabilizam o intercâmbio de informações entre os diversos componentes de um sistema de automação. Em sistemas de automação sempre encontramos elementos sensores, controladores e atuadores, interface homem máquina, e sistemas de supervisão para permitir a interação entre o operador e o sistema.

Esta interação de operador e sistema de controle possui uma evolução constante de acordo os avanços das tecnologias de redes de comunicação. De acordo com (PEREIRA; CARRO, 2007) o crescimento explosivo da computação, comunicação e tecnologias da informação permitem experiências únicas e as plantas industriais também são afetadas por esta "computação ubíqua e pervasiva". Na indústria as novas tecnologias permitem a alta conectividade da planta industrial e as tendências de futuro para a automação e manufatura industrial estão relacionadas a evolução e expansão constante das redes de comunicação, onde cada vez mais conceitos são introduzidos, como a Internet das Coisas (ATZORI; IERA; MORABITO, 2010) (WHITMORE; AGARWAL; DA XU, 2015) e a Computação Pervasiva e Ubíqua (YU *et al.*, 2013).

Na área automotiva o crescente aumento das unidades de controle (ECU – Electronic

Control Units) impulsionou a definição de novas topologias e o uso de diferentes tipos de protocolos de comunicação. O conceito de sistema distribuído é cada vez mais importante para a concepção de redes intra-veiculares devido à grande quantidade de aplicações que dependem desta comunicação, as quais são fundamentais para muitas aplicações de segurança crítica, como ABS (*Anti-lock Braking System*), controle de tração, suspensão ativa, entre outros. Estas redes são baseadas em protocolos responsáveis pela interconexão das ECUs, que se comunicarão via rede e devem executar ações de controle de forma autônoma, respeitando restrições temporais e independente de ações do motorista.

Atualmente as redes e os protocolos de comunicação industriais são comuns e há um crescimento constante em diferentes aplicações de sistemas de controle, como na aviação com os chamados sistemas fly-by-wire – FBW e na área automotiva com os sistemas chamados de drive-by-wire – DBW ou x-By-wire – XBW, onde sistemas eletrônicos ou hidráulicos são substituídos (parcialmente ou totalmente) por sistemas puramente eletrônicos por meio de protocolos de comunicação (TAKARABE, 2009) (LANGE, 2015).

Dentro deste contexto, a seguir são destacados os principais protocolos de comunicação usados na indústria automotiva, que é o foco desta pesquisa, detalhando os aspectos de funcionamento em diferentes tarefas de controle, topologias de interconexão e criticidade das tarefas que usam estes protocolos.

2.1.1 Protocolos CAN, TT-CAN, FTT-CAN e CAN-FD

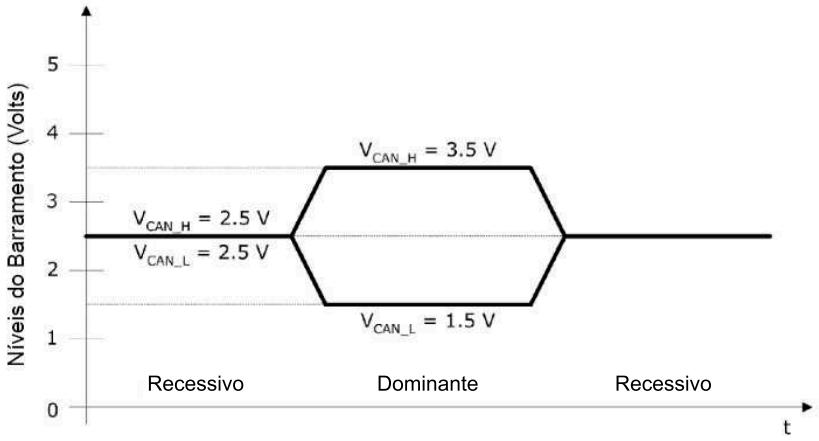
2.1.1.1 CAN

De forma a simplificar os sistemas com múltiplos fios que cresciam ano a ano para interligar os sistemas de controle automotivos, a empresa alemã BOSCH desenvolveu no início dos anos 1980 o protocolo CAN (*Controller Area Network*) (ISO-11898, 2015). A especificação geral de uma rede CAN está presente na ISO 11898, ISO 11898-1 a ISO 11898-5 (*physical, high-speed, low-speed fault-tolerant* / LS FT-CAN, *time-triggered* / TTCAN, *miscellaneous*, respectivamente), além da ISO-16845 (CAN *Conformance Testing*) (NAVET; SIMONOT-LION, 2013). A sua regulamentação específica para aplicações automotivas foi padronizada pela SAE (*Society of Automotive Engineers*).

O protocolo CAN é baseado na transmissão de mensagens, onde cada mensagem possui um identificador que é responsável por indicar a prioridade da mensagem. Esse tipo de abordagem permite a resolução de conflitos durante o acesso ao meio de transmissão, onde a mensagem com o menor identificador possui maior prioridade sobre as demais. A camada física do protocolo é definida pelos padrões ISO 11898-2 e ISO 11898-3. A transmissão de bits utiliza o padrão NRZ (*Non-Return to Zero*), a taxa de transferência de dados pode ser de até 1 Mbit/s, e a maior distância entre dispositivos não pode passar de 40m. A Figura 2 ilustra este modo de codificação usado na camada física do protocolo.

Na codificação NRZ existem dois níveis de tensão para representar os bits 0 e 1, sendo uma forma muito simples de codificação que consiste em associar um nível tensão a cada

Figura 2 – Codificação NRZ na camada física do protocolo CAN.



Fonte: ISO 11898 (ISO-11898, 2015).

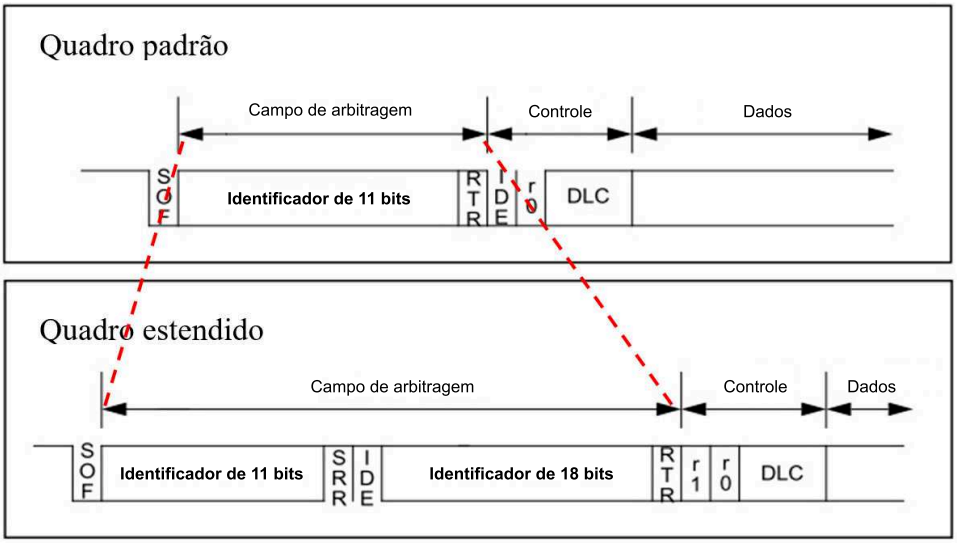
bit. No protocolo CAN somente as camadas 1 e 2 do modelo OSI são definidas, sendo que para a camada 2 o controle de acesso é do tipo CSMA/CD-AMP, onde as colisões de mensagens são resolvidas por um mecanismo de arbitração bit a bit de natureza não destrutiva baseado em prioridade. Para o funcionamento deste mecanismo de arbitração são definidos na camada 1 o nível recessivo – valor binário "1", e dominante – valor binário "0". Assim, no momento da transmissão de uma mensagem os controladores monitoram o nível do barramento. Se o controlador transmite um bit recessivo, o nível lógico do barramento é lido e comparado no mesmo tempo de bit. Caso o nível lido seja dominante então o controlador suspende a transmissão e se torna um receptor da mensagem.

As mensagens transmitidas pela rede CAN possuem duas versões, a 2.0A com campo identificador de 11 bits e a 2.0B com campo identificador de 29 bits. A seguir a Figura 3 apresenta uma breve descrição destes quadros de mensagens.

Apesar de ser eficiente e apresentar bom desempenho na maioria das aplicações o protocolo CAN apresenta deficiências quando erros de transmissão ocorrem com maior frequência. O determinismo do sistema é afetado pelos mecanismos de sinalização global de erros e retransmissão automática de mensagens. Com a alta taxa de retransmissões o protocolo é afetado por outros problemas, como perda de pacotes e a geração de atrasos. Assim, com o aumento do tráfego na rede mensagens de menor prioridade podem sofrer atrasos cada vez maiores. Neste contexto, o protocolo CAN possui 5 métodos para verificação de erros de transmissão ou de formatação das mensagens, os quais são:

- Monitoramento de Bit (*Bit monitoring*): O emissor compara o nível do bit enviado com o nível do barramento real. Um erro de bit existe se o emissor detectar uma discrepância entre os dois níveis. O monitoramento de bits garante que todos os erros globais e todos os erros locais que ocorrem no emissor sejam detectados. (tratamento no emissor).

Figura 3 – Quadro de mensagens padrão e estendido do protocolo CAN.



Fonte: ISO 11898 (ISO-11898, 2015).

- Monitoramento do formato da mensagem (*Form Check*): As ECUs receptoras verificam os bits de quadro das mensagens que possuem valores fixos. Cada mensagem CAN sempre exibe as mesmas sequências de bits em determinadas posições. No delimitador de CRC, no delimitador de ACK e no EOF, os remetentes sempre transmitem bits recessivos. O frame de início da mensagem sempre possui valor dominante. Caso esses valores não estejam de acordo com a padrão um erro de quadro é emitido. (tratamento do receptor).

- Monitoramento de fluxo de bits (*Bit stuff check*): O protocolo CAN especifica que o emissor deve transmitir um bit complementar (invertido) após cinco bits homogêneos, para fins de sincronização. Se uma ECU precisa transmitir mais de 5 bits idênticos a unidade adiciona um bit complementar (*stuff bit*), a cada sequência de 5 bits, assim, a ECU receptora deve retirar este bit adicionado para então interpretar a mensagem. O protocolo gera um erro se mais de cinco bits contíguos homogêneos forem recebidos. (tratamento no receptor).

- Verificação de redundância cíclica (*Cyclic Redundancy Check*): Na verificação de CRC o polinômio R(x) associado aos dados que chegam deve ser igual a um múltiplo do polinômio gerador G(x) especificado pela ISO 11898-1. A ECU emissora calcula o valor em função dos bits encaminhados na mensagem e esse valor é enviado junto no respectivo quadro. A ECU receptora faz o mesmo procedimento e se esses valores forem divergentes ocorreu algum erro durante transmissão. (tratamento no emissor e receptor).

- Verificação de recebimento (*ACK Error Check*): O mecanismo de reconhecimento do protocolo CAN especifica que os receptores devem confirmar a mensagem CAN

> recebida, após a verificação de CRC. As ECUs receptoras marcam o campo ACK com um bit dominante e enviam uma mensagem de resposta, assim, se a ECU emissora receber uma mensagem com o campo ACK contendo um bit recessivo significa que a mensagem enviada estava corrompida ou ninguém a recebeu. Um único reconhecimento positivo é suficiente para sinalizar que pelo menos um receptor recebeu a mensagem CAN corretamente. Se nenhuma confirmação positiva chegar ao remetente, ocorrerá um erro de confirmação (ACK *error*). (tratamento do emissor).

Estas verificações fazem com que o protocolo possua grande capacidade de se adaptar erros que podem ocorrer durante o funcionamento da rede, porém a frequência de ocorrência destes tratamentos pode indicar ruídos persistentes e degradações de desempenho na rede e consequentemente nos sistemas que dependem dela.

2.1.1.2 TT-CAN

O protocolo CAN padrão é orientado a eventos o que significa que as mensagens são geradas em resposta a situações/eventos que ocorrem na rede. No entanto, devido à natureza da arbitragem de mensagens CAN, onde o resultado e o tempo necessário para resolver cada arbitragem é completamente dependente do valor dos identificadores de mensagem no momento da arbitragem, os tempos necessários para enviar e receber mensagens não podem ser caracterizados deterministicamente. Isto é considerado insuficiente para aplicações "em tempo real", em que as restrições "*hard*" em tempo real impõem requisitos rigorosos sobre a capacidade dos nós de rede de se comunicar (TALBOT; REN, 2009) (ZENG; KHALID; CHOWDHURY, 2016).

Nas aplicações de tempo real é desejável que as mensagens sejam transmitidas em instantes precisos, executando essa transmissão e recepção durante um instante de tempo determinado. Para permitir esta abordagem uma camada superior chamada TT-CAN (Time-Triggered CAN) foi desenvolvida (LEEN; HEFFERNAN, 2002). O TT-CAN também contribui para evitar o jitter de acesso ao meio através da utilização desta técnica de divisão do tempo (XIA *et al.*, 2013). O padrão ISO 18898 foi estendido para dar suporte à este tipo de comunicação sob o código ISO 18898-4.

O instante de tempo periódico definido denominado "Ciclo Básico", é composto por algumas janelas de tempo de tipos e tamanhos diferentes. A janela exclusiva é utilizada para transmissão de mensagens periódicas. Para mensagens esporádicas as janelas de arbitragem são utilizadas. Neste caso, quando múltiplos dispositivos competem pelo acesso à rede, o mecanismo de arbitragem por prioridades padrão do protocolo CAN é utilizado. O último tipo de janela é a chamada janela de tempo livre, a qual é reservada para extensões futuras da rede, sendo possível alterar esta janela para uma do tipo exclusiva ou de arbitragem, de acordo com a necessidade (LEEN; HEFFERNAN, 2002) (XIA *et al.*, 2013).

2.1.1.3 FTT-CAN

De outro modo, outra atualização importante no CAN diz respeito a flexibilidade. Assim, o protocolo FTT-CAN (*Flexible Time-Triggered on Controller Area Network*) foi proposto com vistas a atender os requisitos de flexibilidade inerentes aos sistemas de tempo real. Essa flexibilidade implica em requisitos de comunicação dinâmica que permitam em tempo de execução a adição, remoção e adaptação de mensagens, sendo esta a principal característica que diferencia o FTT-CAN dos demais protocolos (ALMEIDA; PEDREIRAS; FONSECA, 2002) (ATAIDE; PEREIRA, 2012).

Para o seu funcionamento o protocolo faz uso do conceito de ciclo elementar (*elementary cycle* - EC) com duas fases de transmissão, para assim, combinar ambos os paradigmas *time-triggered* e *event-triggered* com um isolamento temporal entre eles. O tráfego *time-triggered* é escalonado em tempo de execução por um nó específico denominado nó mestre, que provê um controle da entrada mensagem no segmento e ainda garantindo os requisitos temporais das mensagens já presentes no segmento, caracterizando assim a flexibilidade do protocolo (ALMEIDA; PEDREIRAS; FONSECA, 2002).

De modo geral esse controle flexível se dá da seguinte forma: o nó mestre ativa as transmissões nos nós escravos seguindo um modelo denominado mestre-escravo flexível, que requisita de forma simultânea as mensagens a serem transmitidas no segmento *time-triggered* de um ciclo elementar. Uma mensagem específica denominada de TriggerMessage (TM) - é transmitida pelo nó mestre a fim de ativar o início de um ciclo elementar (EC) dentro de cada nó escravo, que por sua vez transmitirá mensagens nos segmentos time e *event-triggered*. A mensagem TM transporta informações que indicam quais mensagens serão transmitidas no segmento *time-triggered*.

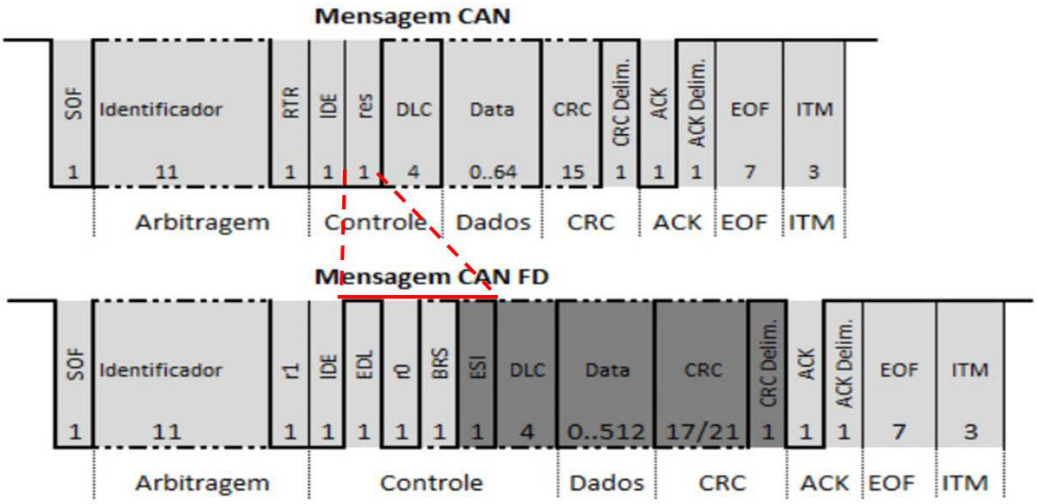
Apesar das características destacadas no FTT-CAN, o protocolo não tem sido usado comercialmente, além de ter a concorrência de outros protocolos que surgiram ao longo dos anos. Diversas pesquisas abordam as suas possibilidades de aplicação no contexto automotivo, como em (ATAIDE; PEREIRA, 2012), onde um experimento procurou melhorar o desempenho do protocolo FTT-CAN para aplicações automotivas, com um método baseado em offset para forçar a ordenação correta das mensagens de forma a reduzir o jitter inerente do efeito bloqueio e do método *bit stuffing*. Em (MARQUES *et al.*, 2013) outra verificação relacionada ao uso e melhorias aplicadas ao protocolo é realizada, agora com foco nas retransmissões de mensagens controladas por um servidor. Nesse trabalho os autores avaliaram o impacto de diferentes políticas de agendamento para o servidor, apresentando uma avaliação qualitativa das alternativas, complementada por um estudo de simulação, para verificar suas vantagens e pontos fracos. Desta forma, é possível verificar que o FTT-CAN é mais uma alternativa que busca prover confiabilidade ao processo de comunicação em sistemas distribuídos de tempo real, levando em consideração políticas de escalonamento flexíveis sem deixar de respeitar as restrições temporais rígidas que aplicações críticas exigem.

2.1.1.4 CAN-FD

A complexidade crescente das redes de comunicação pressiona cada vez mais o protocolo CAN ao seu limite, elevando as taxas de comunicação no barramento. Como pode ser visto em suas diversas extensões e evoluções, a aplicação do protocolo para interconectar sistemas distribuídos de tempo real, tem demandado modificações no formato das mensagens e nos métodos de escalonamento. Com esta motivação, em 2011 a empresa alemã BOSCH iniciou o desenvolvimento do CAN FD (*CAN with Flexible Data-Rate*), que é uma evolução do CAN clássico. O objetivo deste protocolo é atender as novas demandas da indústria automotiva com relação a maior quantidade e velocidade de transmissão de dados (HARTWICH, 2012).

O CAN FD iniciou com uma proposta similar a abordagem de (CENA; VALENZANO, 1999) que tratava do aumento da largura de banda por meio da modificação do formato do quadro da mensagem, melhorando o cabeçalho para possibilitar campos de dados mais longos e acelerando os quadros encurtando o tempo do bit. As inovações do CAN FD incluem o aumento da taxa de transmissão de dados para velocidades superiores a 1 Mbps e o aumento na quantidade de dados transmitidos de 8 para até 64 bytes. Assim, através da manutenção dos mecanismos de arbitragem e reconhecimento, torna-se possível aumentar a velocidade de transmissão dos demais campos da mensagem: *Data Lenght Code*, *Data Field* e *Checksum* (CRC) (BORTH, 2016). A seguir, a Figura 4 apresenta as diferenças entre o quadro de mensagem do CAN e do CAN FD.

Figura 4 – Quadro de mensagens CAN e CAN FD.



Fonte: (BORTH, 2016).

Destaca-se como principal diferença entre os protocolos o fato de que no padrão CAN-FD o campo de dados e o campo de CRC podem ser maiores que no padrão CAN. A diferenciação ocorre no bit reservado “res” do CAN logo após a arbitragem da mensagem. No CAN FD este bit é substituído pelos bits EDL (*Extended Data Length*), “r0”, BRS

(*Bit Rate Switch*) e ESI (*Error State Indicator*). Atualmente o protocolo já vem sendo integrado a novos projetos da indústria automotiva, tendo o padrão revisado e publicado pela ISO 11898-2 em 2016 e consolidado pela SAE sob as novas normas J2284-4 (junho de 2016) e J2284-5 (setembro de 2016).

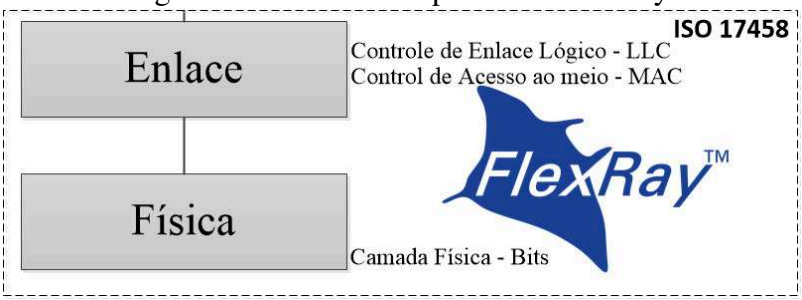
2.1.2 Protocolo FlexRay

O protocolo FlexRay surgiu em 2000 formado pelo *FlexRay Consortion*, que incluía BMW, Daimler-Chrisler, Philips e Freescale, sendo atualmente parte de um conjunto de padrões sob a ISO 17458 (FLEXRAY CONSORTIUM, 2010). O padrão ISO é formado por:

- ISO 17458-1:2013 - General information and use case definition.
- ISO 17458-2:2013 - Data link layer specification.
- ISO 17458-3:2013 - Data link layer conformance test specification.
- ISO 17458-4:2013 - Electrical physical layer specification.
- ISO 17458-5:2013 - Electrical phisical layer conformance test specification.

O protocolo FlexRay teve o seu primeiro uso em aplicações automotivas no ano de 2006 e possui como principais características ser *time-triggered*, possuir flexibilidade, e atender conceitos de segurança (*Safety*). A Figura 5 apresenta camadas do protocolo.

Figura 5 – Camadas do protocolo FlexRay.



Fonte: Autor.

O objetivo central deste protocolo é atender as demandas de largura de banda, confiabilidade, determinismo e sincronização em sistemas automotivos X-By-Wire (MICHELIN, 2014) (LANGE, 2015). O meio físico utilizado pelo protocolo FlexRay pode ser composto por fio de cobre (par trançado) ou fibra óptica. A conexão física é realizada levando em consideração a diferença de tensão entre os terminais BP (Bus Plus) e BM (Bus Minus), tendo 4 (quatro) níveis de tensão para identificação de estados, conforme apresentado na tabela 1.

Tabela 1 – Identificação de sinais e estados no protocolo FlexRay.

Estado	CAN-H e CAN-L	Sinal
Idle low power bus level	-0,2 e 0,2	0V
Idle bus level	1,8 e 3,2	0V
Data 1 bus level	3,5 e 1,5	2V
Data 0 bus level	1,5 e 3,5	-2V

A codificação para representação dos sinais do ciclo de comunicação para controle de acesso ao meio (MAC) é do tipo NRZ, onde existem dois níveis de tensão para representar os bits 0 e 1, sendo uma forma de codificação muito simples e amplamente utilizada. De acordo com a especificação o protocolo permite a conexão de até 64 nós, sendo que a distância máxima entre os cabos não deve ser maior que 24 metros, e de acordo com a topologia usada devem ter no máximo 22 ECUs no mesmo barramento. Outro ponto importante a destacar é o uso de redundância de canais de transmissão o que permite flexibilidade e maior segurança nas transmissões de dados.

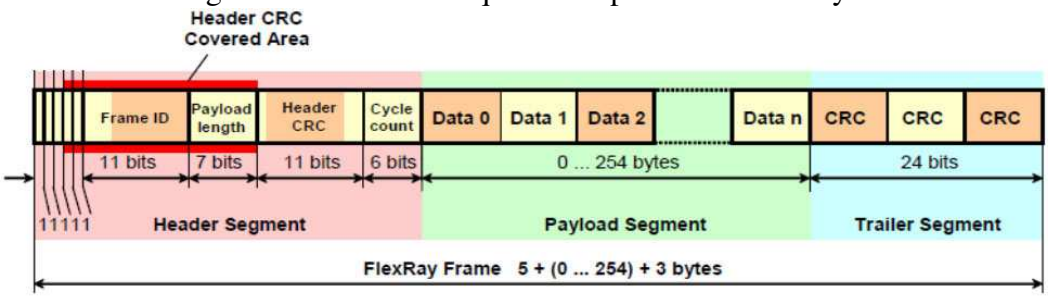
No protocolo FlexRay as topologias de interconexão podem ser do tipo barramento linear passivo, estrela ativa, estrela ativa em cascata, estrela passiva e topologia híbrida (FLEXRAY CONSORTIUM, 2010). Na topologia híbrida o atraso de sinal não deve ser superior a 450ns. De acordo com as topologias e as características do meio físico (com dois canais de transmissão - Channel A e B) é possível relacionar as taxas e bit rates de transmissão. Para a taxa de 10 Mbps a duração do bit deve ser de 100ns, para taxa 5 Mbps a duração do bit deve ser de 200ns, para taxa de 2.5 Mbps a duração do bit deve ser de 400ns. Sem redundância de canais é possível atingir taxas de até 20 Mbps.

A camada de enlace do protocolo FlexRay permite a comunicação através de quatro tipos de segmentos, o Segmento Estático, o Segmento Dinâmico, a Janela de Símbolos e o Tempo Inativo de Rede. No segmento estático, o objetivo é prover comunicação com garantias temporais, já no segmento dinâmico comunicações eventuais. A comunicação é baseada em uma hierarquia temporal dividida em Macroticks (específico número inteiro de microticks) e Microticks (tempo de acordo com cada oscilador da ECU), onde cada ciclo de comunicação possui um número fixo de macroticks (FLEXRAY CONSORTIUM, 2010) (ZENG; KHALID; CHOWDHURY, 2015).

O segmento estático usa acesso múltiplo por divisão de tempo ou TDMA (*Time Division Multiple Access*) para acesso ao meio. O segmento estático possui um número fixo de slots, configurável como uma constante para determinado cluster. O segmento dinâmico usa acesso flexível na divisão do tempo ou FTMA (*Flexible Time Division Multiple Access*) para acesso ao meio. Neste segmento é possível a alocação de um número de "mini-slots", onde este é configurado de acordo com o tamanho do pacote a ser transmitido. Este segmento ainda usa prioridade na transmissão dos quadros baseados no menor ID e segue a contagem de slots do segmento estático. O formato do quadro usado no pro-

tocolo FlexRay pode ser visualizado na Figura 6, onde é possível observar as três partes, o cabeçalho (header), a carga (payload) e a cauda (trailer).

Figura 6 – Formato do quadro do protocolo FlexRay.



Fonte: (FLEXRAY CONSORTIUM, 2010).

Para detalhar e exemplificar este tipo de organização pode ser referenciada a tecnologia Adaptive Drive Chassis Control System presente em veículos da BMW. Este sistema foi usado pela primeira vez em 2007 para controle de suspensão eletrônica em veículos da série X5 da BMW. O Flex Ray controla o sistema de barramento óptico do sistema de controle efetivando o controle dos estabilizadores e válvulas eletromagnéticas dos amortecedores, permitindo que a tecnologia elimine o efeito do rolamento do corpo no veículo (body roll effect) (LEE; KIM; JEON, 2017).

2.2 Conceitos de Tolerância a Falhas

Esta seção apresenta uma visão geral dos principais conceitos de Tolerância a Falhas ou Dependabilidade, dentro do contexto da presente pesquisa, com o intuito de destacar como falhas são eventos que podem ser tratados ou até mesmo evitados, considerando diferentes etapas do projeto e desenvolvimento de sistemas de controle.

O termo Tolerância a Falhas foi apresentado originalmente por Algirdas Avizienis durante a conferência AFIPS - *Fall Joint Computing Conference*, em 1967. Apesar da ampla adoção acadêmica, na indústria esse termo não teve boa aceitação, sendo mais comum encontrar o uso dos termos Sistemas Redundantes ou Sistemas de Alta Disponibilidade. De qualquer modo, muitas pesquisas vem sendo desenvolvidas com foco em métodos e técnicas de tolerância a falhas, visando por exemplo, o diagnóstico, detecção e prevenção a falhas que degradam os sistemas de controle, considerando diferentes áreas da indústria (GAO; CECATI; DING, 2015)

O uso do termo “Tolerância a Falhas” já foi bastante discutido, pois muitos autores destacam que um sistema computacional não deve ser “tolerante a falhas”, mas o uso do termo foi associado a área devido as diversas técnicas que permitem que um sistema continue em operação quando tais falhas ocorrem. Assim, estas técnicas permitem alcançar um nível de qualidade e confiabilidade do sistema, a “dependabilidade”. Os principais atributos de dependabilidade são confiabilidade, disponibilidade, segurança de funcionamento

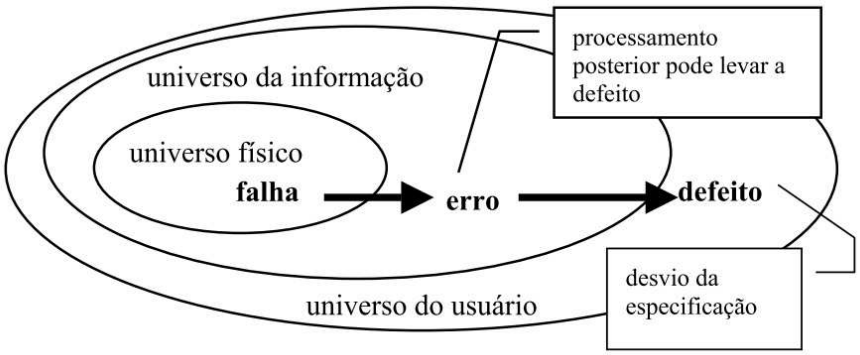
(*safety*), segurança (*security*), mantenabilidade, testabilidade e comprometimento do desempenho (*performability*) (PRADHAN, 1996). De acordo com (GUNES *et al.*, 2014), a dependabilidade se refere a propriedade de um sistema realizar suas funcionalidades, durante um tempo de operação, sem degradação significante no desempenho, refletindo em um grau de confiabilidade de todo o sistema.

Todo e qualquer tipo de sistema computacional, embarcado e/ou distribuído é concebido para satisfazer requisitos funcionais específicos, considerando um cenário ideal e controlado, onde entradas e saídas são conhecidas. Sendo assim, um cenário ideal de operação não reflete as condições reais de aplicação de um sistema, onde diversas variáveis podem influenciar e interferir na operação usual (baseada no ambiente controlado), gerando perturbações nos valores das entradas e saídas.

A origem de tais interferências que alteram estes valores podem ser diferentes tipos de falhas. Conceitualmente em sistemas computacionais os termos falha ou falta (*fault*), erro (*error*) e defeito (*failure*) estão associados a três contextos: a falha ou falta ao contexto físico, onde um hardware ou meio de comunicação é interrompido devido a interferências externas; o erro, ao contexto computacional devido a uma falha que gerou um erro no processamento de uma informação; e o defeito, que está no nível do usuário, relacionado ao desvio do funcionamento ideal ou esperado, que gera um impacto ou percepção direta no usuário (KOPETZ, 2011).

Outra abordagem amplamente usada para ilustrar e caracterizar falhas, erros e defeitos, é a baseada no modelo de 3 universos (PRADHAN, 1996). Falhas estão associadas ao universo físico, erros ao universo da informação e defeitos ao universo do usuário. A seguir a Figura 7 ilustra esse modelo de classificação.

Figura 7 – Abordagem baseada no modelo de 3 universos: falha, erro e defeito.



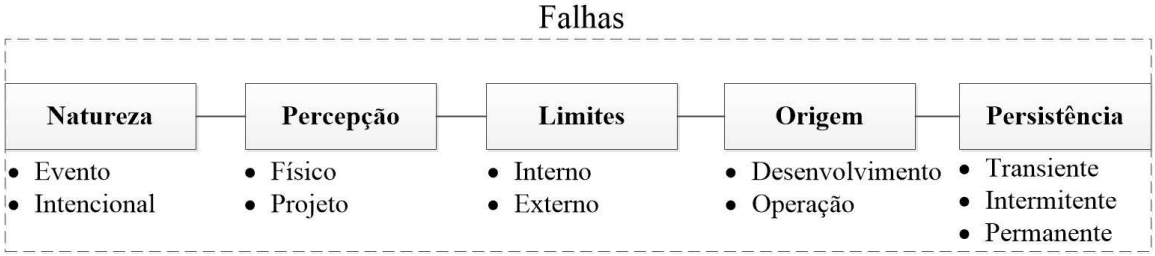
Fonte: Adaptado de (PRADHAN, 1996).

De acordo com esse modelo, é possível exemplificar a situação em que uma interferência, como transientes elétricos ou interferências eletromagnéticas, podem gerar danos como disrupção de sinais (falha no universo físico) em um protocolo de comunicação, alterando valores de seus bits, o que gera uma interpretação errada da informação contida

em uma mensagem (erro no universo da informação), e por consequência, gera resultados indesejados em tarefas de controle críticas (controle de tração, controle de vôo, entre outros) que são perceptíveis na saída final do sistema (defeito no universo do usuário). Em todas as situações, destaca-se que uma falha pode ser tratada, detectada e assim não levar aos problemas subsequentes.

Falhas, erros e defeitos ainda possuem classificações que caracterizam cada domínio. Devido ao contexto da presente pesquisa de doutorado, é detalhada a classificação das falhas ou faltas em nível físico, as quais afetam e degradam o desempenho dos protocolos de comunicação. As falhas ou "*Faults*" podem ser classificadas em cinco diferentes categorias que são ilustradas a seguir na Figura 8. Dentro do escopo desta classificação, a subseção 2.3 detalha as falhas que degradam os protocolos de comunicação.

Figura 8 – Classificação das Falhas em relação ao contexto físico.



Fonte: Adaptado de (KOPETZ, 2011).

Outros conceitos relacionados são os de sistemas de controle tolerantes a falhas, pois estes são responsáveis por tarefas críticas em áreas como a aviônica, espacial e automotiva. Falhas nestes sistemas levam a problemas catastróficos e normalmente são chamados de sistemas de segurança crítica (safety). De acordo com (JIANG; YU, 2012), um sistema de controle que pode automaticamente tratar falhas entre os componentes do sistema enquanto ao mesmo tempo mantém a sua estabilidade, com um nível desejado de desempenho, é denominado um sistema de controle tolerante a falhas – FTCS (*Fault Tolerant Control System*). Neste mesmo caminho também emergem os conceitos de sistemas de controle distribuídos e de tempo real com características de tolerância a falhas, onde técnicas são aplicadas desde o projeto, agregando mecanismos de redundância de hardware, técnicas de escalonamento de tarefas, isolamento de falhas, sistemas multi agentes, entre outros (PEREIRA; CARRO, 2007).

Todavia, para permitir que técnicas e metodologias de tolerância a falhas possam ser aplicadas, previamente é necessário aplicar metodologias de teste e estresse nos sistemas de controle embarcados, com o objetivo de conhecer e aprender com os problemas detectados. Nesse ponto, emergem muitas técnicas, padrões e ferramentas que podem ser aplicadas, as quais normalmente são específicas a cada cenário de aplicação (indústria automotiva, por exemplo). De acordo com (GAO; CECATI; DING, 2015), diversas abordagens para estas etapas são destacadas, como a exploração de métodos de injeção de

falhas (*Fault Injection - FI*), detecção e isolamento de falhas (*Fault detection and isolation - FDI*) ou com recuperação do estado de falhas (*Fault detection, isolation and recovery - FDIR*), diagnóstico e detecção de anomalias em sistemas de comunicação, análise de falhas em componentes eletrônicos e análise de sinais.

Dentre estas abordagens destaca-se o FDIR, que é uma metodologia de controle que garante a continuidade da operação do sistema, de forma segura e aceitável, quando uma falha ocorre e é detectada por um mecanismo de FDI, o que possibilita uma reconfiguração do sistema de controle em resposta a situação anômala detectada (HWANG *et al.*, 2010). O método FDI consiste de uma tomada de decisão binária, verificando que algo esta errado em dado momento, procurando a localização e identificação da falha.

Deste modo, o constante avanço da eletrônica embarcada torna o diagnóstico e detecção de falhas (*faults*) um constante desafio. De acordo com a Federação Iternacional de Controle Automático - IFAC (IFAC-CONTROL, 2019) uma falha (*fault*) é um desvio não permitido de pelo menos uma propriedade ou parâmetro do sistema, da sua condição aceitável, usual ou padrão. Tal mal funcionamento pode ocorrer na unidade individual das plantas de controle, em sensores, atuadores ou componentes lógicos de comutação.

Muitas falhas são inevitáveis, mas as consequências indesejadas destas podem ser tratadas com diferentes técnicas em hardware e software. O estudo e conhecimento destas técnicas é fundamental para o aumento da confiabilidade dos sistemas. A próxima seção destaca os pontos principais relacionados aos conceitos de tolerância a falhas, aplicados especificamente no contexto dos protocolos de comunicação usados em redes intraveiculares.

2.3 Tipos de Falhas em Protocolos de Comunicação

A confiabilidade de um sistema computacional é definida pela combinação de diferentes aspectos, enfatizando a capacidade do sistema de atender as suas especificações, sob um conjunto de condições, durante um período de tempo definido e também condicionado a estar operacional quando o tipo de controle ou função é requisitado. Os aspectos de confiabilidade são inerentes ao conceito de tolerância a falhas ou dependabilidade em sistemas computacionais. A tolerância a falhas é muito importante em sistemas críticos de tempo real (*safety-critical real-time systems*) porque uma falha em um componente pode levar a uma falha catastrófica no sistema. Neste contexto, em um sistema de controle distribuído, um nó da rede é responsável por uma tarefa de controle especifica e pode estar associada a um ou mais tipos de falhas.

Como destacado na seção anterior, os conceitos de tolerância a falhas estão associados a três domínios (falha, erro e defeito). O presente trabalho aborda as falhas (do inglês *faults*) em seu contexto físico, de acordo com sua persistência, que afetam os protocolos de comunicação (por meio de interferências, nos seus limites internos e externos), e que

por sua vez, podem gerar posteriormente erros e defeitos. A seguir a subseção 2.3.1 apresenta conceitos e detalha os tipos de falhas com relação a sua persistência (transiente, intermitente ou permanente).

2.3.1 Falhas transientes, intermitentes e permanentes

Na indústria automotiva um processo de controle pode ser composto de diversos sistemas computacionais embarcados que caracterizam complexos sistemas ciber físicos, onde estes, são compostos por sistemas computacionais colaborativos que controlam entidades físicas, as quais se comunicam e dependem de protocolos de comunicação específicos.

Diferentes tipos de falhas afetam os protocolos de comunicação. De acordo com (MARQUES *et al.*, 2013) estas falhas podem ser permanentes, intermitentes ou transientes. As permanentes são falhas de software e hardware que sempre produzem erros, enquanto as temporárias são aquelas que podem produzir erros por falhas externas com efeitos transitórios (*transient*) ou por falhas internas esporádicas (*intermittent*) (AVIZIENIS *et al.*, 2004).

Falhas transientes são temporárias ou eventuais por natureza e normalmente ocorrem devido a fatores externos como interferências eletromagnéticas (*electromagnetic interference* - EMI), efeitos de radiação ou variação de temperatura. Por outro lado, falhas permanentes são persistentes e resultantes de defeitos físicos em partes do sistema, necessitando de reparos ou substituição de componentes (MAHAPATRO; KHILAR, 2013) (GAO; CECATI; DING, 2015).

Falhas transientes podem afetar a rede de comunicação em situações de controle críticas. Quando a falha transiente ocorre durante um processo de comunicação, uma mensagem que está sendo transmitida pode ser perdida devido a erros de bit. A taxa de erros na recepção das mensagens pode ser avaliada por erros de bits (or BER do inglês "*Bit-Error Rate*") ou também por erros/perda de pacotes (or PLR, do inglês "*Packet-Loss Rate*"), definida também em função do ambiente de operação. Normalmente é um importante ponto a ser considerado e agregado em técnicas de modelagem de comunicação para impulsionar o desenvolvimento de sistemas mais confiáveis (MAHAPATRO; KHILAR, 2013).

Falhas transientes são causadas por eventos oriundos do ambiente onde o sistema está em operação e nem sempre implicam em uma falha efetiva, mas podem degradar o desempenho de forma silenciosa, podendo em algum momento gerar uma falha grave. Uma falha intermitente (*soft-fault*) é originada da parte interna do sistema quando o software/hardware entra em estado de falha. Por natureza uma falha intermitente não ocorre de forma consistente, o que torna o diagnóstico um evento probabilístico ao longo do tempo.

É difícil determinar a diferença entre uma falha transiente e intermitente por uma simples observação do sistema. Uma falha gerada por eventos externos pode ter as mesmas características das geradas por eventos internos. A importância desta distinção é de que uma falha transiente nem sempre implica no estado falho do sistema, embora este estado

justifique ações de prevenção como a reinicialização do sistema. Por outro lado, se a falha é intermitente o sistema deve ser declarado falho até que o problema seja corrigido (AVIZIENIS *et al.*, 2004).

A nível de chip as falhas transientes são muito mais comuns que as falhas permanentes. A taxa de falhas transientes em chip pode ser da ordem de 10 a 100.000 vezes maior que a taxa de falhas permanentes em chip, dependendo do ambiente de operação e instalação. A causa mais comum de falhas transientes são as interferências eletromagnéticas (EMI), distúrbios ou ruídos de fonte de alimentação e a exposição a partículas de alta energia (KOPETZ, 2011).

2.3.2 Análise dos efeitos de interferências em protocolos de comunicação

Como apresentado na seção anterior, diferentes tipos de falhas e interferências podem degradar um sistema de comunicação, e para verificação deste impacto algumas métricas são comumente utilizadas, como por exemplo, a taxa de erros de bits - BER, a taxa de erros/perda de pacotes - PLR, vazão, latência e jitter. Outros fatores como a disposição dos dispositivos na rede, a topologia de interconexão, distância dos nós e consumo de energia também influenciam e servem de métricas de verificação da qualidade do processo de comunicação. A seguir, de acordo com (BREED, 2003) e (BERBER, 2004) é apresentada uma breve descrição destas principais métricas utilizadas para avaliar o desempenho dos protocolos de comunicação com relação a erros de bits e perda de pacotes.

Taxa de Erros de Bit - BER - A taxa BER é caracterizada pelo número de bits errados por unidade de tempo e calculada verificando esse número de erros e dividindo pelo número total de bits transferidos durante um intervalo de tempo específico. Geralmente, essa grandeza é representada como uma potência de 10. Por exemplo, para a qualidade de transmissão das redes locais de Ethernet a 10Mbit/s, é normalmente esperado um valor melhor do que um bit errado em um bilhão de bits transmitidos, ou uma BER de 10^{-9}. Essa taxa é bastante usada para avaliar a qualidade de comunicações seriais, a qual é definida pela Equação 1.

$$BER = \frac{transmission.errors}{total.of.transmitted.bits} \tag{1}$$

A taxa de erros também é comumente estabelecida por probabilidade de ocorrência de erros - POE, de acordo com a Equação 2.

$$POE = \frac{1}{2}(1 - erf)\sqrt{Eb/No} \tag{2}$$

Onde, *erf* é a função do erro, *Eb/No* é a relação entre a energia presente em um bit e a densidade de ruido espectral na largura de 1 Hz. A função de erro é diferente para cada método de modulação, sendo mais importante considerar que o POE é proporcional a *Eb/No*, formando uma relação sinal-ruído.

Taxa de perda de pacotes - PLR - de forma similar, a taxa PLR pode ser obtida pelo número de pacotes recebidos incorretamente dividido pelo número total de pacotes recebidos, ou seja, o número de pacotes descartados na camada MAC em função de erros nos bits do pacote, de acordo com a Equação 3.

$$PLR = \frac{P_F}{P_{tx}} \tag{3}$$

Onde P_F é o número de pacotes com falhas e P_{tx} número de pacotes transmitidos. Esta métrica é utilizada para avaliar as condições de ruído ou interferência no meio.

Em sistemas embarcados de tempo real, o papel de um barramento de comunicação é distribuir dados entre nós conectados ao barramento. Assim, existem requisitos de tempo nesta transmissão, por exemplo, no que diz respeito ao atraso de transmissão (latência/delay) ou *jitter* (a diferença entre o melhor e o pior caso de atraso de transmissão) (BROSTER; BURNS; RODRIGUEZ-NAVAS, 2005) (BURNS; DAVIS, 2013). Também de acordo com as definições dadas por (STALLINGS, 2007), os conceitos de atraso, *jitter* e vazão são detalhados a seguir.

Atraso: do inglês "*Delay end-to-end*" caracteriza o tempo decorrido para transmitir um bloco de dados de um nó/ponto de origem até que ele alcance o nó/ponto de destino. Em uma rede de comunicação é uma informação muito importante, pois definido o tempo entre mensagens enviadas de um nó a outro da rede. Esse parâmetro depende da velocidade do meio de transmissão, com relação ao tipo de meio físico utilizado. Em sistemas de tempo real é uma métrica fundamental de desempenho da rede de comunicação, pois determina o tempo entre o início e termino de uma tarefa de controle, que contribui para definição de restrições temporais do sistema.

Jitter: representa a magnitude da variação do atraso. Em aplicações de tempo real esta é uma medida importante, pois os pacotes são esperados em intervalos fixos de tempo no destino. Pacotes que chegam atrasados são normalmente descartados, principalmente em aplicações que requerem uma malha de controle em laço fechado, em que pacotes atrasados podem resultar em uma resposta errônea ao sistema de controle.

Vazão: do inglês "*throughput*" é a largura de banda efetiva ou carga, que representa a quantidade de dados isentos de erros transmitidos por unidade de tempo. Teoricamente, a vazão deve aumentar à medida que a carga oferecida à rede cresce, até o máximo da capacidade de rede. Mas na prática a vazão da rede depende do método de acesso ao meio, da carga atual da rede e da taxa de erros. Também usa-se o termo inglês "*busload*" referindo-se a variação da carga sob diferentes aspectos.

Desta forma, estas métricas auxiliam na análise do impacto de falhas geradas por interferências internas e externas ao sistema, que são preponderantes para as características de confiabilidade. A seguir a Seção 2.3 apresenta os principais conceitos dos sistemas embarcados distribuídos que são relevantes para o desenvolvimento da presente pesquisa.

2.4 Sistemas Embarcados Distribuídos e de Tempo Real

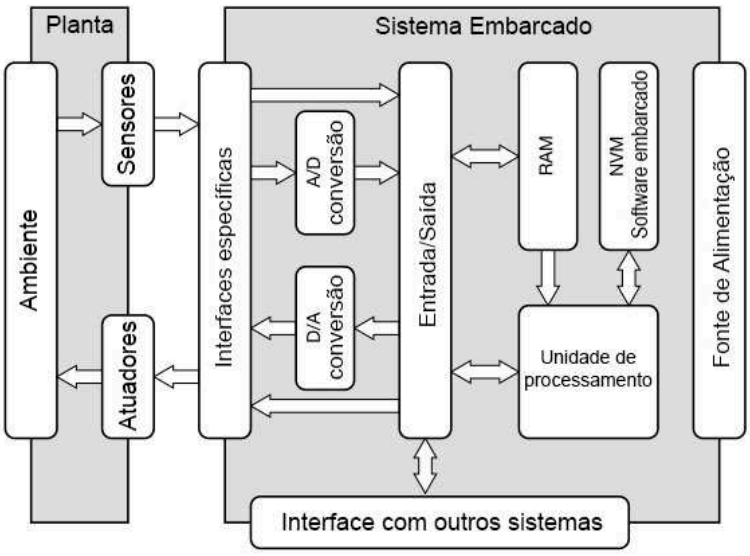
2.4.1 Sistemas Embarcados

Existem algumas definições para sistemas embarcados, também chamados de sistemas embutidos ou sistemas dedicados. De modo geral as definições destacam que são sistemas microprocessados com a finalidade de gerenciar funcionalidades pontuais com ou sem interação com o ambiente (WOLF, 2008). Diferente de computadores de propósito geral, um sistema embarcado realiza um conjunto de tarefas predefinidas e geralmente com requisitos de desempenho específicos.

Sistemas embarcados são sistemas com a finalidade de automatizar e controlar um ambiente ou dispositivo físico. Segundo (BROEKMAN; NOTENBOOM, 2003), um sistema embarcado pode ser definido como um sistema computacional especializado que faz parte de uma máquina ou sistema maior.

Estes conceitos são inerentes ao ambiente onde estes sistemas são aplicados, pois são importantes e responsáveis por diversas atividades humanas, e em muitos casos, fazendo parte de artefatos eletrônicos comuns e que passam despercebidos por grande parte das pessoas, como os celulares, equipamentos médicos, e uma lista extensa de produtos que moldam nosso mundo atual (RODRIGUEZ-SÁNCHEZ *et al.*, 2016). Como exemplo para ilustrar de forma resumida a composição de projetos de sistemas embarcados, a Figura 9 ilustra um diagrama de blocos com as principais partes que compõem este tipo de sistema.

Figura 9 – Diagrama geral que exemplifica um sistema embarcado.



Fonte: Adaptado de (BROEKMAN; NOTENBOOM, 2003).

No exemplo ilustrado na Figura 9 observa-se que um sistema embarcado interage com sensores e atuadores para efetivar um controle específico no ambiente ou planta de atuação. Esse sistema embarcado pode ainda ser interconectado com outros formando um sistema maior, distribuído em um ambiente maior de atuação, onde estes sistemas coope-

ram entre si. Nos dias atuais é evidente a inserção dos sistemas computacionais embarcados e distribuídos em diferentes áreas, destacando neste trabalho o uso no controle de diversas tecnologias e processos industriais, incluindo a grande quantidade de tecnologias aplicadas na área automotiva, por meio das unidades de controle eletrônicas - ECUs.

De acordo com (WOLF, 2008) na área automotiva um sistema embarcado pode ser responsável pelo controle de diferentes funcionalidades do veículo, ou auxiliar na realização de alguma função, como por exemplo, controle do motor, de freios, transmissão e suspensão. Nas primeiras redes veiculares era atribuído um processador para cada dispositivo físico. Atualmente, os projetistas e engenheiros tendem a combinar várias funções em uma unidade de processamento e realizar a comunicação com os demais dispositivos.

Nos automóveis as ECUs são exemplos de sistemas embarcados servindo como fonte de uma parte da computação do sistema. Nos sistemas modernos são atribuídas múltiplas tarefas a uma ECU para reduzir o número elementos de processamento e seu hardware de suporte associado (TUOHY *et al.*, 2015).

2.4.2 Sistemas de Tempo Real

Em muitas áreas os sistemas embarcados se caracterizam em relação ao impacto de suas restrições de funcionamento em função do seu tempo de processamento. Desta forma, sistemas de tempo real caracterizam-se por possuírem fortes requisitos no atendimento de restrições ou requisitos temporais, como por exemplo, tempo máximo de execução, jitter e deadline, além de robustez e requisitos de segurança.

Estes sistemas normalmente podem ser divididos em *Hard Real-Time* ou *Soft Real-Time*, ou seja, dependendo da criticidade com relação ao tempo de resposta de suas saídas, que normalmente resultam em ações de controle específicas. No sistema *Hard* há a necessidade de garantia que todas as restrições de tempo sejam satisfeitas. No sistema *Soft* não há exigências como garantia e ainda possui uma abordagem de menor esforço, pois o impacto de falhas nas restrições de tempo não é catastrófico (WOLF, 2008).

“Sistemas de tempo real são aqueles em que a corretude do mesmo não depende apenas dos resultados lógicos da computação, mas, também, no tempo em que esses resultados são produzidos” (STANKOVIC, 1996).

Em sistemas embarcados críticos as restrições temporais devem ser satisfeitas de forma rígida, já outros tipos de sistemas embarcados provêm certa flexibilidade quanto aos intervalos de tempo suportados pelo sistema. Os sistemas embarcados de tempo real devem ser projetados de forma que todos os *deadlines* de comunicação sejam obedecidos, para tal torna-se necessário incluir escalonadores ou núcleo de sistemas operacionais que garantam a execução concorrente de tarefas de acordo com as restrições temporais. A criticidade e os efeitos colaterais de falhas nos sistemas embarcados caracterizam outra linha mais específica de pesquisa, os sistemas embarcados críticos e de tempo real (*Critical Real-Time Embedded Systems*).

Em sistemas de tempo real um importante aspecto a ser considerado é a previsibilidade de execução das tarefas, pois essa característica determina que tais sistemas sejam determinísticos. Assim, o principal ponto a considerar em um sistema de tempo real não está relacionado ao seu desempenho mas sim à capacidade do sistema ser previsível (STANKOVIC, 1996) (KOPETZ, 2011). Neste contexto de criticidade e determinismo na execução das tarefas, os sistemas embarcados de tempo real são cada vez mais responsáveis por funções de segurança (safety) crítica para os usuários. No cenário tecnológico atual os sistemas embarcados de tempo real podem ser tão complexos quanto necessários, devido a produção em massa e a fabricação de componentes a custos reduzidos.

Este cenário permitiu cada vez mais a inserção de sistemas de tempo real no controle de funções importantes na indústria e também nos produtos por elas gerados. Como por exemplo na indústria automotiva, onde aplicações computacionais só eram utilizadas em veículos em tarefas não críticas e de conforto dos veículos, e atualmente diversos fatores impulsionaram a utilização de sistemas de tempo real em aplicações críticas como o controle de tração, suspensão e transmissão de veículos. Em (KOPETZ, 2002) o autor enfatizava que no futuro observaríamos a integração de muitas funções com o objetivo de aumentar a estabilidade do veículo em manobras críticas de condução. Não só chegou-se a este ponto, mas surgiram outros avanços com as pesquisas relacionadas a carros autônomos, onde cada vez mais os sistemas de tempo real são fundamentais para a execução das tarefas de controle, e por consequência, preocupa-se cada vez mais com os erros e a segurança na execução destas funções.

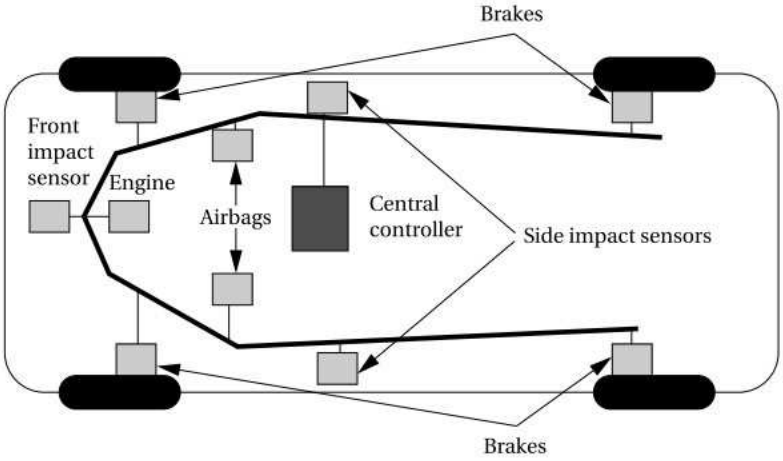
2.4.3 Sistemas distribuídos e o contexto automotivo

Sistemas distribuídos caracterizam-se pela distribuição espacial de unidades de processamento que em conjunto gerenciam um processo. Um sistema distribuído tem como objetivo dividir o trabalho de processamento proporcionando o resultado desejado sem que o usuário final do sistema perceba a descentralização (TANENBAUM; VAN STEEN, 2007). Além de propiciar a união das unidades de processamento para prover um maior poder computacional final, os sistemas distribuídos também são empregados em aplicações que se encontram descentralizadas, especialmente por necessidades específicas de projeto, como ocorre na amostragem de dados e na ação de atuadores em sistemas eletrônicos de direção de automóveis (SbW - *steer-by-wire*) (WANG *et al.*, 2014).

Uma característica importante de sistemas distribuídos esta relacionada ao fato de que usar várias unidades de processamento (CPUs), onde uma parte do sistema pode ser usada para ajudar a diagnosticar problemas em outra parte. Se há a necessidade de depurar um protótipo, diagnosticar um problema, ou isolar o erro em uma parte do sistema, essas tarefas se tornam difíceis quando tudo é feito em uma única CPU. Ao usar várias CPUs no sistema, é possível dedicar uma para gerar entradas de monitoramento de comunicação e fazer verificações nas saídas (WOLF, 2008) (KOPETZ, 2011).

Em sistemas embarcados distribuídos essas unidades de processamento são conectadas via rede usando protocolos de comunicação específicos (exemplificados na Seção 2.1) que viabilizam tarefas de controle distribuídas. O conceito de nó de processamento responsável por executar tarefas individuais é introduzido e faz parte de um conjunto complexo de sistemas de controle. Esse tipo de organização ainda permite ao sistema o uso de mais de um tipo de rede de comunicação, com topologias distintas usando elementos de interconexão (*gateways*). A Figura 10 ilustra um exemplo de rede veicular usada para interconexão de sensores para situações de impacto.

Figura 10 – Sistema distribuído para monitoramento de sensores na rede veicular.



Fonte: (WOLF, 2008).

Redes automotivas são um exemplo claro e importante da adoção de sistemas distribuídos, com diversos componentes interconectados e com restrições temporais rígidas para muitas funções. Carros e aviões modernos necessitam de confiabilidade de seus componentes eletrônicos para operar. Aproximadamente um terço do custo total de um avião e de um carro vem do uso de diversos dispositivos eletrônicos. Sistemas eletrônicos são usados em aplicações críticas em veículos, além de sistemas de monitoramento do conforto do passageiro. Estes dispositivos são conectados via sistema distribuído em rede e devem respeitar requisitos funcionais e temporais, garantindo ainda confiabilidade e desempenho (WOLF, 2008).

2.5 Especificação de Requisitos e Modelagem Orientada a Aspectos

2.5.1 Especificação de requisitos para sistemas de tempo real

A engenharia de requisitos compreende tarefas para o processo sistemático de elicitar, especificar, analisar, validar e gerenciar requisitos, considerando os usuários, objetivos e necessidades técnicas do sistema, como por exemplo, controlar um dispositivo, efetuar uma compra e procurar uma informação (DICK; HULL; JACKSON, 2017). Em sistemas embarcados de tempo real a especificação de requisitos segue esse processo sistemático

de análise, mas levando em consideração outros aspectos importantes, como as restrições temporais da aplicação, questões de desempenho, distribuição de tarefas e consumo de energia.

A especificação de requisitos é considerada o processo mais crítico para o desenvolvimento de software, pois impacta diretamente nas características do produto de software resultante. De acordo com (SOMMERVILLE, 2011), tendo como foco o sistema, os requisitos podem ser classificados em requisitos funcionais e não funcionais:

- Requisitos funcionais: funcionalidades que o sistema deve prover ou oferecer, como o sistema deve reagir perante determinadas entradas, como deve reagir perante determinadas situações, e de modo geral definindo o que o sistema deve ou não fazer.
- Requisitos não funcionais: diz respeito as restrições impostas pelo sistema a determinados serviços ou funções, como por exemplo, restrições temporais, restrições relativas ao processo de desenvolvimento e desempenho do produto. São as restrições que normalmente afetam e são associadas ao sistema como um todo.

Como destacado na literatura um grande número de problemas surgem devido à má especificação de requisitos, considerando a complexidade com o advento dos novos sistemas ciber físicos (RIBEIRO *et al.*, 2013) (KHAITAN; MCCALLEY, 2015), principalmente quando o impacto negativo da definição destes requisitos está relacionado a sistemas com restrições rígidas em tempo real, como por exemplo, em aplicações de sistemas embarcados na agricultura e na área automotiva (PATTANAIK; CHANDRASEKARAN, 2012) (KATERIS *et al.*, 2014). Nestes sistemas de tempo real o processo de especificação de requisitos requer uma análise particular e apurada dos requisitos não funcionais, cada vez mais nas fases iniciais do projeto, pois estes impactam de forma negativa em todo o sistema, afetando funções de segurança crítica (KOPETZ, 2011) (WEHRMEISTER; PEREIRA; RAMMIG, 2013).

Dentro deste contexto, o presente trabalho aborda a especificação de requisitos não funcionais relacionados a falhas em sistemas de tempo real, especificamente interferências e falhas transientes que afetam os protocolos de comunicação e geram efeitos globais no sistema, ocasionado degradação de desempenho, interferindo nos requisitos temporais da aplicação.

2.5.2 Modelagem Orientada a Aspectos

Em sistemas de tempo real embarcados e distribuídos a especificação de requisitos ligadas ao cumprimento de restrições temporais, distribuição de tarefas e relacionadas ao desempenho do sistema que está embarcado em chip são extremamente importantes para o sucesso do projeto. Essas características afetam diversos componentes do sistema de maneira não uniforme, fato que as torna difíceis de serem tratadas com metodologias tradicionais de desenvolvimento, como a orientação a objetos (FREITAS, 2007).

Diversos requisitos de projeto afetam não somente um componente isolado, mas de forma global os sistemas embarcados distribuídos. Requisitos ligados à consumo de energia, desempenho e tolerância a falhas afetam de forma transversal o sistema, fato que impulsionou a utilização de novos paradigmas de modelagem para especificação de requisitos, como a orientação a aspectos (AOM - *aspect-oriented modeling*). Estes requisitos normalmente eram tratados somente nas fases finais de projeto, sendo considerados a partir das fases de codificação, porém atualmente é cada vez mais importante considerar a modelagem de características transversais nas fases iniciais do projeto. Este paradigma propõe a separação de conceitos no tratamento dos requisitos não-funcionais contribuindo com a modularização do sistema (FREITAS, 2007) (WEHRMEISTER; PEREIRA; RAMMIG, 2013).

A modelagem orientada a aspectos pode ser considerada uma extensão da modelagem orientada a objetos. De acordo com (KIENZLE *et al.*, 2010) AOM foca na aplicação de técnicas baseadas em aspectos com o objetivo de modularizar os conceitos transversais, com diferentes notações de modelagem e níveis de abstração, e também, durante diferentes momentos durante o processo de desenvolvimento de software. Abordagens de AOM vem sendo aplicadas em diferentes contextos, baseados em modelos UML (*Unified Modeling Language*) (IQBAL *et al.*, 2012) (SALDIVAR *et al.*, 2015), adicionando aspectos para o desenvolvimento de sistemas mais seguros (NGUYEN; KLEIN; LE TRAON, 2014), e ainda trabalhando o reuso de uma modelagem de aspectos genéricos com base em atributos de dependabilidade (HOFFMANN *et al.*, 2014).

Sob o ponto de vista do desenvolvimento de software esse paradigma de modelagem surgiu a nível de programação com a o AspectJ, uma extensão java que implementou a orientação a aspectos (KICZALES *et al.*, 1997). Após isso, a aplicação da orientação a aspectos não se restringiu a programação, mas se estendeu para a fases iniciais do ciclo de desenvolvimento de software, como a engenharia de requisitos (WIMMER *et al.*, 2011) (VYAS; VISHWAKARMA; JHA, 2016).

As características transversais de requisitos não funcionais geram um problema relacionado ao reuso de componentes em sistemas de tempo real, pois nesta especificação de requisitos é difícil separar ou associar estes a apenas uma parte do sistema. Requisitos não funcionais estão normalmente espalhados pelo sistema, o que gera mudanças globais e dificulta a verificação do impacto da mudança. Assim, técnicas de engenharia de software como a orientação a aspectos permitem a especificação destes requisitos e a maior modularização do sistema de tempo real, contribuindo para o reuso de componentes.

A modularização do sistema também contribui para melhor manutenibilidade e produtividade, características que impulsionam o uso da modelagem orientada a aspectos em diferentes áreas de desenvolvimento de software, como a automação industrial. Em (VYATKIN, 2013) são destacadas as diferentes abordagens de engenharia de software aplicadas na automação industrial, destacando as ferramentas e técnicas de AOM, gera-

ção automática de código com base em modelos UML, engenharia dirigida por modelos (MDE – *model-driven engineering*), que indicam o crescimento e o foco de pesquisas no ciclo de vida e dependabilidade, mais do que em questões ligadas a performance.

O uso da orientação a aspectos cresce em conjunto com a MDE que emprega modelos como artefatos primários do desenvolvimento de software. O emprego da MDE em conjunto com a AOM tem ajudado os engenheiros a trabalhar com a crescente complexidade dos sistemas, propondo modelos que abstraem a complexidade de componentes de software, criando assim, artefatos reusáveis para funções específicas que aceleram o processo de desenvolvimento de software.

Como destacado os requisitos não funcionais afetam de forma global o sistema, sendo importante o tratamento destes requisitos para aumentar o reuso de componentes de software. Neste sentido, o ponto chave da AOM são os interesses transversais (*crosscutting concerns*) que tratam dos conceitos que não podem ser mapeados ou associados a um único módulo, mas sim estão espalhados em todo o sistema. Desta forma, requisitos não funcionais podem ser vistos como *crosscutting concerns*, devido ao fato de estes normalmente estarem vinculados a requisitos que afetam a funcionalidade de vários módulos do sistema (WEHRMEISTER; PEREIRA; RAMMIG, 2013) (AKKAYA *et al.*, 2016).

De acordo com (CLARKE; BANIASSAD, 2005) existem diferentes abordagens usadas na modelagem orientada a aspectos: Assimétrica, que separa aspectos das funcionalidades principais tratando-os de forma individual; e Simétrica, que trata os conceitos em um mesmo nível de hierarquia, onde aspectos e conceitos base tem a mesma importância.

As abordagens tradicionais baseadas em orientação a objetos (OO) não tratam de forma adequada os conceitos transversais, pois a forma de decomposição da OO é incapaz de encapsular requisitos transversais ocasionando o emaranhamento e dispersão destes requisitos (WEHRMEISTER, 2009). Desta forma, a orientação a aspectos representa uma importante forma de modularizar conceitos transversais, encapsulando em entidades individuais conceitos que afetam de forma global o sistema, como por exemplo, diferentes tipos de falhas e interferências, que é o foco deste trabalho. A presente pesquisa aplica os preceitos da modelagem orientada a aspectos seguindo a abordagem assimétrica, mas neste caso procurando separar os aspectos de diferentes tipos de falhas das funcionalidades principais do sistema, para agregar confiabilidade a protocolos de comunicação usados em situações críticas de controle.

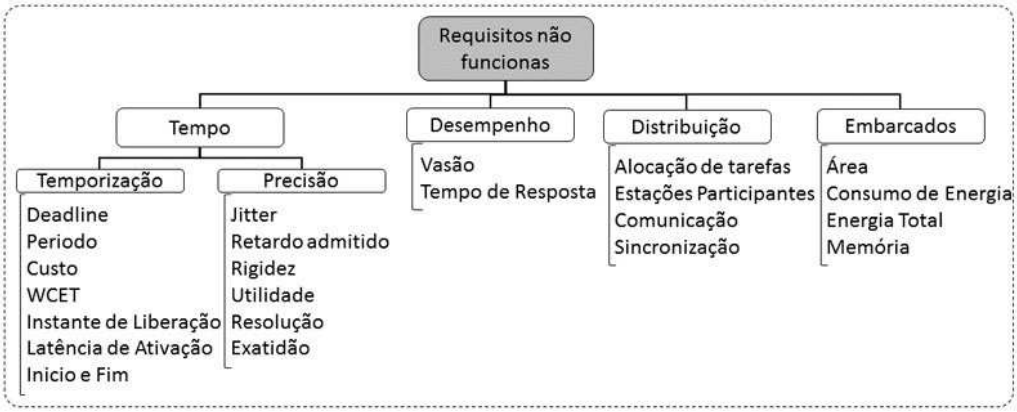
2.5.3 Especificação de requisitos com o RT-FRIDA framework

Com base no estudo sobre as lacunas inerentes ao tratamento de falhas em protocolos de comunicação, e também sobre a modelagem destas, foi estudada a pesquisa que propõe o Framework RT-FRIDA (*Real-Time From Requirements to Design using Aspects*) (FREITAS, 2007) (FREITAS *et al.*, 2007) relacionada à especificação de requisitos não funcionais em sistemas de tempo real embarcados e distribuídos - DERTS, que estende e

tem como base a pesquisa de (BERTAGNOLLI, 2009). Outro trabalho importante neste contexto é a proposta de (WEHRMEISTER, 2009) que visa integrar as fases de projeto de DERTS focando em aplicações de automação e usando técnicas de engenharia guiada por modelos (MDE) em conjunto com o projeto orientado a aspectos (AOD). Nesta pesquisa o autor propõe uma ferramenta de geração de código, que suporta a transição automática das fases iniciais de especificação para as fases seguintes de implementação.

Nestas pesquisas o propósito principal é o mapeamento de requisitos não funcionais considerados desde o princípio do projeto do sistema. O framework RT-FRIDA especifica requisitos não funcionais relacionados a tempo, desempenho, distribuição e embarcados, seguindo os preceitos da orientação a aspectos. A Figura 11 ilustra esta classificação de requisitos não funcionais.

Figura 11 – Classificação de RNF para Sistemas de Tempo Real.



Fonte: Adaptado de (FREITAS, 2007).

Essa classificação foi proposta para tratar os principais problemas inerentes ao projeto de sistemas de tempo real distribuídos. Estes requisitos são diretamente mapeados em propriedades que regem a execução de atividades em um DERTS. De acordo com (FREITAS, 2007) a descrição de cada um destes requisitos é dada da seguinte forma:

Tempo - Temporização:

- Deadline: limite temporal para a execução de uma atividade do sistema;
- Período: intervalo entre duas ativações sucessivas de uma atividade;
- Custo: tempo médio necessário para se executar uma atividade no sistema;
- WCET: tempo de execução do pior caso;
- Instante de Liberação: instante em que o sistema está pronto para executar;
- Latência de Ativação: corresponde ao tempo entre o instante de liberação e o instante no qual uma atividade começa a ser executada;

- Início e Fim: instantes de início e fim da execução de uma atividade.

Tempo - Precisão:

- Jitter: variação na medida de determinado requisito temporal;
- Retardo Admitido: tempo excedente máximo admitido para o início da execução de uma atividade do sistema;
- Rigidez: define a precisão de outros conceitos, como o de deadline por exemplo, que pode ser hard ou soft. Indica, as consequências do não atendimento do requisito temporal;
- Utilidade (Prazo de Validade): é associado à ideia de lapso temporal no qual o valor de um dado/variável é considerado válido. Como exemplo o dado lido por um sensor, onde este possui uma validade temporal que pode tornar o dado inutilizável;
- Resolução: identifica a granularidade de tempo mais fina na qual um elemento de temporização do sistema pode trabalhar.

Desempenho:

- Vazão: taxa na qual um recurso deve executar sua função, ou quantas chamadas por unidade de tempo um recurso deve ser capaz de tratar;
- Tempo de Resposta: representa o tempo necessário para que o sistema retorne uma resposta final que dependa da execução de atividades locais ou remotas.

Distribuição:

- Alocação de Tarefas: este requisito se relaciona à questão da distribuição e escalonabilidade das tarefas nas diferentes unidades de processamento (“hosts” ou estações participantes);
- Estações Participantes: constitui os requisitos de monitoramento de estações participantes do sistema. Este critério encontra-se relacionado às questões que envolvem a alocação de tarefas;
- Comunicação: este requisito engloba as restrições relativas à comunicação no sistema, tais como exigência de utilização de determinado barramento ou tecnologia especifica;
- Sincronização: definição das políticas e restrições de acesso a recursos do sistema.

Embarcados:

- Área: corresponde à restrição de área máxima (em silício ou placa de circuito impresso) que pode ser ocupada por determinado componente do sistema;

- Consumo de Potência: corresponde à restrição de consumo de potência associada a componentes do sistema;

- Energia Total: quanto de energia seria disponibilizado para o sistema. Este requisito trata de quanto deve durar a energia disponibilizada para o sistema;

- Memória: corresponde às restrições quanto ao uso de memória no sistema.

O framework RT-FRIDA usa templates e checklists para a especificação e descrição de requisitos não funcionais, sendo dividida em três fases principais:

1 - Identificação e especificação de requisitos

- identificação e especificação de requisitos funcionais, e

- identificação e especificação de requisitos não-funcionais.

2 - Mapeamento de requisitos em elementos de projeto

- Extrair do diagrama de casos de uso e dos templates de requisitos funcionais os conceitos e atributos responsáveis por compor a parte funcional do sistema;

- Realizar a extração de aspectos, que consiste na análise dos aspectos candidatos e decisão de que aspectos serão utilizados na fase de projeto para tratar cada requisito não-funcional levantado;

- Composição da informação levantada nos passos anteriores em uma tabela de mapeamento que relaciona os requisitos aos elementos de projeto (classes e aspectos) especificados para tratá-los.

3 - Projeto do sistema

- Utiliza-se primeiramente a tabela de mapeamento construída na segunda fase para povoar o diagrama de classes;

- Constrói-se então a hierarquia de classes do sistema;

- As classes são estereotipadas com elementos UML;

- Constrói-se um novo diagrama composto apenas pelas classes afetadas por requisitos não-funcionais e os respectivos aspectos que as afetam, usando o diagrama ACOD (*Aspect Crosscutting Overview Diagram*).

Sendo assim, destaca-se que a proposta do framework RT-FRIDA tem como objetivo principal elicitar claramente os requisitos não-funcionais referentes ao domínio de interesse mapeando o seu tratamento em aspectos, permitindo a representação destes aspectos em modelos UML adaptados. É enfatizado que o processo de mapeamento dos requisitos não funcionais deve fazer parte das fases inicias do projeto de DERTS, mantendo o elo entre requisitos e projeto, promovendo a rastreabilidade, o reuso e a manutenção dos sistemas. Tais características destacam a possibilidade de uso deste framework no contexto do presente trabalho, permitindo a especificação de requisitos relacionados a confiabilidade dos protocolos de comunicação usados em redes veiculares. A seguir a Seção 3 apresenta as pesquisas recentes que servem de base para o desenvolvimento da presente tese.

3 ESTADO DA ARTE

Este capítulo apresenta a revisão de trabalhos relacionados com a temática desta tese, dando ênfase a avanços e direcionamentos futuros, bem como também, às lacunas e oportunidades de pesquisa encontradas. Primeiramente, são apresentados trabalhos relacionados aos protocolos usados em redes intra-veiculares, baseadas no protocolo CAN e FlexRay, relacionando as típicas falhas que afetam e degradam o desempenho destas redes de comunicação. Na sequência são apresentados trabalhos relacionados ao uso da modelagem orientada a aspectos no contexto de controle e automação industrial, especificamente no que tange às aplicações críticas de tempo real, para em seguida verificar a sua aplicabilidade na modelagem de falhas em protocolos de comunicação. Por fim, são apresentados trabalhos sobre especificação de requisitos para sistemas embarcados de tempo real, onde para tal, são apresentadas as possibilidades de uso e extensão do framework RT-FRIDA.

3.1 Redes intra-veiculares e a suceptibilidade a falhas

3.1.1 Redes baseadas no protocolo CAN

Existem lacunas relacionadas à confiabilidade dos processos de comunicação em redes intra-veiculares, muitas destas relacionadas a mecanismos de diagnóstico de falhas que permitem a sobrevivência ou manutenção do sistema em operação. A possibilidade de modelar estas falhas e integrá-las nas fases iniciais de desenvolvimento dos sistemas embarcados também é um desafio constante.

Com a crescente complexidade das redes de comunicação e a grande quantidade de dispositivos interconectados, aspectos relacionados à confiabilidade ganham em importância. Nas pesquisas de (ATAIDE, 2010) e (ASSIS, 2011), são destacadas as demandas por desempenho e a crescente complexidade dos sistemas automotivos, bem como as possibilidades de uso do protocolo FTT-CAN, que permite a configuração de uma rede de tempo real com garantias temporais e flexibilidade. Em (ATAIDE, 2010) o autor apresenta uma discussão sobre os problemas relacionados ao escalonamento de mensagens no protocolo FT-CAN, que geram atrasos e aumento do jitter. Estes problemas afetam e

geram interferências nas tarefas, consequentemente prejudicando seus respectivos tempos de resposta e a confiabilidade do sistema. Já em (ASSIS, 2011), o autor destaca o uso de um padrão que agrega características de flexibilidade e reuso, chamado de AUTOSAR (*AUTomotive Open System ARchitecture*), em conjunto com um sistema de hot-plugging para a rede que agrega características de plug-and-play à rede automotiva. O trabalho não discute problemas específicos de confiabilidade, porém destaca que a adoção de padrões pode contribuir para o desenvolvimento de sistemas mais confiáveis, onde sistemas tolerantes a falhas podem ser reusados em diferentes sistemas de controle.

Em (PATTANAIK; CHANDRASEKARAN, 2012) é proposto um método de predição da ocorrência de falhas analisando-se sequências temporais de eventos no sistema, tais como tarefas de controle com restrições temporais, que são repetidos em determinada frequência e por um período determinado de tempo. A verificação de um evento é tratada analisando-se um histórico dos eventos, para assim, definir a probabilidade de falhas ocorrerem por meio de um sistema de votação, que faz a análise e decide se um mecanismo de redundância ou de correção será acionado. Essa decisão é baseada na proposta de um modelo de recuperação colaborativa (*Collaborative recovery model*) que correlaciona múltiplos mecanismos de recuperação com a verificação do tempo e dados dos eventos. Em um sistema com eventos heterogêneos, como um sistema automotivo, as taxas de sucesso ou falhas de eventos durante um período de tempo são usadas para definir o grau de confiabilidade do sistema.

Continuando a pesquisa anterior, em (PATTANAIK; CHANDRASEKARAN, 2013) os autores reportam um problema no protocolo CAN relacionado ao processamento dos bits do final do quadro de mensagem (EOF – *end of frame*), pois os nós CAN recebedores validam uma mensagem processando uma sequência de bits até o sexto bit do campo EOF. Mas os nós CAN remetentes só validam a transmissão no último bit (sétimo bit) do EOF, assim, se um subconjunto de nós recebedores detectar um erro no sexto bit, estes rejeitarão a mensagem e iniciarão um processo de retransmissão, sinalizando erro no sétimo bit. Todos os outros nós podem já ter validado esta mensagem ocasionando erros de entrega inconsistente e mensagens duplicadas. Os autores destacam que diversas técnicas em software são aplicadas para minimizar estes problemas, e que estas geram aumento no uso da largura de banda do canal. Assim, o trabalho faz uma análise comparativa do uso de ECUs normais com FTUs (*fault tolerant units*) na resolução de erros típicos como o citado anteriormente, e princípios de confiabilidade são discutidos em termos de manutenibilidade dos sistemas automotivos, destacando os tipos de abordagens reativas, proativas, preditivas que impulsionam os aspectos de confiabilidade.

Em (MARQUES *et al.*, 2013) os autores destacam e focam nas falhas transientes que afetam a rede CAN, abordando um mecanismo de escalonamento em tempo real (*on-line traffic scheduling*) em conjunto com servidores de escalonamento, os quais tem por objetivo definir o momento de retransmissão de uma mensagem afetada por erros. Essa

proposta é implementada junto ao protocolo FTT-CAN, pois as mensagens são retransmitidas de forma flexível, em intervalos distintos que respeitam as restrições temporais das tarefas, e acionadas após a ocorrência de falhas (*event-triggered*). Essa abordagem busca reduzir o impacto das retransmissões geradas por falhas que sobrecarregam o barramento, mas mesmo assim, os autores destacam que até mesmo as retransmissões podem ser afetadas por falhas, necessitando a aplicação de outros algoritmos de escalonamento em conjunto, para minimizar o efeito das retransmissões sem afetar as restrições temporais. Desta forma, dando continuidade a esta pesquisa, os mesmos autores avaliam em (MARQUES *et al.*, 2014) diferentes políticas de escalonamento, visando reduzir o impacto das retransmissões, ainda no contexto da ocorrência de falhas transientes. Neste estudo o processo de avaliação dos algoritmos de escalonamento é realizado apenas por simulação, destacando a interação do servidor responsável por retransmitir a mensagem, onde este tem prioridade e se comunica com o escalonador principal, a fim de incluir uma mensagem afetada por erro, antes do deadline pré-definido para a comunicação. Os resultados mostraram que as políticas de escalonamento impactam para a efetividade do servidor de escalonamento proposto pelos autores no trabalho anterior.

Neste mesmo contexto, o crescente empenho em agregar maior segurança em aplicações de tempo real baseadas no protocolo CAN, motiva outros métodos de verificação de falhas baseado na detecção e injeção de falhas no sistema. No trabalho de (GESSNER *et al.*, 2014) é apresentado o projeto e implementação do sfiCAN, um hardware para injeção de falhas no protocolo CAN, que possibilita diversos testes para verificação de uma rede baseada na topologia estrela. O trabalho possui características de flexibilidade, com a configuração remota de injeção de falhas e o retorno de informações do comportamento dos nós da rede. O hardware projetado injeta as falhas no protocolo por meio de bits de erros para representar ou simular situações de falhas, não tendo uma ação física ou interferência sobre o hardware, e normalmente após o sistema de controle estar pronto e em operação.

Outra abordagem de injeção e análise de falhas é tratada em (NAKAMURA *et al.*, 2015) onde é proposta uma arquitetura híbrida para o controle de erros baseado em repetição automática de mensagens *(automatic repeat request - ARQ*) em redes CAN submetidas a altas interferências eletromagnéticas. O protocolo CAN padrão já permite o controle de erro ARQ, mas com o aumento da complexidade dos veículos modernos e a adoção de carros elétricos, o protocolo não trata as situações de alta interferência eletromagnética. Os autores destacam que em veículos modernos como os carros elétricos e carros híbridos os ruídos eletromagnéticos causados pelos conversores DC podem ser altos, na ordem de centenas de volts, degradando de forma significativa o protocolo CAN. Para tal, neste trabalho é estruturado um experimento para a injeção de falhas em uma rede CAN real, para simular estas situações de falha e testar a arquitetura híbrida. Essa arquitetura é baseada em três estados, ARQ (repetição automática), FEC (correção automática) e

HALT (retirada do nó), onde é detectado o nível de ruído e associado ao modo apropriado para otimizar as retransmissões de mensagens. Os resultados conduzidos em redes reais, mostram que a rede CAN alternou entre os modos esperados, destacando a redução do volume de retransmissões geradas por cada situação de testes. É importante destacar que a abordagem novamente é baseada no número de retransmissões de mensagens afetada por ruídos e falhas, após estes ocorrerem, e que efetivamente as retransmissões degradam o desempenho dos processos de controle que usam o protocolo de comunicação.

Outro tipo de falha que afeta de forma significativa o protocolo CAN são os transientes elétricos rápidos. No trabalho de (FONTANA; HUBING, 2015) é apresentado um estudo sobre a susceptibilidade do protocolo CAN aos transientes elétricos rápidos (*electrical fast transientes - EFT*), especificamente nos sinais das portas dos transceptores. É desenvolvido um circuito para a injeção de falhas do tipo EFT na rede CAN, sendo este circuito montado de acordo com as especificações da norma IEC TS 62228 (IEC-TS-62228, 2007). Nos veículos os circuitos de proteção dos transceptores são afetados por um evento EFT, que pode ser representado por um pulso ou por rajadas de tensão de amplitude variada, podendo chegar a 70 volts. Esses pulsos podem ocasionar erros na geração dos sinais do transceptor CAN, afetando o cálculo diferencial de sinal, ocasionando erros de identificação dos sinais CAN-High e CAN-Low. Experimentos foram realizados para verificar a susceptibilidade dos transceptores CAN e um mecanismo para tentar reduzir os efeitos dos EFT é proposto. Todavia, o trabalho não faz nenhuma análise dos efeitos dos transientes elétricos nas restrições temporais de tarefas de controle específicas, ou no desempenho de uma rede CAN em operação.

Os transientes elétricos rápidos podem ser gerados por diferentes fontes, e um estudo extensivo sobre este tipo de falha se faz necessário. Em (PANNILA; EDIRISINGHE, 2014) é apresentado um estudo por meio de uma série de experimentos sobre os efeitos dos transientes gerados no acionamento de diversos sistemas elétricos automotivos, especificamente os efeitos de comutação de energia (*power switching transients*). Os dados de transientes medidos são usados como parâmetros para verificar a sensibilidade e imunidade de vários dispositivos eletrônicos de um automóvel aos transientes conduzidos. Após a montagem de um chassi veicular real em laboratório, uma série de acionamentos de dispositivos eletrônicos é realizada e dados estatísticos de transientes gerados são gravados, especificamente sobre o acionamento de três operações principais (ignição, ar-Condicionado e luzes). Os dados são apresentados especificando o tempo de duração do transiente e o pico de amplitude de tensão em volts. Apesar de diversas proteções a ruídos e interferências já serem tratadas no processo de fabricação destes equipamentos, é importante destacar que este tipo de transiente pode ser ocasionado por diferentes motivos, incluindo equipamentos de terceiros, falhas elétricas em componentes ou até mesmo ataques maliciosos na rede. A seguir a Figura 12 apresenta uma parte das medições realizadas nesta pesquisa.

Figura 12 – Parte das medições dos transientes elétricos realizadas durante experimentos.

Parameter	Operation	Number Of observations	Switching Direction	Range		Average
				Min	Max	
Burst Duration	Ignition	244	OFF-ON	47 µs	1.2 ms	224 µs
			ON-OFF	80 µs	938 µs	486 µs
	AC	158	OFF-ON	0.33µs	675 µs	423 µs
			ON-OFF	0.68µs	1.4 ms	112 µs
	Headlight	100	OFF-ON	35 µs	198 µs	159 µs
	Total	**502**	ON-OFF	16 µs	335 µs	108 µs

Fonte: (PANNILA; EDIRISINGHE, 2014).

Todos esses dados são importantes e servem de prova dos efeitos danosos dos transientes elétricos em uma rede automotiva, mas este volume de dados gerado, não é associado e analisado em relação aos possíveis efeitos de degradação de desempenho que podem causar na rede (analisados e medidos por exemplo, por efeitos na vazao, jitter e perda de pacotes da rede) e o seu impacto em restrições temporais de tarefas críticas que operam sob estes protocolos.

Com a perspectiva de verificar o impacto dos transientes elétricos no desempenho da rede CAN, em (ROQUE *et al.*, 2016) é apresentado um estudo sobre o impacto dos EFTs em uma rede CAN experimental, realizando a comunicação entre três nós de uma malha de controle (um nó sensor, um nó atuador e um nó controlador). Baseado no método de injeção de falhas proposto em (FONTANA; HUBING, 2015) e nas especificações de falhas medidas no trabalho de (PANNILA; EDIRISINGHE, 2014), este trabalho realiza análise do impacto deste tipo de falha no atraso gerado entre os laços de controle, tendo como métrica base a variação no atraso entre mensagens - jitter, para assim verificar a influência do EFT no desempenho da rede CAN. O método de teste aplicado no trabalho mostrou que os efeitos das falhas do tipo EFT vão além das interferências físicas geradas nos transceptores, gerando picos de atraso nos laços de controle com maior frequência, resultando assim, na degradação do desempenho da rede CAN e podendo afetar as restrições temporais de tarefas de controle específicas.

Neste contexto, verifica-se que algumas pesquisas preocupam-se em destacar os efeitos de falhas transientes, independente da fonte, em tarefas de controle específicas em redes veiculares. A pesquisa desenvolvida em (HUANG *et al.*, 2016) destaca este ponto, pois, um estudo sobre o efeito de falhas transientes em um sistema de controle de frenagem veicular é realizado (*brake-by-wire system* - BBW). Nesta pesquisa é proposta uma arquitetura hierárquica para tratar falhas transientes que afetam o sistema BBW, incluindo um algoritmo de monitoramento dos nós de controle e realocação de tarefas de acordo com a presença de falhas. A análise das falhas transientes é tratada de acordo com características de propagação e de atraso gerado nos laços de controle. Um experimento

é conduzido para avaliar o desempenho do sistema BBW com a comunicação baseada no protocolo TT-CAN e também com a simulação da comunicação baseada na linguagem AADL (*Architecture Analysis and Design Language*), uma linguagem de descrição de arquiteturas de sistemas embarcados críticos de tempo real padronizada pela SAE (*Society of Automotive Engineers*) e muito usada na área automotiva. É importante destacar que a pesquisa não utiliza mecanismos de injeção falhas reais e não realiza experimentos em situações reais de funcionamento, tendo as falhas simuladas apenas desligando e ligando os nós controladores para destacar que um nó saiu de operação. Outro ponto importante é a característica de uma solução voltada apenas para este sistema de frenagem, não tendo características de modelagem que permitam ser reusadas em outros sistemas de controle, caracterizando mais uma abordagem reativa a um tipo de falha específica.

Seguindo as abordagens de análise de falhas transientes e seu impacto em sistemas de controle veiculares, em (ROQUE *et al.*, 2017), uma pesquisa mais detalhada do impacto das falhas transientes do tipo EFT é realizada em uma rede CAN experimental, tendo como foco tarefas de controle críticas, com restrições temporais definidas. Neste trabalho o mesmo método de injeção de falhas é aplicado, porém agora em um estudo de caso de uma rede CAN para controle um sistema veicular de suspensão ativa, baseado no modelo de planta de suspensão *quarter-car-model* proposto em (MICHELIN, 2014). Para a realização dos experimentos o modelo foi implementado e simulado em uma ferramenta de desenvolvimento para aplicações veiculares amplamente utilizada no setor, a ferramenta VECTOR CANoe, da empresa alemã Vector Informatik (VECTOR INFORMATIK, 2018). Os resultados apresentados na pesquisa destacam o impacto das falhas transientes na degradação de desempenho da rede CAN, afetando assim, as tarefas de controle críticas de tempo real. Como esta pesquisa é um dos resultados e contribuições da presente tese de doutorado, um detalhamento maior de tal estudo é apresentado no Capítulo 4.

Outro tipo importante de falha que afeta os protocolos de comunicação são as interferências eletromagnéticas – EMI. Pesquisas recentes abordam técnicas para detectar e analisar os efeitos destas interferências, como o trabalho de (JUAN; JEONG; KIM, 2016) que analisa o efeito de interferências eletromagnéticas em uma rede CAN, principalmente os efeitos em sistemas de assistentes de direção avançados (ADAS – *Advanced Driver Assistance System*). Nesse contexto é proposto um método de entrelaçamento que minimiza a necessidade de bit stuff devido a erros causados por EMI, reduzindo a probabilidade de erros e retransmissões com o objetivo de aumentar a eficiência do sistema. Outras pesquisas como a de (ANKARSON *et al.*, 2015) e (NA; DAO; CHO, 2016) destacam os efeitos de EMI em sistemas de entretenimento veiculares e sistemas baseados em comunicação sem fio integrados as tecnologias veiculares. Nestas pesquisas os autores analisam os efeitos de EMI em sistemas LTE (*Long Term Evolution*) que fazem parte de modernas redes de sistemas de entretenimento veiculares (IVIN – *In-vehicle infotainment*

network) (TUOHY *et al.*, 2015). São destacados os efeitos na degradação de desempenho da comunicação destes sistemas, que são praticamente inevitáveis devido ao aumento dos dispositivos eletrônicos usados nos veículos, da conectividade com outros dispositivos e da possibilidade de distúrbios que estes podem gerar. Algumas técnicas são abordadas, como salto de canais, controle de potência e cooperação com nós vizinhos, visando mitigar os problemas de desempenho analisados e constatados após a injeção de interferências na rede de comunicação.

Em (SHIRAI; SHIMIZU, 2019) também é enfatizado o problema de interferências do tipo EMI, nesse caso gerado por conversores buck (redutor de tensão/conversor CC/CC), na rede CAN. O estudo aborda as falhas de comunicação geradas por estes conversores na rede CAN, chamados de ruídos em modo comum - CM. Para tratar esse problema de erro de transmissão no CAN, os autores propõem um método de controle do conversor de tensão para não coincidir o ruído CM com o tempo de amostragem da rede CAN. Testes experimentais mostraram a aplicabilidade do método na redução dos efeitos deste ruído. Todavia, os autores destacam que a controlabilidade do conversor não é mantida e aprimoramentos no método são necessários no futuro.

3.1.2 Redes baseadas no protocolo FlexRay

Assim como no protocolo CAN, pesquisas recentes também abordam a análise e susceptibilidade a falhas de outros protocolos de comunicação, como o FlexRay (FLEXRAY CONSORTIUM, 2010). Em (SEDAGHAT; MIREMADI, 2010) é destacado que o FlexRay é usado em aplicações de segurança crítica, destacando os efeitos de falhas transientes nos controladores. Neste protocolo quando falhas são injetadas, estas resultam em um ou mais erros, como violação de restrições temporais, congelamento de nós, erros de sincronização de tarefas, perda de pacotes, entre outros. Nesta pesquisa, para estudar e verificar se as falhas injetadas são detectadas, um experimento com quatro nós foi modelado em Verilog HDL (linguagem de descrição de hardware). Após a injeção de 135.600 falhas transientes resultados mostraram que apenas 9.342 falhas foram detectadas (6.9 %), e que a taxa de erros de sincronização também aumentou chegando a 70%, dados que mostram a importância de estudos que venham a mitigar o impacto destas falhas no desempenho da rede.

Em (ZENG; KHALID; CHOWDHURY, 2015) são destacadas algumas deficiências do protocolo FlexRay relacionadas à detecção e diagnóstico de falhas. Alguns problemas típicos que ocorrem são a inconsistência de dados e o atraso de reação nas notificações de falhas, enfatizando que o FlexRay não provê mecanismos adequados de reconhecimento e notificação de falhas. Este tipo de situação faz com que os nós recebedores descartem mensagens sem informar a causa aos nós que enviaram a mensagem.

Outros problemas são destacados em (MARQUES *et al.*, 2014) que tratam dos efeitos de falhas transientes na tarefas de controle que operam sob o protocolo FlexRay. Os au-

tores propõem um mecanismo que usa redundância temporal para permitir que os nós se recuperem de erros causados por falhas transientes em mensagens time-triggered, usando para tal, a retransmissão de mensagens por meio do segmento dinâmico do protocolo FlexRay. O propósito do trabalho é agregar no FlexRay o mesmo princípio usado no protocolo FTT-CAN que consiste em escalonar mensagens dinamicamente, fazendo retransmissões baseadas em eventos. Este tipo de abordagem objetiva reduzir o uso de largura de banda do canal, minimizar os erros e limitar o tempo necessário para retransmitir mensagens.

A migração de tarefas para outros nós é sugerida em (HUANG; LEU; CHEN, 2015) com utilização da replicação de hardware onde um nó extra é usado como backup para nós falhos em uma rede FlexRay. Nessa pesquisa é proposto um framework de tolerância a falhas para aplicações de segurança crítica em redes FlexRay, com o objetivo de demonstrar como nós de backup podem ser usados, espelhando e migrando tarefas para permitir a manutenção da operação de um sistema de controle. É destacado que métodos que analisam a confiabilidade de sistemas FlexRay críticos nas fases iniciais de projeto, contribuem para o aumento da segurança destes sistemas, além de reduzir significativamente o custo e o tempo de re-design inerente a soluções desenvolvidas depois das falhas afetarem o sistema.

Aspectos relacionados à confiabilidade em protocolos para aplicações veiculares são discutidos em (DO SOUTO; PORTUGAL; VASQUES, 2016), onde os autores apresentam uma abordagem baseada em políticas de reconhecimento e retransmissão para analisar a confiabilidade de protocolos de broadcast implementados na última camada do protocolo FlexRay. Protocolos de broadcast confiável (*RB Protocols*) utilizam técnicas de inundação de dados por meio da disseminação de mensagens em toda a rede, permitindo um maior raio de cobertura e redução de atrasos em retransmissões. Em cenários de baixa e média densidade de mensagens esses protocolos garantes que um nó selecionado vai receber a mensagem, porem em cenários de alta densidade ocorre o excesso de mensagens desnecessárias que podem afetar o desempenho da rede. Na pesquisa é avaliado o impacto de políticas de reconhecimento de padrões de falhas que afetam e degradam a confiabilidade destes protocolos. Como metodologia de análise foi desenvolvido um modelo analítico baseado em cadeias de Markov de tempo discreto, que considera um conjunto de falhas (permanentes, transitórias, omissivas e assimétricas) que afetam os nós e seus efeitos na execução do protocolo. O principal ponto a destacar nesta pesquisa é que novamente a política verifica os erros de bits gerados no protocolo, associando aos tipos de falhas, mas efetivamente não considera a fonte das falhas, aplicando novamente técnicas que geram um grande número de retransmissão de mensagens, favorecido agora pelo excelente desempenho do protocolo FlexRay.

Outra abordagem que visa a confiabilidade da comunicação no protocolo FlexRay é apresentada em (SAHA; ROY; RAMESH, 2016), onde é feita uma análise e verificação formal de algoritmos de inicialização, que são comparados também com o protocolo TT-

CAN. Em arquiteturas time-triggered os nós usam a estratégia TDMA para acesso ao barramento, e estas fazem uso de um algoritmo de inicialização para alcançar um modo de operação estável após a sua ativação ou para se recuperar de falhas. Além disso cada nó possui um clock local que deve ser sincronizado com os demais para permitir a divisão igual dos slots de tempo na rede, assim, este processo de sincronização é agravado quando falhas transientes afetam a rede e consequentemente estes algoritmos de inicialização. Em aplicações de segurança crítica esses algoritmos demandam rigorosas etapas de verificação, que devem fazer parte das fases de modelagem e projeto do sistema. Com relação a modelagem, os autores destacam que a experiência dos engenheiros contribui muito para definição de requisitos relacionados aos tipos de falhas que afetam o protocolo, porém, apesar de ser muito importante agregar esta modelagem nas fases iniciais, é muito difícil de prever todas as possibilidades de falhas. Desta forma, esta questão é tratada normalmente especificando e modelando hipóteses de falhas que farão parte do sistema tolerante a falhas. Como resultado é destacado que uma hipótese de falha é de interesse global do sistema e deve ser avaliada no projeto como um todo.

A pesquisa de (LIU; BAI; ZHEN, 2017) destaca que o protocolo FlexRay não possui mecanismos para mitigar falhas transientes e realizar o tratamento dos erros resultantes, assim, o trabalho propõe um mecanismo de retransmissão de mensagens durante a ocorrência destas falhas. O mecanismo chamado de PRTM visa reduzir a probabilidade de ocorrência de falhas em sistemas críticos, com a aplicação de um método estatístico em conjunto com uma análise temporal para definir o ciclo em que as mensagens serão retransmitidas. Os autores destacam que a proposta reduz a probabilidade de falhas em sistemas críticos, mas com um pequeno custo de projeto e desperdício de largura de banda. Com abordagem similar, em (LEE *et al.*, 2018) os autores também enfatizam os problemas relacionados a falhas transientes que podem afetar o protocolo FlexRay e as tarefas críticas que normalmente dependem dele. Na pesquisa é proposto um método para reduzir os efeitos negativos das falhas transientes usando novamente um mecanismo de retransmissão de mensagens, em conjunto com tabelas de referência LUT (*lookup tables*) para melhorar o escalonamento em tempo de execução, gerando melhorias de até 70,76% com uma redução de custo de 13.33% em relação a outros métodos.

Na revisão apresentada por (HUANG *et al.*, 2019) os autores destacam os problemas nos sistemas SbW (*Steer-by-wire systems*) assistentes de direção, tipicamente dependentes de protocolos com redundância de canais e maior largura de banda como o FlexRay. Os autores destacam que falhas podem causar atrasos e degradação de performance em sistemas SbW, gerando instabilidade e redução da confiabilidade destes sistemas. Neste sentido, a pesquisa apresenta uma visão geral de técnicas de tolerância a falhas que vem sendo aplicadas em protocolos de comunicação visando aumentar a confiabilidade da comunicação entre ECUs em redes intra-veiculares, enfatizando principalmente técnicas de detecção e isolamento de falhas - FDI.

3.1.3 Pesquisas e análises com o emergente protocolo CAN-FD

Devido à crescente demanda por maior largura de banda de comunicação em redes intra-veiculares e à necessidade de atender aos requisitos dos modernos sistemas de controle, um novo protocolo foi recentemente desenvolvido pela BOSCH, o CAN com taxa de dados flexível (CAN-FD) (HARTWICH, 2012). Como já detalhado na Seção 2, o CAN-FD mantem os mecanismos de arbitragem e o padrão clássico de mensagens CAN, permitindo o aumento na velocidade de transmissão dos outros campos de mensagens e também a quantidade de dados transmitidos. No entanto, nenhum desses recursos desenvolvidos são garantidos se houver interferência ou perturbações no protocolo de comunicação.

Como o protocolo é relativamente novo e sua adoção ainda é dúvida na indústria, algumas trabalhos mostram a performance em substituição ao tradicional CAN, porém, por enquanto sem considerar cenários críticos e de tolerância a falhas. Em (NGUYEN; CHEON; JEON, 2014) uma avaliação de desempenho do CAN-FD é desenvolvida focada na reprogramação de ECUs com pacotes de dados de 8 bytes, mesmo que o CAN-FD suporte comunicações de 12, 16, 20, 24, 46 e 64 bytes. Resultados experimentais mostraram que, ao trabalhar com grandes quantidades de dados em ECUs, o CAN FD apresenta um desempenho significativamente melhor que o CAN, com campo de dados de 64 bytes o tempo de reprogramação foi reduzido em 68% e com 8 bytes foi reduzido em 89%. Com o objetivo de mostrar que o CAN-FD pode atender aos requisitos temporais de aplicações de tempo real, em (TENRUH *et al.*, 2015) é apresentada uma análise de desempenho do CAN FD com um conjunto de mensagens baseado no Benchmark SAE, composto por mensagens periódicas e eventuais (*time-triggered e event-triggered*). Para realizar a análise, modelos de sistema CAN e CAN-FD foram desenvolvidos em MATLAB. Os resultados avaliaram os atrasos de mensagens e a utilização de barramento usando ambos os modelos, e mostraram que o protocolo CAN-FD fornece alto desempenho atendendo os requisitos de sistemas de controle em tempo real.

Outros estudos vem sendo apresentados com o objetivo de avaliar a confiabilidade e o desempenho do protocolo CAN-FD sob diferentes aspectos de segurança, seja segurança de dados ou segurança funcional. Em (BORTH, 2016) foi apresentado um estudo e avaliação de desempenho da utilização do protocolo CAN-FD em substituição ao CAN tradicional. O sistema avaliado foi uma comunicação CAN em duas redes (de chassi e carroceria) de um caminhão fora-de-estrada, utilizado na indústria de mineração. Essas redes utilizam mensagens formatadas pelo protocolo de aplicação SAE J1939. Resultados do estudo mostraram uma redução significativa das taxas de ocupação e nos atrasos entre mensagens para as redes de comunicação que utilizaram o padrão CAN FD. Em (WOO *et al.*, 2016) uma arquitetura de segurança é proposta para o projeto de veículos baseados no protocolo CAN-FD, que inclui recursos para verificação da integridade de dados e segurança automotiva baseado também na ISO-26262. O estudo apresenta o

projeto e gerenciamento de chaves de criptografia de dados e protocolos de autenticação para o CAN-FD. Uma análise foi conduzida para verificar o desempenho da arquitetura de segurança baseada em modelo de ataque a rede.

Em (AN; JEON, 2017) é proposto o estudo de redes mistas CAN e CAN-FD, para tal os autores apresentam um sistema de gateway com um método de roteamento. O trabalho analisa o tempo de transmissão e o tempo de processamento do gateway para cada mensagem, com cargas úteis diferentes e de acordo com o estado da rede CAN-FD. Assim, uma memória intermediária foi alocada para transmitir os dados da rede CAN-FD, e assim evitar a falta de dados. Testes foram realizadas na plataforma CANoe (Vector Informatik GmbH) e mostram que durante as transmissões nenhum dado entre as duas redes foi perdido. Na pesquisa de (DE ANDRADE *et al.*, 2018) os autores focam na análise de desempenho, realizando comparações de desempenho analítico e experimental do CAN-FD e o CAN tradicional. Na pesquisa são consideradas mensagens obtidas por engenharia reversa de um sistema real baseado em CAN, com exemplos de mensagens de interferência de alta prioridade adicionais para teste de estresse do sistema com cargas de barramento diferentes. Assim como nas outras pesquisas, com grandes volumes de dados os resultados experimentais validam as vantagens de desempenho do CAN-FD sobre CAN.

Outra abordagem apresentada recentemente foca novamente nos problemas relacionados a vulnerabilidades do protocolo CAN-FD, a ataques maliciosos e interferências geradas intencionalmente. A pesquisa de (AGRAWAL *et al.*, 2019) enfatiza uma nova arquitetura de autenticação para a comunicação entre ECUs por meio da proposta de um gateway de ECU (GECU). Os resultados experimentais apresentados demonstram que o uso do GECU com um padrão de criptografia proporciona melhor desempenho do que a aplicação de primitivas individuais para a criptografia de dados de ECUS, contribuindo para a verificação da integridade das informações.

Recente estudo destaca a preocupação com as garantias temporais para aplicações críticas usando o CAN-FD. Na pesquisa de (LIM *et al.*, 2019), os autores destacam que as alterações realizadas no protocolo CAN-FD, para prover maior flexibilidade no campo de dados, não só leva a problemas de maior susceptibilidade a interferências eletromagnéticas, como também a problemas de distorção de sinais ("*ringing effect*"), devido ao curto tempo de bit que torna o protocolo mais sensível a assimetrias no atraso de propagação na rede. Esta assimetria causa variação no tempo de bit e consequentemente limita a taxa de transferência (bit rate). Assim, nesta pesquisa é proposto uma análise baseada em uma modelagem deste efeito que gera distorção de sinais, considerando as características físicas de um transceptor CAN. Essa análise permite a verificação do desempenho da comunicação, estimando por simulação a ocorrência deste problema, podendo assim ajustar parâmetros do barramento físico para um projeto específico.

3.1.4 Problemas e oportunidades de pesquisa encontradas

Protocolo CAN:

Muitos dos trabalhos relacionados a falhas no protocolo CAN não abordam a relação de causa ou origem das falhas, mesmo em testes, sendo considerados mecanismos reativos. Esse tipo de abordagem tem demandado maior esforço de projeto, pois tratar as falhas após o seu registro gera incertezas com relação ao desempenho dos futuros sistemas de controle. A literatura recente destaca que ainda existem lacunas com relação a métodos para teste e modelagem de falhas (MO *et al.*, 2017), mesmo em fases iniciais de projeto (LU *et al.*, 2018), para aprender com as mesmas e mitigar perdas de desempenho, destacando a carência da aplicabilidade destas abordagens nas redes e protocolos de comunicação veiculares. Normalmente o tratamento é em nível sistêmico, com soluções algorítmicas que não consideram o comportamento dos sinais do protocolo de comunicação sob falhas ou efeitos de distúrbios internos ou externos a rede.

Pesquisas como as de (ATAIDE, 2010) e (ASSIS, 2011) destacam o desenvolvimento e a adoção de protocolos que aumentam a flexibilidade das tarefas de controle na rede distribuída, adotando ainda, padrões de desenvolvimento da indústria que agregam possibilidades de reuso. Porém estes deixam lacunas relacionadas as possibilidades de integração de mecanismos tolerantes a falhas, de modelagem e métodos de testes. Métodos estes, que possam ser reutilizados para agregar maior confiabilidade e reduzir problemas recorrentes relacionados aos atrasos gerados pelos escalonadores na retransmissão de mensagens na rede.

Abordagens especificas são tratadas nos trabalhos de (MARQUES *et al.*, 2013) e (MARQUES *et al.*, 2014), que são duas abordagens centradas em mecanismos em nível de software, tratando apenas de redundância temporal, avaliadas usando simulação, e procurando minimizar o problema recorrente das retransmissões de mensagens. De acordo com os testes, as abordagens se mostraram eficientes, mas não tratam sob nenhum aspecto a correlação com a fonte das falhas ou monitoramento de possíveis fontes de falhas, sendo uma abordagem totalmente reativa, com ações realizadas após as falhas e a degradação de desempenho ocorrerem.

Assim, diversos trabalhos sobre injeção de falhas no barramento de comunicação surgem (GESSNER *et al.*, 2014), (PANNILA; EDIRISINGHE, 2014), (NAKAMURA *et al.*, 2015), (FONTANA; HUBING, 2015), (ANKARSON *et al.*, 2015), (HUANG *et al.*, 2016), (JUAN; JEONG; KIM, 2016), e (NA; DAO; CHO, 2016), os quais procuram de diferentes formas analisar o impacto e susceptibilidade dos protocolos a diferentes falhas, incluindo EFT e EMI. Nesta mesma tendência, a abordagem de (SHIRAI; SHIMIZU, 2019) propõe um método para um problema específico de EMI, os ruídos gerados por conversores de tensão, porém, a abordagem é pontual, destacada pelos próprios autores que precisa ser melhorada, não contemplando uma metodologia de solução replicável e capaz de ser inserida em novos projetos.

Foi observado que as pesquisas destacam o problema relacionado a uma modelagem mais precisa e que deve ser realizada cada vez mais cedo, com as transições entre modos de operação e controle podendo ser otimizadas baseadas, por exemplo, em análises probabilísticas, preditivas e com especificação de requisitos para tratamento de falhas. Para corroborar com estas análises, os experimentos realizados em (ROQUE *et al.*, 2016) e (ROQUE *et al.*, 2017) contribuem e destacam os efeitos das falhas transientes no desempenho de tarefas críticas de controle, que dependem de protocolos de comunicação como o CAN, direcionando para as lacunas de pesquisa com foco em métodos de testes, obtenção de informações de falhas e anomalias na rede de comunicação, especificação de requisitos a cerca das mesmas e a verificação da integração destas análises nas fases iniciais de projeto.

Protocolo FLEXRAY:

Assim como nos trabalhos citados e relacionados ao protocolo CAN, diversas pesquisas recentes destacam que ainda existem lacunas de pesquisa relacionadas a confiabilidade em outro importante protocolo, o FlexRay, ainda que este tenha sido desenvolvido justamente para prover maior confiabilidade e maior largura de banda nos processos de comunicação. Tendo como ponto negativo o custo, o protocolo FlexRay ainda não é consenso nas aplicações automotivas, e pesquisas recentes destacam que assim como o protocolo CAN, este também é afetado por diferentes tipos de falhas, especificamente falhas transientes.

As pesquisas de (SEDAGHAT; MIREMADI, 2010), (ZENG; KHALID; CHOWDHURY, 2015), (HUANG; LEU; CHEN, 2015), (MARQUES *et al.*, 2014) e (DO SOUTO; PORTUGAL; VASQUES, 2016) destacam que as falhas transientes causam perda de pacotes e degradam o desempenho da rede, principalmente afetando as tarefas mais críticas que possuem restrições temporais rígidas, tendo ainda, a política comum da retransmissão de mensagens. Uma abordagem importante citada nos trabalhos que se mostra eficiente e minimiza o uso da largura de banda do canal é a retransmissão de mensagens orientada a eventos incluída no segmento dinâmico do protocolo FlexRay. A inundação de dados também é tratada, porém os próprios autores citam que também sobrecarrega o canal gerando atrasos, tendo como ponto importante, o fato destes trabalhos não relacionarem o impacto de atrasos e da grande quantidade de retransmissões em tarefas de controle críticas de tempo real.

Muitas destas falhas podem ser especificadas e modeladas previamente como destacado em (SAHA; ROY; RAMESH, 2016), mas a maioria das abordagens tratam de políticas sem considerar o ambiente e a origem das falhas, também não levando em conta o reuso de técnicas para outros sistemas de controle veiculares. Falhas em protocolos de comunicação geralmente tem efeitos globais ou transversais, afetando toda a rede. Neste sentido, a próxima seção dá atenção a trabalhos que abordam a possibilidade de mapeamento e modelagem de falhas em protocolos, servindo de alternativa as propostas

anteriores e com a vistas a inserir esta atividade cada vez mais cedo no projeto de sistemas de controle de tempo real, especificamente na área automotiva.

As pesquisas de (LIU; BAI; ZHEN, 2017) e (LEE *et al.*, 2018), destacam e abordam os efeitos negativos das falhas transientes no protocolo FlexRay. Novamente, destaca-se que ambas as propostas focam em retransmissão de mensagens baseadas em análise da frequência de tratamento de erros do protocolo, que inevitavelmente causam desperdício de largura de banda. Outro ponto é que tais métodos não foram validados em cenários com injeções reais de falhas na rede de comunicação e não abordando a detecção destas anomalias na rede, gerando assim oportunidades de pesquisas que possam abordar a detecção, diagnóstico e verificação do impacto das falhas em específicos sistemas de controle críticos. Trabalho recente como o de (HUANG *et al.*, 2019), também destaca os efeitos negativos de falhas que geram atrasos em sistemas de controle críticos como os assistentes de direção, com destaque a carência de abordagens de detecção e diagnóstico das falhas.

Protocolo CAN-FD

Devido ao protocolo CAN-FD estar ainda em fase de estudos, existem poucas pesquisas focadas em técnicas de tolerância a falhas e avaliação de desempenho perante falhas. A principais pesquisas encontradas tratam principalmente da análise de desempenho do protocolo com as novas características flexíveis dos pacotes de transmissão de dados. Em (NGUYEN; CHEON; JEON, 2014) é verificado o desempenho do protocolo para processos de reprogramação de ECUs, mostrando o ótimo desempenho do CAN-FD. Segundo sos autores, essas características contribuem para a restauração do sistema em tempo real perante falhas melhorando a segurança do sistema, porém ainda são considerados os cenários ideais de funcionamento e em simulações, não verificando o comportamento do sistema perante situações reais. Em (TENRUH *et al.*, 2015) os resultados de análise de desempenho são simulados em modelos MATLAB e revelaram que o protocolo CAN-FD fornece melhorias consideráveis no desempenho na velocidade de transmissão de mensagens, por exemplo, com redução na média de atraso entre mensagens de 1,67 a 4,16 vezes. Porém o destaca que os testes foram em situações controladas e deixa possibilidades de análises dos mesmos cenários de controle sob influências de distúrbios que normalmente ocorreriam em situações reais e que demandariam ajustes no sistema de controle.

De forma similar em (BORTH, 2016) é apresentado um estudo do uso do CAN-FD em aplicações de caminhões pesados, na indústria de mineração, avaliando o desempenho do protocolo no roteamento de informações de chassi e carroceria, e também por meio de simulações. Em (DE ANDRADE *et al.*, 2018) os autores avaliam novamente o desempenho do CAN-FD sob o CAN tradicional, porem tratando dados oriundos de aplicação real e também evidenciando o ganho de desempenho. Essas pesquisas mostram como o protocolo tem potencial para aplicações futuras que demandam grande volume de dados e também garantias temporais, ainda assim, as pesquisas apresentam carências

naturais de análises do protocolo perante falhas, distúrbios e também em situações reais de funcionamento.

Em (WOO *et al.*, 2016), (AN; JEON, 2017) e (AGRAWAL *et al.*, 2019), os trabalhos destacam questões relacionadas a segurança e a coexistência de redes intra-veiculares, destacando a demanda futura de melhores análises de desempenho em situações críticas e quais mecanismos podem ser agregados para melhorar a confiabilidade dessas redes. No entanto, os estudos avaliam o protocolo CAN-FD sob uma perspectiva de segurança e integridade, geradas ou não por falhas e interferências, não relacionando qual o impacto de distúrbios de comunicação maliciosos e seus efeitos sobre o desempenho do protocolo. Porém, os autores destacam que falhas locais ou de interferências geradas intencionalmente na rede são aspectos que devem ser relacionados e influenciam questões de segurança.

A modelagem de efeitos de degradações no protocolo CAN-FD é destacada em (LIM *et al.*, 2019), onde são apresentados e analisados os efeitos de distorção de sinais ("*ringing effect*"). A pesquisa mostra que ao trabalhar com tempos de bit menores, os protocolos de comunicação são mais susceptíveis a interferências, sendo importante a modelagem destes efeitos para aprimorar o projeto dos sistemas de controle e definição de um meio físico mais robusto. Todavia, a pesquisa realiza todos os estudos com simulações, não tendo verificado esses efeitos em redes reais. A pesquisa destaca carências de metodologias que permitam a modelagem destas falhas, para assim, contribuir para o aumento da confiabilidade destes protocolos.

3.2 Abordagens sistêmicas de análise de falhas e segurança funcional em redes veiculares

Diferentes trabalhos vem sendo propostos com o intuito de agregar confiabilidade as redes veiculares, em diferentes níveis, com abordagens puramente simuladas, novos hardwares, novos métodos de análise de falhas, tudo em função das crescentes demandas da área. Em (BARONTI *et al.*, 2011) é apresentado o projeto e verificação de um hardware para redes veiculares de alta velocidade com características de tolerância a falhas. O hardware consiste de uma transceptor Flexray e uma interface SpaceWire (SpW) que é um padrão de link de alta velocidade usado na comunicação de alta velocidade de naves espaciais. O hardware foi testado com simulações e implementações de comunicação locais entre nodos, para mostrar a capacidade de comunicação e atender as demandas de altas velocidades. Na ocasião a pesquisa já destacava a tendência atual de recursos de alta velocidade com o aumento de dados e funções na eletrônica automotiva, pois, a proposta foi motivada pela necessidade de altas taxas de dados das aplicações automotivas de prevenção de colisão e assistência ao motorista baseados em sensores de câmera e radar. Nesse mesmo sentido, trabalhos apresentados por (DARDAR *et al.*, 2012) e (BIRCH

et al., 2013), destacam a importância de verificação funcional e do impacto de falhas em sistemas de controle automotivos tendo como base o padrão ISO-26262 (*functional safety standard*). Em (DARDAR *et al.*, 2012), é descrita a experiência obtida ao aplicar o padrão ao sistema de estimativa de nível de combustível em caminhões da Scania, sistema este, que em conjunto com as funcionalidades de outras partes do veículo desempenha um papel significativo em termos de segurança global dos caminhões pesados. Na ocasião a norma não era obrigatória para aplicação em veículos pesados, mas atualmente a sua adoção é requisito fundamental em qualquer aplicação automotiva. Já em (BIRCH *et al.*, 2013) os autores destacam que o padrão não fornece diretrizes práticas para categorizar e analisar o impacto de problemas nos componentes, e o posterior risco gerado. Assim é proposta uma forma de categorizar e analisar as principais estruturas para um caso de segurança crítica e também apresentado relações existentes entre os componentes veiculares.

Em (OETJENS *et al.*, 2014) é apresentada uma revisão sobre desafios de pesquisa para a avaliação de segurança (*safety*) em eletrônica automotiva, com base em protótipos virtuais – VPs. Os VPs são mais frequentemente aplicados como plataformas para o desenvolvimento de software, na exploração do escopo de projeto e na verificação do sistema. Os autores destacam os desafios, no contexto de desenvolvimento de software para ECUs, com relação a análise de segurança de funções nestes sistemas complexos, destacando que o VPs contribuem para o desafio de verificação funcional em fases iniciais de projeto (*early-desgin phases*). Segundo a pesquisa, o maior desafio é garantir a consistência entre o VP e o sistema real, sendo que uma inconsistência a resolver é que falhas não são modeladas no VP, mas seu efeito na funcionalidade do sistema real.

De acordo com esses exemplos de abordagens que mostram a preocupação da comunidade cientifica com aspectos se segurança funcional e confiabilidade dos sistemas de controle automotivos, outras pesquisas procuram apresentar e propor formas de sistematizar o processo avaliação, por meio da especificação de framework ou métodos. Em (MACHER *et al.*, 2015) é apresentada uma combinação de abordagens, uma para análise e avaliação de risco e outra no domínio de segurança de dados, descrevendo os impactos de diferentes problemas de segurança (*security and safety*) no nível do sistema automotivo. O trabalho classifica a probabilidade de ameaças à segurança, que pode ser usada para determinar o número de contramedidas que precisam ser consideradas. De forma similar em (LIN *et al.*, 2015) os autores destacam a importância de considerar questões de segurança funcional e dados nas redes CAN veiculares. A pesquisa trata a tendência das comunicações inter-veiculares (V2V) e a correção com o problemas de segurança e falhas internas a rede veicular, afetando suas restrições temporais, indicando a preocupação em atender as restrições temporais do sistema e também considerar questões que afetam a segurança funcional durantes os estágios de projeto.

Outro tipo de avaliação também verificada no contexto das redes veiculares é a verifi-

cação especifica de uma função da ECU. Em (HERNANDEZ-ALCANTARA *et al.*, 2016) é apresentada a modelagem, diagnóstico e estimativa de falhas em amortecedores semiativos (SA) automotivos, e são propostos dois modelos para especificar o efeito da falha do amortecedor no domínio da frequência (FFE) e outro no Espaço da Paridade Robusta (RPS) que consiste num gerador residual sensível à falha no domínio do tempo, validando ainda com resultados experimentais e mostrando como as funções das ECUs são críticas para determinados sistemas de controle. Outro trabalho neste sentido é o apresentado por (AZIMI; MORAMARCO; STERPONE, 2017) que avalia a confiabilidade de *System-on-Chip* de ECUs por meio de injeções de falhas. No trabalho é desenvolvida uma ferramenta de Injeção de Falha capaz de injetar falhas eventuais (*Single Event Upsets* - SEUs) usando um ambiente de simulação. O método desenvolvido observa a confiabilidade do sistema e verifica a capacidade de tolerância a falhas de um módulo de sistema veicular. Já em (THEISSLER, 2017) uma abordagem de detecção de anomalias é proposta para detectar falhas e classificar falhas conhecidas e desconhecidas, destacando o uso de base de dados registradas em cenários de condução. O foco é analisar e classificar informações por meio de bases de dados do próprio veículo para fins de diagnóstico de falhas.

Recentes pesquisas destacam que devido a avanços em sistemas de apoio ao motorista, como sistemas assistentes de direção e direção autônoma, a segurança funcional dos sistemas eletrônicos é cada vez mais crítica, sendo importante pesquisar métodos de teste e diagnóstico que coloquem as ECUs a funcionar em condições advervas e não usuais. Neste sentido, em (LU; CHEN; HUANG, 2018) foi desenvolvido um framework para FMEDA (*Failure Mode Effect and Diagnostic Analysis*) baseado em injeção de falhas e análise de dados "on-chip", de acordo com as características funcionais, de risco e segurança definidas no padrão ISO-26262 (*functional safety standard*) para sistemas embarcados automotivos. O foco da pesquisa é reduzir o esforço e gerar mais rapidamente os dados de FMEDA, assim, uma ferramenta foi desenvolvida para coleta das informações de uma unidade de controle perante *benchmarks* de cálculos realizados durante injeções de falhas. Desta forma, em seguida é gerada uma classificação e análise da falhas, para assim, o projetista poder analisar os dados, reconhecer as fraquezas do sistema e projetar futuras soluções.

Seguindo padrões usados na indústria para teste e certificação de ECUs, em (MATSUSHIMA *et al.*, 2018) são destacadas as tendências atuais para verificação de imunidade e compatibilidade eletromagnética, nos principais protocolos de comunicação, mas com maior ênfase em aplicações futuras com o protocolo *ethernet*. A pesquisa tem como referência a norma IEC 62228 (*EMC Evaluation of CAN Transceivers*), verificando a influência de distúrbios na qualidade da comunicação "100BASE-TX". Por meio de experimentos práticos foram injetados distúrbios em um cabo *ethernet* para ver como a qualidade da comunicação é realmente afetada. Resultados mostraram que perturbações com uma largura de pulso próxima ao tempo de um 1 bit degradaram mais a qualidade da

comunicação.

Em (BECKER; VOSS; SCHÄTZ, 2018) os autores destacam um formalismo em nível sistêmico de como requisitos de dependabilidade são importantes para a efetividade de funções críticas em sistemas de controle automotivos. São consideradas falhas operacionais, falhas funcionais e degradações funcionais. Assim, é descrito um modelo de sistema automotivo, com base em um meta-modelo UML, contendo os recursos funcionais do veículo, componentes de software, e possíveis degradações de recursos em unidades de controle. Com base neste modelo os autores apresentam uma análise estrutural dos níveis de degradação e os cenários que podem levar as unidades de controle a degradação e falhas. O modelo formal e as restrições são apresentados usando expressões aritméticas e lógicas. Para validação das expressões foi aplicada a teoria de módulo de satisfabilidade (SMT), para assim obter soluções para o modelo de problema e assim satisfazer as restrições operacionais.

De outro modo, a pesquisa de (NASRI *et al.*, 2019) destaca a adoção e desenvolvimento de um sistema decentralizado de diagnóstico de falhas em tempo real (*Real time analysis - RTA*), avaliando a temporização de mensagens e o efeito de atrasos gerados em uma rede CAN. A proposta é validada analisando a modelagem do sistema em uma rede automotiva com um sistema anti-colisão. Os resultados destacam a redução do tráfego de mensagens em função da aplicação do sistema que otimiza o escalonamento de tarefas entre os nós da rede. A validação ocorre por meio de simulações realísticas (*hardware-in-the-loop*), realizadas em laboratório e sem a injeção de distúrbios reais.

3.2.1 Problemas e oportunidades de pesquisa encontradas

A pesquisa de (BARONTI *et al.*, 2011) destaca o projeto de um novo hardware para satisfazer as necessidades de altas de taxas de comunicação e também de tolerância a falhas, destacando a demanda futura das aplicações automotivas devido aos novos e complexos sistemas que estão surgindo, como os assistentes de direção e detecção de obstáculos. Esses mesmos requisitos são cada vez mais importantes para diferentes industrias, como destacado em (DARDAR *et al.*, 2012) e (BIRCH *et al.*, 2013) que evidenciam os problemas gerados por falhas, mas com foco na adoção de padrões como a ISO-26262 para a certificação e verificação funcional de sistemas de controle em veículos. Em (OETJENS *et al.*, 2014) os autores destacam os desafios com relação a verificação funcional do sistemas automotivos, mas destacando a modelagem de erros e falhas nas ECUs o mais cedo possível. É sugerido na pesquisa a adoção cada vez maior de VPs (protótipos virtuais) mas que estes ainda carecem de contribuições principalmente na modelagem de erros que acontecem em sistemas reais, enfatizando os desafios futuros de obtenção de métodos para teste e simulação de erros, para assim, identificar pontos fracos do sistema em fases de projeto.

Em (HERNANDEZ-ALCANTARA *et al.*, 2016) e (AZIMI; MORAMARCO; STER-

PONE, 2017) são destacadas pesquisas relacionadas as funcionalidades ao hardware das ECUs, uma sobre a modelagem de falhas em atuadores de suspensão e outra com na injeção de falhas no SoC de uma ECU. Ambos os trabalhos mostram como falhas oriundas da detecção de anomalias funcionais ou de fases de teste com injeções de falhas, são importantes para agregar robustez ao projeto dos sistemas de controle, deixando lacunas no que diz respeito a forma de obtenção destas informações e sobre considerar somente testes funcionais locais da ECU e não da rede de comunicação. A proposta de (THEISSLER, 2017) é uma das mais abordagens mais promissoras e desafiantes das pesquisas futuras, também abordada na presente tese, que é a de trabalhar dados registrados no veículo ou registrar dados de falhas em logs funcionais do veiculo. Porém no trabalho é destacada uma análise em locais da ECU, sem considerar a rede de comunicação e de forma off-line, não fazendo ainda vinculo com fases de projeto. Assim, destaca-se que pesquisas futuras precisam correlacionar métodos de testes e dados gerados por estes, com informações já armazenadas em ECUs e com requisitos de projeto dos novos sistemas de controle.

Mais recentemente em (LU; CHEN; HUANG, 2018), os autores destacam a importância de coletar informações sobre o comportamento do sistema perante falhas, também usando injeção de falhas e analisando os riscos segundo o padrão ISO-26262, porém os dados são analisados após a execução do sistema (off-line) e não considera dados oriundos da rede de comunicação, são apenas tratadas as situações locais a unidade de controle. Na pesquisa de (MATSUSHIMA *et al.*, 2018) são destacadas as tendências atuais de verificação de imunidade e compatibilidade eletromagnética, nos principais protocolos de comunicação, seguindo a norma que é referência nesta área, a IEC 62228. Outro importante trabalho apresentado em (BECKER; VOSS; SCHÄTZ, 2018) mostra por meio de formalismos matemáticos os aspectos críticos de funcionamento de um sistema automotivo, destacando como o sistema deve atender e tolerar falhas sistêmicas. A pesquisa de (NASRI *et al.*, 2019) apresenta uma proposta que é uma tendência recente referente a importância de monitorar em tempo real a rede intra-veicular. Os autores propõem um sistema de diagnóstico para redes CAN, que foca na melhoria do escalonamento de mensagens de um sistema de controle quando falhas ou degradações ocorrem, validando por meio de simulações, porém deixando lacunas para testes do modelo em redes reais. A pesquisa também não aborda falhas físicas que podem gerar atrasos na comunicação e as possibilidades de detectar seus efeitos de degradação com injeções de falhas reais.

Estes trabalhos discutem a complexidade crescente dos sistemas de controle automotivos, destacando os desafios no diagnóstico, modelagem e tratamento de falhas que podem degradar as unidades controle, mesmo aquelas falhas silenciosas e que geram degradação em função do tempo. Apesar do foco central da presente tese estar nos protocolos de comunicação veiculares, este eixo de pesquisas sistêmicas é apresentado para mostrar que mesmo pesquisas que tratam especificamente aspectos da rede de comunicação, consideram a criticidade do efeito de falhas no funcionamento das ECUs e de suas respectivas

funcionalidades. Estas pesquisas são apenas alguns exemplos do contexto de segurança funcional, onde muitas abordagens são baseadas na IEC 62228 e ISO-26262, avaliando as interferências e problemas do atendimento dos requisitos do sistema de controle e os riscos a segurança dos usuários. Estas questões são oportunidades de pesquisa abordadas na presente tese.

3.3 Modelagem de falhas e especificação de requisitos

A modelagem de requisitos não funcionais é considerada uma etapa muito importante para o projeto de sistemas de tempo real mais confiáveis. Diferentes abordagens têm sido utilizadas para a modelagem destes requisitos, sendo que mais recentemente a modelagem orientada a aspectos caracteriza-se como uma das técnicas emergentes, amplamente usada para modelar diferentes problemas em aplicações industriais.

Pesquisas recentes mostram que a modelagem baseada nos conceitos de transversalidade inerentes à modelagem orientada a aspectos (AOM) pode adicionar robustez no projeto de sistemas de controle complexos, como os que compõem os sistemas ciber físicos. Neste contexto, o trabalho apresentado por (ALI; BRIAND; HEMMATI, 2012) destaca que a AOM pode ser usada para modelar e agregar robustez em sistemas complexos combinando o uso de aspectos na forma de máquinas de estado. O trabalho apresenta a metodologia RUMM (*Robustness Modeling Methodology*) que usa aspectos para modelar o comportamento do sistema, apresentando e discutindo os benefícios, como por exemplo, a redução de tempo e esforços de modelagem de software para estes sistemas. Segundo os autores a modelagem de sistemas complexos requer uma descrição comportamental detalhada a fim de tornar a modelagem mais completa possível. O uso de máquinas de estado permite a especificação de modelos comportamentais do software, mas quando este software faz uso de informações de sensores que dependem do funcionamento pleno de um ambiente físico, a descrição passa a ter características transversais que afetam diferentes partes do sistema (desempenho, consumo de energia, falhas), assim a modelagem orientada a aspectos contribui para a formação de máquinas de estado mais complexas. Como resultado a metodologia é representada como uma extensão UML (*UML profile*) permitindo a modelagem de máquinas de estado com aspectos. Modelar e testar sistemas industriais complexos com base em modelos de alto nível são benefícios que a AOM agrega nas fases de projeto de software.

Em (WASICEK; DERLER; LEE, 2014) é apresentada uma revisão sobre as formas e possibilidades de aplicação da AOM para descrever ataques em sistemas ciber físicos na área automotiva. É destacado que os conceitos da AOM podem ser usados para representar conceitos de segurança (*safety*) que são integrados a modelos que caracterizam determinados ataques ao sistema, para assim prover maior confiabilidade em sistemas de controle automotivos. Os autores sugerem o uso da AOM de forma sistemática para for-

mar modelos comportamentais dos ataques, onde estes, são modelados de acordo com as características transversais que formam os aspectos relacionados aos ataques. Neste trabalho a modelagem baseada em atores é integrada com aspectos e aplicada usando a ferramenta Ptolemy II (PTOLEMAEUS, 2014), que usa modelos de computação (*Models of Computation - MoCs*) para representar a comunicação e permitir a interação dos atores que compõem o sistema ciber físico. No Ptolemy II os MoCs são constituídos por implementações especificas que caracterizam funcionalidades do sistema, as quais podem ser reusadas. Como estudo de caso foi conceitualmente modelado um sistema de controle adaptativo de velocidade de cruzeiro (*adaptive cruise control*) onde este é modelado especificando quatro modelos de ataques que representam distúrbios (veículos a frente que reduzem a velocidade, por exemplo) que geram alteração na velocidade de um veículo durante um determinado trajeto. Nesta pesquisa é importante ressaltar que os autores enfatizam a importância da modelagem de ataques ou distúrbios no funcionamento do sistema de controle, cada vez mais cedo no projeto. Porém não abordadas soluções especificas considerando os protocolos de comunicação veiculares, questões de desempenho e métodos de testes para a injeção de distúrbios, questões estas que são consideradas na presente tese.

A pesquisa apresentada em (NGUYEN; KLEIN; LE TRAON, 2014) destaca a utilização de metodologias para o desenvolvimento de sistemas seguros usando a modelagem dirigida a segurança (*Model-Driven Security* - MDS), como uma abordagem especializada integrada à modelagem dirigida a modelos (*Model-Driven Engineering* - MDE). A MDS fornece meios para melhorar a produtividade e qualidade no processo de desenvolvimento de software, fazendo uso de modelos como o principal artefato. Os autores propõem uma MDS exploratória de acordo com um conjunto de padrões de segurança modelados usando orientação a aspectos, os quais servem de guia para os desenvolvedores nas etapas de especificação de requisitos. É destacado que estes padrões de segurança são independentes de domínio e comprovados ao longo do tempo por conhecimento e experiências práticas de desenvolvedores. Como resultado, é proposto um framework (denominado SoSPa) que permite a seleção destes padrões de segurança, especificados como modelos de aspectos reusáveis, que podem ser automaticamente incluídos na modelagem do sistema que esta sendo projetado. A contribuição do trabalho enfatiza a importância da segurança como elemento nativo no processo de desenvolvimento, aliado aos esforços da representação e reuso de componentes baseados em modelos, os quais abstraem a complexidade inerente as diferentes implementações.

Outras pesquisas mostram os esforços em manter a confiabilidade tratando de falhas em sistemas de controle automotivos, principalmente no que tange os requisitos temporais de aplicações críticas. Em (PIPER *et al.*, 2015) é apresentada uma metodologia para monitorar e proteger tarefas de controle críticas em sistemas automotivos sob interferências que causam falhas, com o objetivo de prover garantias temporais de execução. Nesta

metodologia também é discutida uma técnica de monitoramento de tarefas, que verifica os atrasos que podem prejudicar a execução das tarefas. O trabalho tem como base o padrão de segurança baseado na gestão de riscos, a ISO 26262 (ISO-26262, 2011). O objetivo é destacar e comparar características de monitoramento existentes em ferramentas usadas na área, como o AUTOSAR, que também permite a análise de tarefas, mas não considera o monitoramento de tarefas menos críticas, que em diversas situações pode propagar erros, os quais geram atrasos de comunicação e podem afetar as restrições temporais das tarefas mais críticas.

Outro trabalho recente que aborda a modelagem orientada a aspectos é apresentado por (AKKAYA *et al.*, 2016) que mostra como o uso da orientação a aspectos em conjunto com o projeto baseado em modelos MBD (*Model Based Design*) caracteriza uma forma sistemática de especializar domínios de conhecimento com a separação de características transversais dentro de um modelo de sistema. Conceitos baseados em modelagem orientada a atores (usando também a ferramenta PtolemyII) são apresentados para ilustrar como gerenciar a complexidade dos sistemas. Porém a pesquisa não especifica áreas de aplicações e não faz estudos de casos pontuais da sua aplicabilidade. Especificamente sobre a modelagem de falhas os autores destacam que a orientação a aspectos em conjunto com a orientação a atores contribui para modelar o comportamento de falhas que possuem impacto transversal, afetando diferentes requisitos não funcionais como desempenho e consumo de energia.

Assim como a MDE e a AOM, outras abordagens vem sendo estudadas com o objetivo de representar o processo de modelagem e análise de confiabilidade. Em (MO *et al.*, 2017), os autores destacam a aplicação de modelos para verificar e analisar a confiabilidade nas fases iniciais de projeto, antes dos testes e aplicações reais. É proposto um novo modelo estocástico representado por uma abordagem linear de tempo discreto, considerando transmissões de pacotes de dados entre controlador-atuador e sensor-controlador. Os comportamentos históricos de degradações da rede são modelados por cadeias de Markov de multi-estados com incertezas, destacando que as falhas de todos os períodos são independentes umas das outras. Os autores focaram em sistemas de controle digitais em rede, considerando um estudo de caso de aplicação industrial e avaliação estatística baseada no método Monte Carlo.

Em (LU *et al.*, 2018) os autores apresentam uma abordagem para detecção de falhas, baseadas em um modelo prévio do sistema e extração de dados do modelo com base em redes neurais profundas (DNN) e memórias de longo prazo. Com esta abordagem os autores destacam que é possível representar uma distribuição complicada e intrínseca de dados, que é adequada para manipular os dados de falhas precoces mascaradas por ruídos pesados, tendo ainda o uso da memória de longo prazo para auxiliar na descoberta de dependências entre dados temporais do sistema de controle. O modelo é avaliado e comparado com outros modelos de DNN, com base em um data set de falhas registradas

de um sistema de rolamentos de um equipamento industrial, mostrando contribuições em relação aos outros trabalhos.

Neste contexto, abordagens que corroboram com as pesquisas anteriormente citadas são aquelas que focam na engenharia de requisitos. Degradações causadas por falhas podem ser especificadas também como requisitos não funcionais, em diversas aplicações industriais. Pesquisas recentes destacam técnicas de modelagem e especificação de requisitos em sistemas automotivos, direcionando a necessidade cada vez maior de considerar testes e efeitos de degradação para verificar a conformidade destes sistemas críticos. A pesquisa de (ANICULAESEI *et al.*, 2018) apresenta um estudo de caso de aplicação de um modelo de geração automatizada de requisitos baseados em casos de teste, direcionados a um protótipo de sistema de controle automotivo para velocidade de cruzeiro. Os autores aplicaram uma linguagem natural para eliminar as imprecisões na formulação dos requisitos do sistema, utilizando critérios e armadilhas (pontos de discussão ou reavaliação) para definir o quanto o requisito satisfaz aos testes do sistema. Os resultados definem um modelo abstrato do sistema em estudo, formalizando e reduzindo o tempo da especificação dos requisitos, por meio de uma ferramenta automatizada desenvolvida durante a pequisa. Os autores ainda destacam que a forma com que os requisitos foram definidos no inicio do projeto são fundamentais para a compreensão dos objetivos que o sistema de atingir. Verificar se os requisitos e as funções agregadas podem aumentar a confiabilidade depende de novas avaliações e análises, para isso, em (ADEDJOUMA; YAKYMETS, 2019) os autores apresentam um *framework* chamado "Sophia", composto por um conjunto de ferramentas baseadas em modelos de referência, que avaliam os requisitos de dependabilidade de sistemas ciber físicos. Entre tais ferramentas destacam-se o uso de modelos UML e SysML como entradas do framework e também análises baseadas em causa e efeito, como o FMEA (*Failures Modes and Effects Analysis - FMEA*). O *framework* permite executar análises de segurança e confiabilidade, avaliação de riscos e gerenciamento de requisitos de segurança usando técnicas semi-formais e formais.

Dando ênfase a ferramentas, modelagem e especificação de requisitos em sistemas embarcados, em (STARON, 2019) são apresentadas as principais técnicas e padrões da engenharia de requisitos que são aplicadas para o desenvolvimento de sistemas embarcados no contexto automotivo. Os autores apresentam uma visão geral das principais abordagens para tratamento de requisitos nessa área, são elas:

- o modelo V, recomendado pela norma ISO 26262, descrevendo as principais características dos sistemas usados na área automotiva, principalmente focados em questões de segurança e confiabilidade;

- especificação textual de requisitos, usando tabelas, templates, checklists de verificação, frameworks, com destaque ao AUTOSAR (AUTOSAR, 2019), uma arquitetura de referência para o desenvolvimento de software na área automotiva;

- especificação de diagramas de casos de uso, que descrevem a interação entre os principais atores do sistema automotivo, destacando o uso da modelagem UML para estas representações formais. O principal motivo da adoção destes modelos de alto nível deve-se a facilidade de representação e discussão entre diferentes projetistas, com relação as principais propriedades do sistema;

- especificação de requisitos baseados em modelos, usando modelos formais e matemáticos, baseado em UML ou Simulink, com representações da dinâmica destes sistemas (ex: ABS, Suspensão Ativa, Controle de Tração). Amplamente adotado por engenheiros na indústria por apresentar representações evoluídas, que já passaram por grande estudo e esforço de projeto, sendo usados como referência para algoritmos de controle.

- requisitos como modelos, especificamente com a adoção perfis evoluídos e adaptados da UML, como por exemplo SysML (modelagem de aplicações em engenharia de sistemas), para a descrição formal dos requisitos e sua interação com as diferentes partes da aplicação fim.

3.3.1 Problemas e oportunidades de pesquisa encontradas

Os trabalhos estudados destacam o uso da modelagem orientada a aspectos como importante expertise para metodologias de desenvolvimento de sistemas mais seguros, considerando também a possibilidade e tendência de uso desta modelagem nas fases de especificação de requisitos de projeto. Em (ALI; BRIAND; HEMMATI, 2012), (WASICEK; DERLER; LEE, 2014) e (NGUYEN; KLEIN; LE TRAON, 2014), os autores apresentam técnicas que usam a AOM como uma abordagem que permite o desenvolvimento de sistemas mais seguros, unindo estas aos conceitos de MDE, permitindo produtividade por meio do reuso de componentes. Os trabalhos destacam a robustez agregada, a segurança da informação, o reuso de padrões, porém algumas propostas são centradas em nível arquitetural, não sendo especificada para a modelagem de aspectos relacionados ao meio físico de comunicação ou inerente ao processo de comunicação específico de um sistema de controle. A proposta de (WASICEK; DERLER; LEE, 2014) realiza estudo de caso na área automotiva e modela ataques maliciosos ao sistema de controle por meio de interferências externas, não considerando uma falha física no sistema ou a possível fonte de falhas que afetam o protocolo de comunicação, o qual degrada de também o desempenho, deixando assim lacunas para a modelagem de falhas seguindo este mesmo modelo e também usando a orientação a aspectos na especificação de requisitos.

Na pesquisa de (PIPER *et al.*, 2015) os autores destacam a importância do tratamento e dos problemas gerados pela violação de requisitos temporais em protocolos usados em redes veiculares, mas não consideram a modelagem destas violações como falhas nas fases de projeto dos sistemas de controle embarcados, tipicamente abordando uma forma

de mitigar o impacto das falhas perante o monitoramento de ocorrências eventuais. Já em (AKKAYA *et al.*, 2016) os direcionamentos dados pelos autores indicam como a combinação de técnicas de modelagem baseadas em modelos e orientação a aspectos podem contribuir para a modelagem de falhas, devido aos seus efeitos transversais no desempenho dos protocolos de comunicação.

Falhas e degradações em sistemas de controle de rede são destacadas em (MO *et al.*, 2017), onde os apresentam um modelo para análise de confiabilidade, considerando aplicações de controle industrial, mas sem destacar pontos específicos do tipo de comunicação utilizada. Um importante ponto a destacar em (MO *et al.*, 2017) é a ênfase na possibilidade e no desafio de modelar falhas ou anomalias em fases iniciais do projeto, no qual os autores destacam ser um dos importantes desafios de futuras pesquisas. Em (LU *et al.*, 2018) os autores destacam que é crucial a análise e detecção de falhas o mais cedo possível para reduzir o tempo de inatividade e manutenção em aplicações industriais, destacando os desafio de se obter informações das falhas, por exemplo, como extrair informações de incipientes sinais de falhas, detectar possíveis anomalias com base na correlação de dados sequenciais e gerar possíveis análises e notificações.

Além da ênfase em diagnostico das degradações, outras pesquisas abordam as possibilidades de utilizar modelos de referência para especificar e avaliar requisitos relacionados a confiabilidade dos sistemas. Em (ANICULAESEI *et al.*, 2018), (ADEDJOUMA; YAKYMETS, 2019) e (STARON, 2019) são tratadas técnicas para modelagem e verificação de requisitos de dependabilidade, com *frameworks* que apoiam a avaliação e análise de confiabilidade, porém as pesquisas são generalistas, onde os próprios autores destacam a necessidade de adaptação a cenários específicos, deixando lacunas na especificação de requisitos relacionados a falhas físicas que afetam sistemas de controle distribuídos.

Apesar dos esforços realizados nessas pesquisas, foi identificado que há a carência de métodos de modelagem e diagnóstico de falhas que afetam o desempenho de sistemas de controle distribuídos, não especificando metodologias de teste e análise focadas nos protocolos de comunicação veiculares. Apesar de alguns trabalhos considerarem dados reais e outros usarem representações formais de sistemas de controle, há a carência de trabalhos com estudos de casos práticos com dados de injeção de falhas reais ou com análise de dados históricos de anomalias no processo de controle. Sendo assim, estes caracterizam os principais pontos e lacunas que podem ser exploradas para agregar maior robustez e confiabilidade aos sistemas de controle no contexto das redes intra-veiculares.

3.3.2 Problemas encontrados e oportunidades em relação ao framework RT-FRIDA

Após um estudo aprofundado do *framework* RT-FRIDA e de trabalhos relacionados, foi identificado que o *framework* não trata de uma forma específica requisitos relacionados a falhas que afetam a comunicação de sistemas embarcados de tempo real, os quais podem caracterizar um novo nível para a classificação proposta por (FREITAS, 2007). Diferentes

tipos de falhas podem afetar as restrições temporais de tarefas críticas com efeito transversal, podendo afetar vários requisitos ao mesmo tempo, como tempo (afetando deadline e jitter), desempenho (aumentando o tráfego na rede e o tempo de resposta) e distribuição (impossibilitando a alocação de tarefas nos tempos definidos). Assim, falhas que afetam protocolos de comunicação usados em redes veiculares podem ser especificadas como requisitos não funcionais e modeladas de acordo com os conceitos de orientação a aspectos. Desta forma, a extensão do framework RT-FRIDA para prover suporte a modelagem de falhas de comunicação em sistemas de tempo real caracteriza também uma contribuição da presente pesquisa.

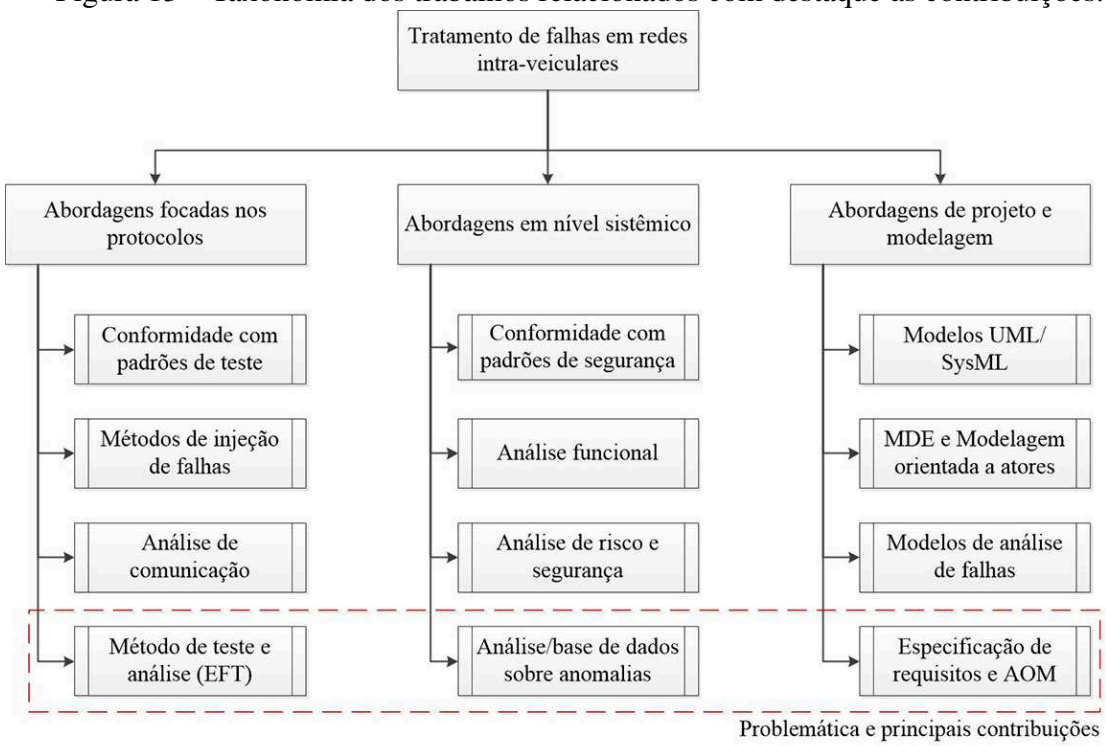
3.4 Taxonomia dos trabalhos relacionados e caracterização do problema

Com base no estudo realizado esta seção apresenta uma síntese dos principais trabalhos relacionados, com o intuito de destacar a lacuna e o problema de pesquisa que está sendo abordado. A pesquisa bibliográfica teve como base os principais veículos de publicação, como o portal de periódicos da CAPES, as bases IEEE Explorer, ELSEVIER e ACM, além do repositório digital LUME de Teses e Dissertações da Universidade Federal do Rio Grande do Sul – UFRGS. Ao todo mais de 600 trabalhos acadêmicos (incluindo artigos, teses e dissertações) relacionados a confiabilidade em sistemas de automação, protocolos de comunicação veiculares, diagnóstico de falhas e modelagem de falhas, foram selecionados e organizados com apoio da plataforma MENDELEY (MENDELEY, 2017), passando por um processo de refinamento até chegar aos trabalhos específicos listados e discutidos no estado da arte.

Para representar as temáticas abordadas na presente tese e a problemática tratada, após a análise dos referidos documentos foi desenvolvida uma taxonomia, que contribui não somente para elucidar o escopo da pesquisa, mas também para mostrar possíveis áreas de estudos futuras e pesquisas relacionadas. A Figura 13 apresenta a taxonomia dos trabalhos relacionados apresentados.

O problema de pesquisa é centrado na modelagem de falhas que degradam o desempenho da rede intra-veicular, focado em protocolos de comunicação, análise de dados de funcionamento e de comunicação das ECUs, integrando assim, estas informações às fases iniciais de projeto de novos sistemas de controle. A análise dos trabalhos relacionados indica a carência de métodos de teste e modelagem que contemplem especificamente a análise das principais falhas que afetam e degradam o protocolos de comunicação usados no contexto automotivo. Destaca-se também que há carências na análise de bases e logs do funcionamento técnico de ECUs (fornecidos em alguns casos de forma restrita pelos próprios fabricantes), carências na correlação e análise de falhas em nível de modulação de sinais, verificando também os impacto nos pacotes de dados transmitidos na

Figura 13 – Taxonomia dos trabalhos relacionados com destaque às contribuições.



- (SEDAGHAT; MIREMADI, 2010)
- (PATTANAIK; CHANDRASEKARAN, 2012)
- (HARTWICH, 2012)
- (PATTANAIK; CHANDRASEKARAN, 2013)
- (MARQUES et al., 2013)
- (MARQUES et al., 2014a)
- (MARQUES et al., 2014b)
- (GESSNER et al., 2014)
- (PANNILA; EDIRISINGHE, 2014)
- (NGUYEN; CHEON; JEON, 2014)
- (NAKAMURA et al., 2015)
- (FONTANA; HUBING, 2015)
- (ANKARSON et al., 2015)
- (ZENG; KHALID; CHOWDHURY, 2015)
- (HUANG; LEU; CHEN, 2015)
- (TENRUH et al., 2015)
- (BORTH, 2016), (WOO et al., 2016)
- (SOUTO; PORTUGAL; VASQUES, 2016)
- (SAHA; ROY; RAMESH, 2016)
- (ROQUE et al., 2016), (HUANG et al., 2016)
- (JUAN; JEONG; KIM, 2016)
- (NA; DAO; CHO, 2016)
- (ROQUE et al., 2017)
- (NA; JEON, 2017)
- (LIU; BAI; ZHEN,2017), (LEE et al, 2018)
- (DE ANDRADE et al., 2018)
- (LIM et al. 2019), (AGRAWAL, 2019)
- (SHIRAI; SHIMIZU, 2019)
- (HUANG, 2019)

- (BARONTI et al., 2011)
- (DARDAR et al., 2012)
- (BIRCH et al., 2013)
- (OETJENS et al., 2014)
- (MACHER, et al., 2015)
- (LIN et al., 2015)
- (HERNANDEZ-ALACANTARA ET AL., 2016)
- (AZIME; MORAMARCO; STERPONE, 2017)
- (THEISSLER, 2017)
- (LU; CHEN; HUANG, 2018)
- (BECKER; VOSS; SCHÄTZ, 2018)
- (MATSUCHIMA et al, 2018)
- (NASRI et al, 2019)

- (BERTAGNOLLI, 2004)
- (FREITAS et al., 2007)
- (WEHRMEISTER, 2009)
- (ALI; BRIAND; HEMMATI, 2012)
- (WASICEK; DERLER; LEE, 2014)
- (NGUYEN; KLEIN; LE TRAON, 2014)
- (PIPER et al., 2015)
- (AKKAYA et al., 2016)
- (MO et al, 2017)
- (LU et al, 2018)
- (ANICULAESEI et al, 2018)
- (STARON, 2019)
- (ADEDJOUMA; YAKYMETS, 2019)

Fonte: Autor.

rede e como requisitos podem ser modelados nas fases de projeto do sistema de controle. Muitos desses problemas emergem da crescente complexidade dos sistemas ciber físicos que compõem as redes veiculares, com o advento por exemplo, de carros elétricos (geram alta interferência eletromagnética), carros autônomos (caracterizados por tarefas críticas), cidades inteligentes (coexistência de múltiplas redes sem fio comunicando-se com veículos), a grande quantidade de equipamentos de terceiros que de forma complementar podem ser instalados nos veículos, além dos possíveis ataques maliciosos que podem ocorrer nestas redes.

Diversos autores destacam a necessidade de modelagem de falhas nas fases iniciais de projeto de sistemas de controle distribuídos e de tempo real. Sistemas de controle automotivos operam e dependem de protocolos de comunicação, caracterizando a importância de mitigar o impacto de falhas em restrições temporais de tarefas críticas. Dentro deste contexto, foi identificado também a possibilidade de integração e extensão do framework RT-FRIDA para permitir um quinto nível de requisitos não funcionais relacionados a falhas. Este framework é usado, em conjunto com a modelagem orientada a aspectos, na fase de especificação de requisitos relacionados a falhas do estudo de caso realizado nesta tese. A seguir, o Capítulo 4 detalha como estas lacunas foram trabalhadas, destacando os estudos realizados e as contribuições de pesquisa da tese.

4 METODOLOGIA PARA TESTE E ANÁLISE DE FALHAS EM DCS VEICULARES

Este capítulo apresenta o detalhamento da pesquisa elaborada na presente Tese de Doutorado, com a especificação da metodologia desenvolvida, os métodos de teste e injeção de falhas aplicados para análise de susceptibilidade a falhas EFT, concluindo com a apresentação do processo de modelagem e especificação de requisitos não funcionais relacionados a falhas, o qual é suportado pela extensão do framework RT-FRIDA.

4.1 Caracterização dos elementos que compõem a metodologia

A análise de confiabilidade de sistemas embarcados distribuídos têm sido tema de muitas pesquisas, como as abordadas no estado da arte, e muito se deve às características heterogêneas do ambiente onde estes sistemas são aplicados, como por exemplo, na indústria automotiva. Neste cenário de estudo as unidades de controle eletrônicas (ECUs) são responsáveis por tarefas de controle críticas que geram problemas também em aspectos de segurança dos usuários. Essas redes são comumente chamadas de redes intra-veiculares, constituídas por topologias e protocolos de comunicação responsáveis pela interconexão de vários nós de controle. Diferentes tipos de falhas podem afetar estas tarefas críticas, e de acordo com a literatura recente, muitos mecanismos de tolerância ou diagnóstico de falhas são totalmente reativos, não tendo sido projetados considerando as possíveis degradações silenciosas causadas por falhas no contexto físico.

Esforços recentes de pesquisa estão focados em mecanismos que podem prever alguns comportamentos de falha e para esta tarefa são necessários processos e métodos de teste, em conjunto com análises de dados tanto dos processos de teste quanto de informações geradas em tempo real pelas unidades de controle. Desta forma, a presente proposta contribui tratando lacunas destacadas em pesquisas recentes (AKKAYA *et al.*, 2016), (MO *et al.*, 2017), (LU *et al.*, 2018) que enfatizam a carência de mecanismos de interligação destas etapas com fases de projeto, permitindo por exemplo, um bom nível de especificação de requisitos funcionais e não funcionais, tendo como foco falhas que degradam o desempenho dos sistemas de controle distribuídos, especificamente no contexto do pre-

sente estudo, caracterizado pelas redes intra-veiculares.

As redes intra-veiculares representam complexos sistemas ciber-físicos - CPS, principalmente devido às recentes tecnologias que envolvem o uso de assistentes de direção, sistemas para detecção e desvio de obstáculos, carros autônomos e elétricos, entre outras tecnologias. Com essa complexidade o número de erros de comunicação entre os componentes tende a aumentar significativamente, assim, o mapeamento de diferentes tipos de falhas na fase de projeto é muito importante e contribui para melhoria no desempenho e na confiabilidade do sistema.

Protocolos de comunicação veiculares tem sofrido atualizações frequentes (TTCAN, FTT-CAN, CAN-FD), possuem proteções físicas e alguns mecanismos nativos para tratamento de erros. Embora essas características forneçam maior confiabilidade nas comunicações, ainda há oportunidades de melhoria, pois muitas situações de falhas e de degradação de desempenho só são percebidas com testes de operação e de estresse. Desta forma, a presente tese contribui com um novo processo de teste e modelagem, baseado no mapeamento de comportamentos originados por falhas e anomalias de comunicação, adicionando estas informações às fases de projeto. Assim, é possível contribuir para o desenvolvimento de sistemas de diagnóstico pró-ativo e sistemas de controle mais robustos.

A presente tese é dividida em algumas etapas importantes, que caracterizam os diferenciais da proposta em relação a outras abordagens da literatura. Estas etapas são apresentadas a seguir e descritas em seguida com apoio de um diagrama representativo.

- Estudo das vulnerabilidades dos protocolos de comunicação veiculares, suas evoluções e tendências futuras.

- Desenvolvimento de um método de injeção de falhas transientes, análise de susceptibilidade dos protocolos, e implicações no desempenho de sistemas de controle críticos de tempo real.

- Análise e geração de dados referente a falhas, anomalias e degradação de desempenho, por meio de métricas e ferramentas usadas na indústria automotiva.

- Mapeamento de informações que caracterizam as anomalias de comunicação, com apoio dos conceitos de orientação a aspectos e especificação de requisitos com o framework RT-FRIDA. Modelagem de um mecanismo de diagnóstico.

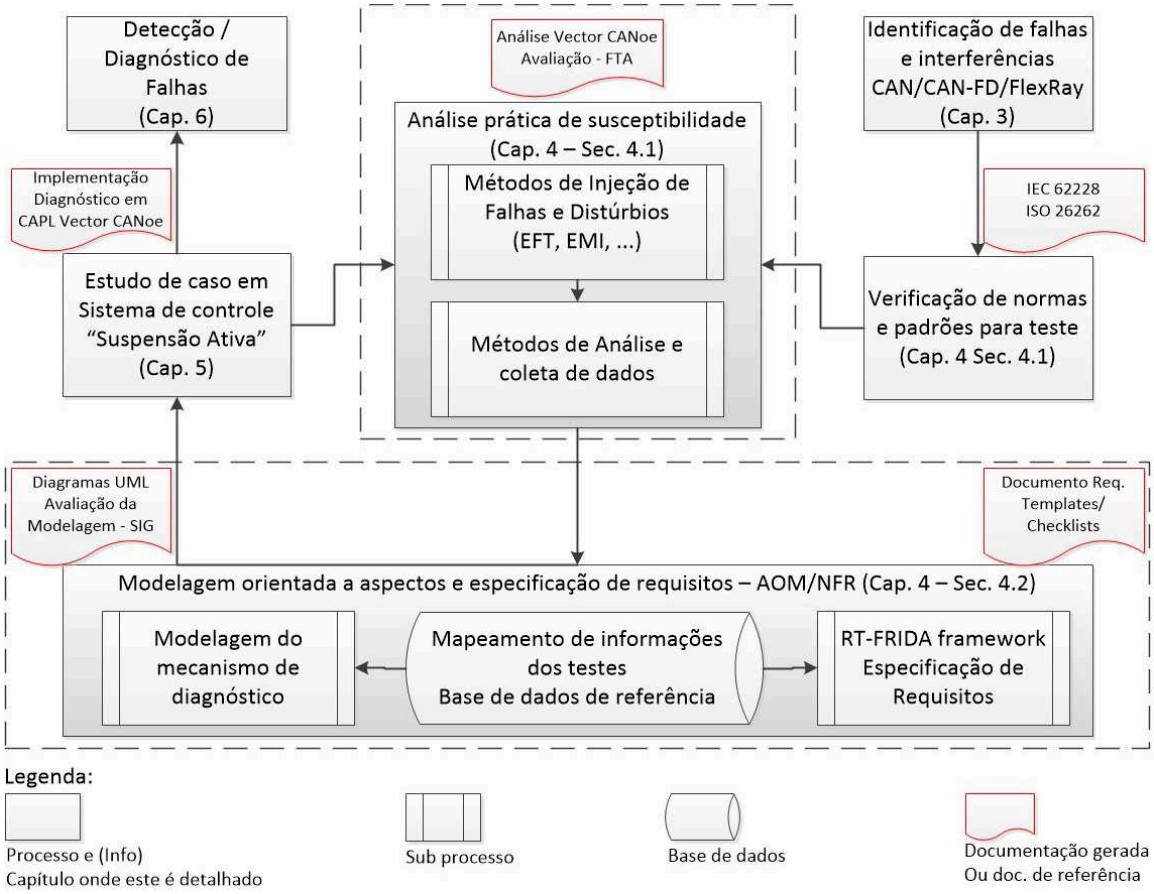
- Desenvolvimento de estudo de caso para aplicação do método em sistema de controle veicular, realizando a validação com análises baseadas em métricas de desempenho, e diagnóstico das ocorrências de degradações causadas por falhas.

Estas etapas são destacadas e podem ser visualizadas na Figura 14, que procura representar visualmente os processos que compõem a metodologia proposta. A representação apresentada segue os preceitos de um fluxograma clássico, destacando as informações e a

comunicação entre processos, subprocessos, base de dados e documentações geradas ou que são de referência. Na Figura 14 são destacadas duas áreas pontilhadas, caracterizando o ponto central da metodologia, a conexão permanente entre fases de teste e modelagem de falhas.

A etapa inicial compreende o estudo das principais características de funcionamento e de vulnerabilidades a falhas dos protocolos de comunicação veiculares CAN, CAN-FD e FlexRay, usados na interconexão de ECUs. O estudo levou em consideração características técnicas relativas as camadas física e de enlace, verificando problemas em nível modulação de sinais, na representação dos bits de transmissão, seguindo para os níveis de quadro, analisando a formação de mensagens com periodicidade e eventualidade. Questões de desempenho relativas ao uso do canal de comunicação, atrasos e interferências são especificadas. É importante destacar também a verificação técnica da composição típica dos laços de controle entre ECUs, com topologias barramento e estrela sendo mais comumente usadas, e restrições temporais de acordo com o tipo de aplicação. Todas essas questões são apresentadas e analisadas nos estudos iniciais que embasam a proposta desta teste. Na Figura 14 essa atividade é representada no processo inicial "Identificação de falhas e interferências - CAN/CAN-FD/FlexRay".

Figura 14 – Representação das partes que compõem a metodologia para testes e análise de falhas em redes intra-veiculares.



Fonte: Autor.

Na etapa seguinte, para os tipos de falhas e interferências identificadas, normas e padrões de teste devem ser analisados e aplicados como referência aos procedimentos de teste. Neste estudo, a IEC 62228 (versão de 2007 e também a nova revisada em 2018 e 2019) é a referência principal, incluindo também suas respectivas bases a IEC 61000-4 e ISO 7637-3. Em conjunto, softwares, métodos de análise e métricas de desempenho são aplicadas para quantificar os impactos das falhas no desempenho dos sistemas de controle veiculares, tendo como base métricas definidas anteriormente, como a variação no atraso de pior caso (*difference jitter*), a variação no atraso médio (*average jitter*), e variações na carga da rede (*busload*). Nesse caso as análises foram baseadas em dados obtidos do software CANoe/CANAnalyser (VECTOR INFORMATIK, 2018) que são amplamente usados na indústria automotiva. Dados são analisados por meio de logs obtidos em períodos de amostragem, bem como também de logs de funcionamento das ECUs durante os testes. Estas atividades são representadas na Figura 14 pelos processos "Verificação de normas e padrões para teste" e "Análise prática de susceptibilidade", este último composto pelos subprocessos "Métodos de Injeção de Falhas e Distúrbios (EFT, EMI, ...)" e "Métodos de análise e coleta de dados".

Após estes testes e análises da rede intra-veicular, é realizada uma compilação de dados referentes aos impactos das falhas no desempenho dos protocolos de comunicação, caracterização de anomalias e situações críticas, definindo limites de operação ou pontos de alerta que são representados por meio dos conceitos de orientação a aspectos. Com estas informações requisitos específicos para tratamento, análise, detecção ou diagnóstico de falhas podem ser especificados, para futuramente serem agregados aos sistemas de controle. Esta especificação foi desenvolvida com suporte do framework RT-FRIDA, com a extensão do mesmo para as especificidades relacionadas às falhas e seus impactos na degradação de desempenho. Estas atividades são representadas na Figura 14 por meio do processo "Modelagem orientada a aspectos e especificação de requisitos - AOM/NFR", o qual é composto pela base de dados "Mapeamento de informações dos testes - Base de dados de Referência" e pelos sub processos "RT-FRIDA framework/Especificação de Requisitos" e "Modelagem do mecanismo de diagnóstico".

Por fim, um estudo de caso é conduzido para a validação da metodologia proposta. Para esta etapa, foi realizado um estudo com dois hardwares de injeção de falhas, tendo como base o Sistema de Controle de Suspensão Ativa (MICHELIN, 2014), que sofreu os ajustes e adaptações necessárias para a validação da proposta. O desenvolvimento e implementação do estudo foi realizado na plataforma Vector CANoe, com feedbacks obtidos durante o estágio de doutorado realizado na Alemanha, no Instituto de Automação e Engenharia de Software da Universidade de Stuttgart. Durante esse período de estudos um estágio dentro do setor de R&D da empresa Vector Informatik foi realizado, com o objetivo de aprender as ferramentas de testes usadas pela indústria automotiva e obter contribuições por meio de discussões com profissionais do setor. Na Figura 14 esta ativi-

dade é representada pelos processos “Estudo de Caso em Sistema de Controle Suspensão Ativa” e “Detecção/Diagnóstico de Falhas”.

As próximas seções descrevem com detalhes as etapas que compõem a metodologia proposta, destacando na primeira parte (subseção 4.1) o processo de coleta de informações sobre distúrbios e anomalias na rede intra-veicular, métodos de injeção de falhas e seus efeitos na degradação de desempenho. Na segunda parte (subseção 4.2) é apresentado como os conceitos de orientação a aspectos podem ser aplicados em conjunto com o framework RT-FRIDA, no processo de modelagem e especificação de requisitos relacionados a falhas. Ao observar a Figura 14 destaca-se que no centro estão os principais processos detalhados neste capítulo, a análise prática de susceptibilidade a falhas transientes e a modelagem orientada a aspectos com especificação de requisitos não funcionais.

Em sequencia, os Capítulos 5 e 6 apresentam especificamente o estudo de caso e resultados obtidos durante a última etapa da pesquisa.

4.2 Teste e análise de degradação de desempenho em redes intra-veiculares

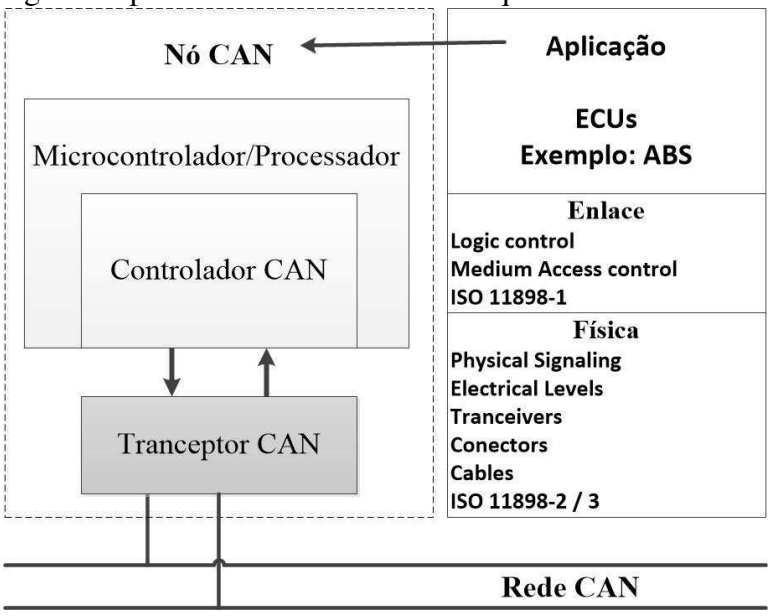
Com o intuito de detalhar os processos de análise das redes intra-veiculares, enfatizando ferramentas e normas associadas, as próximas subseções detalham os procedimentos e métodos de testes utilizados nos três protocolos de comunicação abordados nesta tese (CAN, CAN-FD e FlexRay).

4.2.1 Análise de impacto de EFT em redes CAN

Protocolos de comunicação usados em redes intraveiculares são responsáveis por interconectar várias unidades de controle eletrônico, tipicamente nos carros mais modernos, mais de 80 ECUs de acordo com uma topologia específica. Algumas tarefas críticas se comunicam por meio de mensagens nesta rede e devem executar suas ações em tempo hábil. Além disso, cada nó de uma rede veicular inclui um ou mais transceptores (*tranceivers*). Os transceptores são a interface entre o meio físico e a unidade de processamento do nó host, e também são responsáveis pela implementação das especificações da camada física (PEREIRA; NEUMANN, 2009), (GALLOWAY; HANCKE, 2013). A Figura 15 ilustra esse tipo de conexão de uma ECU com o barramento CAN.

Neste cenário, os protocolos de comunicação podem ser afetados por várias fontes de interferência eletromagnética que podem acoplar energia transitória a circuitos através de cabos de dados e causar diferentes tipos de perturbações nas ECUs. Diferentes tipos de situações de falha podem ser geradas por descargas eletrostáticas, como transitórios de chaveamento (*power switching transients*), definidos como transientes elétricos rápidos (EFTs) (IEC-61000, 2012). De acordo com (ZHANG *et al.*, 2014) os problemas de imunidade eletromagnética podem ser causados por uma variedade de mecanismos e os

Figura 15 – Diagrama típico de uma ECU com destaque à conexão do transceptor CAN.



Fonte: Autor.

projetistas de sistemas estão aplicando cada vez mais testes rigorosos de imunidade, com o objetivo de prevenir esses problemas antes que eles ocorram.

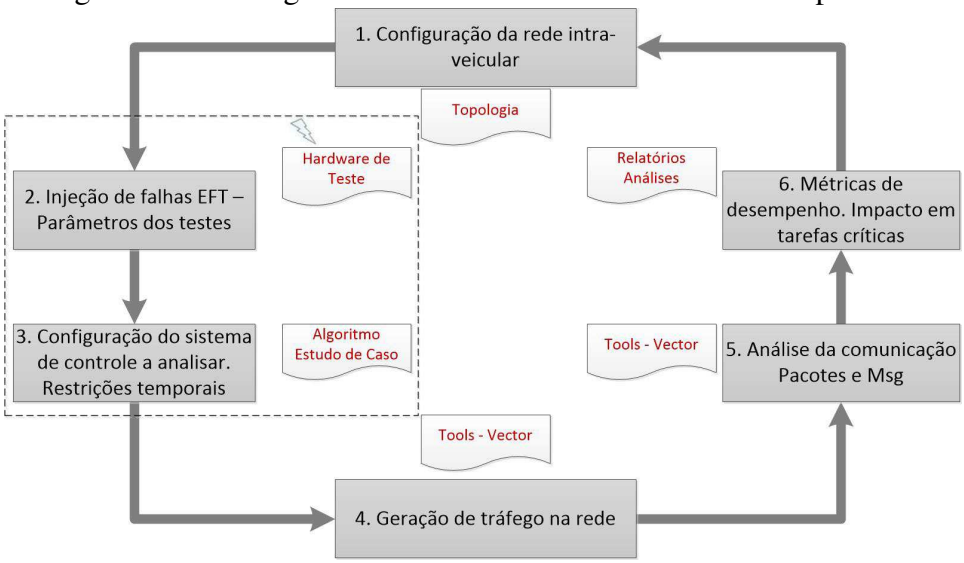
Assim, pesquisas que analisam a susceptibilidade dos sistemas aos EFTs são de grande importância. Um EFT é a manifestação externa de uma mudança repentina nas condições do circuito e geralmente é causada por um pulso muito pequeno e imprevisível (GIES, 2013), (BAUER; DEUTSCHMANN; WINKLER, 2015). Representados por um único pulso ou por uma explosão de picos (*bursts*), os EFTs podem interromper a sinalização diferencial do protocolo CAN, causando falhas de transmissão. Segundo (URSACHI; HELEREA, 2014), os distúrbios eletromagnéticos, como os EFTs de baixa energia gerados por fenômenos de chaveamento em redes de baixa e média tensão, são um desafio presente e permanente para equipamentos elétricos e eletrônicos.

Muitos fatores tornam esse tipo de falha difícil de medir e constituem fontes potenciais de falhas críticas. Portanto, medidas muito precisas para entender esses eventos e sua periodicidade são essenciais para a confiabilidade do processo de comunicação. Assim, na presente pesquisa técnicas de medição, métricas de desempenho e análises baseadas em logs de funcionamento da ECUs foram aplicadas aos experimentos, com o objetivo de registrar os impactos das ocorrências de transientes elétrico rápidos na rede CAN. Assim, este estudo preenche lacunas relacionadas à análise do impacto de EFTs no desempenho dos protocolos de comunicação e dos respectivos sistemas de controle, de acordo com normas como a IEC62228, IEC61000-4.

O estudo desenvolvido descreve e aplica um novo método de teste e injeção de falhas EFT, focado em uma parte que trata por meio de experimentos o impacto de EFTs não apenas em transceptores de ECU, mas também na propagação de erros em redes intra-

veiculares, associando esse impacto a tarefas de controle críticas com restrições temporais definidas. A fim de verificar o impacto das EFTs nos protocolos de comunicação, os distúrbios na rede são gerados e analisados por meio de um hardware de teste dedicado, que compõe a etapa 2 do método de teste. A Figura 16 ilustra uma visão geral do método de teste aplicado.

Figura 16 – Visão geral do método de teste desenvolvido e aplicado.



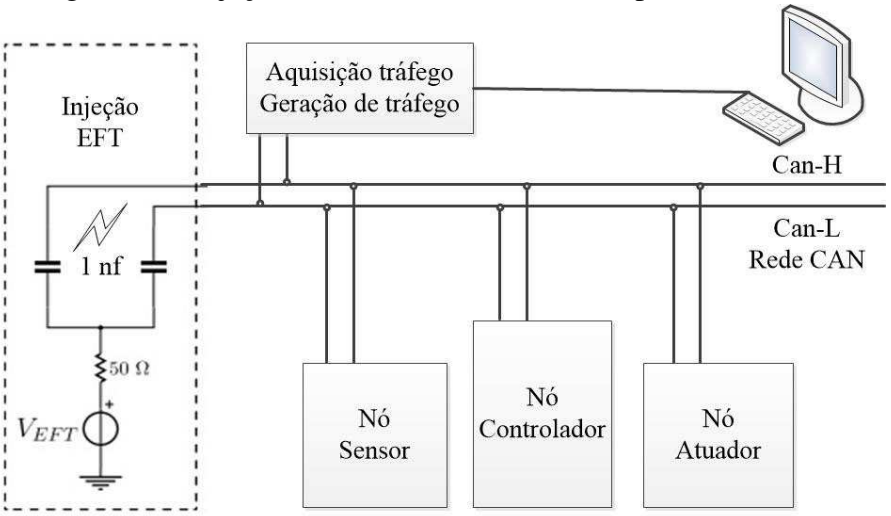
Fonte: Autor.

O método deve ser executado ciclicamente para atualização de parâmetros e realizar refinamentos necessários para aumentar a precisão e a confiabilidade nas medições de dados. Este método é aplicado em seis etapas, conforme descrito a seguir. Na etapa 1, a topologia da rede é configurada (barramento, estrela), bem como o método de interconexão do nó durante o experimento, a fim de caracterizar situações críticas de controle. Na etapa 2, foi utilizado um hardware para injeção de EFT, baseado no padrão de teste IEC TS 62228, com foco na especificação de injeções de falha em rajada (*bursts*), aplicada em uma rede CAN real, com o sistema de controle implementado na plataforma de desenvolvimento Vector CANoe. Neste experimento, um gerador de função foi usado para injetar transientes com diferentes amplitudes.

Contudo nesse primeiro experimento foram necessárias algumas adaptações que tornaram o hardware um pouco diferente da norma IEC TS 62228, por exemplo, tendo como principal diferença o fato de que a norma usa uma única placa com todos os nós CAN acoplados para teste de EFT, e este experimento usou a parte injeção de falhas conectada diretamente em uma rede CAN real. Assim, foi possível verificar o impacto deste tipo de falha analisando os efeitos em um sistema de controle veicular real. Essa decisão possibilitou que este trabalho não apenas aplicasse o teste sugerido na norma, mas também estende-se o mesmo, mostrando de outra perspectiva como a falha pode ser analisada. Essa nova forma de análise pode contribuir para o desenvolvimento de técnicas

futuras para mitigar essa falha nos sistemas de controle. De acordo com a IEC TS 62228, é necessário atender a sete requisitos de teste para este tipo de injeção de falha: 1) Gerador/Injetor de pulsos de teste; 2) Placa de teste; 3) Largura de banda do osciloscópio maior/igual 500 MHz; 4) um gerador de funções; 5) uma fonte de alimentação externa; 6) ao menos um nó em uma unidade de controle; e 7) um PC para registro e controle da ações. Dentre os requisitos dois pontos foram modificados, mas que não prejudicaram os testes que visavam verificar os efeitos destes transientes na rede CAN. O primeiro ponto refere-se ao requisito 2 da norma, onde a injeção da falha foi feita com o circuito de teste, mas nesse caso conectado diretamente na rede estendendo a forma de análise. O segundo ponto é relacionado os pulsos EFT gerados pelo circuito, onde é recomendado que estes tenham tempos de borda de subida e descida em torno de 5 ns (nanosegundos) e neste experimento estes tempos foram um pouco acima, devido aos modelos de transistores utilizados. A Figura 17 ilustra essa topologia de interconexão e como a injeção de EFT é introduzida.

Figura 17 – Injeção de EFT na rede CAN - Etapa 2 do método.



Fonte: Autor.

Na etapa 3, parâmetros relacionados às restrições temporais da aplicação, tipos de tarefas críticas e respectivas mensagens que trafegam pelo protocolo de comunicação são especificados e configurados (por exemplo, restrições para um sistema de controle de suspensão ativa). De acordo com o método para o experimento três nós no mínimo devem ser especificados (sensor, controlador e atuador), com seu comportamento programado de acordo com a referida planta de controle. Após a configuração de rede e do hardware de injeção de falhas, a etapa 4 utiliza uma ferramenta/software para geração de tráfego que simula as condições típicas de uso do barramento. A etapa 5 utiliza outra ferramenta/software para analisar a comunicação e detectar distúrbios. Por fim, na etapa 6, são apresentados relatórios e análises sobre o impacto de EFT nas tarefas críticas definidas na etapa 1. O método e as análises apresentadas servem de suporte para o projeto de no-

vos sistemas de controle, considerando estes tipos de distúrbios, podendo assim adicionar gatilhos de diagnóstico dentro do sistema, reconfigurar os parâmetros relacionados a restrições de tempo e verificar os requisitos de hardware para a tolerância destas falhas. As subseções a seguir apresentam detalhes de procedimentos de testes realizados em estudo de caso prático, com exemplos de informações especificas a cada etapa e os respectivos resultados obtidos com a aplicação deste método.

4.2.1.1 Descrição dos materiais e procedimentos de teste

O método de teste foi aplicado em uma rede CAN procurando representar as condições típicas de comunicação entre três nós, com tráfego de mensagens e carga de uso do barramento em torno de 30% (*busload*). O estudo de caso tem como objetivo verificar especificamente as influências dos transientes EFT e obter informações que sirvam para o mapeamento de anomalias de comunicação.

As ferramentas e configurações usadas em cada etapa foram as seguintes:

- Etapa 1: Rede baseada no protocolo CAN em topologia de barramento;
- Etapa 2: Injeção de EFT com um novo hardware desenvolvido;
- Etapa 3: Detalhamento das restrições temporais para o sistema de controle de suspensão ativa/Implementação do controle no software Vector CANoe;
- Etapa 4: Geração de tráfego com o software Vector CANoe, em conjunto com hardware VN8900 e módulos VN8970 para CAN;
- Etapa 5: Análise de comunicação com o software Vector CANAnalyzer; e
- Etapa 6: Geração de resultados por meio de gráficos e análises do impacto de EFTs no desempenho da rede;

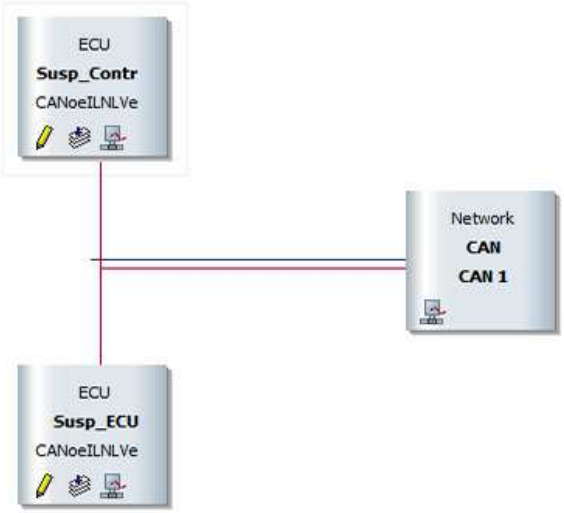
Para estudar os efeitos da falhas EFT em uma rede CAN, optou-se pelo sistema de controle de suspensão ativa, adaptado do trabalho de (MICHELIN, 2014). Esta tarefa compõe a etapa 1 do método de teste ilustrado na Figura 16. Esse sistema foi escolhido por se tratar de um sistema de segurança crítica, o qual depende de mensagens trafegando pelo protocolo de comunicação para atender às suas restrições temporais. O sistema de controle é baseado em um modelo de suspensão (planta) muito conhecido e utilizado na literatura, o modelo *quarter-car-model* (POUSSOT-VASSAL, 2008). Um dos principais e mais relevantes parâmetros é a deflexão da suspensão X_{def}, que aqui é considerada para avaliar o desempenho do sistema de controle.

Para estes experimentos o controlador adotado foi o controle de "realimentação de estados baseada na técnica de dados amostrados", correpondente ao terceiro controlador desenvolvido em (MICHELIN, 2014), mas que nesta pesquisa foi adaptado para o

protocolo CAN. Maiores detalhes sobre o controle de suspensão ativa, o modelo *quarter-car-model* e técnica de controle usada nos testes serão apresentados no Capítulo 5, quando será apresentado o estudo de caso que avalia a metodologia de teste proposta nesta tese em conjunto com técnicas de modelagem e diagnóstico de falhas.

Para avaliação experimental deste estudo sobre o impacto do EFT, foram utilizados dois módulos de hardware Vector VN8910A equipados com um módulo plug-in VN8970. Cada módulo permite a configuração de 4 nós físicos e mais 4 nós emulados, considerando uma rede CAN. Esse hardware permite a configuração de redes híbridas, podendo ainda dentro destes limites usar nós CAN, LIN e FlexRay com os mesmos módulos. Neste cenário do estudo de caso são considerados três nós (sensor, atuador e controlador). Um dispositivo VN8970 é responsável por simular a planta, recebendo e enviando dados da suspensão (sensor e atuador), operando em modo autônomo. O outro dispositivo VN8970 implementa o controlador e registra os dados de comunicação para análise. A rede CAN foi modelada usando a plataforma Vector CANoe, com uma topologia em barramento, como pode ser visualizado na Figura 18.

Figura 18 – Configuração da rede CAN nos testes de injeção EFT.



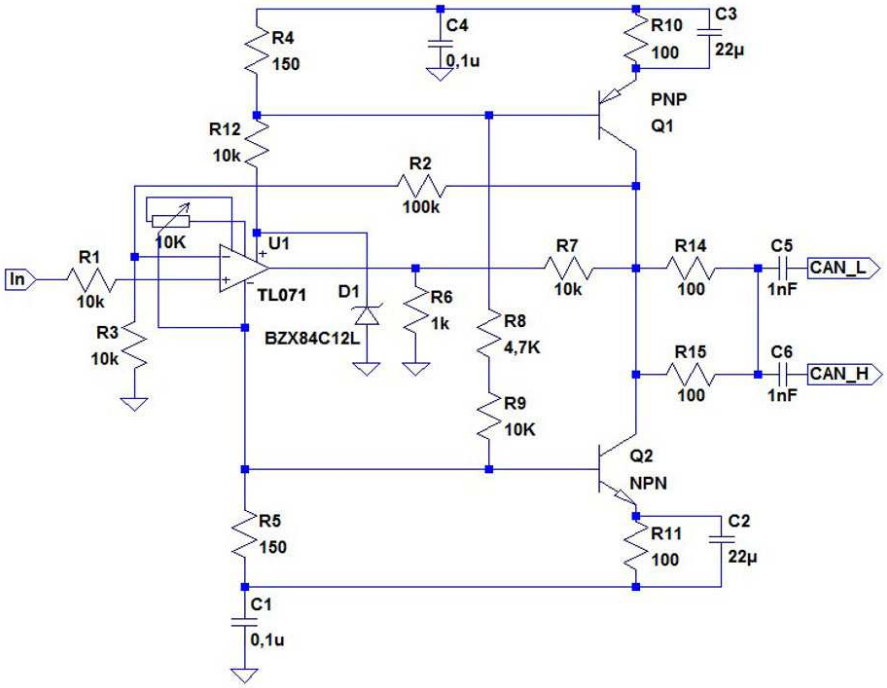
Fonte: Autor.

A ECU “Susp-ECU” implementa o modelo *quarter-car-model*, enquanto a ECU “Susp-Contr” implementa o controlador. O principal objetivo deste teste é analisar a suscetibilidade de uma rede CAN na presença de injeção de EFT, verificando o impacto que ela produz no no tráfego de mensagens críticas. Consequentemente, também foi analisado o efeito do EFT no desempenho do sistema de controle devido ao aumento no atraso entre as mensagens, ou em caso de perda do pacotes.

Para a parte de injeção EFT do experimento, um circuito específico foi projetado. Transientes EFT consistem em rajadas com amplitudes de tensão variáveis geradas em redes CAN por diferentes fontes de ruído (exemplo: ignição, farol, ar condicionado e também outros equipamentos de terceiros). A fim de verificar se foram produzidos distúr-

bios, de acordo com a etapa 2 do método de teste (Figura 16), foi necessário desenvolver um circuito específico para a injeção dos pulsos EFT, o qual é apresentado na Figura 19.

Figura 19 – Circuito para injeção de transientes EFT.



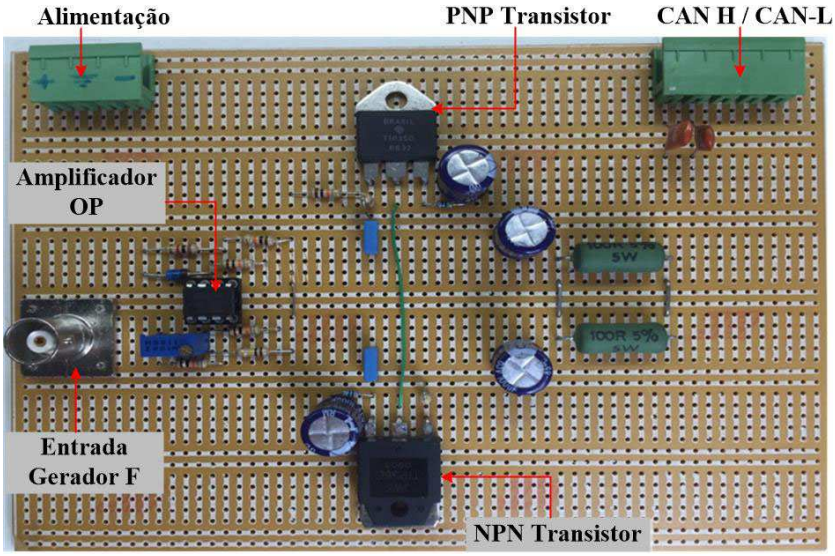
Fonte: Autor.

O circuito de injeção EFT consiste de um amplificador de tensão, usando um amplificador operacional (circuito integrado TL071), alimentado entre -15 e 60 Vdc. A estratégia para permitir o funcionamento do amplificador nesta faixa de tensão é o uso de um divisor de tensão por 10 na saída, em conjunto com uma configuração de não inversor com ganho 10. A Figura 20 apresenta a placa prototipada para a Injeção EFT destacando alguns dos componentes principais.

Com o objetivo de garantir que o valor de tensão esteja correto, um diodo zener foi usado para fixar este valor a 12 Vdc no terminal positivo do amplificador operacional. Isso permite que este sistema use um sinal de entrada de 15 V de pico e forneça a saída de até 55 V de pico. O sinal original veio de um gerador de função arbitrário configurado para fornecer picos com duração de 15us (microsegundos) a 2 ms (milisegundos) e atraso de 5 a 10 ms (milisegundos).

Os tempos de borda de subida e descida do circuito são de aproximadamente 20 ns (nanosegundos) de acordo com os transistores utilizados (modelos TIP35C e TIP36C), mas é importante destacar que o mesmo circuito pode ser atualizado usando-se transistores mais rápidos, para assim atender outras especificações. Com essa atualização é possível atender aos 5 ns de tempo de borda de subida/descida, que são especificados na norma IEC 61000-4-4. Por outro lado, a norma IEC TS 62228 não especifica os tempos de borda de subida e descida, apenas enfatiza como o método de teste e a medição

Figura 20 – Placa que foi prototipada para o circuito de injeção EFT.



Fonte: Autor.

devem ser realizados, tarefas estas que foram realizadas com sucesso neste experimento. O ponto principal deste estudo é mostrar como a injeção de EFT pode ser realizada, seguindo as orientações das normas, permitindo a análise do impacto deste tipo de falhas em redes CAN, para assim desenvolver mecanismos que possam detectar e mitigar esse problema. O circuito desenvolvido é semelhante ao circuito proposto em trabalho recente (FONTANA; HUBING, 2015), que também é baseado na mesma norma, mas limitado ao teste de susceptibilidade dos transceptores ao EFT em uma rede CAN embarcada, com uma unica PCI composta por dois nós, e ainda não considerando um sistema de controle afetado.

As diferenças entre o circuito de teste desenvolvido para com as normas IEC 61000-4-4 e IEC TS 62228 não afetam os objetivos e resultados do estudo apresentado, uma vez que permitem a análise dos transientes EFT na rede de comunicação sob outra perspectiva, considerando o nível de sistema de controle. Conforme detalhado em (PANNILA; EDIRISINGHE, 2014), as falhas do tipo EFT não possuem um padrão ou comportamento específico, são esporádicas e podem ter tempos variados considerando as bordas de subida e descida dos sinais. Os padrões de referência sugerem especificações base para os testes, mas isso não significa que eles não possam sofrer mudanças ou adaptações, porque nem sempre estes atendem as particularidades de todos os cenários de teste. Nesse sentido, as alterações realizadas neste experimento permitiram que a injeção de EFT fosse realizada em uma rede intra-veicular real, com um sistema de controle em operação, baseado em uma plataforma de desenvolvimento usada pela indústria automotiva. Essa análise também permite verificar a importância da modelagem prévia dos efeitos negativos de algumas falhas transientes.

Tendo o circuito definido e especificado, para determinar a imunidade e suscetibili-

dade da rede CAN aos transientes EFT, foram injetados distúrbios na rede com amplitudes de 19, 39 e 57 Volts de pico, tendo como base os experimentos e os registros deste tipo de falha no estudo de (PANNILA; EDIRISINGHE, 2014). Para todos os processos de injeção de falhas, os dados de comunicação e tráfego da rede CAN foram registrados em logs para análise posterior.

De acordo com estudo de caso especificado para a etapa 3 do método (sistema de controle de suspensão ativa), as mensagens que transmitem as informações que compõem a lei de controle são cíclicas e têm um período de 5 ms. Esse intervalo de tempo é crucial para o sistema de controle, uma vez que períodos mais longos podem levar o sistema a uma condição instável. Portanto, essa restrição é analisada após a injeção dos transientes EFT. O método de teste aplicado requer que o processo de comunicação entre os nós seja registrado em condições normais de operação, sendo assim, todo o tráfego de comunicação é monitorado e registrado pelo módulo Vector VN8910A, que é a interface de barramento utilizada durante esse experimento (etapas 4 e 5 do método). Esta ferramenta suporta CAN 2.0b e ISO 11898-2 (CAN de alta velocidade com taxas de transmissão até 1 Mbit/s). Conforme apresentado na Figura 17, os terminais para a injeção de EFT (CAN-H e CAN-L) são conectados à placa prototipada usando um cabo CAN padrão com terminação de 120 Ohms.

4.2.1.2 Resultados

De acordo com o procedimentos de teste definidos foi efetuado o processo de análise de impacto dos EFTs sobre restrições de tempo do sistema de controle (etapa 6 do método aplicado). Os testes foram realizados seguindo o estudo detalhado sobre as ocorrências de transientes elétricos rápidos apresentadas em (PANNILA; EDIRISINGHE, 2014). Assim, a Tabela 2 apresenta sequencia de injeções de falhas que foi efetuada, levando em consideração os tempos de rajada (burst) e amplitudes de tensão medidas nos experimentos.

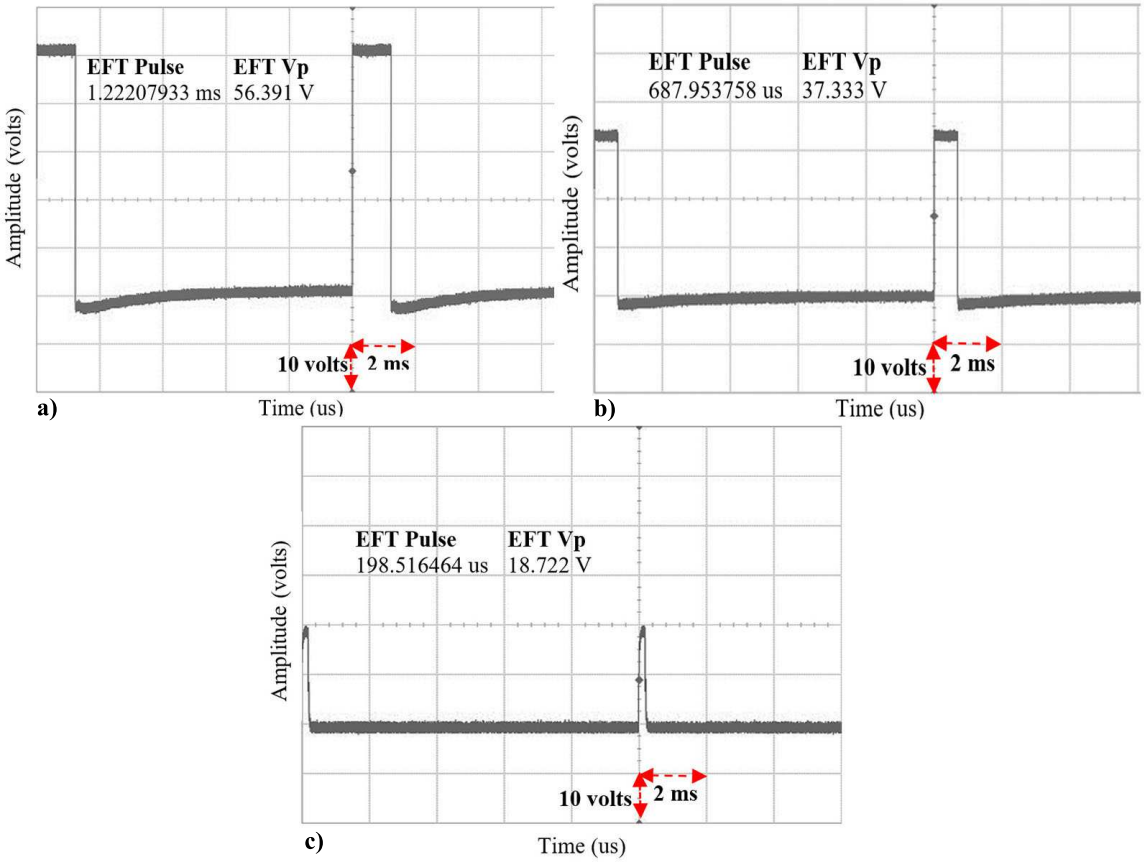
Tabela 2 – Parâmetros de referência para a Injeção de EFT. Baseado em (PANNILA; EDIRISINGHE, 2014).

Parâmetro	Operação	Limite
Duração de Rajada	Ignição	1.2 ms
	AC	687 us
	Luzes	198 us
Amplitude	Ignição	57 V
	AC	37 V
	Luzes	19 V

Como especificando anteriormente as injeções de EFT foram geradas usando um gerador de onda arbitrário. Para verificar a interconexão e efetividade de geração dos sinais pelo gerador de função foi realizada uma aferição prévia dos pulsos gerados. A Figura 21

a), b) e c) apresenta a sequência de medição desses pulsos gerados pelo osciloscópio.

Figura 21 – Medições dos pulsos EFT geradas pelo Osciloscópio. a) 57Vp e 1.2ms; b) 37Vp e 687us; c) 19Vp e 198us



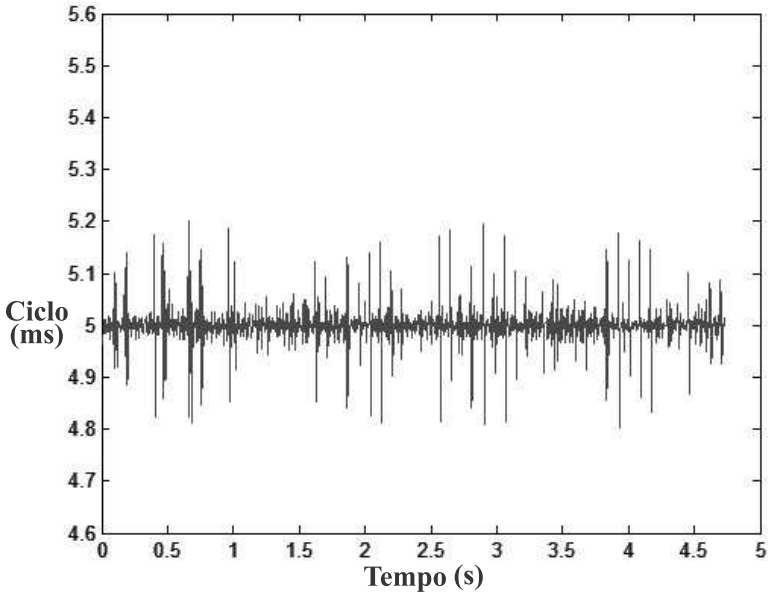
Fonte: Autor.

Tendo como base essa verificação dos sinais gerados, foi executada uma sequência de testes de comunicação, verificando a periodicidade das mensagens da lei de controle (parâmetros computados por meio de um conjunto de mensagens recebidas da planta, dentre eles a deflexão da suspensão) com 5 ms de intervalo entre as rajadas EFT. Após estes experimentos os logs de comunicação foram registrados pelo software Vector CAN Analyzer e os gráficos correspondentes foram gerados com auxílio do MATLAB, que em seguida foram o suporte para a aplicação das métricas de desempenho.

Para fins de comparação, antes de realizar a sequência de teste, foi realizada uma medição do ciclo de comunicação da rede CAN sem a injeção dos transientes EFT. Vários testes e medições foram realizados para fins de observar se o comportamento e as variações na rede se repetiam de forma similar. O intervalo de amostragem para a gravação de logs foi de 5 segundos, pois como foi usado um intervalo pequeno entre as injeções EFT, e durante esse tempo vários laços de controle são concluídos, um grande volume de registros para a análise foi gerado. De acordo com (FONTANA; HUBING, 2015), análises de susceptibilidade a EFT são realizadas em curtos períodos de estresse nos componentes, tendo em vista ser um tipo de falha que só acontece em curtos espaços de tempo.

A seguir as figuras 22, 23, 24 e 25 apresentam os gráficos gerados a partir dos registros de comunicação da rede, onde foram calculados os atrasos entre as mensagens da lei de controle, sem a injeção dos transientes EFT e com EFT nas três amplitudes de referência usadas para testes de susceptibilidade.

Figura 22 – Gráfico do teste com medição do ciclo de comunicação sem a injeção de EFT.



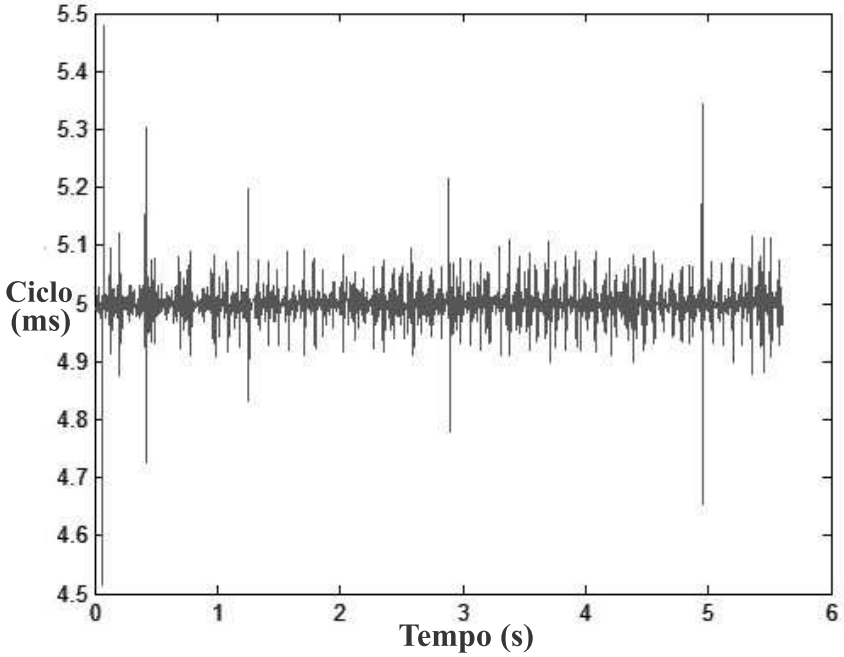
Fonte: Autor.

De acordo com o gráfico apresentado na Figura 22, para o período amostrado, o ciclo de comunicação medido varia entre 4,8 e 5,2 ms. Essa oscilação ocorre devido à variação no tempo de recebimento de mensagens que compõem a lei de controle no ciclo anterior, em relação as mensagens que compõem e são enviadas na próxima lei de controle. O primeiro teste realizado é representado pelo gráfico da Figura 23, que apresenta o desempenho da rede de comunicação com a injeção de pulsos EFT de 19 Volts de amplitude e com 198 microsegundos de duração de rajada (*burst*). De acordo com os picos de variação no ciclo de comunicação observados na Figura 23, é possível observar que o impacto na comunicação com este nível de EFT não é muito frequente, mas significativo em termos de aumento de atraso em alguns ciclos de comunicação, com variações do ciclo médio entre 4,7 e 5,5 ms.

A Figura 24 apresenta o segundo teste realizado, verificando o impacto de transientes de amplitude maior, neste caso com a injeção de 37 Volts de amplitude de EFT com duração de *burst* de 687 microssegundos.

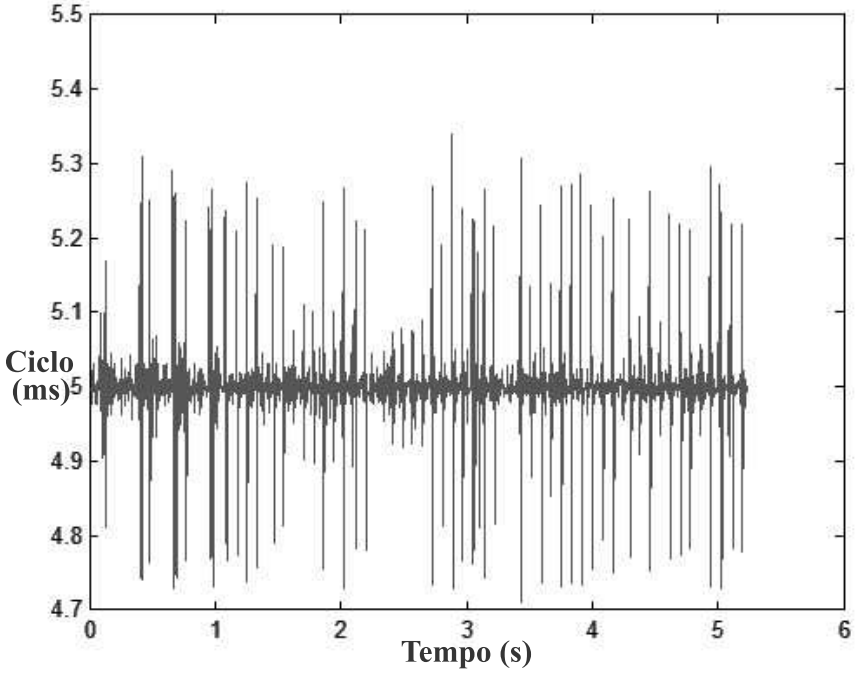
Pode ser observado na Figura 24, de acordo com o aumento na variabilidade no tempo do ciclo de comunicação, caracterizados pelo aumento no picos de sinais no gráfico, que vários ciclos foram afetados pela injeção de transientes EFT, aumentando também a frequência da ocorrência destes atrasos. O ultimo teste realizado é apresentado na Fi-

Figura 23 – Gráfico com as variações do ciclo de comunicação com EFT de 19Vp.



Fonte: Autor.

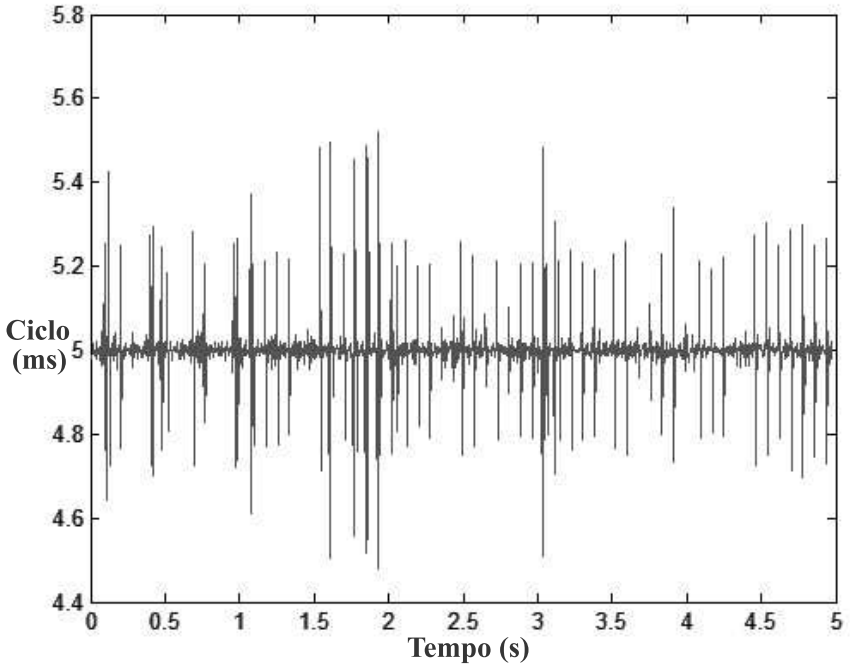
Figura 24 – Gráfico com as variações do ciclo de comunicação com EFT de 37Vp.



Fonte: Autor.

gura 25, que apresenta o resultado da injeção de EFTs com 57 Volts de amplitude e com duração de 1,2 milissegundos de *burst*.

Figura 25 – Gráfico com as variações do ciclo de comunicação com EFT de 57Vp.



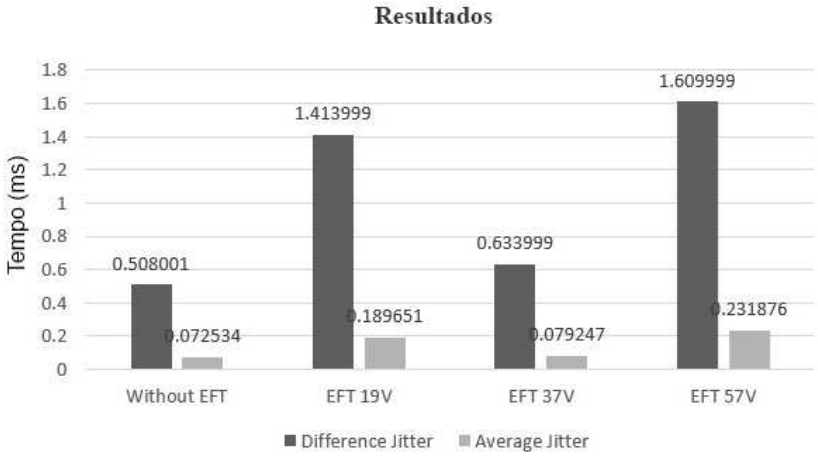
Fonte: Autor.

Nos testes de comunicação com injeção de EFT 57 Volts é possível observar um aumento nos atrasos de comunicação, tanto em frequência quanto em amplitude dos sinais, com uma variação nos ciclos de comunicação entre 4,5 e 5,6 ms. Na Figura 25, é possível verificar que a quantidade de picos de variação é menor quando comparado com a Figura 24, mas neste último teste com maior amplitude, gerando atrasos maiores entre vários ciclos de comunicação. Estes resultados indicam a suscetibilidade da comunicação CAN aos transientes EFT, como já destacado em outros trabalhos recentes, porém agora com uma análise estendida para o impacto no desempenho da rede de comunicação e sistemas de controle específicos.

Para correlacionar e analisar os experimentos realizados este estudo levou em consideração as métricas de variação no atraso médio (*average jitter*) e também a variação nos piores casos (*difference jitter*), também consideradas nas pesquisas de (NAHAS; PONT; SHORT, 2009) e (BURNS; DAVIS, 2013). O *difference jitter* é obtido pela subtração do melhor tempo de transmissão (tempo mínimo) pelo pior caso de transmissão (tempo máximo) a partir dos registros e medições do período de amostragem. O *average jitter* é representado pelo desvio padrão nos registros de tempo médio de transmissão das mensagens. A Figura 26 apresenta os resultados obtidos desta análise.

Com base nestes resultados é possível concluir que os transientes EFT podem afetar as restrições temporais de aplicações críticas, reforçando as análises dos trabalhos rela-

Figura 26 – Resultados dos testes de injeção EFT baseados na variação do *Jitter*.



Fonte: Autor.

cionados e confirmando a importância de verificar as falhas que causam degradação de desempenho e confiabilidade dos protocolos de comunicação. Os resultados demonstraram um aumento no *Average Jitter* de 161% com 19 Vp, 9,25% com 37 Vp e 219% com 57 Vp. Estes resultados podem variar um pouco devido a rajada/burst EFT poder coincidir mais ou menos vezes com os ciclos de controle em diferentes testes de injeção. No entanto, este fato não prejudica os resultados dos testes, cujo o objetivo principal é analisar a suscetibilidade do protocolo CAN aos transientes elétricos.

Assim, conclui-se que os experimentos realizados detalharam um novo método de teste e injeção de falhas, demonstrando neste estudo como os transientes EFT podem afetar o processo de comunicação das redes CAN. Estudos recentes mostraram a suscetibilidade de transceptores CAN aos EFT, mas o presente estudo contribuiu com testes focando na análise destes efeitos em um sistema de controle veicular especifico baseado no protocolo CAN. A verificação foi centrada no estudo de caso de um sistema de controle de suspensão ativa, onde os efeitos de degradação podem afetar o desempenho do sistema, pois este, possui um intervalo de comunicação que deve ser cumprido de forma rígida. Os resultados mostraram que os atrasos são significativos e podem afetar a confiabilidade do sistema de controle e do protocolo de comunicação.

Com o objetivo de comparação e análise deste estudo em outros protocolos de comunicação que podem ser usados na indústria automotiva, outro estudo de injeção EFT também foi efetuado, considerando também melhorias no hardware de injeção de falhas e verificação do impacto dos transientes em cenários de maior uso da largura de banda do canal. Esse estudo é apresentado na Seção 4.1.2, porém agora considerando um protocolo emergente nas redes veiculares atuais, o protocolo CAN-FD.

4.2.2 Análise de impacto de EFT em redes CAN-FD

Com base no estudo desenvolvido e apresentado na subseção anterior, um trabalho mais aprofundado foi realizado com base no hardware de injeção de falhas. O foco central deste novo estudo foi melhorar o hardware de injeção de falhas, atender completamente as normas e analisar a susceptibilidade a falhas em protocolo recente usado em redes automotivas, neste caso, o emergente CAN-FD. Como destacado anteriormente, devido à complexidade crescente dos sistemas eletrônicos e à demanda por computação distribuída, cada vez mais sistemas críticos de tempo-real são afetados por interferências, fato que pode levar a sérios problemas, os quais motivam o presente estudo.

Neste contexto, os novos experimentos realizados tiveram o objetivo de determinar com maiores detalhes o impacto de transientes elétricos rápidos - EFTs em redes intra-veiculares. Assim, o novo estudo procurou atender aos seguintes padrões: IEC 62228 (*EMC evaluation of CAN transceivers*) e ISO 26262 (*Road Vehicle Functional Safety Services*) com suas respectivas bases, os padrões IEC 61000-4-4 (*Testing and measurement techniques - Electrical fast transient/burst immunity test*) e ISO 7637-3 (*Electrical transient transmission by capacitive and inductive coupling through means other than supply lines*).

O hardware de teste projetado para a injeção de falhas é composto por três nós de processamento desenvolvidos pela empresa *Texas Instruments Incorporated* (TEXAS INSTRUMENTS, 2019), que usa um processador da série Hercules (TMS570LS3137), o qual atende os requisitos de segurança (safety) de redes intra-veiculares, de acordo com os padrões IEC 62228 e ISO 61508. Estes padrões também servem de base para determinar se um determinado sistema de controle está de acordo com o AUTOSAR (*AUTomotive Open System ARchitecture*) (AUTOSAR, 2019), que é um padrão mais focado na conformidade da arquitetura de software utilizada em sistemas automotivos. Com este hardware é possível desenvolver rotinas de teste para verificar a confiabilidade de sistemas de controle, seguindo os três protocolos de comunicação mais usados atualmente, CAN, CAN-FD e FlexRay. A segunda parte do hardware foi desenvolvida especialmente para a injeção de falhas nos respectivos transceptores de cada nó conectado na placa. Para a comunicação dentro do padrão do protocolo CAN-FD (ISO 11898-1 e 11898-2) usou-se um controlador externo, o MCP2517FD (Microchip), bem como também o transceptor TCAN332G (*Texas Instruments*).

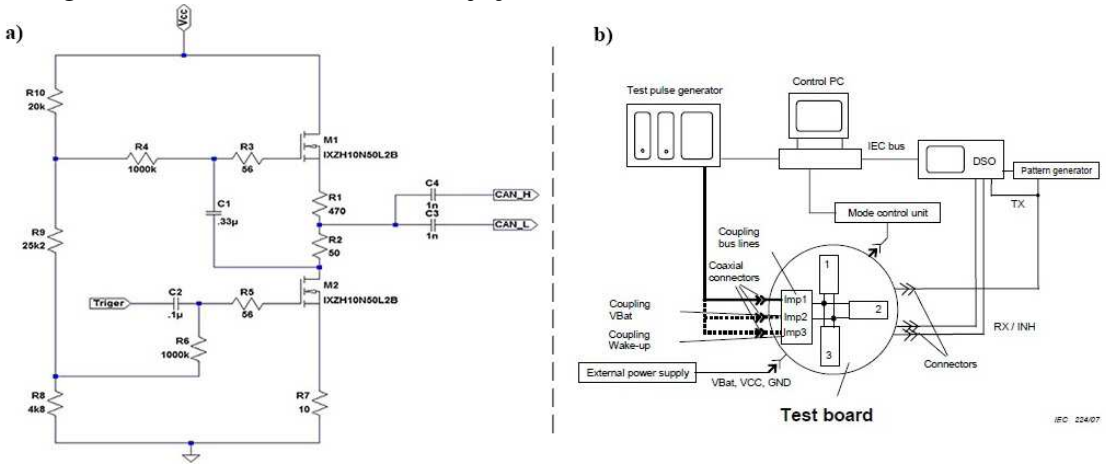
A fim de permitir a realização de todos os testes (com os protocolos CAN, CAN-FD e FlexRay) e cumprir as restrições das normas estudadas, o novo hardware projetado fez uso de componentes eletrônicos mais robustos, especialmente o processador Hercules TMS570LS3137, pois este possui dois canais FlexRay e quatro canais CAN. Este componente permite a comunicação FlexRay tanto no canal A quanto no canal B. Essa flexibilidade permitiu o projeto de uma placa que integrou diferentes nós controladores para os referidos protocolos. Devido o objetivo do presente estudo ser as análises de

susceptibilidade na rede intra-veicular, maiores detalhes do projeto e desenvolvimento deste hardware de teste podem ser obtidos na dissertação de mestrado de Daniel Pohren (POHREN, 2020).

Todos experimentos com o protocolo CAN-FD seguiram os mesmos padrões, sequencia de passos e sistema de controle usado na pesquisa com o protocolo CAN (ROQUE *et al.*, 2017). Esses passos seguem o método de testes apresentado na Figura 16, com destaque a etapa 2 (hardware de injeção de falhas), que neste caso atende perfeitamente as condições impostas nas normas, ainda estendendo o tipo de análise não somente para os transceptores CAN, mas também para os efeitos de degradação no sistema de controle objeto do estudo de caso.

A Figura 27 a) apresenta o diagrama esquemático do gerador de pulsos EFT que é empregado nos testes.

Figura 27 – a) Novo circuito de injeção EFT. b) Placa de acordo com a IEC 62228.



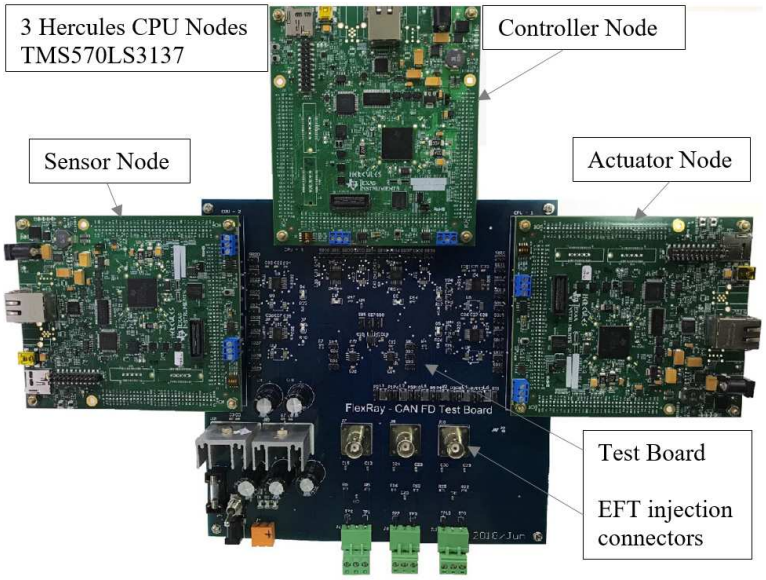
Fonte: Autor.

De acordo com o projeto apresentado por (POHREN *et al.*, 2019), tanto o circuito de injeção EFT quanto a placa que reúne os transceptores e as linhas de comunicação CAN, CAN-FD e FlexRay, devem cumprir os padrões IEC 62228 e ISO 26262. Assim, o uso de um circuito de condicionamento de pulso digital foi projetado em conjunto com uma seção de potência usando o transistor YXIS MOSFET modelo IXZH10N50L2B, que possui tecnologia L-MOS de baixa capacitância. Este transistor atende aos parâmetros de tempo de subida e descida de sinal (não atingido nos testes anteriores), bem como também permite a operação em altas correntes e tensões de trabalho.

Desta forma, a placa de teste foi projetada com três conectores de oitenta pinos, que permitem a conexão de cada CPU ou nó. As injeções de EFT são efetuadas diretamente através de um conector do tipo BNC, onde os três transceptores CAN-FD estão interconectados. Este mesmo esquema de ligação é usado para a ligação de alimentação para todos os transceptores e CPUs. Esse formato de conexão deve seguir o modelo apresentado na IEC 62228, que é apresentado na Figura 27 b).

Ainda segundo a norma a placa de teste deve conter pelo menos três tranceivers CAN, uma alimentação externa e incluir conectores coaxiais para os pontos de injeção dos pulsos EFT. A Figura 28 apresenta a placa de teste para injeção de pulsos EFT projetada no escopo da dissertação mestrado de Daniel Pohren.

Figura 28 – Ilustração da nova placa projetada.



Fonte: (POHREN, 2020).

Como observa-se na Figura 28, a placa de teste segue uma topologia em estrela e destaca a ligação conjunta dos três nós que fazer parte da rede de comunicação. A seguir são apresentados detalhes dos resultados obtidos durante os testes de injeção de falhas EFT em uma rede CAN-FD.

4.2.2.1 Procedimentos de teste e Resultados

Como destacado anteriormente os procedimentos de teste seguiram o método apresentado na Figura 16, tendo como base o mesmo sistema de controle de suspensão ativa, mas agora aplicado com o protocolo CAN-FD. A seguir são apresentados os nós configurados na rede e as referidas mensagens:

- Sensor: Dados do movimento vertical do veículo com base no conjunto massa-mola e no modelo quarter-car utilizado. Mensagens: BodyVertSpeed, SuspDeflection e TireDeflection.

- Atuador: Ajustes na posição vertical do conjunto de suspensão. Mensagem: SuspSetVertSpeed.

- Controlador: Computação dos apropriados níveis de ajuste nos parâmetros da suspensão que caracterizam a lei de controle. Mensagem: ControlLaw.

As mensagens definidas para cada nó são transmitidas ciclicamente (lei de controle com periodicidade de 5 ms) e representam as condições de operação típicas do sistema de controle utilizado. De acordo com as configurações de cada nó, a rede CAN-FD foi configurada para operar entre 1 e 4 Mbps, devido às características do hardware utilizado. Apesar do campo de dados do protocolo CAN-FD permitir tamanhos de mensagens de até 64 bytes, as mensagens utilizadas em todos os experimentos foram de 8 bytes, seguindo as definições do projeto inicial de (MICHELIN, 2014). A não modificação da estrutura original de mensagens do sistema de controle, também é importante para não gerar inconsistências que afetariam a correta análise de desempenho nos três protocolos abordados nesta tese (CAN, CAN-FD e FlexRay).

Outro ponto importante da análise das redes intra-veiculares é a utilização de diferentes cargas de ocupação da rede, para simular situações de alto tráfego, e verificar como falhas durante essas situações podem ser ainda mais críticas. Para tal, nesse estudo mensagens aleatórias para geração de tráfego, com diferentes campos de dados, foram inseridas para aumento da carga da rede. Durante os experimentos a carga da rede permaneceu entre 30% e 60%, justamente pela geração aleatória de mensagens.

Com estas definições os experimentos objetivam determinar as influências dos transientes EFT na rede CAN-FD, registrando também logs de toda a comunicação, durante o período de amostragem. Assim como no experimento anterior com o protocolo CAN, as seguintes ferramentas e configurações foram utilizadas:

- Etapa 1: Vector VN8910A com módulo VN8970, para programação dos scripts de teste e definição da topologia em barramento com o protocolo CAN-FD.
- Etapa 2: Injeção de pulsos EFT com o novo hardware desenvolvido de acordo com as normas IEC 62228 and ISO 26262.
- Etapa 3: Geração de tráfego na comunicação, análise da comunicação durantes as injeções e registro de logs com o software Vector CANoe.

Para a análise de susceptibilidade do protocolo CAN-FD, uma sequencia de pulsos EFT foram injetados de acordo com a configuração apresentada na tabela 3.

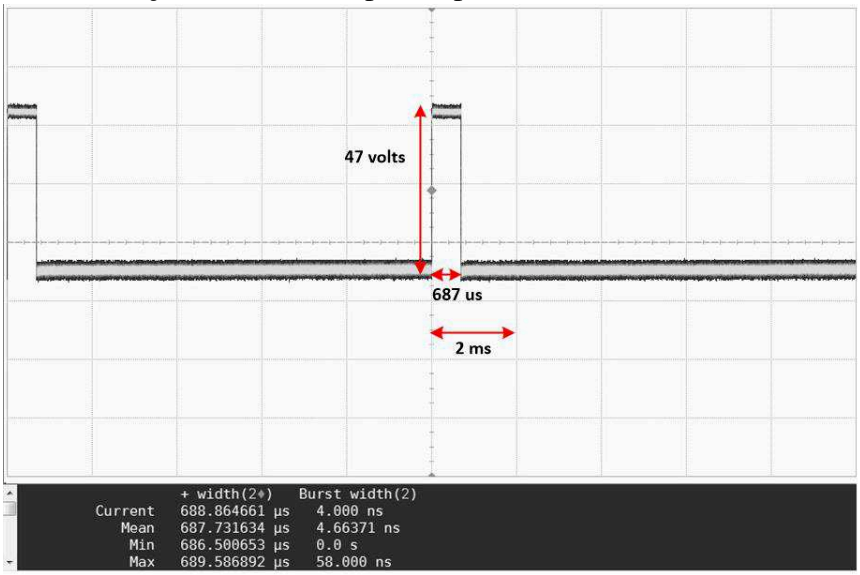
Tabela 3 – Parâmetros para a Injeção de EFT.

Tensão de pico	Duração da Rajada
47 volts	687 us
57 volts	1,2 ms
63 volts	500 us
67 volts	500 us

Para fins de verificar o atendimento aos requisitos da norma IEC 62228, principalmente em relação aos tempos de subida e descida dos sinais durante a geração dos pulsos

EFT, um registro dos sinais foi efetuado previamente durante a preparação dos experimentos. Esse registro pode ser visualizada na Figura 29, que mostra a injeção de um pulso de 47 volts com burst de 687 microsegundos, destacando que os tempos de borda ficaram nos limites de 5 ns definidos na norma.

Figura 29 – Medição no Osciloscópio do pulso de 47 volts com burst de 687 us.

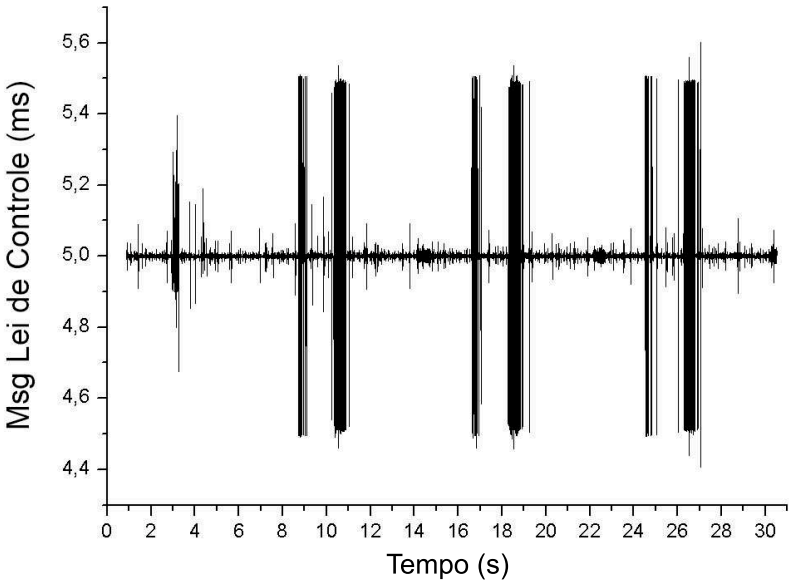


Fonte: Autor.

Em seguida uma série de experimentos foram conduzidos com os sinais e pulsos definidos, para assim avaliar as interferências deste tipo de falha no sistema de controle e consequentemente observando o comportamento da rede de comunicação com o protocolo CAN-FD. A mensagem central analisada é a mensagem da lei de controle, enviada pelo nó/ECU controlador, com base nas informações recebidas da ECU sensor. Os registros de comunicação foram analisados por meio do software CANoe/CANAnalyzer (VECTOR INFORMATIK, 2018), e em seguida, os gráficos correspondentes foram gerados para a análise. A computação das métricas de desempenho é realizada usando como referência os logs de comunicação gerados pelo software. Para fins de comparação, antes de gerar as sequências de teste foi realizada uma medição da comunicação sem a injeção dos pulsos EFT. Todas as medições foram realizadas com base em um período de amostragem de 30 segundos, com todos os registros de logs armazenados e analisados. A Figura 30 mostra essa medição inicial da rede CAN-FD, destacando a mensagem da lei de controle.

O gráfico da Figura 30 mostra o período de amostragem da comunicação com intervalos de ciclo variando entre 4,4 e 5,6 ms. Destaca-se nesse ponto que por meio das ferramentas usadas uma rede real é exemplificada, com ECUs conectadas por meio de hardware e cabeamento certificados, que caracterizam um cenário de aplicação automotiva. Nesta medição as flutuações são normais e frequentes, mas com variações pequenas que são tipicamente registradas durante a operação normal do sistema.

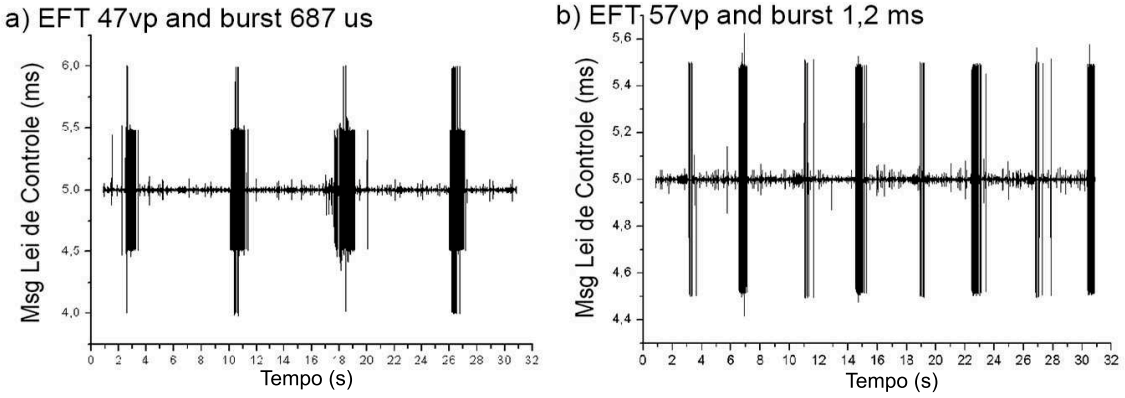
Figura 30 – Medição da rede CAN-FD registrada sem a injeção de pulsos EFT.



Fonte: Autor.

A razão da ocorrência destas flutuações é a variação no tempo entre pacotes recebidos na Lei de Controle anterior e os pacotes enviados na próxima Lei de Controle, pacotes estes, oriundos do grupo de mensagens que compõem os sensores da planta, os quais podem sofrer com interferências e distúrbios durante a comunicação. Para analisar estas possíveis interferências no desempenho, os gráficos das figuras 31 e 32 apresentam resultados das injeções de falhas realizadas, tendo como referência a mensagem da lei de controle.

Figura 31 – Registro da Lei de Controle na rede CAN-FD com EFT de 47 e 57 volts.



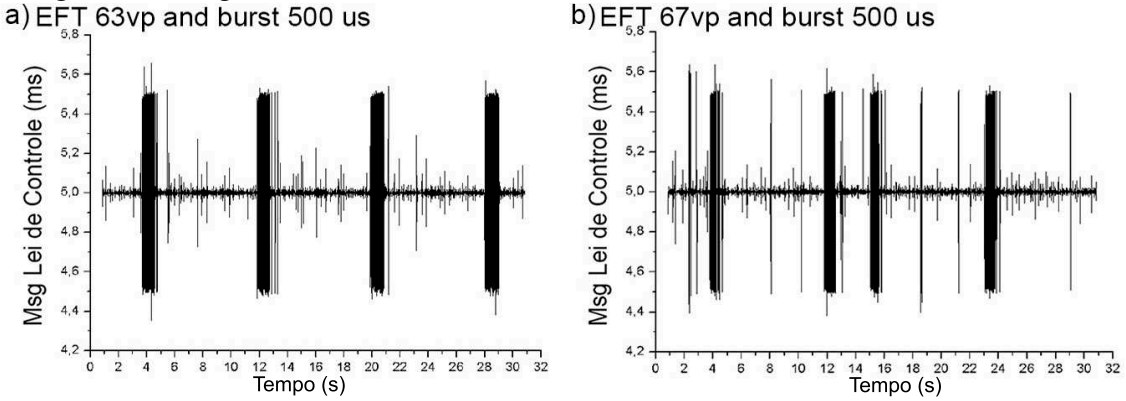
Fonte: Autor.

A Figura 31 a) apresenta o registro da comunicação no período de amostragem definido, com injeção de EFT de 47 volts e 687 microssegundos de duração de rajada. A Figura 31 b) apresenta o registro da comunicação com injeção de EFT de 57 volts e 1,2 milissegundos de rajada. Esses gráficos ilustram o impacto das injeções de falhas no ciclo de comunicação do sistema de controle, destacando os picos de atraso que ocorrem com mais frequência durante as transmissões de mensagens, variando entre 4 e 6 milissegun-

dos. Também é possível observar um aumento no número de ciclos com atraso maior de 5,2 milissegundos, bem como também, o aumento na frequência de ocorrência destas flutuações.

Em seguida a Figura 32 a) e b) apresentam os registros da comunicação com injeções de EFT de 63 e 67 volts de pico com rajadas de 500 microssegundos. Esse teste foi adicionado para verificar o comportamento da rede CAN-FD com EFTs de maior tensão de pico e com rajadas curtas, procurando estender o tipo de análise realizado nos experimentos anteriores, levando em conta que no trabalho de (PANNILA; EDIRISINGHE, 2014), registros de transientes com rajadas abaixo de 500 microssegundos também foram encontrados.

Figura 32 – Registro da Lei de Controle na rede CAN-FD com EFT de 63 e 67 volts.



Fonte: Autor.

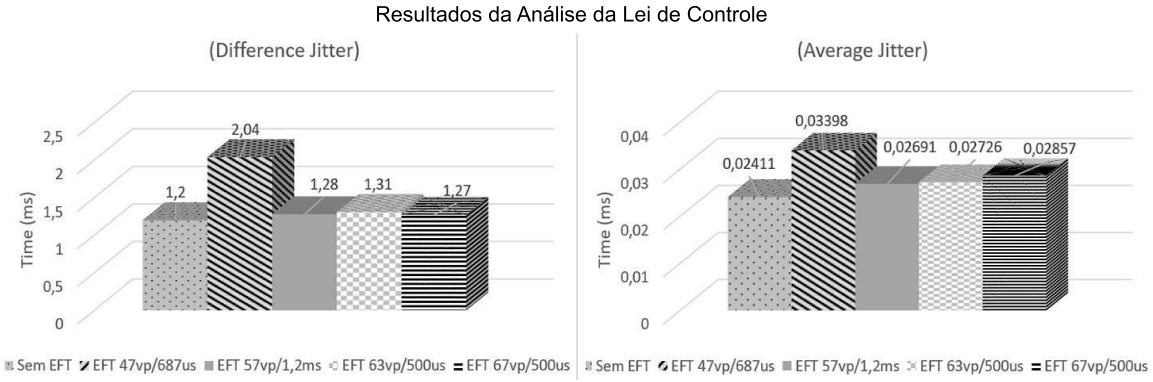
Analisando os experimentos realizados foi observado que, quando havia EFTs de 47 volts e 687 microssegundos de rajada, os efeitos em termos de atraso foram maiores, porém com menor frequência de ocorrência. Por outro lado, com injeções de EFT com tensões mais altas e rajadas menores, o efeito é menor na análise dos sinais, mas a ocorrência de atrasos nos ciclos de controle é maior. A razão para esse contraste é devido ao fato de que com rajadas maiores, as chances de EFTs gerarem interferências e retransmissões em uma determinada sequência de mensagens é maior do que em rajadas menores. Ou seja, os transientes precisam coincidir e afetar transmissões específicas, gerando disrupções na modulação de sinais do protocolo. Assim, foi observado que com uma tensão mais alta de injeção de EFT (acima de 60 volts), a frequência de atrasos nos ciclos de controle é esporádica, enquanto com quantidade e duração de rajadas maiores, pode-se observar uma grande quantidade de quadros de erro. A Figura 33 ilustra a tela de estatísticas do software Vector CANoe durante estes experimentos com maior tensão de EFT, podendo verificar o registro de 131 quadros de erro.

Figura 33 – Estatísticas obtidas no software CANoe durante injeções EFT de 67 volts.

Statistic	Current / Last	Min	Max	Avg
Busload [%]	30.16	30.16	88.47	31.79
Min. Send Dist. [ms]	0.000	n/a	n/a	n/a
Bursts [total]	7414	n/a	n/a	n/a
Burst Time [ms]	1.006	0.498	358.409	1.116
Frames per Burst	4	2	1428	4
Std. Data [fr/s]	1200	1200	3517	1265
Std. Data [total]	47719	n/a	n/a	n/a
CANStress	7554	n/a	n/a	n/a
Susp_Control_ECU	8033	n/a	n/a	n/a
Susp_ECU	32132	n/a	n/a	n/a
Unknown sender	0	n/a	n/a	n/a
Ext. Remote [fr/s]	0	0	0	0
Ext. Remote [total]	0	n/a	n/a	n/a
Errorframes [fr/s]	0	0	21	4
Errorframes [total]	131	n/a	n/a	n/a
Chip State	Active	n/a	n/a	n/a
Transmit Error Count	0	n/a	80	n/a
Receive Error Count	0	n/a	1	n/a
Transceiver Errors	0	n/a	n/a	n/a
Transceiver Delay [ns]	106	106	131	126

Fonte: Autor.

Figura 34 – Resumo dos resultados da análise de desempenho do protocolo CAN-FD.



Fonte: Autor.

Seguindo esta análise, a Figura 34 a) e b) apresentam um resumo dos dados de desempenho obtidos na rede CAN-FD para o sistema de controle que foi estudado. As métricas de avaliação seguiram os mesmos experimentos anteriores (*Difference Jitter* e *Average Jitter*), para fins de comparação em mesmos níveis nos diferentes protocolos.

Com base neste estudo, enfatiza-se a importância de mitigar e diagnosticar os efeitos de transientes elétricos rápidos na rede CAN-FD. Os experimentos realizados destacam a degradação de desempenho que causa atrasos e que muitas vezes pode ser crítico para sistemas de controle de tempo real.

Desta forma, considerando a métrica Difference Jitter, a Figura 34 a) sumariza os picos de atraso que afetam a Lei de Controle, com os seguintes acréscimos registrados: 41,18% com EFT de 47vp, 6,25% com EFT de 57vp, 8,4% com EFT de 63vp e 5,5% com EFT de 67vp. De outro modo, na Figura 34 b) são apresentados os registros de degradação de desempenho de acordo com a métrica Average Jitter, os quais foram os seguintes: 29,05%

com EFT de 47vp, 10,41% com EFT de 57vp, 11,56% com EFT de 63vp e 15,61% com EFT de 67vp. Esses dados confirmam os aspectos críticos apresentados na norma ISO 26262, devido ao fato de que estes efeitos de degradação podem acarretar em riscos que afetam a segurança e a confiabilidade de sistemas de controle distribuídos, como os que compõem as redes intra-veiculares.

Os resultados obtidos também destacam a degradação causada no emergente protocolo CAN-FD, o qual esta em estudo e poderá ser adotado pela indústria em breve. Conforme observado nos experimentos, os EFTs levam à disrupção dos sinais que sinalizam os bits durante a transmissão de mensagens, efeitos constatados durante a análise das mensagens que compõem a lei de controle. Assim, cada vez mais o diagnóstico e registro da ocorrência de falhas torna-se importante para contribuir com a redução da manutenção e também melhorar o projeto de novos sistemas de controle intra-veiculares.

4.2.3 Análise de impacto de EFT em redes FlexRay

De acordo com as análises de susceptibilidade a EFT realizadas anteriormente em CAN e CAN-FD, um terceiro experimento foi realizado com base em uma rede veicular utilizando o protocolo FlexRay (FLEXRAY CONSORTIUM, 2010). Esse último experimento é importante para verificar os efeitos dos transientes elétricos rápidos mesmo em protocolos mais robustos e projetados com características de tolerância a falhas, com largura de banda maior e com possibilidade de comunicação em canais de redundantes.

Como especificado anteriormente, o protocolo FlexRay tem como diferencial em relação aos outros a versatilidade, pois suporta a configuração de redes em diferentes topologias, como barramento, estrela e híbrida. O FlexRay foi desenvolvido para atender as demandas dos sistemas de controle e de segurança críticos (*safety-critical control systems*), aliando largura de banda e menor latência de comunicação. Exemplos destes sistemas críticos são os sistemas de controle eletrônicos (*X-By-Wire systems*), usados no controle de direção (*steer-by-wire*), controle de frenagem (*brake-by-wire*) e outros sistemas de assistência a direção. Mas apesar das características que proveem maior confiabilidade ao FlexRay, pesquisas recentes tem destacado a susceptibilidade à falhas físicas e erros de comunicação (DO SOUTO; PORTUGAL; VASQUES, 2016), (HUANG *et al.*, 2019), levando ao desenvolvimento de métodos e técnicas que procuram mitigar esses problemas (SAHA; ROY; RAMESH, 2016), (LIU; BAI; ZHEN, 2017), (LEE *et al.*, 2018). Neste sentido, devido aos recentes problemas observados com relação a diferentes interferências na comunicação, uma análise de susceptibilidade a EFT no FlexRay ainda não foi verificada, principalmente considerando seus efeitos nos sistemas de controle críticos.

Estas verificações motivam o presente estudo efetuando tais análises, com a aplicação do mesmo hardware projetado e usado nos testes com o protocolo CAN-FD (Figura 27), atendendo as respectivas normas de verificação de imunidade a transientes elétricos IEC 62228, IEC 61000-4-4 e ISO 7637-3. O hardware projetado possui a possibilidade de re-

alizar a comunicação em uma malha de controle, contendo um nó sensor, um nó atuador e um nó controlador, e ainda uma conexão externa para interconexão a outros barramentos. Especificamente para os testes com o protocolo FlexRay o hardware apresentado na Figura 28 possui 6 transceptores (3 para o Canal A e 3 para o Canal B), sendo assim, as injeções de falhas podem ser efetuadas de forma independente por meio de um conector BNC para cada canal de comunicação. A comunicação FlexRay é estabelecida por meio do transceptor TJA1080A da NXP (NXP SEMICONDUCTORS, 2019), que está em conformidade com a ISO 17458-2013 (ISO-17458, 2013) que define a estrutura de sistemas de comunicação com o protocolo FlexRay.

Assim como nos dois experimentos anteriores, todos os procedimentos de testes com o protocolo FlexRay seguem a mesma sequencia definida no método apresentado na Figura 16, utilizando também o mesmo sistema de controle de suspensão ativa como referência de mensagens trafegando no barramento.

4.2.3.1 Procedimentos de teste e Resultados

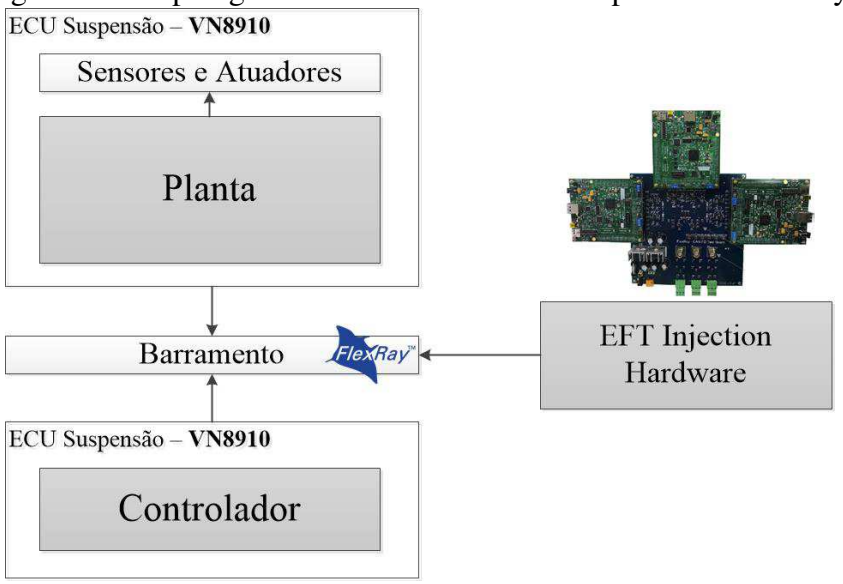
Por questão de coerência e geração de dados comparativos mais adequados, todos os procedimentos de teste com o protocolo FlexRay seguem a definição de grupo de mensagens do sistema de suspensão ativa, de acordo com o estudo de caso realizado nos protocolos CAN e CAN-FD. As principais mensagens são: dados sensoriais de movimento vertical do veículo (nó sensor), sinais de ajustes na posição vertical do veículo (atuador) e a computação dos adequados níveis de ajuste nos parâmetros do sistema de controle, os quais caracterizam a Lei de Controle.

De acordo com as configurações destes três "nós", a rede FlexRay foi configurada para operar no Canal A a 10 Mbps. Devido a maior robustez do protocolo (características do circuitos integrados utilizados) e operando com uma largura de banda maior, as variações médias no tempo do ciclo de controle são da ordem de poucos nanossegundos. De acordo a pesquisa de (MICHELIN, 2014), que é a referência deste estudo com o protocolo FlexRay, informações observadas no experimento 3 (com avaliação da periodicidade das mensagens de controle) mostraram uma variação média de aproximadamente 25 nanossegundos, tendo um limite superior na ordem de 70 nanossegundos. Neste experimento, como o objetivo é analisar o efeito dos transientes nas mensagens periódicas, o segmento dinâmico do protocolo foi mantido em 100% de ocupação. O segmento estático onde as mensagens periódicas são alocadas, foi testado com três variações de carga de ocupação, 2% (somente o sistema de controle), 30% e 80% (com geração de tráfego por software). Como o objetivo é a verificação da susceptibilidade do protocolo aos transientes elétricos não foi necessário usar ambos os canais.

Outro fator importante a destacar, é que neste procedimento de testes a configuração da rede FlexRay, bem como toda a programação dos "nós" foi realizada na plataforma Vector CANoe com as interfaces VN8910A, com conexão via cabo FlexRay de 50 cm até

o hardware de injeção de falhas. As configurações de *time slots* de mensagens usadas no protocolo FlexRay, segmento estático, seguiram as mesmas especificações apresentadas no trabalho de (MICHELIN, 2014), com ciclo de 5ms, sendo a transmissão dos estados da planta no slot 20 e a lei de controle no slot 40. O código fonte (linguagem CAPL) usado nos testes pode ser visualizado no Apêndice A. A seguir a Figura 35 apresenta a topologia em barramento usada nos experimentos.

Figura 35 – Topologia utilizada nos testes com o protocolo FlexRay.



Fonte: Autor.

Nesta configuração os dados amostrados dos sensores são *time-triggered*, disponibilizando suas amostras em intervalos constantes. Com estas definições os experimentos objetivam determinar as influências dos transientes EFT na rede FlexRay, registrando também logs de variação de atrasos na comunicação. As seguintes ferramentas foram utilizadas:

- Etapa 1: Vector VN8910A com módulo VN8970, para programação dos scripts de teste e definição da topologia em barramento com o protocolo FlexRay.
- Etapa 2: Injeção de pulsos EFT com o hardware de injeção de transientes EFT.
- Etapa 3: Geração de tráfego na comunicação, análise da comunicação durantes as injeções de falhas e registro de logs de desempenho com o software Vector CANoe.

Novamente, para fins de análise de susceptibilidade do protocolo, uma sequencia de pulsos EFT foram injetados de acordo com a configuração apresentada na tabela 4.

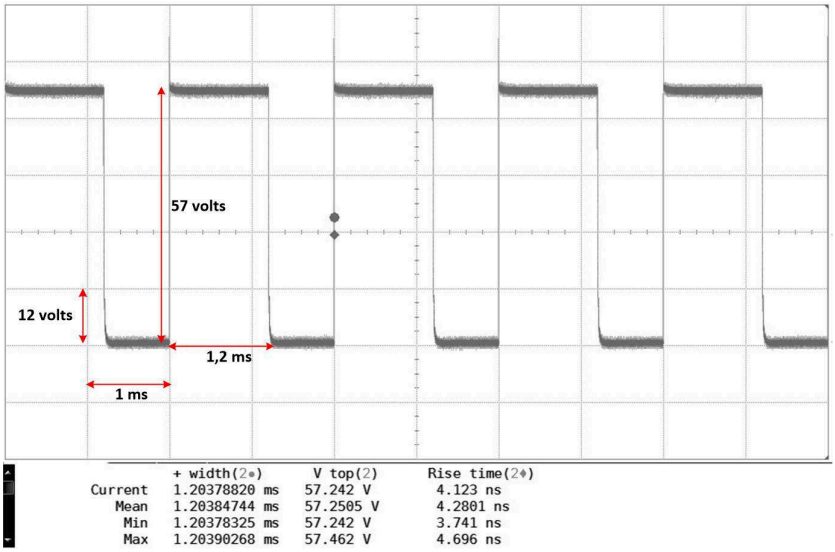
Para fins de verificar se os tempos de subida e descida estão adequados e conforme as especificações da normas de teste, um registro dos sinais de uma injeção EFT foi efetuado previamente durante a preparação dos experimentos. Esse registro obtido no osciloscópio

Tabela 4 – Parâmetros para a Injeção de EFT no protocolo FlexRay.

Tensão de pico	Duração da Rajada
47 volts	687 us
57 volts	1,2 ms
63 volts	500 us

pode ser visualizado na Figura 36, que mostra a injeção de um pulso de 57 volts com burst de 1,2 milissegundos.

Figura 36 – Análise prévia de uma injeção EFT observada com o osciloscópio.



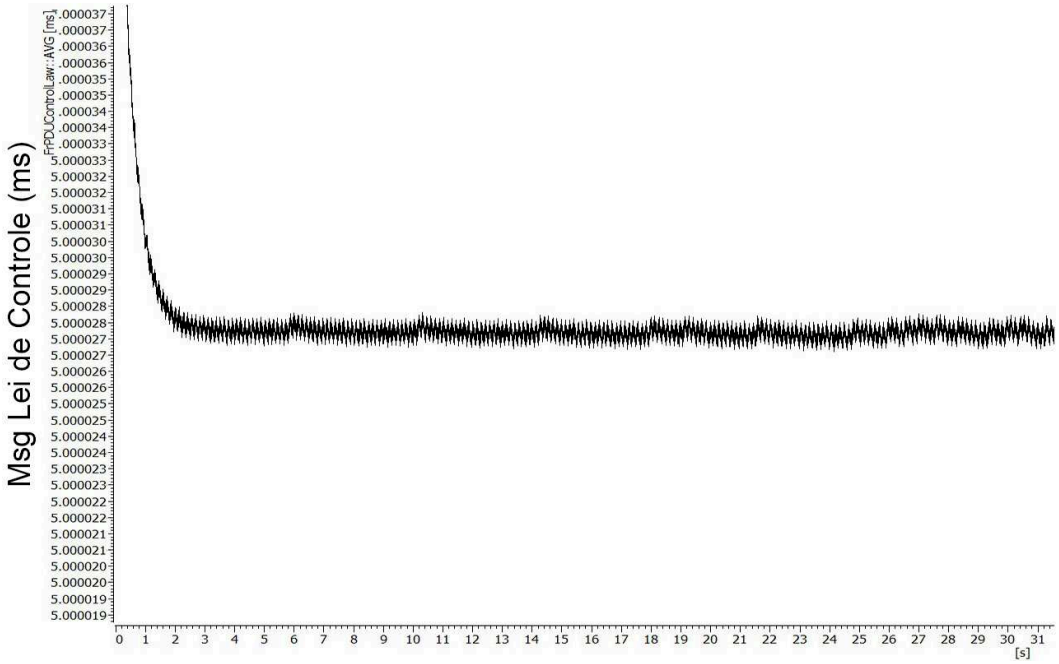
Fonte: Autor.

Após essa observação inicial e o processo de preparação do experimento, uma série de injeções EFT foi conduzida com os sinais e pulsos definidos na Tabela 4, para assim avaliar o impacto dos transientes elétricos na operação normal do sistema de controle, observando o comportamento da rede FlexRay, armazenando logs com o registro da variação no ciclo de comunicação. A mensagem central analisada é a mensagem da lei de controle, enviada pela ECU controlador, com base nas informações da planta recebidas da ECU sensor. A análise da comunicação foi realizada pelo software CANoe, com a plotagem dos gráficos correspondentes a cada injeção de EFT.

A computação das métricas de desempenho (*Average Jitter* e *Difference Jitter*) é realizada usando como referência os logs de comunicação gerados pelo software. As métricas são definidas analisando o registro dos tempos automaticamente gerados na camada de aplicação do software, obtidas em tempo de execução pela função CAPL *timeNowNS()*, que retorna o *timestamp* das mensagens. Como realizado anteriormente, antes de gerar as sequências de injeção de EFT foi realizada uma medição da comunicação sem estas interferências. A Figura 37 mostra essa medição inicial da rede FlexRay, destacando a oscilação média da lei de controle dentro do esperado, em torno de 28 ns. Para exemplifi-

car o impacto dos transientes na rede FlexRay a Figura 38 mostra uma injeção com pulso de 47 volts e 687 us de rajada, destacando o acentuado impacto dos transientes EFT em relação a oscilação normal da rede FlexRay, com picos de atraso acima de 200 us. Como o objetivo é mostrar se há impacto dos transientes na comunicação, as demais figuras apresentadas são aproximadas (*zoom in*) com o intuito de destacar estes pontos.

Figura 37 – Medição da rede FlexRay registrada sem a injeção de pulsos EFT.



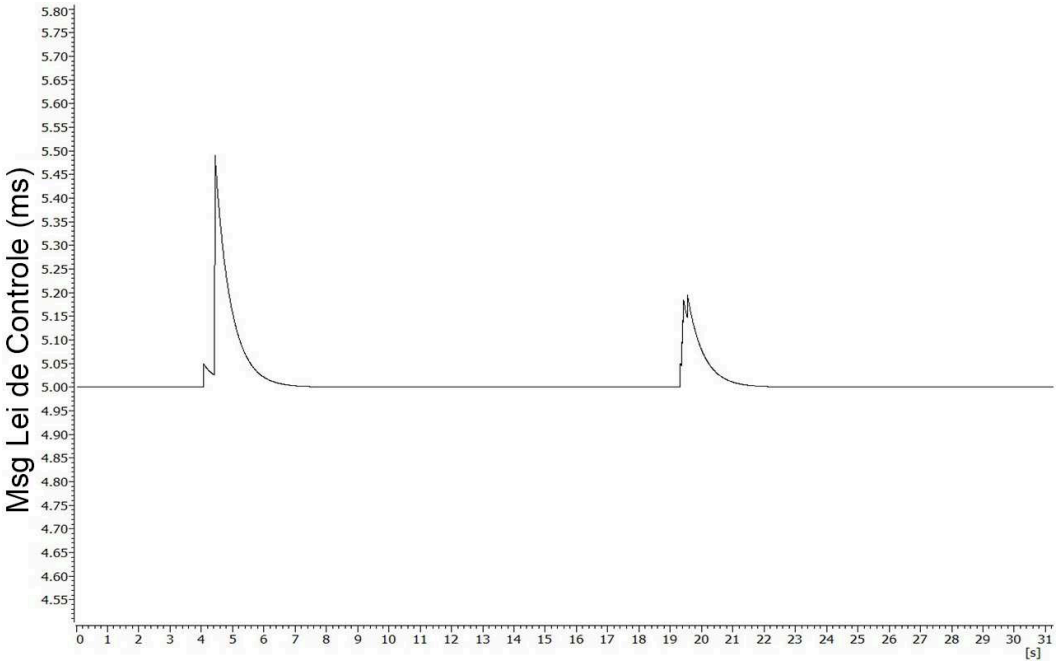
Fonte: Autor.

As medições tem como base um período de amostragem de aproximadamente 30 segundos, devido ao fato de ser um teste de estresse, que gera um grande volume de dados, suficientes para a análise de susceptibilidade ao EFT. Em seguida, os procedimentos de injeção de falhas foram iniciados dentro da sequencia definida na tabela 4, com três tipos de variação de carga de ocupação da rede FlexRay. Os gráficos apresentados na Figura 39 a), b) e c) destacam as oscilações no ciclo da lei de controle, com injeção de EFT com 47 volts e 687 microssegundos, e variação na carga de ocupação da rede no segmento estático. A Figura 40 apresenta para fins de comparação, as estatísticas geradas pelo software CANoe durante estes experimentos.

Observa-se na Figura 39 que com carga de 2% (somente o sistema de controle) a oscilação média do ciclo de controle fica em torno de 26 ns, e com as cargas de 30% e 80% a média aumenta (de acordo com as partes destacadas do gráfico), registrando picos de atraso maiores. Essas informações são destacadas também observando um comparativo das estatísticas da rede FlexRay observados na Figura 40. Verifica-se que a média na taxa de erros em quadros tratados no protocolo dobra, passando de 20 para 55 quadros por segundo.

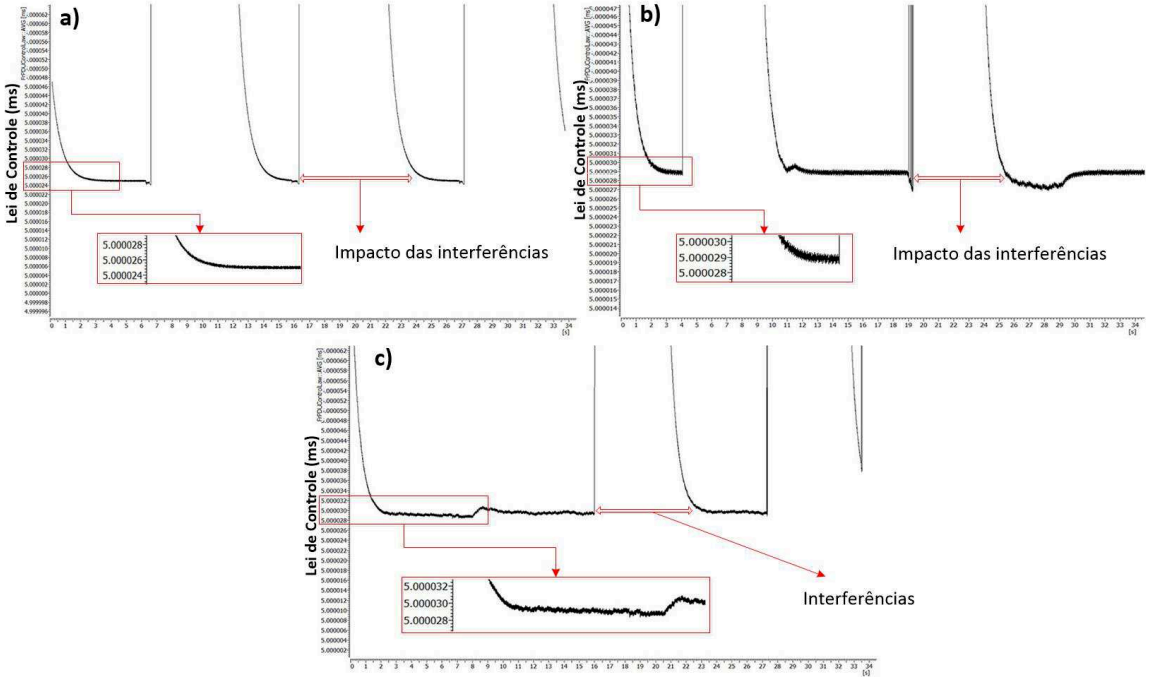
Em sequencia aos testes, o próximo experimento de injeção de falhas EFT é apresen-

Figura 38 – Exemplo do impacto dos transientes EFT na rede FlexRay.



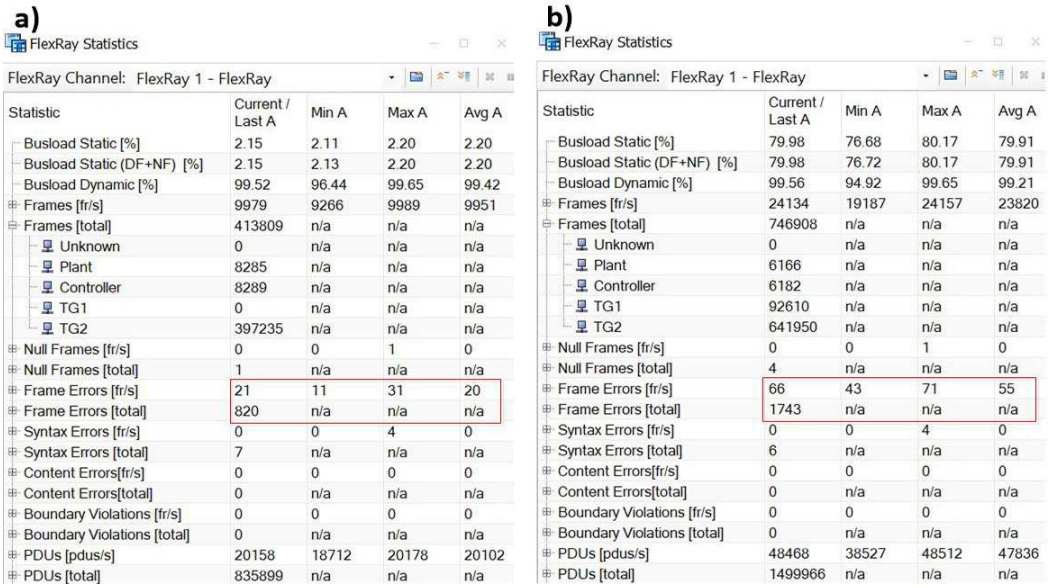
Fonte: Autor.

Figura 39 – Gráficos registrados da rede FlexRay com injeções EFT de 47 volts e 687us. a) Rede em 2% b) Rede em 30% e c) Rede em 80%.



Fonte: Autor.

Figura 40 – Estatísticas geradas pelo software durante as injeções de EFT com 47 volts na rede FlexRay. a) Rede em 2% b) Rede em 80%.



a)

FlexRay Channel: FlexRay 1 - FlexRay

Statistic	Current / Last A	Min A	Max A	Avg A
Busload Static [%]	2.15	2.11	2.20	2.20
Busload Static (DF+NF) [%]	2.15	2.13	2.20	2.20
Busload Dynamic [%]	99.52	96.44	99.65	99.42
Frames [fr/s]	9979	9266	9989	9951
Frames [total]	413809	n/a	n/a	n/a
Unknown	0	n/a	n/a	n/a
Plant	8285	n/a	n/a	n/a
Controller	8289	n/a	n/a	n/a
TG1	0	n/a	n/a	n/a
TG2	397235	n/a	n/a	n/a
Null Frames [fr/s]	0	0	1	0
Null Frames [total]	1	n/a	n/a	n/a
Frame Errors [fr/s]	21	11	31	20
Frame Errors [total]	820	n/a	n/a	n/a
Syntax Errors [fr/s]	0	0	4	0
Syntax Errors [total]	7	n/a	n/a	n/a
Content Errors[fr/s]	0	0	0	0
Content Errors[total]	0	n/a	n/a	n/a
Boundary Violations [fr/s]	0	0	0	0
Boundary Violations [total]	0	n/a	n/a	n/a
PDUs [pdus/s]	20158	18712	20178	20102
PDUs [total]	835899	n/a	n/a	n/a

b)

FlexRay Channel: FlexRay 1 - FlexRay

Statistic	Current / Last A	Min A	Max A	Avg A
Busload Static [%]	79.98	76.68	80.17	79.91
Busload Static (DF+NF) [%]	79.98	76.72	80.17	79.91
Busload Dynamic [%]	99.56	94.92	99.65	99.21
Frames [fr/s]	24134	19187	24157	23820
Frames [total]	746908	n/a	n/a	n/a
Unknown	0	n/a	n/a	n/a
Plant	6166	n/a	n/a	n/a
Controller	6182	n/a	n/a	n/a
TG1	92610	n/a	n/a	n/a
TG2	641950	n/a	n/a	n/a
Null Frames [fr/s]	0	0	1	0
Null Frames [total]	4	n/a	n/a	n/a
Frame Errors [fr/s]	66	43	71	55
Frame Errors [total]	1743	n/a	n/a	n/a
Syntax Errors [fr/s]	0	0	4	0
Syntax Errors [total]	6	n/a	n/a	n/a
Content Errors[fr/s]	0	0	0	0
Content Errors[total]	0	n/a	n/a	n/a
Boundary Violations [fr/s]	0	0	0	0
Boundary Violations [total]	0	n/a	n/a	n/a
PDUs [pdus/s]	48468	38527	48512	47836
PDUs [total]	1499966	n/a	n/a	n/a

Fonte: Autor.

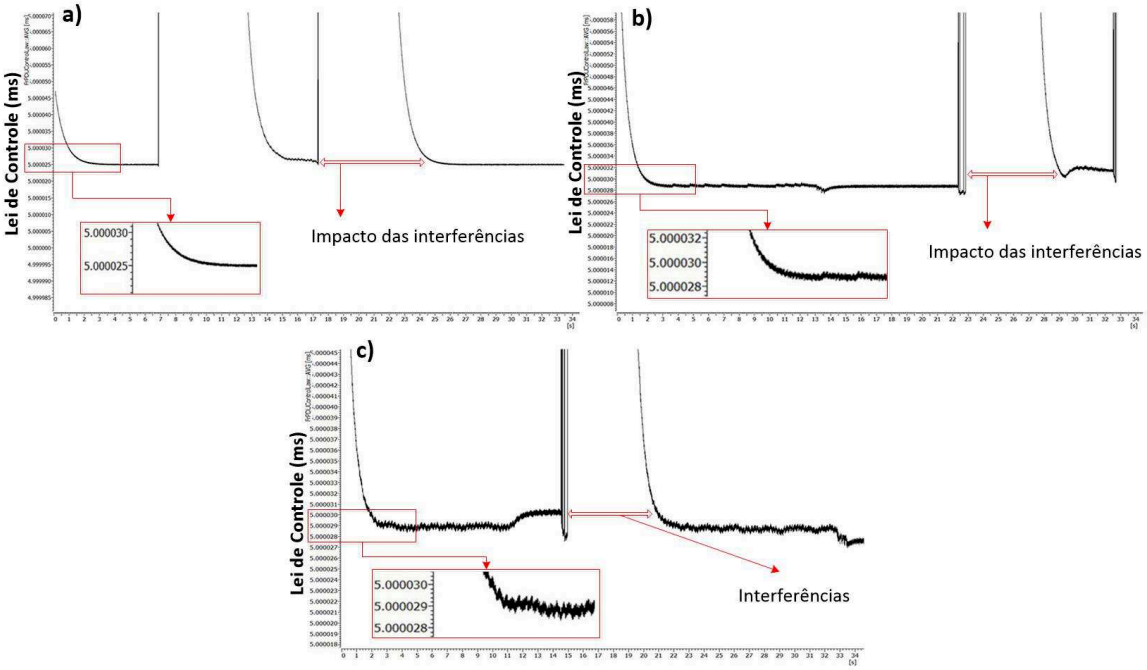
tado na Figura 41 com o registro das injeções de EFT com 57 volts e 1,2 milissegundos de *burst* (duração da injeção).

Neste segundo experimento de injeção de falhas EFT é possível observar pelos gráficos da Figura 41 um impacto similar na rede de comunicação, com distúrbios que elevam pouco a média do ciclo da lei de controle (na mesma faixa de 25 a 30 ns). Porém, cabe destacar que essa elevação é observada somente nos segmentos de comunicação em que não há uma interferência significativa, pois, ao observar os gráficos da Figura 41 a), b e c), são visíveis os atrasos gerados por falhas EFT. Esses pontos de interferência elevam o atraso médio do período de amostragem para a escala de microssegundos, com picos de até 37us com carga de 2%, para picos de até 260us com carga de 80%, que são valores altos para este protocolo. O resumo destes atrasos com os valores registrados e computados são apresentados ao final destes testes. A seguir a Figura 42 mostra as estatísticas geradas no software CANoe durante esse segundo experimento.

Com base nas informações apresentadas na estatísticas geradas Figura 42, observa-se o efeito da injeção de EFT com 57 volts e 1,2ms, com impacto menor em comparação ao primeiro experimento, onde a taxa de erros média passa de 15 para 52 quadros por segundo, com respectivamente 2% e 80% de carga de ocupação na rede FlexRay.

Por fim, o terceiro experimento de injeção de falhas EFT é apresentado na Figura 43, com 63 volts e 500 microssegundos de duração de rajada. Esse experimento procura analisar os efeitos de transientes com tempos de rajada menores, mas com maior frequência de ocorrência. Na Figura 43 a), b) e c) são destacados os pontos de ocorrência das interferências e atrasos gerados no ciclo de comunicação, com destaque aos efeitos mais acentuados registrados com a carga de ocupação de rede maior (em 80%).

Figura 41 – Gráficos registrados da rede FlexRay com injeções EFT de 57 volts e 1,2ms. a) Rede em 2% b) Rede em 30% e c) Rede em 80%.



Fonte: Autor.

Figura 42 – Estatísticas geradas pelo software durante as injeções de EFT com 57 volts na rede FlexRay. a) Rede em 2% b) Rede em 80%.

a)

FlexRay Statistics

FlexRay Channel: FlexRay 1 - FlexRay

Statistic	Current / Last A	Min A	Max A	Avg A
Busload Static [%]	2.20	2.11	2.20	2.19
Busload Static (DF+NF) [%]	2.20	2.13	2.20	2.19
Busload Dynamic [%]	99.62	96.38	99.65	99.48
Frames [fr/s]	9988	9501	9995	9973
Frames [total]	411580	n/a	n/a	n/a
Unknown	0	n/a	n/a	n/a
Plant	8227	n/a	n/a	n/a
Controller	8241	n/a	n/a	n/a
TG1	0	n/a	n/a	n/a
TG2	395112	n/a	n/a	n/a
Null Frames [fr/s]	0	0	1	0
Null Frames [total]	1	n/a	n/a	n/a
Frame Errors [fr/s]	12	5	24	15
Frame Errors [total]	616	n/a	n/a	n/a
Syntax Errors [fr/s]	0	0	7	0
Syntax Errors [total]	16	n/a	n/a	n/a
Content Errors[fr/s]	0	0	0	0
Content Errors[total]	0	n/a	n/a	n/a
Boundary Violations [fr/s]	0	0	0	0
Boundary Violations [total]	0	n/a	n/a	n/a
PDUs [pdus/s]	20176	19191	20190	20145
PDUs [total]	831373	n/a	n/a	n/a

b)

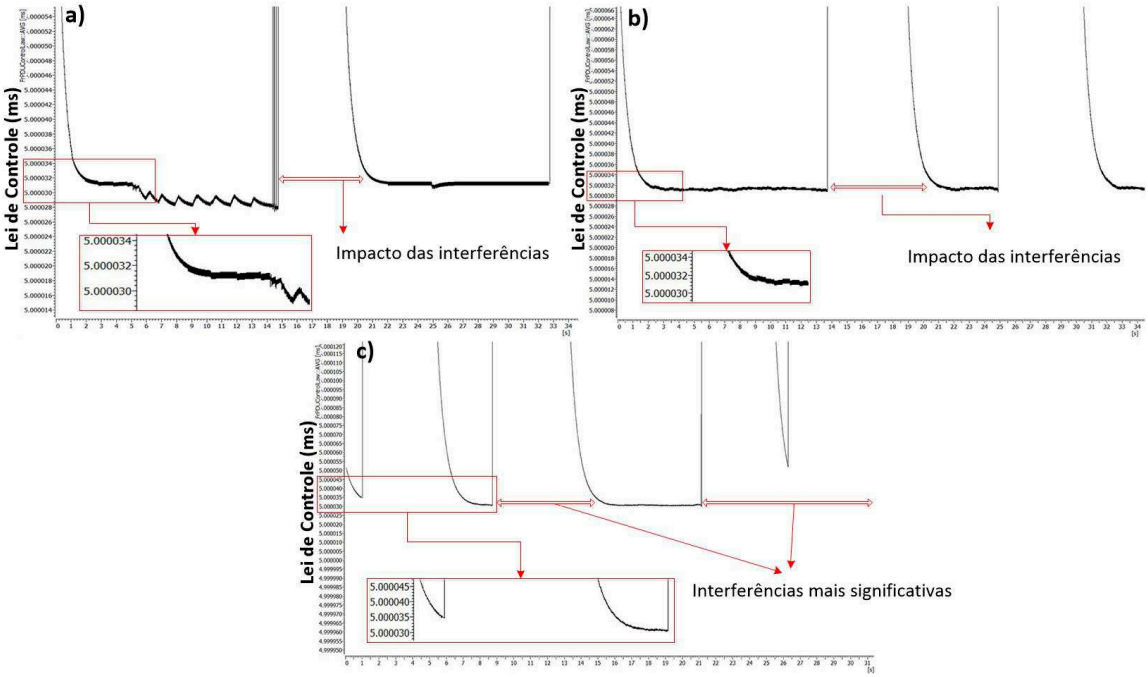
FlexRay Statistics

FlexRay Channel: FlexRay 1 - FlexRay

Statistic	Current / Last A	Min A	Max A	Avg A
Busload Static [%]	80.05	77.54	80.13	79.92
Busload Static (DF+NF) [%]	80.05	77.56	80.13	79.92
Busload Dynamic [%]	99.59	96.28	99.62	99.33
Frames [fr/s]	24147	22998	24165	24064
Frames [total]	670495	n/a	n/a	n/a
Unknown	0	n/a	n/a	n/a
Plant	5548	n/a	n/a	n/a
Controller	5547	n/a	n/a	n/a
TG1	83074	n/a	n/a	n/a
TG2	576326	n/a	n/a	n/a
Null Frames [fr/s]	0	0	1	0
Null Frames [total]	2	n/a	n/a	n/a
Frame Errors [fr/s]	53	35	68	52
Frame Errors [total]	1452	n/a	n/a	n/a
Syntax Errors [fr/s]	0	0	0	0
Syntax Errors [total]	0	n/a	n/a	n/a
Content Errors[fr/s]	0	0	0	0
Content Errors[total]	0	n/a	n/a	n/a
Boundary Violations [fr/s]	0	0	0	0
Boundary Violations [total]	0	n/a	n/a	n/a
PDUs [pdus/s]	48494	46178	48530	48326
PDUs [total]	1346539	n/a	n/a	n/a

Fonte: Autor.

Figura 43 – Gráficos registrados da rede FlexRay com injeções EFT de 63 volts e 500us. a) Rede em 2% b) Rede em 30% e c) Rede em 80%.



Fonte: Autor.

Figura 44 – Estatísticas geradas pelo software durante as injeções de EFT com 63 volts na rede FlexRay. a) Rede em 2% b) Rede em 80%.

a)

FlexRay Statistics

FlexRay Channel: FlexRay 1 - FlexRay

Statistic	Current / Last A	Min A	Max A	Avg A
Busload Static [%]	2.20	1.99	2.20	2.19
Busload Static (DF+NF) [%]	2.20	2.03	2.20	2.19
Busload Dynamic [%]	99.23	94.66	99.65	99.03
Frames [fr/s]	9962	9368	9980	9930
Frames [total]	421672	n/a	n/a	n/a
Unknown	0	n/a	n/a	n/a
Plant	8443	n/a	n/a	n/a
Controller	8457	n/a	n/a	n/a
TG1	0	n/a	n/a	n/a
TG2	404772	n/a	n/a	n/a
Null Frames [fr/s]	0	0	1	0
Null Frames [total]	3	n/a	n/a	n/a
Frame Errors [fr/s]	38	20	44	33
Frame Errors [total]	1423	n/a	n/a	n/a
Syntax Errors [fr/s]	0	0	7	0
Syntax Errors [total]	13	n/a	n/a	n/a
Content Errors[fr/s]	0	0	0	0
Content Errors[total]	0	n/a	n/a	n/a
Boundary Violations [fr/s]	0	0	0	0
Boundary Violations [total]	0	n/a	n/a	n/a
PDUs [pdus/s]	20124	18909	20160	20058
PDUs [total]	851773	n/a	n/a	n/a

b)

FlexRay Statistics

FlexRay Channel: FlexRay 1 - FlexRay

Statistic	Current / Last A	Min A	Max A	Avg A
Busload Static [%]	80.12	76.34	80.13	79.85
Busload Static (DF+NF) [%]	80.12	76.36	80.13	79.85
Busload Dynamic [%]	99.36	94.66	99.62	99.14
Frames [fr/s]	24112	22381	24137	23906
Frames [total]	670827	n/a	n/a	n/a
Unknown	0	n/a	n/a	n/a
Plant	5546	n/a	n/a	n/a
Controller	5562	n/a	n/a	n/a
TG1	83204	n/a	n/a	n/a
TG2	576515	n/a	n/a	n/a
Null Frames [fr/s]	0	0	1	0
Null Frames [total]	2	n/a	n/a	n/a
Frame Errors [fr/s]	88	51	110	76
Frame Errors [total]	2131	n/a	n/a	n/a
Syntax Errors [fr/s]	0	0	5	1
Syntax Errors [total]	14	n/a	n/a	n/a
Content Errors[fr/s]	0	0	0	0
Content Errors[total]	0	n/a	n/a	n/a
Boundary Violations [fr/s]	0	0	0	0
Boundary Violations [total]	0	n/a	n/a	n/a
PDUs [pdus/s]	48424	44937	48474	48010
PDUs [total]	1347184	n/a	n/a	n/a

Fonte: Autor.

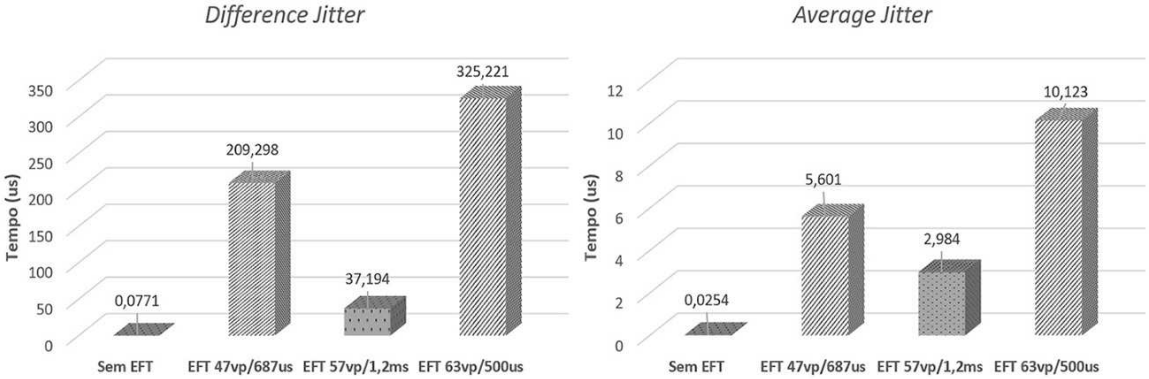
Enfatizando os efeitos negativos dos transientes EFT de maior amplitude de tensão, a Figura 44 apresenta as estatísticas do período de amostragem da rede FlexRay.

Do mesmo modo que no primeiro experimento, observa-se que os transientes EFT de menor tempo de rajada e maior amplitude de tensão, geram mais impactos negativos, refletindo-se em maior atraso no ciclo de comunicação e com um número maior de mensagens afetadas pela disrupção dos sinais do protocolo. A taxa de erros sobe de 33 para 76 quadros por segundo, respectivamente na análise com 2% e 80% de carga de ocupação da rede.

Essa sequencia de injeções de transientes EFT (47, 57 e 63 volts) permite uma análise aprofundada, sob condições de estresse, dos impactos negativos que esse tipo de falha pode gerar em sistemas de controle críticos. A observação destaca que devido ao fato de o protocolo FlexRay trabalhar com tempos de bit menores (para prover maior largura de banda), as chances de ter sinais afetados acaba sendo maior.

Desta forma, o presente estudo aborda a importância das análises de susceptibilidade dos protocolos de comunicação a transientes elétricos rápidos. Métodos e técnicas que visam mitigar ou diagnosticar esses efeitos são de suma importância para agregar confiabilidade aos sistemas de controle com requisito temporal crítico, e nesse caso, mesmo em um meio físico que faz uso de um protocolo de comunicação robusto como o FlexRay. Para evidenciar a degradação de desempenho causada durante as injeções de falhas, de forma coerente, e seguindo a mesma metodologia de análise, as mesmas métricas *Difference Jitter* e *Average Jitter* foram utilizadas na compilação dos dados obtidos durantes os experimentos. As figuras 45, 46 e 47 destacam os dados obtidos após a análise da rede de comunicação FlexRay.

Figura 45 – Resultados da análise da Lei de Controle na rede FlexRay com 2% de carga.

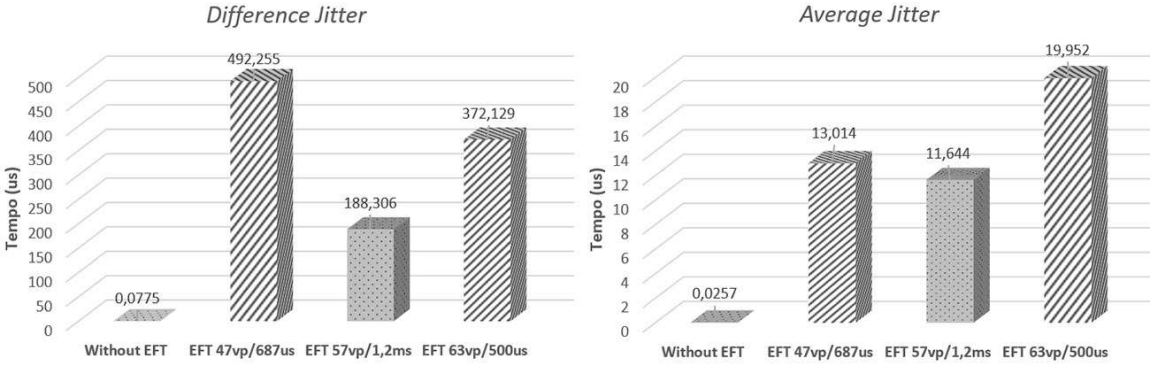


Fonte: Autor.

A Figura 45 sumariza os atrasos registrados na Lei de Controle, com 2% de carga de ocupação da rede, destacando um aumento significativo em relação a medição normal da rede de comunicação. Enquanto a medição normal possui uma oscilação média de 25 a 28 nanosegundos, durante as injeções de falhas esses valores sobem para a escala de microssegundos, chegando a atingir 325us de pico e 10us na média de atrasos. De forma

similar, o gráfico apresentado na Figura 46 destaca os atrasos gerados durante as injeções de EFT com 30% de carga de rede.

Figura 46 – Resultados da análise da Lei de Controle na rede FlexRay com 30% de carga.

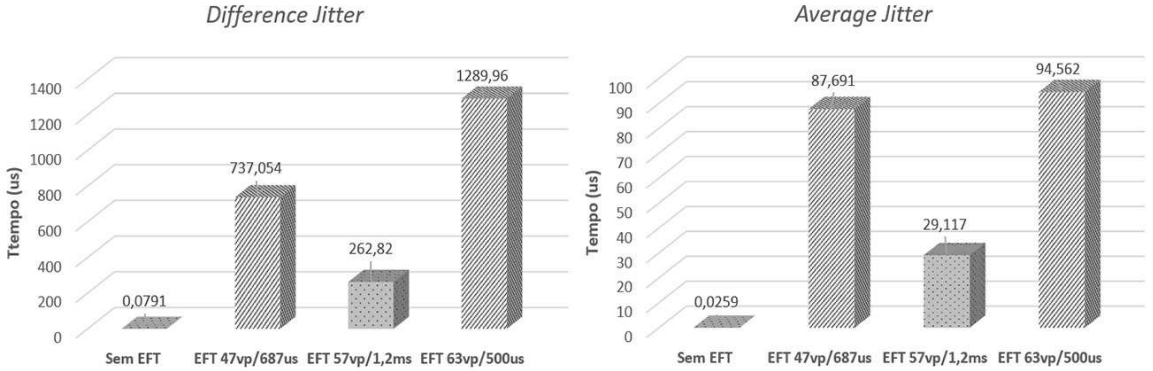


Fonte: Autor.

Nesse experimento observa-se que em todos os casos de injeção EFT, tanto os valores de picos quanto as médias de atraso aumentam, chegando a picos de quase 500us (com EFT de 47vp e 687us) e média de 19,95us segundos no pior caso com a métrica *average jitter* (em EFT de 63vp e 500us).

Na Figura 47 são apresentados o resultados do pior caso de análise, com uma carga de ocupação de rede em 80% no segmento estático. Nesse caso, os valores resultantes das análises com as métricas mostram um efeito de degradação de desempenho ainda maior, chegando a picos de atraso na ordem de 1,2 ms (EFT 63vp e 500us) e oscilação média no atraso de até 94,5 us (EFT 63vp e 500us). Em seguida, a Tabela 5 resume de forma comparativa os efeitos de degradação registrados durante os experimentos realizados, com base na métrica *average jitter*.

Figura 47 – Resultados da análise da Lei de Controle na rede FlexRay com 80% de carga.



Fonte: Autor.

Ao observar os dados da Tabela 5, destaca-se o efeito de degradação maior na rede FlexRay com transientes EFT de menor duração de rajada e principalmente com tensões de pico maiores. Esses valores se justificam pelo fato de o ciclo de comunicação do sistema de controle usado no estudo de caso ser de 5 ms, onde nesse caso, o intervalo

Tabela 5 – Resumo das degradações com injeções EFT na rede FlexRay - Métrica: *Average Jitter*.

	Rede 2%	Rede 30%	Rede 80%
sem EFT	0,0254 us	0,0257 us	0,0259 us
EFT 47 volts	5,6 us	13 us	87,6 us
EFT 57 volts	2,9 us	11,6 us	29,1 us
EFT 63 volts	10,1 us	19,9 us	94,5 us

de geração de transientes menor, permite o aumento da chance desses pulsos afetarem a mensagem cíclica com mais frequência. Por outro lado, com a duração de rajada maior (1,2 ms), durante o ciclo da lei de controle, a quantidade de transientes afetando os dados obtidos da planta do sistema são menores, resultando em degradação também menor. Cabe ressaltar, que as análises foram feitas em um curto período de amostragem, caracterizando uma análise de susceptibilidade a transientes, de acordo com orientações das normas ISO 26262, IEC 61000-4-4 e ISO 7637-3.

Essas informações confirmam os efeitos negativos de falhas como o EFT, em redes de comunicação com mensagens críticas. Mesmo que os testes realizados tenham sido de estresse (com muitas injeções), as degradações de desempenho registradas mostram que tais efeitos, mesmo silenciosos, podem afetar a segurança e confiabilidade de um sistema de controle crítico de tempo real. Sendo assim, é muito importante o monitoramento e registro destas situações, mesmo para análises posteriores ou certificações dos limites de tolerância do sistema de controle em estudo.

Os resultados obtidos também destacam que apesar de protocolos como CAN-FD e FlexRay possuírem maior largura de banda de comunicação, parte dessa comunicação pode ser ocupada por pacotes de erros, reduzindo a largura de banda efetiva e também a confiabilidade da rede. Conforme observado nos experimentos, os EFTs levam a disrupção na modulação de sinais, tendo efeito ainda mais crítico pelo FlexRay trabalhar com tempos de bit muito pequenos. Os atrasos observados na Lei de Controle do estudo de caso (Sistema de Controle de Suspensão Ativa) durante a transmissão de mensagens, evidenciaram a perda de desempenho e a criticidade do problema. Assim, cada vez mais o diagnóstico e registro da ocorrência de falhas, torna-se importante para contribuir com a redução da manutenção e também melhorar o projeto de novos sistemas de controle para redes intra-veiculares.

4.2.4 Avaliação dos transientes EFT com análise de árvore de falhas

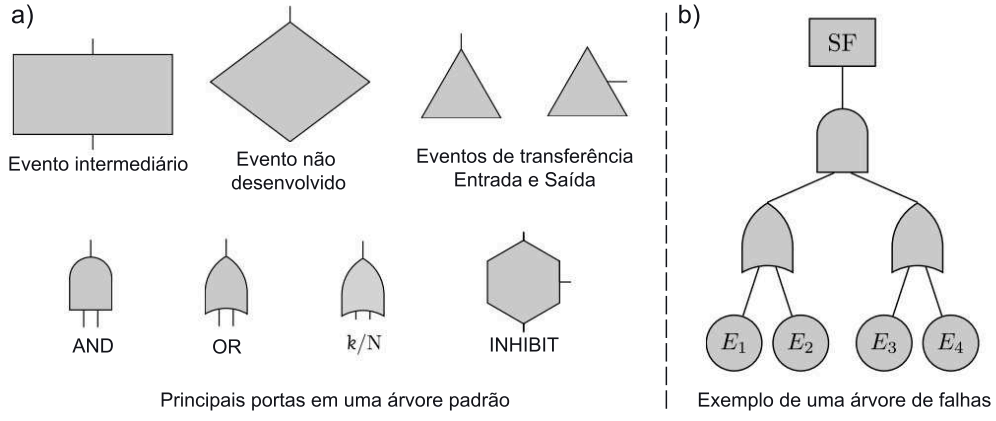
Uma árvore de falhas é um modelo gráfico para analisar as propriedades de confiabilidade de um determinado sistema. De acordo com (RUIJTERS; STOELINGA, 2015), as árvores de falha padrão (*SFT - Standard Fault Tree*) são as árvores mais básicas utilizadas para representar os aspectos de confiabilidade de um sistema. Essa abordagem foi intro-

duzida na década de 1960, no *Bell Labs*, para a análise de um míssil balístico. A Análise de Árvores de Falhas - FTA caracteriza um método importante para destacar os riscos relacionados à segurança funcional em sistemas críticos. Como exemplo, o presente estudo apresenta a análise do impacto de falhas transientes em protocolos de comunicação e consequentemente nos sistemas de controle que dependem deste meio de comunicação em uma rede intra-veicular.

Uma árvore de falhas é representada por um gráfico acíclico direcionado (DAG), que consiste em dois tipos de nós principais: "eventos"e "portas". Eventos são as ocorrências de falhas, por exemplo, uma falha de um componente individual que representa uma parte de todo o sistema. Os eventos são classificados de acordo com seus respectivos eventos básicos - BEs (representados no diagrama por círculos), que ocorrem involuntariamente, e eventos intermediários (retângulos), causados por um ou mais eventos diferentes. No topo da árvore está o principal evento sendo analisado, especificamente em relação as características do sistema sob análise. Eventos intermediários geralmente são usados para documentação e não afetam a análise da árvore de falhas. Se uma árvore de falhas é muito grande, uma representação baseada em triângulos é usada para transferir eventos entre várias árvores (RUIJTERS; STOELINGA, 2015) (IEC-61025, 2006).

Outras partes importantes da árvore de falhas são as portas (*Gates*), que representam como as falhas se propagam pelo sistema, com base em uma combinação de eventos que causam uma falha. Cada porta possui uma saída, com uma ou mais entradas. A Figura 48 a) apresenta as principais portas usadas nessa representação, e b) apresenta um pequeno exemplo de uma árvore de falhas.

Figura 48 – a) Principais eventos e portas de uma árvore de falhas. b) Exemplo de uma árvore de falhas.



Fonte: Adaptado de (RUIJTERS; STOELINGA, 2015).

De acordo com a Figura 48 as portas principais são:

- E/AND - Evento de saída que ocorre se todos os eventos de entrada ocorrerem;
- OU/OR - Evento de saída que ocorre se algum dos eventos de entrada ocorrer;

- k/N - VOTAÇÃO entre N entradas. Nesse caso, o evento de saída ocorre se pelo menos k eventos de entrada ocorrerem. Essa porta pode ser substituída por uma OR entre todos os conjuntos de k entradas, mas o usa-se uma porta k/N para agregar mais clareza a esta representação;

- INIBIDO (*INHIBIT*) - O evento de saída ocorre, se o evento de entrada ocorrer enquanto o evento de uma condição a direita da porta também ocorrer. Essa porta se comporta de maneira idêntica a porta AND, de forma que raramente é utilizada. Essa porta é usada para definir quando os comportamentos da árvore de falhas começam a ser analisados ou verificados (Por exemplo: somente se uma função estiver ativa ou ligada).

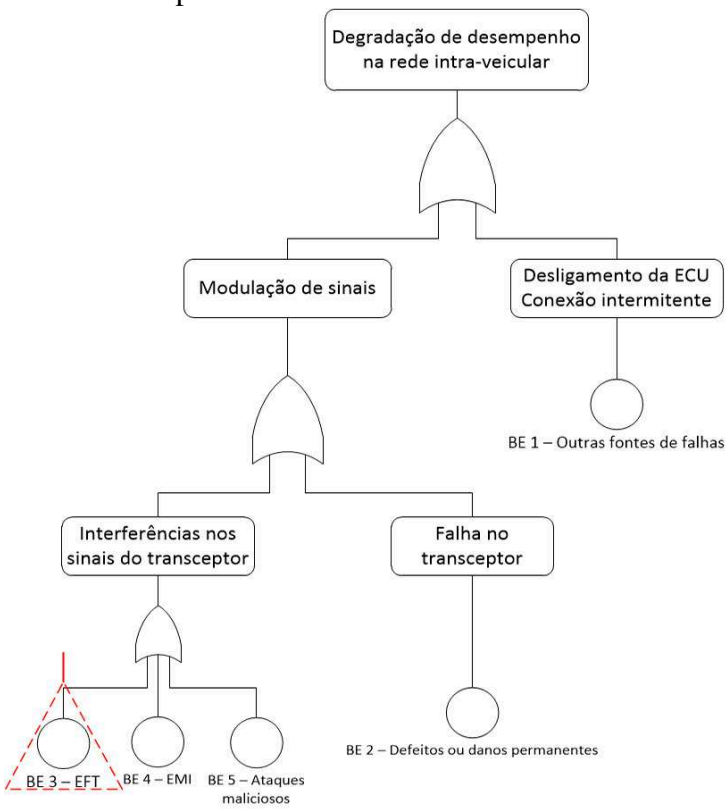
A representação de árvore de falhas apresentada neste estudo, bem como os principais conceitos, são exemplos do formalismo comumente utilizado, baseados nas pesquisas de (RUIJTERS; STOELINGA, 2015) e (KABIR, 2017). Porém, cabe destacar que existem várias extensões e representações, algumas personalizadas ao escopo e tipo de sistema em análise, inclusive adicionando ou criando novas portas para a propagação dos eventos que causam falhas.

Desta forma, este estudo utiliza a representação de uma árvore de falhas para avaliar a propagação dos efeitos de degradação causados por transientes elétricos rápidos em uma rede de comunicação intra-veicular. Nesse universo, as falhas transientes são eventos físicos causados por diversos dispositivos dentro do veículo, por exemplo, conversores buck DC/DC, sistemas de acionamento e comutação de energia (*power switching*), comumente ao ligar e desligar sistemas de ar-condicionado, faróis, ignição, entre outros.

De acordo com os estudos previamente realizados e apresentados nas seções anteriores, os protocolos de comunicação CAN, CAN-FD e FlexRay são susceptíveis aos transientes elétricos rápidos. Se os eventos forem persistentes, estes podem gerar consequências como atrasos nas malhas de controle de sistemas críticos. Desta forma, a análise de árvore de falhas foi realizada com base nos efeitos negativos gerados no sistema de controle de suspensão ativa, utilizado como referência durante as análises das injeções EFT. A Figura 49 apresenta a árvore de falhas desenvolvida.

A Figura 49 destaca o evento central que é objeto do estudo, a degradação de desempenho causada por transientes elétricos. Especificando a análise, a degradação foi classificada em duas fontes principais: problemas elétricos diversos que geram conexão intermitente ou o simples desligamento das ECUs, situações que caracterizam outras fontes de falhas (Evento Básico 1); e problemas na modulação de sinais do protocolo de comunicação, caracterizado pelas interferências nos transceptores dos protocolos. As falhas nos transceptores podem ser defeitos físicos permanentes no circuito integrado (Evento Básico 2) ou interferências na codificação dos sinais para o meio físico (Evento Básico 3, Evento Básico 4 e Evento Básico 5). De acordo com os estudos realizados, as principais

Figura 49 – Árvore de falhas para análise dos efeitos de transientes na rede intra-veicular.



Fonte: Autor.

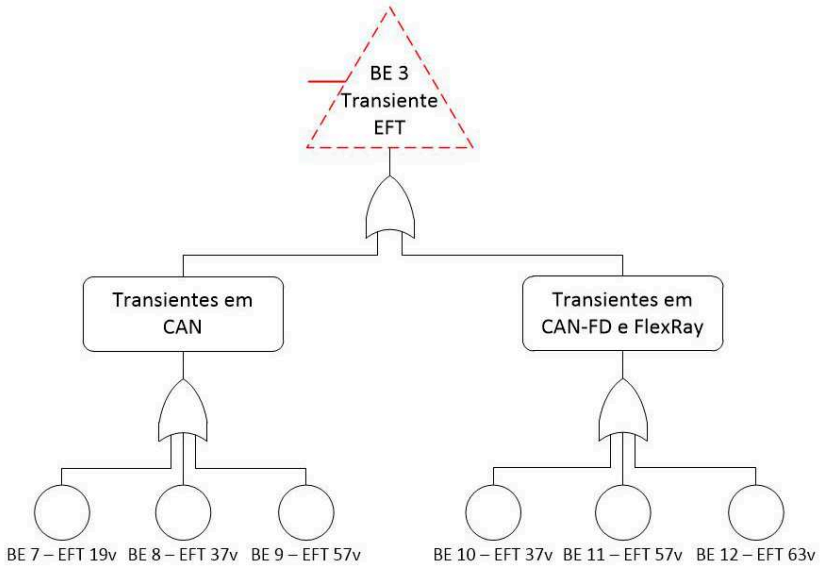
fontes de interferência são transientes elétricos rápidos - EFT, interferências eletromagnéticas - EMI ou falhas similares causadas por ataques maliciosos.

Desta forma, expande-se a análise do Evento Básico 3, que trata das fontes de transientes elétricos rápidos (*power switching transients*), com diferentes amplitudes de tensão e tempos de rajada. A Figura 50 detalha a árvore de falhas para este evento.

Neste ponto, é detalhado como cada tipo de transiente elétrico representa uma fonte potencial e silenciosa de degradação de desempenho, pois esses eventos seguem pela árvore até o evento do topo, impactando em atrasos de comunicação em uma malha de controle crítica. Esses efeitos são diferentes em cada protocolo de comunicação, por este motivo, optou-se pela separação da árvore entre dois eventos intermediários para os protocolos. O protocolo CAN tradicional já sofre interferências acentuadas a partir de EFTs de 19 volts, enquanto os protocolo CAN-FD e FlexRay possuem maior degradação a partir de 37 volts de amplitude, por este motivo as diferentes amplitudes de tensão adotadas.

Com base nesta representação, diferentes abordagens podem ser usadas para mensurar a probabilidade da ocorrência destas falhas e também dos possíveis impactos negativos. Métodos qualitativos consideram mais a estrutura de uma árvore de falhas, e métodos quantitativos proveem importantes valores numéricos a respeito das possibilidades de propagação de falhas na mesma estrutura. Para o presente estudo, medidas estocásticas são

Figura 50 – Expansão da Árvore de falhas especificamente para o Evento Básico 3 - tipos de transientes e protocolos estudados.



Fonte: Autor.

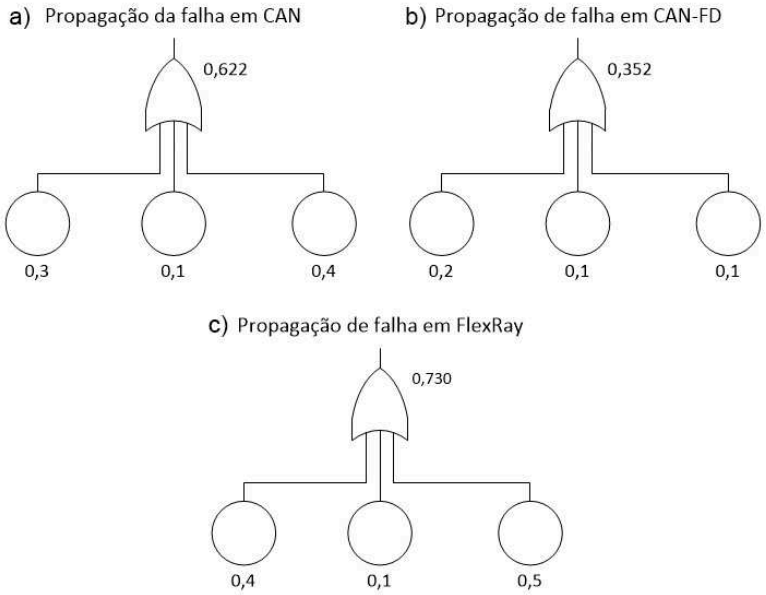
muito comuns, porque fornecem orientações sobre as probabilidades de falha, uma vez que a abordagem depende de fenômenos que variam aleatoriamente (como os EFTs).

Devido ao escopo da presente tese não ser uma análise estatística ou probabilística da ocorrência dos transientes EFT, esta seção objetiva, de forma simplificada e sucinta, apresentar uma análise formal das possibilidades de propagação de uma falha transiente. Desta forma, a fim de demonstrar um exemplo de análise da árvore de falhas construída, é apresentada uma avaliação da probabilidade dos transientes EFT gerarem degradação, com base em uma análise de tempo único, baseado em cortes mínimos (*single-time approach*) (RUIJTERS; STOELINGA, 2015).

A Figura 51 a), b) e c) expõem os três cortes mínimos, que resultam na propagação de falhas EFT especificamente em cada protocolo, de acordo com a abordagem de tempo único. Os dados obtidos e utilizados são baseados nas recentes publicações resultantes dos estudos da presente tese de doutorado (ROQUE *et al.*, 2017) (POHREN *et al.*, 2019).

Um corte mínimo representa um conjunto ou combinações de eventos básicos (BE) que levam a uma falha do sistema. A abordagem de tempo único não considera a evolução de um sistema ao longo do tempo: um horizonte temporal fixo é considerado. Nesse sentido, a probabilidade de impacto dos EFT foi analisada com base na degradação causada na rede intra-veicular, considerando o tempo de amostragem obtido durante as injeções de falhas. De acordo com (RUIJTERS; STOELINGA, 2015), em uma abordagem de tempo único de análise, o evento básico é associado a uma função de probabilidade P : BE [0,1] que define a probabilidade da falha P(e) para cada $e \in$ BE. Assim, cada BE e pode ser associado a uma variável randômica X$e \sim$ Alt(P(e)); que é P(Xe = 1) = P(e) e P(Xe = 0) = 1 – P(e). Assim, de acordo com a teoria de probabilidade, seguindo a lei de Morgan na

Figura 51 – Análise da probabilidade de falhas com a abordagem de tempo único. a) CAN, b) CAN-FD e c) FlexRay.



Fonte: Autor.

teoria dos conjuntos, e assumindo a independência dos eventos, têm-se P(X$e_1 \cup$ Xe_2) = 1 - (1-P(Xe_1) (1-P(Xe_2). A seguinte equação define esta representação:

$$P(X_1 \cup X_2 \cup ... \cup X_n) = 1 - \prod_{i=1}^{n}(1 - P(X_i)) \tag{4}$$

A variável associada a cada BE define o grau de relevância da falha para a propagação ao evento intermediário acima. Assim, neste estudo foi avaliado o impacto dos três tipos de transientes injetados em cada protocolo, definindo um valor de probabilidade, correspondente ao nível de degradação de cada injeção de falha.

Durante o período de amostragem com o protocolo CAN, a degradação registrada em cada injeção de falha com a métrica *average jitter* foi de 161%, 9.25% e 219%. Assim, de acordo com a abordagem utilizada, considerou-se um valor de probabilidade proporcional, que levaria a propagação para o evento intermediário acima. Para o protocolo CAN foram adotados os seguintes valores de probabilidade de acordo com cada tipo de falha: 0.3, 0.1 e 0.4. Nos experimentos com o protocolo CAN-FD, a degradação registrada em cada injeção de falha foi de 29,05%, 10.41% e 11,56%. Da mesma forma, considerou-se um valor de probabilidade proporcional a cada injeção de falha, os quais foram os seguintes: 0.2, 0.1 e 0.1.

No protocolo FlexRay o impacto dos transientes EFT foi mais acentuado em comparação com o seu tempo normal de comunicação, mas com valores de atraso menores em relação aos outros protocolos. Para definir uma referência de propagação em relação ao FlexRay, foram comparados os valores de pico de atraso gerados com os transientes EFT, observando que os transientes de 47 e 63 volts geraram os piores resultados em

comparação com o transiente de 57 volts. Assim, foram adotados como referência para a probabilidade de propagação os seguintes valores: 0.4, 0.1 e 0.5 respectivamente para cada tipo de injeção de falha.

A Figura 51 a), b) e c) destaca um exemplo da probabilidade dos EFT na degradação de cada protocolo. A porta OR indica a ocorrência de degradação de desempenho se qualquer um dos eventos básicos abaixo se propagarem. Então, para o protocolo CAN aplica-se a probabilidade P(Xe = 0) = 1 – (1-0,3)(1-0,1)(1-0,4) = 1 - 0.378 = 0,622. Para o protocolo CAN-FD, aplica-se a probabilidade P(Xe = 0) = 1 – (1-0,2)(1-0,1)(1-0,1) = 1 - 0.648 = 0,352. Para o protocolo FlexRay, aplica-se a probabilidade P(Xe = 0) = 1 – (1-0,4)(1-0,1)(1-0,5) = 1 - 0.27 = 0,730.

Assim, de acordo com a abordagem aplicada, quanto mais próximo de 1, maior a probabilidade de se atingir o evento intermediário acima e consequentemente a propagação até o evento do topo da árvore. A análise confirma os experimentos previamente realizados, mostrando que CAN e FlexRay possuem a maior susceptibilidade aos efeitos das falhas EFT, sendo o CAN-FD o menos susceptível.

4.3 Modelagem Orientada a Aspectos e Especificação de Requisitos com o framework RT-FRIDA

Por meio das análises de desempenho da rede intra-veicular e da susceptibilidade dos protocolos a degradação causada por transientes elétricos, os dados obtidos podem ser representados por requisitos não funcionais da aplicação. Estes requisitos visam compor funções para o monitoramento em tempo real de desempenho da rede, definindo limites de tolerância com relação às restrições temporais do sistema de controle, permitindo ainda que mecanismos de alerta e notificação possam ser desenvolvidos. Outra característica que pode ser agregada à rede veicular por meio destas análises é a redundância de recursos de hardware e mecanismos de escalonamento de mensagens de acordo com as variações e perturbações que ocorrem na rede. Devido à amplitude destes estudos, o escopo da presente pesquisa é restrito ao uso das informações obtidas nos testes para a definição de gatilhos(*triggers*) em tempo real que permitam a geração de alertas e também o diagnóstico de degradação de desempenho causada por falhas e distúrbios na rede.

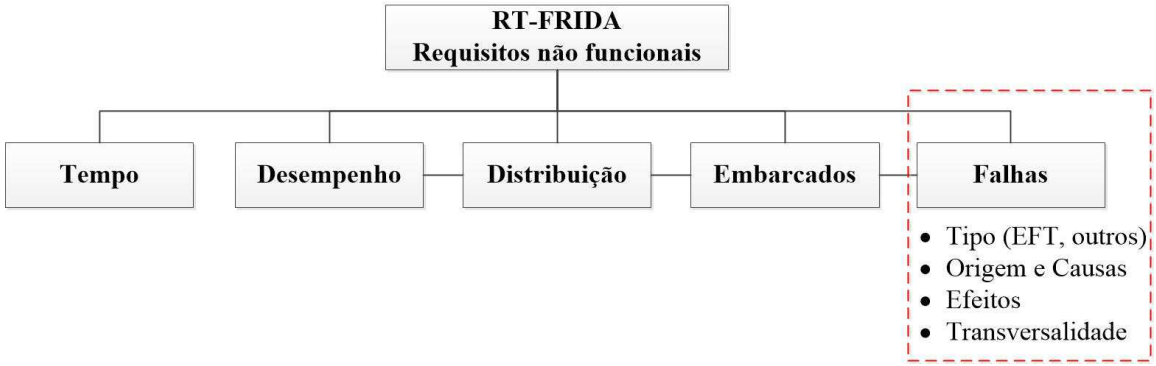
Primeiramente, antes da definição destes parâmetros que servem de base para os gatilhos de diagnóstico, a modelagem de falhas e especificação de requisitos com o framework RT-FRIDA foi estudada. A modelagem orientada a aspectos - AOM permite a verificação dos efeitos transversais de falhas, especificando o efeito destas em diferentes partes do sistema (no caso automotivo em ECUs e sistemas de controle, por exemplo). Assim, a AOM contribui para a especificação proativa de requisitos que mapeiam a degradação de desempenho em sistemas de controle distribuídos, promovendo a manutenibilidade e confiabilidade destes. Em AOM uma falha ou o efeito causado por ela

pode ser descrito como um requisito não funcional, devido ao seu impacto transversal no desempenho do sistema embarcado. Os interesses transversais *(crosscutting concerns)* caracterizam a base da AOM, na qual cada requisito não funcional deve ser considerado pela sua criticidade dentro do domínio de interesse da aplicação, ou seja, o quanto ele impacta ou afeta diferentes partes do sistema.

Neste sentido, o framework RT-FRIDA foi escolhido como ferramenta de apoio na representação dos requisitos não funcionais, devido à sua concepção já levar em conta a orientação a aspectos em sua base. Assim, o framework é usado para modelar falhas que degradam o desempenho dos protocolos de comunicação veiculares e analisar seu efeito por meio da representação e especificação de requisitos relacionados a degradação do sistemas de controle. Essa integração entre fases de teste e análise de desempenho com as fases de projeto e especificação de requisitos do sistema, contribui para o projeto de sistemas de controle mais robustos, atendendo uma lacuna destacada na literatura referente ao tratamento de falhas cada vez mais cedo nas etapas de projeto. Pesquisas vem destacando esse esforço na identificação e tratamento de requisitos não-funcionais no início do ciclo de vida de um sistema, sendo comumente conhecido como *early aspects* (SÁNCHEZ *et al.*, 2010) (CHITCHYAN *et al.*, 2015) (BARRA; MORATO, 2016).

Como o presente estudo está inserido no escopo de sistemas de controle de tempo real em redes intra-veiculares, o framework RT-FRIDA se enquadra perfeitamente como ferramenta de apoio a esta pesquisa, pois permite mapear os requisitos não funcionais, a partir de fases de projeto, usando conceitos de orientação a aspectos para a especificação do sistema, modelos de templates e checklists para a especificação de requisitos, além de representações baseadas em diagramas de caso de uso da aplicação. Para tal tarefa foi necessário estudar a composição do framework RT-FRIDA e adicionar mais um nível na sua estrutura de requisitos não funcionais, a modelagem de falhas. Com base na visão geral do framework apresentado na Seção 3.3 e na Figura 11, a seguir a Figura 52 ilustra esta especialização.

Figura 52 – Detalhamento do nível agregado ao framework RT-FRIDA para tratamento de falhas como NFR.



Fonte: Autor.

Como pode ser visualizado na Figura 52 diferentes requisitos não funcionais precisam ser especificados para atender os objetivos de um sistema em tempo real (tempo, desempenho, distribuição e embarcados). Esses requisitos possuem características de transversalidade, ou seja, afetam e são preponderantes para a correta operação de diferentes partes do sistema. Diversos tipos de falhas podem afetar uma aplicação de tempo real com estas mesmas características, afetando por exemplo, diferentes requisitos de um sistema de controle distribuído ao mesmo tempo. Por exemplo, uma falha transiente pode afetar a periodicidade de uma tarefa que compõe o sistema de controle, implicando em atrasos e *jitter*, impactando o tempo de resposta de uma comunicação entre ECUs, gerando aumento de tráfego na rede, e em muitos casos, o aumento do processamento também afeta o consumo de energia. Sendo assim, foram adicionadas neste quinto nível do framework informações relacionadas ao tipo de falhas, sua origem, efeitos de degradação e seu impacto transversal no sistema. Esta especialização permite a modelagem de falhas e o desenvolvimento de mecanismos de diagnóstico e detecção de falhas. Para entender e coletar informações de determinadas falhas, processos de injeção e estresse foram estudados, para assim, detalhar os efeitos de degradação de desempenho e como estes podem afetar a rede intra-veicular e consequentemente os seus sistemas de controle.

Para o processo de modelagem proposto nesta pesquisa, o estudo tomou como base uma falha especifica (*electrical fast transients - EFT*) que foi estudada e aprofundada caracterizando um exemplo de processo completo dentro da metodologia de testes e análise proposta (Figura 14). Tendo como base o procedimento de modelagem especificado no framework RT-FRIDA, as seguintes etapas foram realizadas:

- Identificação e especificação de requisitos não funcionais referente ao tipo de falha estudado. Criação de templates e checklists para a especificação destas falhas e de outras similares. Povoamento de um diagrama de casos de uso com específicos estereótipos referentes aos requisitos não funcionais.

- Mapeamento de Requisitos em Elementos de Projeto. Extração de conceitos e atributos que serão responsáveis por compor elementos que tratam os requisitos não funcionais.

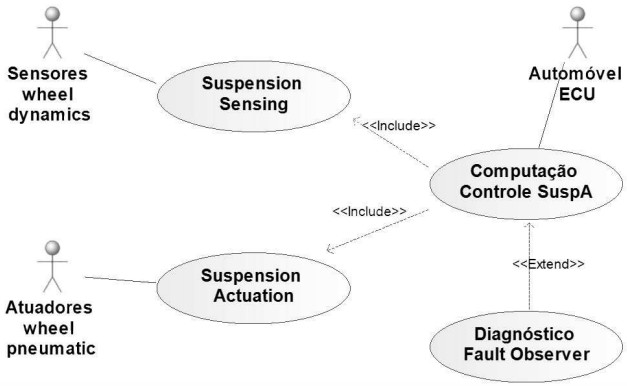
- Projeto do Sistema, desenvolvimento de diagramas representativos, para o projeto do sistema de controle em conjunto com o sistema de diagnóstico de anomalias causadas por falhas.

As subseções seguintes apresentam os detalhes de cada uma destas etapas da modelagem de falhas como requisitos não funcionais, mostrando as possibilidades de reuso e aplicação do framework RT-FRIDA.

4.3.1 Identificação e Especificação dos Requisitos Não Funcionais

Seguindo os procedimentos definidos no framework RT-FRIDA a primeira fase seria composta por especificação de requisitos funcionais e não funcionais, porém como o foco desta pesquisa é especificamente o tratamento de falhas, serão tratados apenas os requisitos não funcionais. Para o estudo de caso de validação desta modelagem, serão assumidos os requisitos funcionais baseados no desenvolvimento de um sistema de controle de suspensão ativa proposto em (MICHELIN, 2014) (detalhes do sistema de controle serão apresentados no Capítulo 5). A modelagem aqui apresentada poderá ser agregada ao sistema de controle ou adicionada a rede como um novo elemento (nó). A Figura 53 apresenta o diagrama de casos de uso do sistema, enfatizando que a modelagem é centrada no controle de suspensão ativa e no mecanismo de diagnóstico aqui nomeado de "FaultObserver".

Figura 53 – Diagrama de Casos de Uso do sistema de controle de suspensão ativa.



Fonte: Autor.

De acordo com as análises do impacto dos transientes EFT nos protocolos de comunicação usados em redes veiculares, foram definidos alguns parâmetros relacionados à degradação causada por estas falhas. Os testes de injeção de falhas EFT foram realizados nos três principais protocolos de comunicação do contexto de redes intra-veiculares, CAN e FlexRay que já são usados na indústria e CAN-FD por ser um protocolo emergente. A largura de banda nominal dos protocolos usados nos testes foi de 1 Mbps para CAN, 2 a 4 Mbps para CAN-FD e 10 Mbps em FlexRay. A tabela 6 apresenta um resumo das informações obtidas durantes os testes e análises realizadas.

De acordo com os dados apresentados na Tabela 6 é possível perceber a maior susceptibilidade do protocolo CAN convencional às falhas transientes em relação ao impacto na variação do jitter. Os protocolos de maior desempenho CAN-FD e FlexRay também apresentaram efeitos negativos, porém na média de variação os atrasos são menores. Por outro lado, se observarmos a diferença entre os atrasos médios sem EFT e com EFT, o protocolo FlexRay é o que sofre as maiores degradações, pois a variação normal sem EFT fica na ordem de poucos nanosegundos, e os atrasos médios registrados chegaram a até

Tabela 6 – Principais informações obtidas dos testes de suceptibilidade a EFT.

Protocolo / EFT burst	Max. Dif. Jitter	Max. Avg. Jitter	Busload / Variação
CAN / 19 vp	1.4 ms	189 us	30%
CAN / 37 vp	0.63 ms	79 us	30%
CAN / 57 vp	1.60 ms	231 us	30%
CAN-FD / 47 vp	2,04 ms	33 us	30% a 60%
CAN-FD / 57 vp	1,28 ms	26 us	30% a 60%
CAN-FD / 63 vp	1,31 ms	27 us	30% a 60%
FlexRay / 47 vp	de 209 us a 737 us	de 5 us a 87 us	2%, 30% e 80%
FlexRay / 57 vp	de 37 us a 262 us	de 2 us a 29 us	2%, 30% e 80%
FlexRay / 63 vp	de 325 us a 1,2 ms	de 10 us a 94 us	2%, 30% e 80%

94 us no pior caso.

Ao monitorar o efeito de degradação de desempenho em diferentes mensagens, como na Lei de Controle (ECU controlador) e deflecção da suspensão (ECU Planta) percebe-se o efeito transversal deste tipo de falha, que afeta tanto a rede de comunicação quanto as tarefas de controle individuais de cada ECU, bem como também gera degradações funcionais nos transceptores do protocolo (como destacado na norma IEC 62228).

Desta forma os efeitos de degradação de desempenho causados por este tipo de falha são mapeados nesta fase de modelagem como requisitos não funcionais. Para tal tarefa um conjunto de questões especificamente desenvolvidas para o domínio de interesse deste estudo (redes intra-veiculares) são desenvolvidas, constituindo o artefato chamado *checklist*. Assim, para o presente estudo cinco *checklists* foram desenvolvidos para cada classe de requisito não funcional, incluindo um novo *checklist* para o tratamento de pontos específicos referentes aos EFTs (Tempo, Desempenho, Distribuição, Embarcados e Falhas).

No Apêndice A são apresentados os detalhes e a composição desses questionários desenvolvidos com base no estudo do sistema de controle de suspensão ativa e na falha especifica detalhada. Os questionários seguem um padrão definido no framework RT-FRIDA, contendo o tipo de requisito, relevância, escala de prioridade (baixa, média e alta) e uma descrição das restrições ou procedimentos para se atender cada questão. Importante ressaltar também, que somente as questões relevantes ao domínio de interesse são tratadas, podendo haver itens que não precisam ser considerados no contexto da presente modelagem.

O questionário adicionado ao framework RT-FRIDA é detalhado na Tabela 7 e contém as principais informações para a especificação de requisitos não funcionais relacionados às falhas. O mesmo questionário também é apresentado no Apêndice A.

O segundo passo após o desenvolvimento destas *checklists* consiste em verificar se algum requisito não-funcional foi esquecido. Para tal o framework RT-FRIDA baseia-se em um artefato chamado léxico, que consiste em um glossário povoado com termos do

Tabela 7 – Checklist para os requisitos de Falhas do sistema de controle de suspensão ativa. Legenda: R - Relevante, P - Prioridade

	R	**P**	**Restrições/Descrição**
Falhas			
Tipo			
Existe um tipo de falha específico que pode degradar o sistema?	X	Alta	Transientes elétricos rápidos - EFT
Caso exista um tipo de falha conhecido, existem normas para testes e verificação dos distúrbios causados por tal falha?	X	Média	Sim. Testes seguindo o padrão de injeção de falhas definido nas normas IEC 62228, ISO 7637
Caso não exista uma norma especificada, existem métodos ou técnicas de stress na rede que pode levar a uma análise aproximada?	X	Média	Existem ferramentas de stress que aproximam o efeito, mas é recomendado seguir as normas.
Origem e Causas			
Existe origem ou causa conhecida do tipo de falha?	X	Média	Transientes gerados por comutação de energia em diferentes componentes espalhados pelo veículo. *Power System Switching Transients*
Existem formas possíveis de detecção deste tipo de falha?	X	Baixa	A adição de sensores ou chips com capacidade de tolerância e detecção é possível, mas os custos podem ser proibitivos.
Efeitos			
Existem rotinas de registro de degradação de desempenho da rede de comunicação?	X	Alta	Registro de de logs para análise posterior.
Existem métricas especificas de desempenho monitoradas em tempo real?	X	Alta	Sim. Carga de ocupação, Jitter e Jitter de pior caso.
Existem limites de tolerância definidos para estas métricas?	X	Alta	Sim. Com base em testes anteriores foi definida uma tolerância de 25% no jitter médio.
Existem ajustes na periodicidade de msg em caso de degradação de desempenho?			
Existem gatilhos de notificação em caso de distúrbios frequentes na comunicação?	X	Média	Sim. Mediante análise de *overhead* na comunicação.
No uso de mecanismos de hardware para detecção, existem rotinas de tratamento?			
Transversalidade			
Existe unidades de controle especificas afetadas?	X	Média	Não. Todas podem ser afetadas.
Existem mensagens críticas especificas afetadas?	X	Alta	Sim. LeiDeControle.

contexto específico do domínio do problema, utilizado para se verificar a presença de detalhes já levantados durante o questionário. Esta verificação foi realizada não encontrando nenhum problema ou redundância de informações. Maiores detalhes da composição dos léxicos podem ser encontradas em (FREITAS, 2007).

Em sequencia à identificação dos requisitos não-funcionais, passa-se então ao processo de especificação. Esta atividade é desenvolvida por meio do preenchimento de artefatos chamados de *templates* de requisitos não-funcionais. O conjunto de *templates* desenvolvido neste estudo é apresentado no Apêndice B. Após o preenchimento destes requisitos outra etapa importante destacada no framework RT-FRIDA é a resolução de conflitos entre requisitos não funcionais, para tal uma matriz é criada para essa verificação. A Tabela 8 apresenta a visualização dos requisitos não funcionais e seus conflitos.

Tabela 8 – Relação dos requisitos não funcionais especificados e conflitos identificados no projeto.

NF	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15
R1															
R2															
R3															
R4															
R5															
R6															
R7															
R8															
R9															
R10															
R11															
R12															
R13															
R14						X									
R15							X								

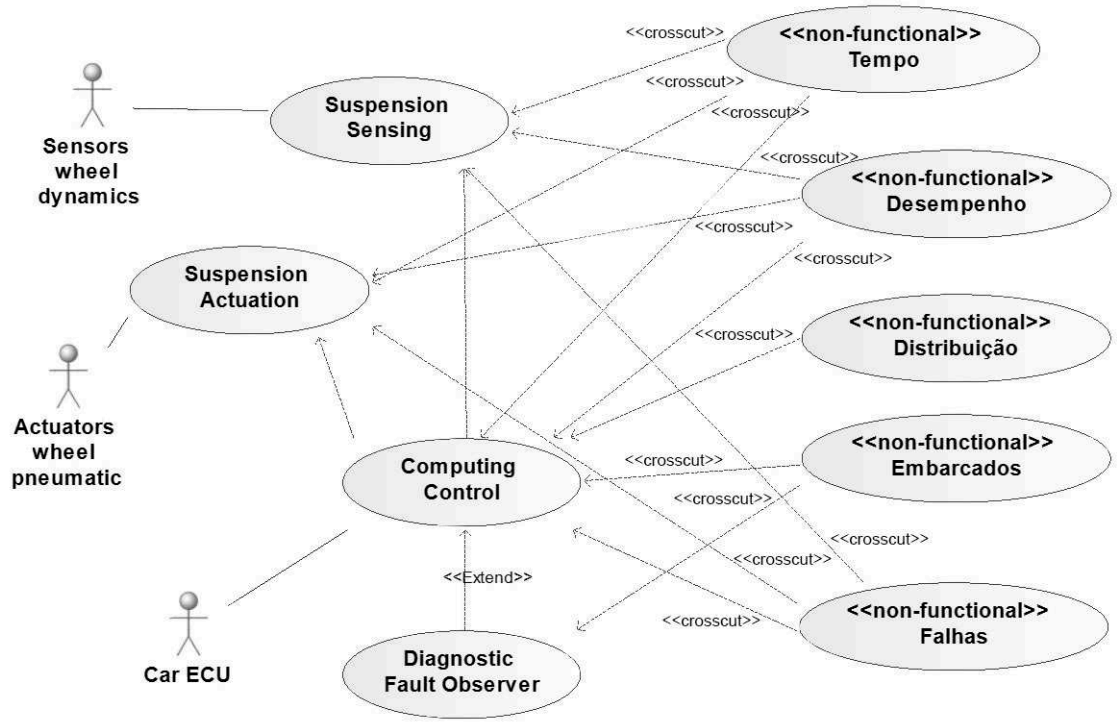
Em relação às questões de desempenho do sistema de controle, dois conflitos no processo de especificação foram encontrados. A definições dos requisitos NFR6 e NFR7 são realizadas de forma mais geral sem considerar perturbações no sistema. Assim, neste estudo a prioridade foi dada para os requisitos especificados no checklist de falhas (NFR14 e NFR15), pois estes tratam o problema de forma mais especializada. Um conflito pode ser observado na Tabela A.3, item vazão e carga, onde para este tipo de degradação serão considerados os limites de desempenho do NFR14. Outro conflito pode ser observado na Tabela A.3, item tempo de resposta, onde no caso de degradação de desempenho os limites definidos no NFR15 terão maior prioridade. Ambas as revisões podem ser vistas nos templates de NFR Tabelas B.6 e B.7. Adicionalmente, é destacado outro possível conflito que pode ocorrer no caso de sistemas que permitem o ajuste da periodicidade de

mensagens em caso de degradação de desempenho (A.5, item Efeitos), que pode ocorrer com o tempo de pior caso, esta possibilidade pode ser vista na Tabela A.2, item temporização. No caso do presente estudo os requisitos relacionados aos efeitos de degradação são usados para fins de diagnóstico e detecção de falhas, não gerando o conflito acima citado.

Outro ponto importante a destacar, é que por meio da adição do quinto *checklist*, o projetista pode especificar separadamente e de forma especializada, os requisitos não funcionais de temporização/desempenho em relação aos efeitos de degradação causados por falhas, que podem afetar estes mesmos requisitos. Este ponto permite a especialização de mais um domínio de interesse no detalhamento do sistema, enfatizado em pesquisas pela possibilidade de tratar comportamento e degradações causadas por falhas em fases inicias de projeto. Nesse caso, cabe ao projetista especificar os requisitos e de forma a deixar claro esta separação.

Após este estudo com a especificação dos requisitos não funcionais relacionados à falha estudada, deve-se atualizar graficamente o diagrama de casos de uso que foi base do estudo (Figura 53), detalhando as relações entre os requisitos e casos de uso afetados. A seguir a Figura 54 apresenta esse diagrama atualizado após o estudo.

Figura 54 – Diagrama de Casos de Uso do Sistema de Controle de Suspensão Ativa atualizado com as relações dos requisitos não funcionais.



Fonte: Autor.

De acordo com o framework RT-FRIDA esta atividade é realizada por meio da adição de um caso de uso com o estereótipo «*non-functional*» para cada requisito não-funcional,

seguido de outro estereótipo «*crosscut*» para a representação do relacionando entre o requisito e o caso de uso afetado. O segundo estereótipo é oriundo dos conceitos de orientação a aspectos tratado no framework, que permite a representação da transversalidade de requisitos que afetam diferentes partes do sistema. Assim, é possível criar um caso de uso para cada requisito não funcional, porém, por questão de clareza, pode ser usado um caso de uso para cada item geral da classificação (exemplo: tempo, desempenho, falhas) que engloba os itens específicos relacionados (exemplo: Jitter, tempo de resposta).

4.3.2 Mapeamento dos requisitos em elementos de projeto

Após o estudo realizado por meio de experimentos práticos de análise dos transientes EFT e das etapas de identificação e especificação dos requisitos do sistema de controle, passa-se à etapa de mapeamento destes requisitos em elementos de projeto para posterior implementação. Esta etapa é importante para as características de reuso do método proposto nesta tese, pois permitirá a verificação dos elementos que implementam os requisitos não funcionais servindo assim de base para outras aplicações futuras.

Um problema recorrente no desenvolvimento de sistemas em geral e que não é desvinculado do projeto e desenvolvimento de sistemas de controle distribuídos, é o uso de documentação adequada a replicação do processo, característica agregada com a aplicação do framework RT-FRIDA nesse trabalho. Outro ponto importante destacado nesta etapa é a associação dos requisitos levantados com os elementos do projeto orientado a aspectos, caracterizados pelos efeitos transversais dos requisitos não funcionais. Assim, o objetivo principal desta etapa é formalizar a ligação entre os requisitos e o projeto do sistema, favorecendo a identificação de quais funções e elementos de projeto são responsáveis por tratar cada requisito, fato que facilita a manutenção futura do sistema.

Essa ligação é representada por meio de uma tabela de mapeamento que dispõe todos os requisitos não funcionais na primeira linha e todos os elementos funcionais na primeira coluna. Assim, o projetista com toda a documentação gerada na fase de especificação de requisitos, verifica e toma a decisão de quais classes ou funções (independente do paradigma de programação) serão responsáveis pela implementação. A Tabela 9 apresenta esse mapeamento desenvolvido para o projeto do sistema de controle de suspensão ativa com o mecanismo de diagnóstico de falhas e degradação de desempenho.

Como pode ser observado na Tabela 9, para fins de facilitar o desenvolvimento e manutenção, vários requisitos são agrupados e tratados por elementos específicos. Na parte inferior pode ser visualizado quais aspectos são implementados para atender aos respectivos requisitos não funcionais. A caracterização do entrelaçamento entre requisitos é identificada assinalando-se a quadrícula que representa a junção e indica qual requisito não funcional afeta outro requisito funcional. A próxima subseção apresenta o projeto do sistema de controle com o mecanismo de diagnóstico de degradação causado por falhas.

Tabela 9 – Mapeamento de requisitos em elementos de projeto. Legenda: T - Tempo, Dp - Desempenho, Ds - Distribuição, E - Embarcados e F - Falhas.

	Requisitos Não Funcionais								
RF	R1	R2	R3 R4 R5	R6 R7	R8	R9	R10	R11 á R15	Elementos que tratam o requisito funcional
RF1	X	X	X					X	SuspensionSensing
RF2	X	X	X					X	SuspensionActuation
RF3	X	X	X	X	X	X	X	X	ComputingControl
RF4				X			X		DiagnosticFaultObs
Aspectos que tratam RNF	T	T	T	Dp e F	Ds	Ds	E	F	

4.3.3 Projeto do sistema de controle com o mecanismo de diagnóstico e detecção de degradação de desempenho

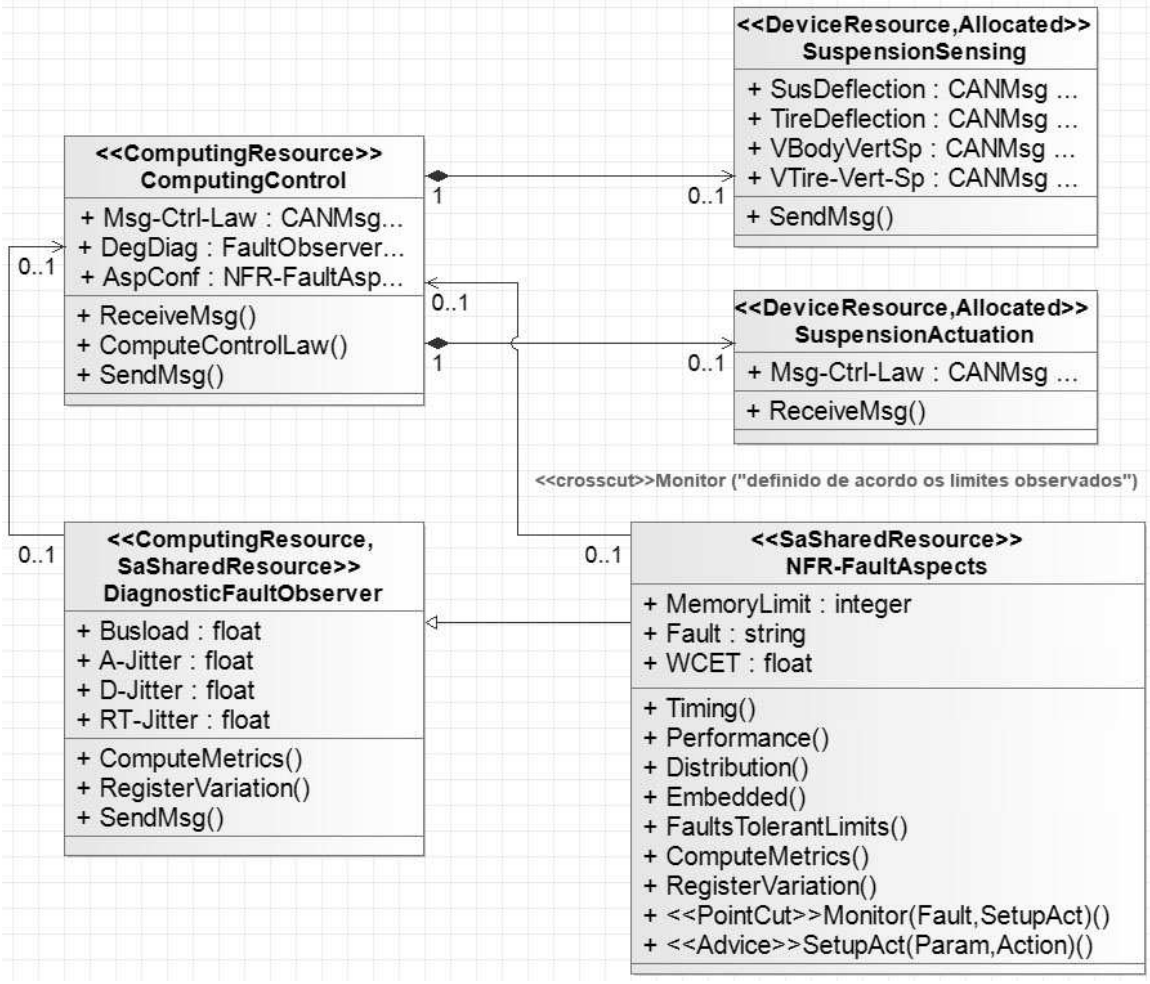
Esta seção apresenta o projeto do sistema de controle de suspensão ativa em conjunto com um mecanismo de diagnóstico de degradação de desempenho causado por falhas. O projeto toma como base e exemplifica a modelagem de degradação causada por um tipo de falha estudado e aprofundado, porém destaca-se que o projeto é expansível, ou seja, no futuro outras falhas podem ser estudadas e ajustes ou adições nos parâmetros do mecanismo de diagnóstico podem ser efetuados. As etapas anteriores de análise de falhas que afetam os protocolos de comunicação veiculares, análise e especificação de requisitos com o RT-FRIDA, mapeamento de elementos de projeto e agora o projeto do sistema, objetivam demonstrar as etapas de aplicação da metodologia proposta nesta tese.

Outra particularidade importante a destacar nesta etapa de projeto é que a técnica de controle aplicada para a dinâmica da planta do sistema de controle de suspensão ativa é baseada na pesquisa de (MICHELIN, 2014) (Controlador: Realimentação de estados baseada na metodologia de dados amostrados), assim detalhes sobre este processo são omitidos nesta etapa. O foco do presente estudo é, com base na dinâmica de planta de controle previamente definida, modelar a dinâmica de comunicação do sistema de controle na rede veicular, em conjunto com um mecanismo de diagnóstico de falhas, considerando aspectos específicos do funcionamento dos principais protocolos de comunicação usados em redes veiculares atualmente. Desta forma, com base nas especificações da seção anterior diagramas foram desenvolvidos para constituir o projeto do sistema.

Para o projeto do sistema os diagramas escolhidos foram o diagrama de classes e o diagrama de transição de estados, suportados por estereótipos do perfil UML MARTE (OMG, 2019). A escolha destes diagramas segue os preceitos de que este projeto poderá ser aplicado futuramente em diferentes plataformas de desenvolvimento para sistemas

embarcados distribuídos do setor automotivo. Muitas plataformas permitem o desenvolvimento do sistema de controle com base nas linguagens C++ e JAVA, suportado pelo paradigma orientado a objetos. Porém existem outras plataformas que permitem o desenvolvimento usando o paradigma estruturado baseado na linguagem C, e ainda o desenvolvimento baseado no paradigma orientado a eventos (o qual é orientado a objetos em sua essência). Esse ultimo paradigma é o utilizado nas ferramentas da VECTOR Informatik, que permite a abstração dos conceitos mais complexos das linguagens e transforma todo o tratamento de processos que ocorrem na rede de comunicação, em eventos disparados quando condições são atendidas. A Figura 55 apresenta o primeiro diagrama de suporte ao projeto, o diagrama de classes.

Figura 55 – Diagrama de Classes do projeto com o mecanismo de diagnóstico e tratamento dos requisitos não funcionais.



Fonte: Autor.

Para a representação dos recursos e detalhes do sistema, o profile UML MARTE fornece diferentes estereótipos que podem ser usados para anotar o modelo. De acordo com a UML MARTE o estereótipo «*deviceResource*» representa um dispositivo externo que pode ser manipulado, escalonado ou requerido pela plataforma ou sistema em execução.

já o estereótipo «*allocated*» representa alocação de recursos, sendo aplicado normalmente a qualquer elemento que mantem a relação de alocação e utilização. Assim, estes estereótipos são usados para representação dos recursos utilizados pelo sistema de controle, neste estudo a planta do sistema, que possui o conjunto de sensores e o sistema de atuação. Outro estereótipo utilizado é o «*computingResource*» que representa qualquer recurso de processamento, virtual ou físico, que armazena e executa programas. Neste caso esse estereótipo é utilizado para representar a ECU do sistema de controle, que realiza a computação da lei de controle e faz uso de outros recursos. O estereótipo «*saSharedResource*» é relacionado a análise de comunicação e recursos compartilhados, muito usado em sistema de tempo real. Esse estereótipo é utilizado em conjunto com o «*computingResource*» para representar sistema de diagnóstico, que neste estudo será agregado a rede como um nó de processamento extra.

O diagrama apresentado na Figura 55 representa a modelagem do sistema de controle com os principais elementos que monitoram o funcionamento da rede e do sistema de controle, destacando informações específicas que podem ser parametrizadas de acordo com uma falha estudada. O diagrama destaca os pontos críticos com relação ao desempenho da rede perante distúrbios e que pontos são responsáveis por tratar estas variações. Deve-se observar que todos os elementos que caracterizam preocupação com requisitos temporais, encontram-se especificados em um único elemento do projeto, dedicado apenas a este tipo de requisito não-funcional. Evita-se com isto o espalhamento deste requisito não-funcional por diversas partes do sistema.

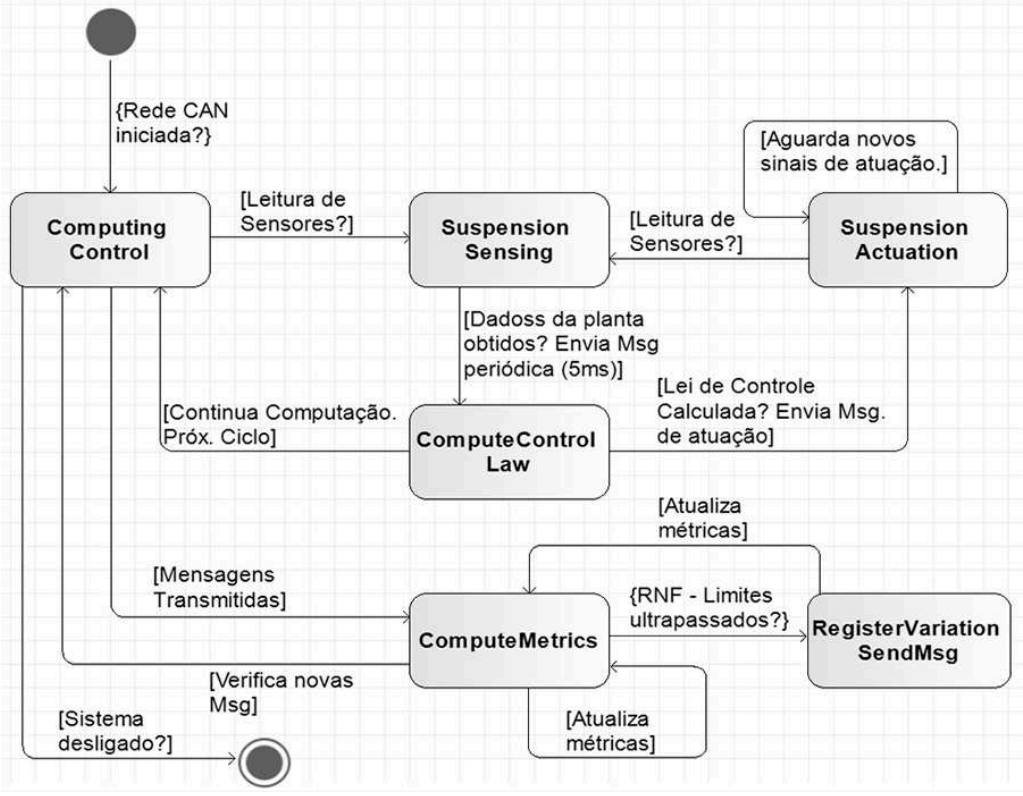
Para representar a dinâmica da interação entre os aspectos e as classes é necessário apresentar o ponto no qual um ou mais aspectos as afetam, chamados de *joinpoints*, os quais definem os locais onde ocorre o entrelaçamento (*pointcuts*). Neste estudo é destacado um ponto que representa todos os aspectos, de forma a deixar o diagrama mais claro e não espalhar pelo sistema o tratamento de cada aspecto, fato que facilita o desenvolvimento e a manutenção. Este ponto é destacado na classe "NFR-FaultAspects", por meio do elemento «*pointcut*» onde ocorre o entrelaçamento. Assim, tem-se o *joinpoint* "Monitor", responsável por realizar as operações de registro, notificação dos distúrbios e variações nas métricas, de acordo com parâmetros definidos previamente e após o estudo de uma falha específica. Esse ponto especificado é responsável por uma chamada a outro elemento chamado de *advice*, nessa representação identificado por "SetupAct", que é responsável pela especificação de que parâmetros/métricas e limites serão observados. Por meio do não atendimento destes limites de operação ou de flutuações acentuadas nas métricas de desempenho, gatilhos de diagnóstico que podem ser acionados, tanto para registro de logs em memória local, quanto para o envio de mensagens de notificação.

O diagrama de classes detalha a comunicação principal entre três elementos do sistema de controle: SuspensionSensing, SuspensionActuation e ComputingControl. Esses elementos são responsáveis pelo ciclo de controle, composto por mensagens de sensoria-

mento da ECU planta enviadas periodicamente para ECU responsável pelo processamento da dinâmica de controle, o qual envia novamente o ajuste necessário no controle para a ECU da planta. No diagrama é destacado e enfatizado o mecanismo de diagnóstico de degradação de desempenho causado por falhas, bem como também todos os aspectos que representam o tratamento dos requisitos não funcionais que afetam o sistema. Adicionalmente, o diagrama de transição de estados é outra forma de detalhamento do mecanismo de diagnóstico, pois contribui para um detalhamento da comunicação e das mudanças de estado do sistema ao longo do tempo, ajudando na compreensão do projeto quando desenvolvido também com base em outros paradigmas, como por exemplo, o estruturado e o orientado a eventos.

A Figura 56 apresenta o diagrama de transição de estados, contribuindo com outra visão e detalhamento de como ocorre o processo de comunicação entre os elementos do sistema, e como as informações da rede são capturadas para análise de desempenho e diagnóstico.

Figura 56 – Diagrama de Transição de Estados do Mecanismo de Diagnóstico.



Fonte: Autor.

De acordo com o framework RT-FRIDA, outro diagrama de suporte ao projeto que pode ser desenvolvido é o diagrama ACOD (*Aspect Crosscutting Overview Diagram*). Este diagrama apresenta funcionalidades afetadas por requisitos não-funcionais, e os aspectos utilizados para tratar estes requisitos. Como já descrito e visualizado anteriormente, o entrelaçamento fica caracterizado através de um relacionamento estereotipado

como «*crosscut*» observado nos diagramas de casos de uso e diagrama de classes. Desta forma, neste projeto o diagrama ACOD não é necessário, tendo em vista a caracterização dos aspectos nos diagramas e o foco do projeto ser a modelagem orientada a aspectos e não a programação orientada a aspectos. Nesse último caso, a linguagem de programação precisa suportar tais recursos ou programador precisa fazer adaptações que caracterizem os aspectos definidos.

Durante a construção desses diagramas o projetista define a representação formal do sistema a ser desenvolvido, que posteriormente pode passar por adaptações e especificações pontuais usadas na linguagem e plataforma de programação definida. Assim, o projeto aqui apresentado pode ainda ser ajustado de acordo com a plataforma escolhida e decisões do desenvolvedor. Como neste projeto adota-se como referência na etapa de validação os equipamentos fornecidos pela VECTOR Informatik (VECTOR INFORMATIK, 2018), o projeto especificado é adaptado ao paradigma orientado a eventos, usado na plataforma Vector CANoe e baseado na linguagem CAPL. Tal decisão de nada afeta os resultados que serão apresentados, tendo em vista que o projeto deve justamente ser uma referência para o desenvolvimento e não restrito a uma linguagem específica. Outro ponto a justificar tal decisão é a ampla adoção da plataforma escolhida pela indústria automotiva, que é justamente o cenário de aplicação desta modelagem. Detalhes sobre esse processo de desenvolvimento e validação do projeto são apresentados no Capítulo 5, durante a validação em diferentes cenários.

4.3.4 Avaliação da modelagem de NFR com gráfico SIG e o método *Softgoal Weight*

O NFR framework é uma abordagem para a análise e modelagem de requisitos orientada a processos ou objetivos (*goal-oriented technique*) que contribui e ajuda o projetista a evidenciar a extensão dos objetivos que serão atingidos com um determinado projeto. De acordo com Chung (2012) com o NFR framework os requisitos não funcionais são apresentados e avaliados como metas a serem atingidas, onde estes estão relacionados ao comportamento do sistema e não às suas funcionalidades. Estas características são importantes para avaliar e analisar o presente projeto porque considera diretamente as propriedades atingidas pelo sistema (confiabilidade, manutenibilidade e flexibilidade), após as mudanças que são efetuadas com o tratamento de requisitos não funcionais relacionados a falhas, apoiadas pelo framework RT-FRIDA.

Para a análise de especificações de projeto diferentes abordagens baseadas na decomposição de objetivos podem ser aplicadas. O NFR framework usa a representação baseada no gráfico de interdependência entre meta/objetivos (*Softgoal Interdependence Graph* - SIG), que é aplicado para a representação dos objetivos alcançados com mudanças nos RNF (CHUNG *et al.*, 2012). Neste contexto, para a especificação de objetivos relacionados a RNF e falhas, outras abordagens também podem ser utilizadas, como por exemplo, a análise de árvore de falhas (*Fault Tree Analysis*), normalmente aplicada para decompor

falhas em suas subsequentes causas e efeitos, baseando-se em uma estrutura *top-dow* de uma árvore (RUIJTERS; STOELINGA, 2015). Outro exemplo é a Notação estruturada por metas (*Goal Structuring Notation - GSN*), que pode ser utilizada para representar o sistema decompondo uma meta principal em sub-objetivos que serão suporte para evidenciar o atendimento dos requisitos (SPRIGGS, 2012).

Muitas dessas abordagens requerem uma análise qualitativa, onde tal estudo demanda um esforço que pode distanciar os engenheiros e projetistas dos objetivos práticos do sistema. Em (SUBRAMANIAN; ZALEWSKI, 2016) destaca-se que com abordagens quantitativas obtêm-se uma avaliação mais efetiva do impacto de mudanças no sistema em função do atendimento ou não de requisitos não funcionais. Por outro lado, com o uso do SIG é possível representar quantitativamente como a especificação de requisitos de diferentes tipos de falhas podem afetar e impactar em melhorias no projeto do sistema. Adicionalmente, este tipo de avaliação contribui para rapidamente identificar o esforço associado ao projeto e desenvolvimento de uma solução que atenda tais requisitos.

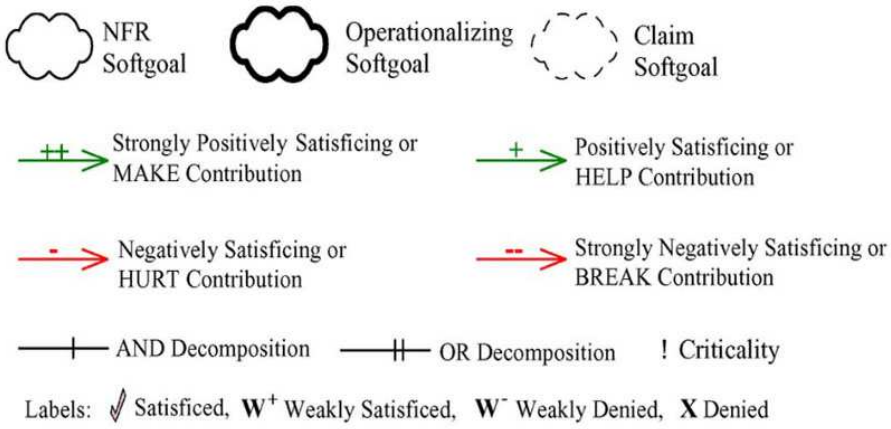
Desta forma, para o presente estudo foi optado pelo uso do SIG para a representação dos RNF que o framework RT-FRIDA permite modelar, em conjunto com as mudanças efetuadas para atender especificamente requisitos relacionados a falhas que degradam o sistema. Os gráficos SIG descrevem as dependências entre os *softgoals* e como eles são decompostos. Por *softgoal* entende-se como uma condição, ou estado do sistema que os *stakeholders* desejam alcançar (meta parcial, meta-objetivo), que deve ser cumprida para se chegar a um objetivo central (goal). Devido a representação formal dos elementos do NFR framework ser amplamente utilizada sem traduções, os termos de representação do SIG serão mantidos em Inglês.

De acordo com (SUBRAMANIAN; ZALEWSKI, 2016) para desenvolver um SIG que represente os requisitos não funcionais do projeto é necessário obter e classificar as propriedades do sistema a ser projetado. Sendo assim, o SIG foi desenvolvido com base na classificação de requisitos do RT-FRIDA e da especificação de falhas adicionada. Estes requisitos compõem um objetivo central (especificação de RNF para um sistema de controle automotivo) que é decomposto em vários *softgoals* e *subsoftgoals*. Em um gráfico SIG a representação de operacionalização das metas (*operationalizing softgoals*) devem satisfazer os softgoals acima (*parent softgoals*), assim, o SIG permite a observação do atendimento e cumprimento de cada meta, ajudando o engenheiro na tomada de importantes decisões relacionadas ao projeto.

A Figura 57 apresenta um sumário desta classificação de acordo com a ontologia usada no NFR framework. Os diferentes tipos de nuvens representam as metas parciais a serem atendidas para efetivar o objetivo da especificação de requisitos não funcionais.

Conforme apresentado em (CHUNG *et al.*, 2012), para a representação do SIG, o NFR framework utiliza uma ontologia específica baseada na representação de como ou de que forma as metas serão operacionalizadas *operationalizing softgoals*, como elas são

Figura 57 – Classificação base da ontologia usada no NFR framework.



Fonte: (SUBRAMANIAN; ZALEWSKI, 2016).

racionalizadas ou afirmadas *claim softgoals*, quais as contribuições *contributions*, e regras de propagação e decomposição *propagation rules*. As contribuições são formas de destacar o quanto um requisito impacta ou é atendido no sistema, sendo representados por fortemente atendido (*Make contribution*), positivamente atendido (*Help contribution*), fracamente atendido (Hurt contribution) e totalmente não atendido (*Break contribution*) (SUBRAMANIAN; ZALEWSKI, 2016).

Como apresentado na Figura 57 a ontologia é a regra de construção do SIG, onde esta deve ser adequada aos objetivos do projeto. De acordo com (KOBAYASHI *et al.*, 2016), estas regras são avaliadas para quantificar o atendimento dos requisitos não funcionais especificados, sendo assim, um método de identificação por pesos do quanto cada requisito é importante para o sistema deve ser utilizado (*softgoal weight method*).

Para o desenvolvimento do SIG "G" esse método é definido pela seguinte equação:

$$< g0, S, O, D, Pw, Cw, Aw > \tag{5}$$

onde *S* é o conjunto de softgoals do SIG "G",

o objetivo raiz *g0* é elemento principal de *S*,

O é um conjunto de *operationalization softgoals*,

D representa o relacionamento de dependência entre *S* e *O*,

P_w é o peso de prioridade (*weight*) para a decomposição de *softgoals*,

C_w define a contribuição entre os *parent and child softgoals*. Nesse caso segue-se os tipos de contribuição positiva e negativa especificados na ontologia, que são representados por linhas sólidas ou pontilhadas,

Por fim, A_w indica o quanto a especificação é atendida pelos meios de operacionalização (*operationalization softgoals*).

De acordo com (YAMAMOTO, 2015) o cálculo do atendimento referente a operacionalização depende da análise de todos os níveis acima, assim, o $A_w(g)$ é calculado pela

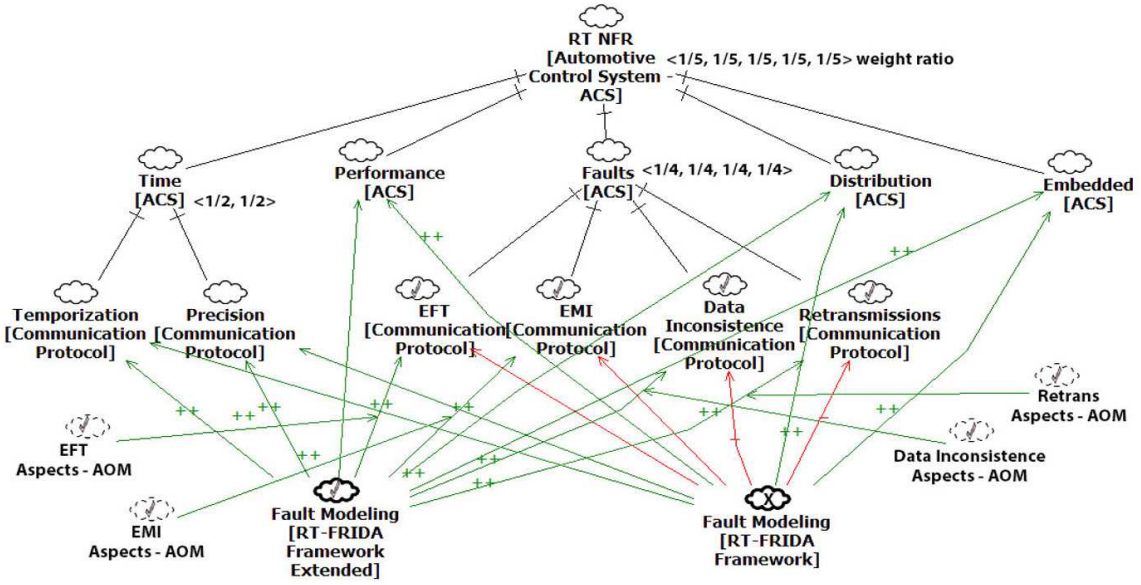
seguinte equação:

$$A_w(g) = \sum_{h} {}_{inChild(g)} P_w(h) * C_w(h) * A_w(h) \quad (6)$$

onde *Child(g)* é o conjunto de sub softgoals para atender o objetivo g.

Em seguida, com base nestas especificações o método *softgoal weight* foi aplicado para fins de verificação do impacto do uso do framework RT-FRIDA e de suas mudanças para especificação de requisitos relacionados a falhas. Com a análise é possível observar o esforço positivo de se tratar falhas em fases iniciais de projeto, considerando os requisitos especificados com base em um estudo prático de impacto de falhas EFT em sistemas de controle distribuídos. De acordo com o NFR framework apresentado por (CHUNG *et al.*, 2012) e recentes trabalhos relacionados a avaliação quantitativa de requisitos não funcionais (SUBRAMANIAN; ZALEWSKI, 2016) (YAMAMOTO, 2015), o SIG para o presente estudo foi desenvolvido, representando como seu objetivo principal no topo do SIG, a modelagem de RNF para sistemas de controle de tempo real, em redes intra-veiculares. A Figura 58 apresenta o SIG desenvolvido.

Figura 58 – Diagrama SIG desenvolvido para a avaliação da modelagem de RNF com o RT-FRIDA.



Fonte: Autor.

A base do SIG é representada pelos elementos de operacionalização, RT-FRIDA original e modelagem estendida com análise de falhas. Os pesos e prioridades são destacados ao lado do *softgoal*. A racionalização de como ocorre o tratamento das falhas e efeitos negativos das mesmas, é representada pelos *claim softgoal* estereotipados com aspectos, devido ao tratamento dado com conceitos de AOM. O principal RNF é decomposto nos *softgoals* "*Time Aspects*", "*Performance Aspects*", "*Distribution Aspects*", "*Embedded Aspects*", e aquele que caracteriza as mudanças mais importantes, o "Fault Aspects". Os

sub *softgoals* caracterizam o peso e prioridades no atendimento do *parent softgoal*. Nesse caso o peso foi definido de forma igual para cada *softgoal* devido a decisão de adotar uma prioridade igual para os requisitos do referido projeto, sendo assim, foram usados os pesos 1/5 para cada sub softgoal do SIG desenvolvido. Pesos iguais também foram definidos para os sub *softgoals* de *Time* e *Faults*. Essa decisão foi tomada com base nos efeitos da falha estudada, sendo considerado de igual importância o atendimento de todos os requisitos não funcionais. Nada impede o projetista de adotar outros valores em caso de haver prioridades maiores em outro projeto. Por exemplo, pode ocorrer de o requisito de performance ser mais crítico em determinada aplicação, perante determinada falha.

Assim, aplicando o método *softgoal weight* no SIG é definido o valor de impacto de atendimento da especificação de requisitos com o RT-FRIDA, incluindo com tratamento especifico de falhas. O valor obtido é apresentado a seguir:

(1/2+1/2)/5+ (1)/5 + (1/4+1/4+1/4+1/4)/5 + (1)/5 + (1)/5 = 1

E valor de impacto obtido com a versão original do RT-FRIDA é de:

(1/2+1/2)/5+(1)/5- (1/4+1/4+1/4+1/4)/5 + (1)/5 + (1)/5 = 3/5

Com essa análise destaca-se que se todos os requisitos não funcionais são atendidos, para o projeto em questão, o framework RT-FRIDA com modelagem especifica das falhas teria peso 1, ou seja, 100% de contribuição positiva no projeto. Por outro lado, a versão original do framework se aplicada ao mesmo projeto, teria um peso de 3/5 ou 60% de contribuição positiva no projeto. Cabe ressaltar que essa análise considera como foco central a modelagem de falhas, de tal modo que essa especifidade não é tratada pontualmente no framework original. Tal comparação não diminui ou invalida a aplicação do framework original, apenas destaca que se o tratamento de falhas for pontualmente tratado, as contribuições positivas para o projeto são maiores.

Neste sentido, de acordo com o SIG apresentado na Figura 58 é possível observar as contribuições do tratamento de falhas como aspectos, caracterizados pelos claim softgoals relacionados aos RNF abaixo do softgoal “Faults”. Esta contribuição mostra os benefícios de operacionalização da modelagem usando o framework RT-FRIDA com tratamento de falhas. É importante destacar que no SIG os sub *softgoals EFT, EMI, Data Inconsistency e Retransmissions*, são exemplos de questões que a modelagem pode tratar com o referido estudo. O estudo de caso pontual trata os os efeitos negativos dos transientes EFT, mas mostra a aplicabilidade da modelagem para outras falhas e em outros contextos.

4.3.4.1 Análise da aplicação do método

A análise desenvolvida com a aplicação do método *softgoal weight* em conjunto com o gráfico SIG destaca as contribuições da estrutura de modelagem de falhas proposta com base no framework RT-FRIDA. O gráfico SIG apresenta a versão estendida do framework, enfatizando o quinto elemento de especificação de requisitos relacionados a falhas e degradações de desempenho, o “Fault-Aspects”.

Este estudo apresenta uma análise de como as falhas podem ser tratadas nos estágios iniciais do projeto, e como seus efeitos de degradação podem ser mapeados para fornecer um mecanismo proativo de diagnóstico, detecção ou prevenção. Após experimentos práticos de análise de suscetibilidade a falhas, o framework RT-FRIDA estendido permite, por meio do desenvolvimento de checklists específicos, detalhar o tipo, a causa e os efeitos da degradação de falhas em estudo. Assim, estas informações fazem parte dos novos requisitos não funcionais especificados para o sistema de controle em estudo, os quais são apresentados em detalhes nos Apêndices A e B.

Enfatiza-se por meio desta análise a possibilidade de correlação de informações, mantendo um link permanente entre as fases de teste e conformidade dos sistemas de controle (com base nos dados obtidos dos testes susceptibilidade a falhas), com a abordagem de modelagem e especificação de requisitos não funcionais baseada em conceitos de orientação a aspectos. Essa melhoria ocorre devido à análise do sistema que está em teste, por meio de injeções de falhas, observando e coletando informações que podem comprometer o correto funcionamento do sistema de controle, e desta forma, adicionar requisitos aos novos projetos do sistema sob análise. Outro ponto importante desta abordagem de modelagem de falhas é a possível correlação em tempo de execução entre o aumento das retransmissões de mensagens e tratamento de erros pelo protocolo de comunicação, com a degradação de desempenho do sistema. Essa análise é possível devido à injeção de falhas e distúrbios durante a operação normal da rede de comunicação. Essas questões são exemplos de como a abordagem atual pode melhorar o projeto do sistema de controle com modelagem antecipada de falhas e seus efeitos indesejados (*early-modeling*).

O estudo realizado também contribui para mostrar a vantagem da presente abordagem usando um gráfico SIG, para assim melhorar a análise de requisitos. De acordo com (SUBRAMANIAN; ZALEWSKI, 2016), os diferentes tipos de relações entre *softgoals*, e entre *softgoals* e a operacionalização destes em um SIG, trazem a oportunidade de conduzir uma análise sistemática, propagando o impacto das decisões ao longo das relações durante a especificação de requisitos. Assim, combinado com o método quantitativo *softgoal weight* aplicado, é possível observar a contribuição da especificação de falhas como RNF e a modelagem cada vez mais cedo nas fases de projeto. Este tipo de estudo foi muito importante, pois validou a análise e especificação de requisitos baseado nos estudos de susceptibilidade a falhas que foram realizados, mostrando assim o impacto que estas mudanças podem causar nos sistemas projetados.

Neste contexto, o presente estudo resultou na proposta de uma solução para efetivar essa abordagem, caracterizado pela adição de um mecanismo de diagnóstico em tempo de execução, para monitorar (com registro de logs), detectar e notificar situações anômalas que possam degradar a performance de sistemas de controle críticos de tempo real. Tal mecanismo chamado de “Fault-Observer” é proposto (figuras 54, 55 e 56) e integrado ao sistema de controle de suspensão ativa, suportado pela modelagem de falhas com o

framework RT-FRIDA e conceitos de orientação a aspectos. De acordo com o estudo de caso, são registradas oscilações em métricas chave para a análise de desempenho do sistema de controle, como o average jitter e o difference jitter, seguindo gatilhos de diagnóstico previamente definidos. Durante testes práticos de susceptibilidade a falhas dos protocolos de comunicação intra-veiculares, que foram apresentados em pesquisas recentes (NAKAMURA *et al.*, 2015) (SHAH *et al.*, 2016) (ROQUE *et al.*, 2017) (POHREN *et al.*, 2019), as métricas average jitter e difference jitter são as mais usadas e contribuem para a avaliação do impacto negativo de falhas na degradação de performance dos sistemas de controle. Para os registros de anomalias de desempenho a proposta leva em conta os limites de memória local existente nas atuais ECUs, que futuramente devem sofrer alterações e estarem conectadas com sistemas de computação em nuvem.

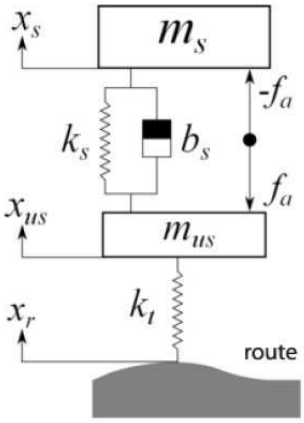
O estudo e os resultados apresentados no presente capítulo são destacados em publicação recente (ROQUE *et al.*, 2019), os quais foram discutidos e aprimorados com os feedbacks fornecidos durante o processo de revisão do periódico.

5 ESTUDO DE CASO EM SISTEMA DE CONTROLE INTRA-VEICULAR

Este capítulo apresenta de forma detalhada o processo experimental de validação da metodologia proposta nesta tese, dando ênfase ao desenvolvimento e integração de um mecanismo para diagnóstico de degradações no desempenho de um sistema de controle veicular, com restrições temporais críticas. Tal mecanismo foi desenvolvido de acordo com a modelagem apresentada anteriormente, tendo como base para o referido estudo de caso, o sistema de controle de suspensão ativa proposto em (MICHELIN, 2014), o qual segue o modelo Quarter-Car-Model (POUSSOT-VASSAL, 2008).

O principal objetivo de um sistema ativo de suspensão é isolar o chassi do veículo de perturbações geradas por irregularidades no terreno, proporcionando assim maior conforto. Sistemas de suspensão ativa tipicamente são constituídos por um conjunto composto por sistema sensorial e atuador eletro-hidráulico (ELATTAR; METWALLI; RABIE, 2016). A Figura 59 ilustra o conceito geral do sistema de suspensão ativa baseado no modelo Quarter-Car-Model.

Figura 59 – Modelo de suspensão Quarter-Car-Model.



Fonte: (MICHELIN, 2014).

Este modelo é constituído por um conjunto massa-mola-amortecedor normalmente usado para representar 1/4 do veículo, objetivando simplificar a análise e modelagem do

sistema de controle. Por meio de um atuador externo, este sistema é empregado para controlar o movimento vertical do corpo do veículo.

Este tipo de planta de controle é composta e descrita por diversos parâmetros, sendo um dos mais importantes o cálculo de deflexão da suspensão. A deflexão da suspensão é definida como a diferença entre a posição vertical do corpo do veículo X_s e a posição da linha de base do conjunto da suspensão X_{us}, como pode ser observado na Figura 59. Assim, a deflexão da suspensão é definida por:

$$X_{def} = X_s - X_{us} \tag{7}$$

A pesquisa de (MICHELIN, 2014) apresenta o desenvolvimento e análise de três sistemas de controle para a planta do sistema de suspensão ativa. São eles:

- 1- Alocação de Pólos via realimentação de estados.
- 2- Realimentação dinâmica de saída baseada no framework H-∞, e
- 3- Realimentação de estados baseada na metodologia de dados amostrados.

O referido trabalho verifica o desempenho dos sistemas de controle em uma rede FlexRay e com foco na dinâmica do sistema de suspensão ativa. Para o desenvolvimento do presente estudo de caso, foi utilizado como base o controlador 3 (Realimentação de estados baseada na metodologia de dados amostrados), devido este ter obtido um melhor desempenho nas análises apresentadas no estudo, priorizando o controlador que tolerou um maior atraso antes de se tornar instável.

Destaca-se que o presente estudo toma como ponto de partida a dinâmica da planta e o sistema de controle previamente estudado. Desta forma, são omitidos o detalhamento da planta e do controle modelado, os quais podem ser obtidos em (MICHELIN, 2014). Por conseguinte, foi agregado a este sistema de controle o mecanismo de diagnóstico desenvolvido de acordo com a metodologia proposta nesta tese, o qual permite efetuar a análise do comportamento da dinâmica de comunicação da rede intra-veicular, considerando três importantes protocolos de comunicação (CAN, CAN-FD e FlexRay).

O estudo de caso desenvolvido contribui apresentando um exemplo de aplicação da metodologia, considerando um cenário real de uma rede veicular perante distúrbios e falhas injetadas por dois tipos de hardwares: o primeiro baseado em um conjunto de ferramentas de teste amplamente usado na indústria automotiva - Vector *Tools* (VECTOR INFORMATIK, 2018); e o segundo por um hardware desenvolvido em conjunto no grupo de pesquisa GCAR, especificamente para a injeção de transientes elétricos rápidos (EFT), tipo de falha que foi objeto de estudo dentro do escopo da presente tese. A seguir, são apresentados detalhes das ferramentas usadas, da configuração da rede veicular e dos processos de teste efetuados.

5.1 Materiais e Métodos

5.1.1 Softwares utilizados

Para o desenvolvimento deste estudo foram utilizados os softwares CANoe / CANAnalyser e hardwares desenvolvidos pela empresa alemã Vector (VECTOR INFORMATIK, 2018), devido a ampla adoção na indústria automotiva para simulação, testes e desenvolvimento de sistemas de controle em redes veiculares. Outra razão da adoção destas ferramentas é oriunda de contatos prévios realizados com a empresa e o grupo de pesquisa do Instituto de Automação da Universidade de Stuttgart, em Stuttgart, Alemanha, para o desenvolvimento de pesquisas durante estágio que foi realizado na mesma Universidade. Esta parceria de pesquisa possibilitou a obtenção de licenças usadas durante o período de estudos, além de treinamento presencial e *feedback* obtido com profissionais do setor.

O software CANoe é muito conhecido por seus recursos de simulação de rede veicular, com a capacidade de simular vários nós, várias redes e vários tipos de barramento, como CAN, LIN, MOST e FlexRay. Assim, é possível testar cenários de controle exatamente como eles devem ser em situações reais, incluindo comunicação *powertrain*, LIN para eletrônica do corpo do veículo/iluminação, MOST para entretenimento/navegação GPS, e FlexRay para chassis e tarefas de alto desempenho ou confiabilidade. É considerada uma ferramenta de análise, teste e desenvolvimento de sistemas de controle para redes intra-veiculares, pois também possui uma linguagem de programação própria, permitindo a programação de processos de comunicação complexos.

A linguagem utilizada é chamada de CAPL. CAPL é uma linguagem de programação procedural baseada na linguagem C. A execução de blocos de programa é controlada por um paradigma orientado a eventos. Isso torna possível acessar todos os objetos contidos em uma base de dados contendo os componentes da rede (mensagens, sinais, variáveis de ambiente), bem como variáveis do sistema. A linguagem e o ambiente de programação ainda fornecem muitas funções predefinidas que trabalham em conjunto com as ferramentas de desenvolvimento, teste e simulação CANoe e CANalyzer.

5.1.2 Equipamentos utilizados

Os equipamentos utilizados na realização dos experimentos desta tese foram cedidos pela empresa Vector durante pesquisa anterior realizada no GCAR-PPGEE, por meio de contato realizado pelo prof. Dr. Carlos Eduardo Pereira. Em conjunto com o software CANoe foram utilizados dois equipamentos para a configuração dos nós que compõem a rede. Cada equipamento é composto por uma interface de hardware denominada VN8900 (sistema de tempo real disponível com processador Intel ATOM ou INTEL Core I7) e um módulo plug-in denominado VN8970, que possui a interface de conexão para com os tipos de protocolos configurados. A Figura 60 apresenta o principal equipamento utilizado nas implementações e experimentos realizados.

Figura 60 – Interface VN8910 com Módulo VN8970.

Fonte: (VECTOR INFORMATIK, 2018).

Outro hardware da Vector utilizado foi o equipamento VH6501 (*CAN disturbance interface*), um hardware específico para a injeção de distúrbios lógicos na rede veicular, que atualmente possui compatibilidade com redes CAN e CAN-FD. Este equipamento foi utilizado no primeiro cenário de testes apresentado neste estudo de caso. A seguir a Figura 61 ilustra o equipamento VH6501.

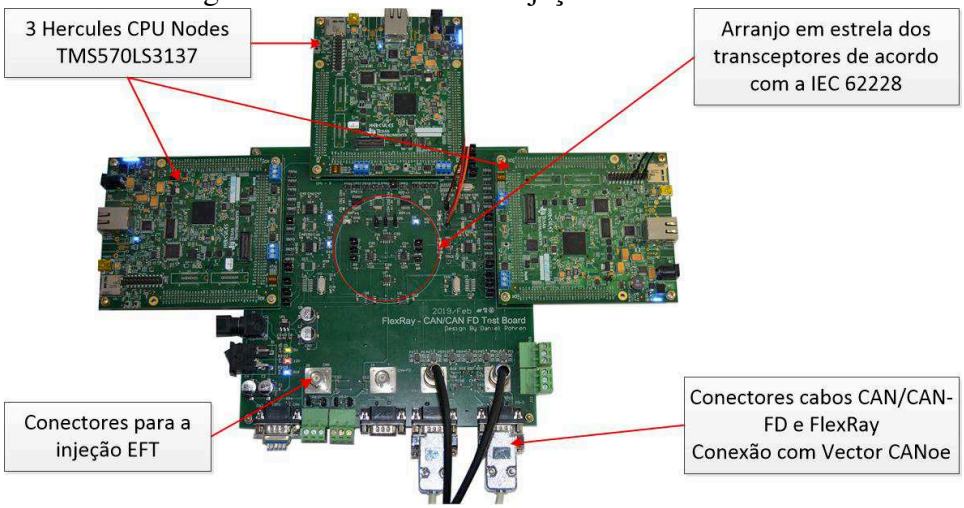
Figura 61 – Equipamento para injeção de distúrbios nas redes CAN - VH6501.

Fonte: (VECTOR INFORMATIK, 2018).

Adicionalmente, o segundo cenário de testes do estudo de caso utiliza outro tipo de hardware para a injeção de falhas na rede, hardware este que foi desenvolvido no mesmo grupo de pesquisa (GCAR-PPGEE) e faz parte da dissertação de mestrado de Daniel Pohren (POHREN, 2020). A Figura 62 apresenta o hardware de injeção de falhas EFT utilizado no segundo cenário de testes, após recentes atualizações realizadas.

O hardware originou do primeiro estudo realizado sobre a susceptibilidade do protocolo CAN aos transientes elétricos rápidos (EFT) (ROQUE *et al.*, 2017), sendo atualizado e ampliado para automatizar o processo de teste, cumprindo também os requisitos dos padrões IEC 62228, IEC 7637 e ISO 26262. Cabe ressaltar que o hardware projetado permite a implementação de uma planta de controle com os três nós especificados na Figura 62, mas também pode conter nós configurados e implementados em outros hardwares, conectados ao mesmo barramento pelos conectores especificados. Devido ao escopo de implementação deste trabalho estar na utilização dos hardwares da Vector, dois hardwares VN8900 são utilizados para a implementação do sistema de controle de suspensão ativa e conectados a este barramento para a injeção e análise das falhas EFT. Ambos os

Figura 62 – Hardware de Injeção de falhas EFT.



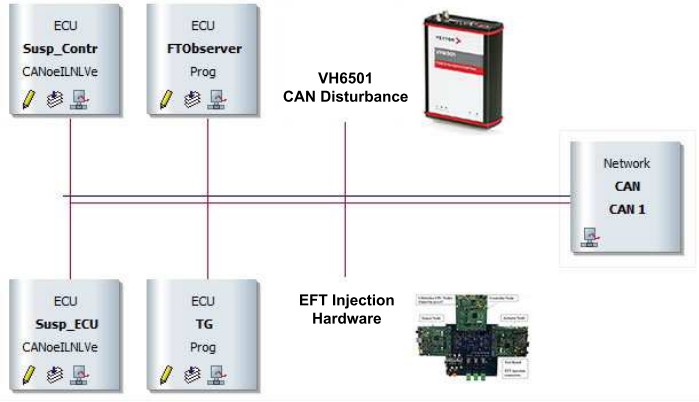
Fonte: Adaptado de (POHREN *et al.*, 2019).

projetos de hardware para injeção EFT desenvolvidos tiveram resultados e contribuições destacadas, as quais foram apresentadas nas seções 4.1.1, 4.1.2 e 4.1.3.

5.1.3 Especificação da rede intra-veicular e análise de desempenho

De acordo com o presente estudo de caso uma rede veicular foi configurada para a realização dos experimentos, sendo composta por seis nós, três da planta de controle, um nó para injeção de tráfego, um para o mecanismo de diagnóstico e um especificamente para a realização da injeção de distúrbios. Essa configuração segue a topologia em barramento e é configurada dentro do software Vector CANoe. A seguir a Figura 63 apresenta a topologia de rede utilizada nos experimentos.

Figura 63 – Topologia de rede com as ECUs e hardwares de Injeção de falhas usados nos experimentos.



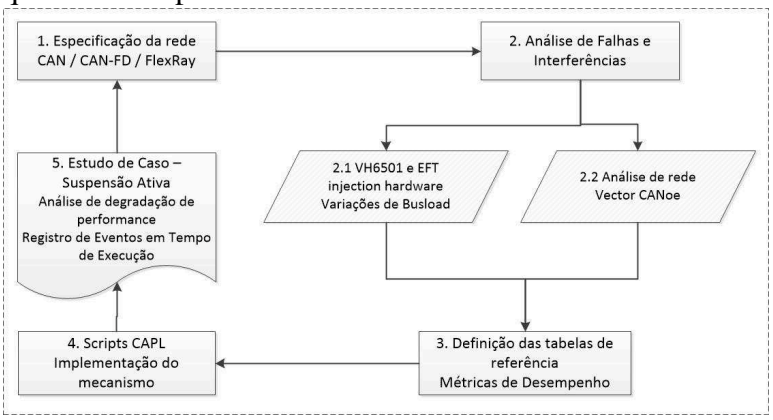
Fonte: Autor.

De acordo com a figura 63, dentro do software CANoe são configuradas 4 ECUs, com seus respectivos códigos CAPL. A ECU “Susp-Contr” representa o nó e a programação do sistema de controle de suspensão ativa que está em execução. A ECU “Susp-ECU”

representa a programação referente aos nós Sensor e Atuador no escopo da planta de controle. A ECU ”TG“ contém os códigos para a geração de tráfego de mensagem na rede. A ECU “FTObserver” é responsável pelo monitoramento da rede em tempo real, com o mecanismo de diagnóstico modelado com base nos testes previamente efetuados. Por fim, mais um nó/ECU é adicionado a essa topologia, o qual representa em cada cenário de testes os respectivos hardwares de injeção de falhas e distúrbios na rede de comunicação.

Para a sequencia de experimentos foi definido um método de teste baseado na contribuição apresentada em (ROQUE *et al.*, 2017), o qual é apresentado na Figura 64. É importante destacar que todo o processo de experimentação possui um período de calibração das ferramentas, por meio de um processo de repetição (em torno de 10 vezes) das injeções de falhas e também coleta de dados das ferramentas de análise. Esse procedimento garante que os dados obtidos e analisados são consistentes e passíveis de reprodução.

Figura 64 – Sequencia de experimentos realizados no desenvolvimento do estudo de caso.



Fonte: Autor.

O primeiro passo da sequencia de experimentos (“1. Especificação da rede CAN/CAN-FD/FlexRay”) é a definição da topologia de rede em barramento, os protocolos de comunicação que serão analisados e configurados para os testes (CAN, CAN-FD e FleRay), e as mensagens que compõem o sistema de controle distribuído. A Tabela 10 especifica as mensagens e detalha as respectivas funções de cada nó configurado.

De acordo com o sistema de controle em estudo, as mensagens apresentadas na Tabela 10 são transmitidas ciclicamente, sendo a mensagem referente à Lei de Controle a principal em análise dentro do estudo (MSG-Control-Law, periodicidade de 5 ms). A configuração de mensagens representa as condições normais de operação do sistema de suspensão ativa objeto do estudo anterior de (MICHELIN, 2014). Seguindo esta configuração dos nós, os testes foram realizados com os seguintes protocolos:

- protocolo CAN, operando a 1 Mbps;
- protocolo CAN-FD, operando de 1 (campos de arbitragem) a 4 Mbps,
- protocolo FlexRay, operando a 10 Mbps.

Tabela 10 – Especificação da mensagens que compõem a rede de comunicação.

Nó	Função	Dados / TX–RX mensagens
Sensor	Conjunto massa-mola de amortecimento. Parâmetros do modelo de sistema de suspensão ativa.	MSG–Body–Vert–Speed, MSG–Susp–Deflection e MSG–Tire–Deflection
Atuador	Ajustes na posição vertical do conjunto de suspensão.	MSG–Susp–Set–Vert–Speed
Controlador	Computação dos níveis apropriados para ajustes dos parâmetros da suspensão. Lei de Controle.	MSG–Control–Law
TG	Gerador de tráfego de mensagens.	MSG-StressN - CAN/CAN-FD DummySS/DN - FlexRay
FTObserver	Envio de mensagens de diagnóstico. Especificado para uso futuro.	MSG-Diagnostic-N

Devido às características do sistema de controle utilizado, todos os pacotes de dados foram de 8 bytes, não necessitando uma variação maior de utilização do campo de dados úteis (*payload*) no protocolo CAN-FD. A implementação de cada código CAPL foi realizada no equipamento Vector VN8910A com o plugin VN8970. O nó controlador é responsável pelo cálculo da dinâmica de controle simulada e representa o link entre o nó controlador e os nós da suspensão (sensores e atuadores). Nas mensagens de tráfego o campo de dados das mensagens se manteve no padrão de 8 bytes, tendo a variação aumentada para 16 e 32 bytes no CAN-FD apenas para fins de aumento do tráfego e carga da rede nos diferentes cenários simulados. Essa decisão foi adotada para avaliar o mecanismo de diagnóstico e o comportamento do sistema em situações críticas e também com o uso amplo da largura de banda do canal. Outro ponto importante a destacar é a utilização do protocolo FlexRay somente nos testes com o hardware de injeção de EFT, devido ao hardware VH6501 ainda não suportar o uso do protocolo FlexRay.

O segundo passo (“2. Análise de Falhas”) representa a injeção de falhas e distúrbios na rede veicular, injeções estas que no caso do hardware VH6501, foi feita usando gatilhos específicos para as mensagens do sistema de controle e também com um nó extra adicionado para geração de tráfego na rede. A configuração e análise de desempenho da rede foi realizada com o software CANoe/CANAnalyser. É comum o uso destas ferramentas pela indústria automotiva para o projeto e testes de conformidade de sistemas de controle, provendo características idênticas ao cenário real de operação. O conjunto de ferramentas segue a seguinte sequencia de configuração:

- VH6501 - Hardware para injeção de distúrbios em CAN/CAN-FD.
- *Hardware* de injeção de transientes EFT (CAN/CAN-FD e FleRay).
- VN8910A/VN8970 - Implementação das ECUs e configuração dos nós da rede.

- Vector CANoe - implementação dos scripts de funcionamento do sistema de controle, scripts de análise da rede e scripts de injeção de falhas. Todo o desenvolvimento na linguagem CAPL.

Maiores detalhes sobre a configuração de hardware de injeção de falhas e distúrbios serão apresentados nas subseções seguintes que apresentam os experimentos realizados.

O terceiro passo ("3. Definição de tabelas de referência.") é caracterizado pela composição das tabelas de referência para os testes, preenchidas levando em consideração os dados obtidos no testes de susceptibilidade previamente realizados. Com o objetivo de verificar os típicos atrasos de comunicação durante a operação normal do sistema de controle (sem interferências ou distúrbios acentuados), vários testes preliminares foram realizados, obtendo os dados de desempenho da rede para cada um dos protocolos estudados e em diferentes cargas de ocupação (12% - somente o sistema de controle em CAN/CAN-FD, 2% - somente sistema de controle em FlexRay, 30%, e 80% com a adição de tráfego gerado por software).

Os dados que compõem as tabelas de referências são guias para a definição dos gatilhos de diagnóstico, especificamente com o hardware de injeção de EFT. O presente estudo de caso objetiva demonstrar a capacidade de detecção dos distúrbios na rede, sendo assim, não são necessários testes com todas as cargas de ocupação de rede, pois apenas demandariam um estudo similar e repetitivo. Assim, nos experimentos realizados com o hardware de injeção de EFT, devido a variação tripla de injeções e rajadas de EFT (47, 57 e 63 volts), os testes do mecanismo de diagnóstico foram realizados considerando 2% (FlexRay), 12% (CAN e CAN-FD) e 30% (em todos os protocolos) de carga de ocupação da rede, que já são suficientes para verificar a capacidade de detecção das degradações de desempenho na rede veicular. De outra forma, no estudo com o hardware VH6501, como não existem grandes variações dos tipos de distúrbios, os testes foram realizados com 12%, 30%, 50% e 80% de carga de ocupação da rede.

As tabelas 11, 12 e 13 mostram os dados obtidos nos testes preliminares e os gatilhos de diagnóstico definidos para cada um dos protocolos (valores em microsegundos).

Tabela 11 – Dados de referência para os testes com o protocolo CAN.

Metric/Trigger	Bus 12%	Bus 30%
Difference Jitter (us)	240	790
Ref Trig1 (us)	250	800
Average Jitter (us)	33,12	72,52
Ref Trig2 +25% (us)	42	90

A tabela 13 possui particularidades claras referentes ao desempenho superior do protocolo FlexRay, onde as variações medidas sem a injeção de distúrbios geram pequenos atrasos e tipicamente na ordem de nanosegundos, como também apresentados no estudo

Tabela 12 – Dados de referência para os testes com o protocolo CAN-FD.

Metric/Trigger	Bus 12%	Bus 30%
Difference Jitter (us)	185,45	297,56
Ref Trig1 (us)	190	300
Average Jitter (us)	20,34	49,50
Ref Trig2 +25% (us)	25	62

Tabela 13 – Dados de referência para os testes com o protocolo FlexRay.

Metric/Trigger	BusSS 2%	BusSS 30%
Difference Jitter (us)	0,0771	0,0775
Ref Trig1 (us)	0,080	0,080
Average Jitter (us)	0,0254	0,0257
Ref Trig2 +25% (us)	0,032	0,032

de (MICHELIN, 2014). Outro ponto a destacar nos testes com FlexRay, é que por questão de coerência com os testes realizados no estudo anterior, as variações de carga ocorreram no segmento estático (neste caso com 2% e 30%), sendo mantido o segmento dinâmico em 100% de ocupação para fins de análise de estresse no protocolo FlexRay.

O valor de referência para a métrica de desempenho *diference jitter* foi definida baseada nos picos de atraso observados, tendo em vista que essa métrica registra as oscilações abruptas entre as mensagens do ciclo de controle. Para a métrica *average jitter* os valores de referência foram acrescidos de aproximadamente 25%. Esta decisão foi tomada devido à mudança na média de atraso levar um período maior de tempo para se alterar, assim, um gatilho nesse limite indica degradações de desempenho contínuas e mais duradouras (que podem indicar defeitos em componentes ou ataques maliciosos na rede), de acordo também com observações de degradações causadas por falhas analisadas em recentes pesquisas (ROQUE *et al.*, 2017) (POHREN *et al.*, 2019). Apesar destes gatilhos serem configurados com foco neste estudo de caso, os ajustes destes parâmetros podem facilmente ser alterados em futuras aplicações da metodologia.

Com base nestas informações, os passos 4 e 5 (“4. Scripts CAPL - Implementação do mecanismo” e “5. Estudo de Caso - Suspensão Ativa”) representam a implementação do mecanismo de diagnóstico em linguagem CAPL (de acordo com modelagem apresentada no Capítulo 4), além de processo de teste e validação com experimentos conduzidos no estudo de caso. A implementação leva em consideração as métricas de desempenho definidas para a avaliação da rede e os gatilhos acionados de acordo com as tabelas de referência para cada protocolo.

Adicionalmente e para fins de análise posterior, que podem servir para futuros trabalhos, todos os eventos de diagnóstico detectados com estes gatilhos são armazenados em

logs, identificando o nó, a mensagem e os valores das métricas. Essas informações ainda podem ser acrescidas e configuradas pelo engenheiro de teste, com informações sobre a taxa de perda de pacotes/taxa de erros, carga total da rede ou carga efetiva, dependendo dos objetivos e do sistema de controle que está sob análise. Essas informações podem facilmente ser adicionadas com rotinas já mapeadas no mecanismo atual, as quais são obtidas também por funções pré-definidas dentro do software CANoe. A Figura 65 ilustra um exemplo dos dados armazenados em arquivo de log sobre os eventos diagnosticados na rede.

Figura 65 – Exemplo de arquivo de log com eventos diagnosticados e registrados.

```
-----------------------------------------------------
Node | DifTime | DifJit | AvgJit | WCET | Trig1 | Trig2
-----------------------------------------------------
1 ; 5000047 ; 47 ; 23.500 ; 5000047 ; 0 ; 0
1 ; 5000000 ; 47 ; 22.156 ; 5000047 ; 0 ; 0
1 ; 5000078 ; 78 ; 33.116 ; 5000078 ; 0 ; 1
1 ; 5000000 ; 78 ; 32.149 ; 5000078 ; 0 ; 1
1 ; 5000047 ; 47 ; 30.472 ; 5000078 ; 0 ; 0
1 ; 5000000 ; 47 ; 29.942 ; 5000078 ; 0 ; 0
1 ; 5000078 ; 78 ; 33.116 ; 5000078 ; 0 ; 1
1 ; 5000000 ; 78 ; 32.730 ; 5000078 ; 0 ; 1
1 ; 5000047 ; 47 ; 31.581 ; 5000078 ; 0 ; 0
1 ; 5000000 ; 47 ; 31.299 ; 5000078 ; 0 ; 0
```

Fonte: Autor.

Neste sentido, métricas para avaliação de desempenho de rede são essenciais para as fases de teste e normalmente o *jitter* é usado para avaliar o tempo de oscilação entre as mensagens recebidas e transmitidas. O *jitter* é calculado com base na variação do tempo de transmissão de uma mensagem no barramento (MARY; ALEX; JENKINS, 2013). Por exemplo, uma mensagem CAN com campo de dados padrão de 8 bytes leva em torno de 130 us para ser transmitida a 1 Mbps com *BitStuff* de 5 bits. Em teoria este é o maior tempo de resposta de uma mensagem no barramento, considerando uma situação ótima.

Essas métricas são calculadas com auxílio do software Vector CANoe, que fornece rotinas para a captura do tempo de transmissão entre as mensagens. Assim, de acordo com (NAHAS; PONT; SHORT, 2009) foram utilizadas as seguintes métricas para a avaliação temporal de desempenho da rede:

- *Average Jitter* - caracterizado pelo desvio padrão na medida do tempo médio de transmissão de mensagens. No entanto, o jitter médio é calculado instantaneamente e atualizado de acordo com a evolução do desempenho do sistema.

- *Difference Jitter* - calculado subtraindo o melhor tempo de transmissão do pior tempo de transmissão de um período medido. Neste caso, em função do monitoramento em tempo de execução, o valor também é obtido registrando os picos de transmissão de uma mensagem crítica durante todo o período monitorado.

- *Busload* - a variação de carga da rede é obtida por rotinas internas do software CANoe, de acordo com a programação dos nós que injetam tráfego na rede.

Os referidos termos das métricas foram mantidos em inglês devido a ampla adoção na literatura. Em seguida, de forma complementar aos dados obtidos na ferramenta, é necessário a realização dos cálculo em tempo de execução e continuamente do *Average Jitter*, o qual é realizado de acordo com a abordagem de (NAHAS; PONT; SHORT, 2009) e detalhado na seguinte equação:

$$InstantAvgJit = \sqrt{\sum_{i}^{n} \frac{(JM_{[i]} - MT_A)^2}{nMsg}} \tag{8}$$

onde,

$JM_{[i]}$ é o jitter até um dado ciclo de controle que pertence ao período/janela medido; O jitter das mensagens é calculado de acordo com a rotina "DiffTime" provida pela ferramenta CANoe e acionada também dentro do código CAPL do mecanismo. Cada variação registrada até o respectivo tempo é registrada em um array e em arquivos de logs, permitindo o cálculo das oscilações de desempenho durante todo o período monitorado;

MT_A é a média aritmética no respectivo ponto no tempo;

$nMsg$ numero total de uma dada mensagem.

É importante destacar que o cálculo das métricas é associado e registrado de acordo com as mensagens críticas que compõem o sistema de controle sob análise, neste caso referente ao sistema de suspensão ativa.

Para efetivar esse processo um algoritmo foi desenvolvido incorporando o processamento das métricas, a ativação dos gatilhos de diagnóstico, envio de mensagens de diagnóstico e também todas as rotinas de registro de logs do sistema de controle. O algoritmo permite a análise de sistemas de controle de tempo real registrando os eventos que indicam degradação de performance que podem ser críticas para o seu correto funcionamento. A seguir é apresentado o pseudo código do algoritmo desenvolvido.

Algorithm 1 RT-Jitter monitor

```
1:  procedure CRITICALMSGMONITOR(MESSAGEID)
2:      RangeLimit ← Percentual ou limite de tolerância para os gatilhos
3:      while rede comunicando - checar ID da Mensagem Crítica do
4:          DifJit ← Cálculo da métrica Difference Jitter
5:          AvgJit ← Cálculo da métrica Average Jitter
6:          if DifJit/AvgJit > RangeLimit then
7:              Log ← Registro de Logs dos eventos
8:              Time ← Registro do timestamp da ocorrência
9:              MsgID ← MessageID                ▷ Id da msg afetada por distúrbio.
10:             SendMsgDiagnostic()                          ▷ Possibilidade.
11:         end if
12:         checkECUMemory()        ▷ Verificar limite de armazenamento interno.
13:         SendMessagetoCloud()                             ▷ Possibilidade.
14:     end while
15: end procedure
```

A presente técnica permite o diagnóstico do sistema sob falhas e interferências, registrando todos os eventos de distúrbios de desempenho, associando o nó e o sistema de controle que esta em análise. Destaca-se que o foco é monitorar os sistemas de controle críticos de tempo real com restrições de tempo rígidas, no cenário automotivo normalmente com tempo de ciclo abaixo de 20 milissegundos. Para aplicações futuras, foi previsto a possibilidade de o algoritmo também enviar mensagens de diagnóstico, permitindo o desenvolvimento de aplicações de alto nível, por exemplo, para gerar notificações e também enviar dados de diagnóstico para sistemas em nuvem (*cloud computing*). Estas possibilidades não foram implementadas devido as restrições de hardware e também escopo da atual pesquisa.

Os eventos armazenados permitem ao engenheiro a observação de como o sistema de controle opera durante diferentes situações críticas (durante falhas e interferências), verificando também a influência das variações de carga da rede (*busload*) no sistema de diagnóstico. Da mesma forma, com a presente metodologia também é possível verificar o desempenho dos protocolos de comunicação nas mesmas situações, indicando os limites e os parâmetros críticos para a operação correta em sistemas de controle de tempo real críticos.

As próximas seções detalham os testes e análise de desempenho realizadas com os dois tipos de hardware usados para injeção de distúrbios e falhas na rede de comunicação intra-veicular, com foco nos protocolos CAN, CAN-FD e FlexRay. Os detalhes qualitativos e quantitativos destes experimentos são compilados e discutidos na Seção 6, com a análise dos dados dos experimentos realizados e validação da metodologia proposta.

5.2 Cenário de Teste 1 - Análise do mecanismo de diagnóstico com distúrbios gerados pelo hardware Vector VH6501

O primeiro cenário de teste avalia a metodologia de teste proposta e o mecanismo de diagnóstico modelado (Capítulo 4) utilizando uma ferramenta usada e certificada pela indústria automotiva, o hardware Vector VH6501. Uma sequencia de experimentos para análise prévia da rede foi realizada com a ferramenta VH6501, com o sistema de controle de suspensão ativa, para fins de definir os parâmetros de referência dos gatilhos. Uma nova tabela de referência é necessária pela natureza diferente do distúrbio (em relação ao hardware de injeção de falhas usado no segundo estudo de caso) e consequentemente gerar um comportamento diferente na rede de comunicação. As tabelas 14 e 15 apresentam os dados obtidos e os valores de referência definidos. Os experimentos foram realizados somente com os protocolos CAN e CAN-FD, devido a ferramenta VH6501 ainda não suportar a comunicação FlexRay. Os scripts de teste que foram desenvolvidos na linguagem CAPL podem ser visualizados no Apêndice E.

O hardware Vector VH6501 (*CAN disturbance interface*) possui funcionalidades es-

Tabela 14 – Dados de referência para os testes com o protocolo CAN com a ferramenta VH6501.

Metric/Trigger	Bus 12%	Bus 30%	Bus 50%	Bus 80%
Difference Jitter (us)	198	979	1989	4930
Ref Trig1 (us)	200	1000	2000	5000
Average Jitter (us)	47,05	110,92	234,62	673,12
Ref Trig2 +25% (us)	60	140	294	842

Tabela 15 – Dados de referência para os testes com o protocolo CAN-FD com a ferramenta VH6501.

Metric/Trigger	Bus 12%	Bus 30%	Bus 50%	Bus 80%
Difference Jitter (us)	185	289	1052	4970
Ref Trig1 (us)	190	300	1060	5000
Average Jitter (us)	19,78	49,16	95,42	672,84
Ref Trig2 +25% (us)	25	62	120	842

pecíficas para a geração de distúrbios nos sinais de redes CAN e CAN-FD. Essa ferramenta permite a observação do comportamento do sistema de controle sob degradação de performance, pois, possibilita a geração de distúrbios em partes específicas do quadro CAN. Para o uso da ferramenta VH6501 existem pelo menos 9 pré-configurações de distúrbios lógicos, chamados dentro da ferramenta de "*Desktops*". Cada pré-configuração se difere apenas no tipo de gatilho "*trigger*" do distúrbio, sendo todas elas focadas em bits específicos dentro do quadro CAN. Apesar de ser possível usar o painel de acionamento para cada um dos *desktops* de teste, para fins de automatizar os experimentos realizados, para todos os testes os gatilhos de distúrbios foram acionados por temporizadores ("*timers*") que foram programados dentro do script CAPL de injeção dos distúrbios. Cada *desktop* ainda possui parâmetros que permitem configurar uma variedade de sequências/-repetições de injeção aumentando o nível de perturbação na rede. Desta forma, foram escolhidos os desktops que geram os tratamentos de falhas mais comuns nos protocolos de comunicação. Sendo assim, foram escolhidos os seguintes *desktops*:

Software Trigger - Um *desktop* configurado por software que imediatamente envia uma sequencia de bits ou uma sequencia de quadros na rede CAN. Os referidos distúrbios são:

- ***Send One Dominant Bit Once*** - Um bit dominante de tamanho de 2 us ou 1 us é enviado diretamente no barramento. Isso leva a uma situação de *Stuff Bit Error*, uma vez que as interfaces CAN entendem que é um inicio de frame - SOF, mas nada é enviado na sequencia.

- ***Send One Dominant Bit Multiple*** - Essa opção apenas se difere da anterior por

usar um parâmetro extra que permite o envio de uma repetição de bits dominantes do tamanho pré-definido, gerando assim um tratamento maior de *Stuff Bit Errors* por parte do protocolo.

- ***Send Frame Once*** - Essa opção permite o envio de um distúrbio em um frame específico, usando o seu identificador como parâmetro, porém sem dados, definindo DLC = 0.

- ***Send Frame Multiple*** - Com essa opção uma sequencia de frames para a específica mensagem são enviados com DLC = 0. Ambos os distúrbios de frame fazem com que erros de processamento e inconsistência de dados ocorram em mensagens específicas.

Frame Trigger - Com esse *desktop* é configurado um gatilho para uma mensagem específica, gerando uma sequencia de distúrbios em posições especificas do quadro CAN. Essa opção permite 6 tipos diferentes de distúrbios, com configurações e efeitos similares. Neste experimento são usadas as quatro primeiras configurações, devido as duas últimas configurações serem repetições idênticas, porém ativadas por um gatilho externo que pode ser conectado a porta de I/O do hardware. Os referidos gatilhos usados são:

- ***Disturb Ack Slot Once*** - Um gatilho é configurado para um mensagem com base em seu identificador, tendo sua posição definida no delimitador CRC. Assim, é gerado um distúrbio no bit seguinte de *Ack slot*, enviando um nível recessivo ao invés de um dominante.

- ***Disturb Ack Delimiter Multiple*** - Essa opção apenas se difere da anterior por usar o gatilho no bit seguinte de *Ack Slot* para assim gerar uma sequencia de distúrbios no bit de *Ack Delimiter*, enviando um nível dominante ao invés de um recessivo.

- ***Start Sending Frame in IFS Once*** - Essa configuração permite usar um gatilho para uma mensagem específica, tendo como posição do gatilho o segundo bit do espaço de inter frames - IFS. Assim uma mensagem de DLC = 0 é enviada sem respeitar o espaço de inter frames gerando quadros de erros e incrementando o contador de transmissões (bursts), afetando em consequência a carga da rede.

- ***Start Sending Frame in IFS Multiple*** - Com esta opção tem-se o mesmo efeito de distúrbio da opção anterior, mas gerando uma sequencia maior de quadros de erros.

Missing Bit Trigger - Esse desktop é configurado como uma extensão do desktop anterior, porém agora configurando um gatilho para o campo CRC da mensagem. Se o trigger for habilitado, a mensagem com um identificador previamente especificado e com o offset configurado se torna o ponto do distúrbio. Nesse desktop duas opções são utilizadas e habilitadas via código CAPL. Esta opções são apresentadas a seguir:

- ***MissingBitTrigger*** - Esta opção é utilizada apenas para habilitar e desabilitar o gatilho que verifica uma mensagem específica.

- ***Disturb CRC Delimiter Multiple*** - O gatilho é configurado para a mensagem com base em seu identificador, tendo sua posição definida como sendo o campo CRC do frame. Assim, são geradas sequencias de distúrbios de tamanho de um bit do barramento, no campo *CRC Delimiter*, enviando um nível dominante ao invés de um recessivo.

No presente experimento a injeção de distúrbios foi feita em duas sequências, uma com o *desktop* "*Software Trigger*" e outra com a união dos *desktops* "*Frame Trigger*" e "*Missing Bit Trigger*". É importante notar que os *scripts* de teste CAPL foram codificados para o presente estudo, devido a ferramenta apresentar apenas códigos de exemplo que devem ser adaptados para cada estudo e aplicação.

De acordo com descrições dos *desktops* utilizados nos experimentos, campos específicos dos quadros CAN são afetados, como por exemplo, o SOF, *Ack Slot*, *Ack Delimiter*, IFS, *CRC frame* e *CRC Delimiter*. Todos os distúrbios geram um tratamento excessivo de erros, acionando os típicos mecanismos de tratamento nativos do protocolo. A seguir são apresentados os resultados de aplicação destes *desktops* nos protocolos CAN e CAN-FD.

5.2.1 Injeção dos transientes no protocolo CAN e CAN-FD

Como especificado anteriormente, o sistema de diagnóstico modelado tem como base o sistema de controle de suspensão ativa operando em diferentes condições de cargas da rede (com 12%, 30%, 50% e 80% de carga de ocupação). Os testes seguem os dados prévios que foram apresentados nas tabelas 14 e 15, tendo um período de amostragem de 120 segundos, com injeções de distúrbios em intervalos de 5 segundos, para fins de observar os impactos de cada desktop no tratamento de erros do protocolo. A decisão deste tempo de amostragem também leva em consideração que, dentro deste período, foi possível observar todas as funções programadas no mecanismo de diagnóstico.

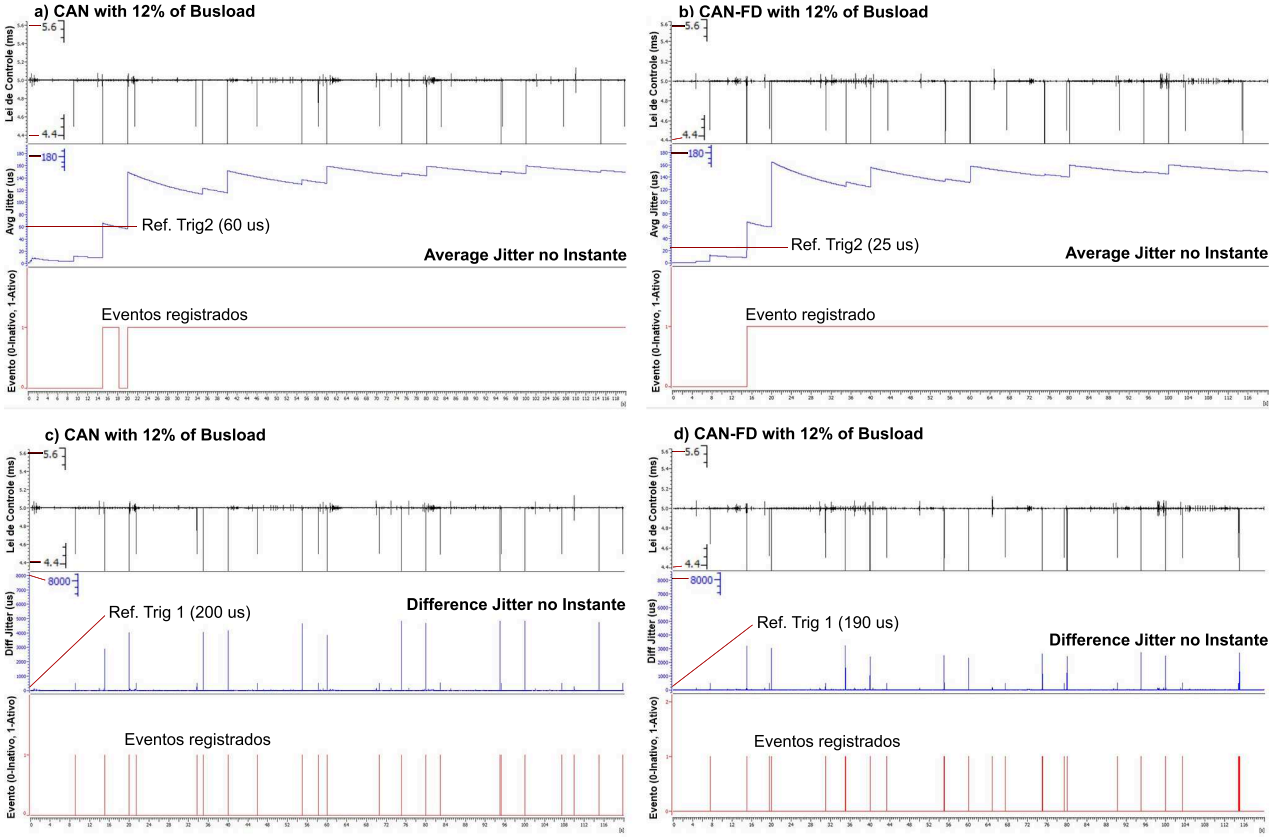
De acordo com (KOPETZ, 2002) a observação de um sistema de tempo real deve priorizar os eventos chave a se analisar, permitidos por um período de amostragem. Desta forma, no presente estudo o período de amostragem está relacionado com a análise de oscilações no tempo de ciclo de controle, influenciado por mensagens específicas da rede de comunicação. Neste caso, em apenas um segundo podem ser registrados 200 ciclos de controle, assim, as análises podem ser realizadas em poucos segundos de monitoramento da rede. Outro ponto a destacar, é que grande parte dos testes de susceptibilidade a falhas e interferências destacados na literatura, normalmente são realizados em poucos segundos, seguindo também as limitações das ferramentas de análise utilizadas na indústria.

Os distúrbios são injetados pela ferramenta VH6501 com uma periodicidade de 5 segundos, para ambas as sequencias "*Software Trigger*" e "*Frame/Missing Bit Trigger*".

Esse intervalo leva em consideração que no código CAPL de cada *desktop*, repetições/rajadas de distúrbios são geradas, ocasionando uma interferência mais duradoura na rede. Em conjunto aos distúrbios injetados o mecanismo de diagnóstico desenvolvido verifica as variações de performance da rede e do sistema de controle, identificando e registrando todas as situações anômalas. A seguir a Figura 66 a) e b) apresentam os gráficos resultantes dos testes na injeção de distúrbios com 12% de carga (somente o sistema de controle operando na rede), com a métrica *average jitter*. Logo abaixo a Figura 66 c) e d) mostra a geração dos dados com base na métrica *difference jitter*.

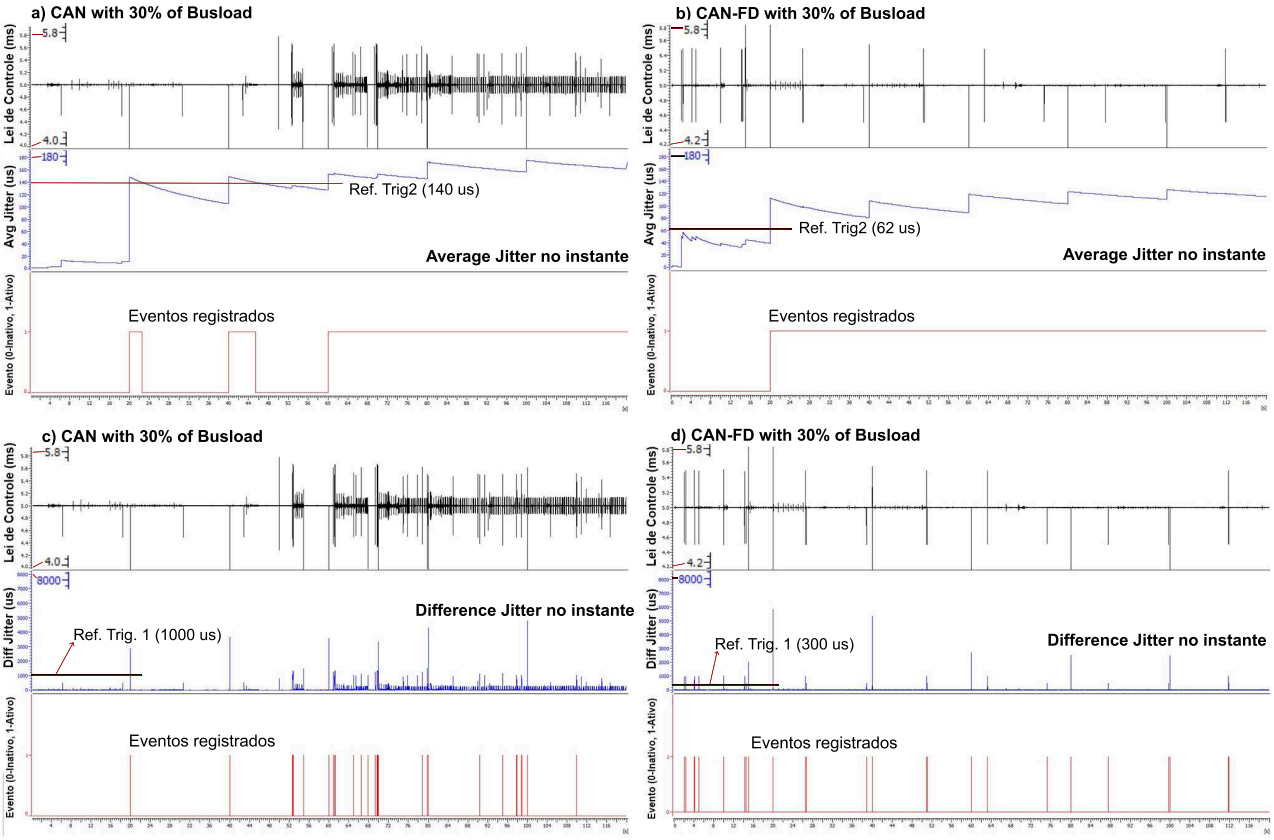
Do mesmo modo, nas figuras 67 a), b), c) e d) são apresentados os testes com 30% de carga, com ambas as métricas de análise. Neste caso, inclui-se o sistema de controle em conjunto com mensagens adicionais geradas pelo nó "TG", responsável por gerar tráfego na rede CAN/CAN-FD. Os gráficos gerados nas figuras 66 e 67 foram obtidos diretamente da ferramenta de análise integrada ao software Vector CANoe. Para cada gráfico gerado a ferramenta também gera o respectivo arquivo de log da comunicação. Adicionalmente, foram gerados arquivos de logs específicos para os eventos oriundos do mecanismo de diagnóstico, registrados durante o período de monitoramento da rede CAN. Destaca-se nestes dois primeiros experimentos, com 12% e 30% de carga da rede, que todos os distúrbios gerados pela ferramenta VH6501 foram registrados pelo mecanismo.

Figura 66 – Gráficos dos testes com carga de 12% - métricas average e difference jitter.



Nestes primeiros experimentos com carga de rede baixa é possível observar que o nú-

Figura 67 – Gráficos dos testes com carga de 30% - métricas average e difference jitter.



mero de eventos gerados de acordo com métrica *average jitter* é menor, pois seu objetivo é observar um distúrbio mais duradouro na rede. Esse monitoramento ocorre deste o inicio da análise e depende da quantidade de memória interna da ECU, fato que em caso de um monitoramento de longos períodos (dias, meses) levaria a criação de uma janela temporal de monitoramento. Por outro lado, os eventos gerados com a métrica *difference jitter* estão relacionados a distúrbios repentinos e de curta duração que afetam o desempenho do sistema de controle. Neste caso a geração de eventos é maior e reflete diretamente as oscilações que estão ocorrendo na rede de comunicação, como pode ser observado nas figuras 66 e 67. A seguir a Figura 68 exemplifica o registro das estatísticas geradas pelo software CANoe durante a análise com 30% de carga da rede.

Dando continuidade à análise, mais dois experimentos foram realizados com aumento da carga da rede para 50% e 80%. As figuras 69 e 70 apresentam estes testes.

O objetivo destes testes é a observação do comportamento do mecanismo de diagnóstico modelado em situações de alta ocupação da rede. Do mesmo modo, analisa-se os efeitos das injeções de falhas nas diferentes situações, e seus efeitos no desempenho do sistema de controle de suspensão ativa. É possível observar na Figura 69 a) e c) que no teste com 50% de carga de ocupação da rede CAN, o sistema de controle se torna instável e após 80 segundos a rede entra em estado "*BussOff*", não obtendo mais nenhum dado de tráfego da rede por meio do software CANoe. No mesmo tipo de análise, com as mesmas

Figura 68 – Estatísticas geradas pelo software CANoe com 30% de carga de rede.

CAN

CAN Statistics

CAN Channel: CAN 1 - CAN

Statistic	Current / Last	Min	Max	Avg
Busload [%]	30.33	30.28	30.45	30.34
Min. Send Dist. [ms]	0.000	n/a	n/a	n/a
Bursts [total]	38743	n/a	n/a	n/a
Burst Time [ms]	1.517	0.250	1.517	0.880
Frames per Burst	12	2	12	7
Std. Data [fr/s]	2396	2394	2408	2400
Std. Data [total]	289118	n/a	n/a	n/a
FTObserver	3	n/a	n/a	n/a
CANStress	168630	n/a	n/a	n/a
Susp_Control_ECU	24125	n/a	n/a	n/a
Susp_ECU	96360	n/a	n/a	n/a
Unknown sender	0	n/a	n/a	n/a
Ext. Data [fr/s]	0	0	0	0
Ext. Data [total]	0	n/a	n/a	n/a
Std. Remote [fr/s]	0	0	0	0
Std. Remote [total]	0	n/a	n/a	n/a
Ext. Remote [fr/s]	0	0	0	0
Ext. Remote [total]	0	n/a	n/a	n/a
Errorframes [fr/s]	3	0	13	1
Errorframes [total]	133	n/a	n/a	n/a
Chip State	Active	n/a	n/a	n/a
Transmit Error Count	0	n/a	180	n/a
Receive Error Count	0	n/a	128	n/a
Transceiver Errors	0	n/a	n/a	n/a
Transceiver Delay [ns]	0	0	0	0

CAN-FD

CAN Statistics

CAN Channel: CAN 1 - CAN

Statistic	Current / Last	Min	Max	Avg
Busload [%]	30.39	30.30	30.42	30.34
Min. Send Dist. [ms]	0.000	n/a	n/a	n/a
Bursts [total]	46884	n/a	n/a	n/a
Burst Time [ms]	1.392	0.250	1.392	0.714
Frames per Burst	11	2	11	6
Std. Data [fr/s]	2404	2397	2405	2400
Std. Data [total]	288435	n/a	n/a	n/a
FTObserver	1	n/a	n/a	n/a
CANStress	168238	n/a	n/a	n/a
Susp_Control_ECU	24056	n/a	n/a	n/a
Susp_ECU	96140	n/a	n/a	n/a
Unknown sender	0	n/a	n/a	n/a
Ext. Data [fr/s]	0	0	0	0
Ext. Data [total]	0	n/a	n/a	n/a
Std. Remote [fr/s]	0	0	0	0
Std. Remote [total]	0	n/a	n/a	n/a
Ext. Remote [fr/s]	0	0	0	0
Ext. Remote [total]	0	n/a	n/a	n/a
Errorframes [fr/s]	0	0	5	0
Errorframes [total]	43	n/a	n/a	n/a
Chip State	Active	n/a	n/a	n/a
Transmit Error Count	0	n/a	152	n/a
Receive Error Count	0	n/a	82	n/a
Transceiver Errors	0	n/a	n/a	n/a
Transceiver Delay [ns]	156	156	156	156

Fonte: Autor.

Figura 69 – Gráficos dos testes com carga de 50% - métricas average e difference jitter.

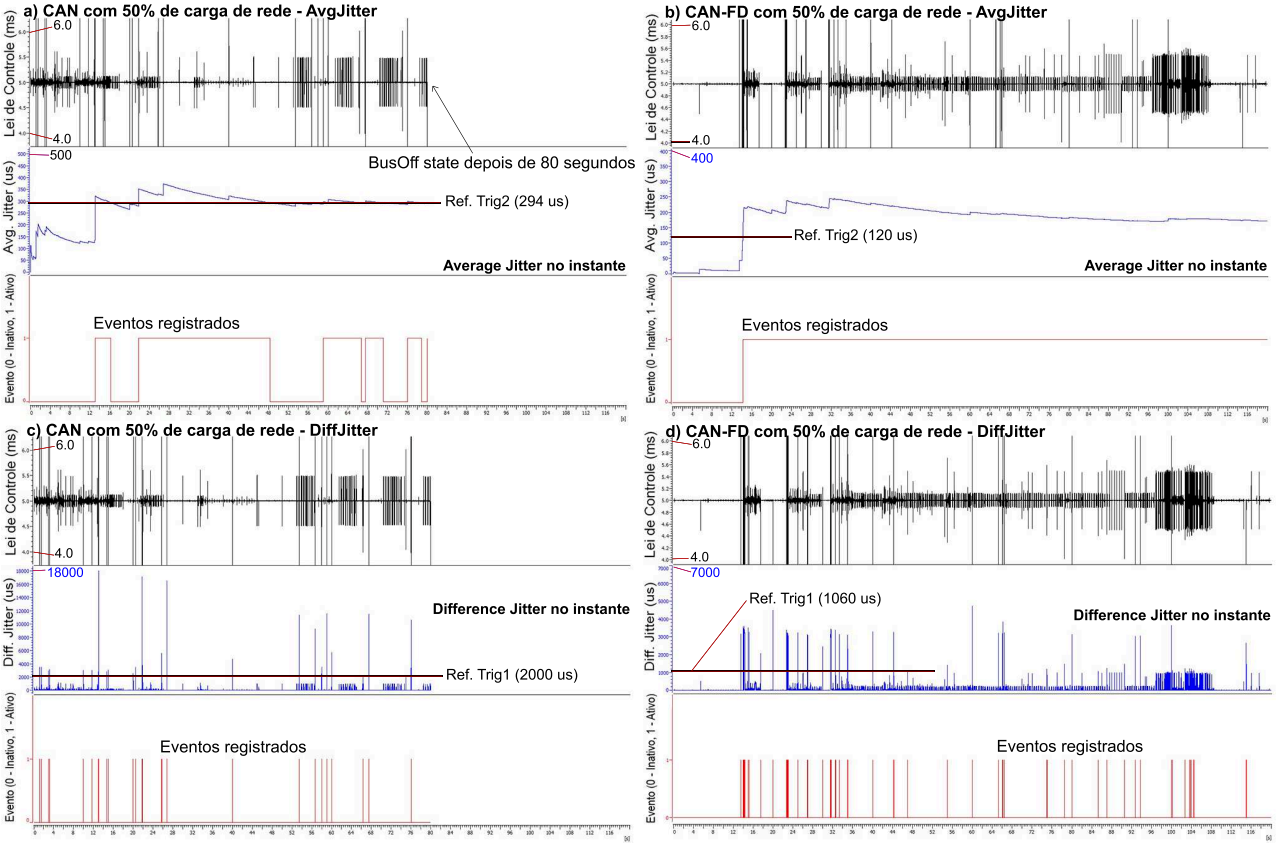
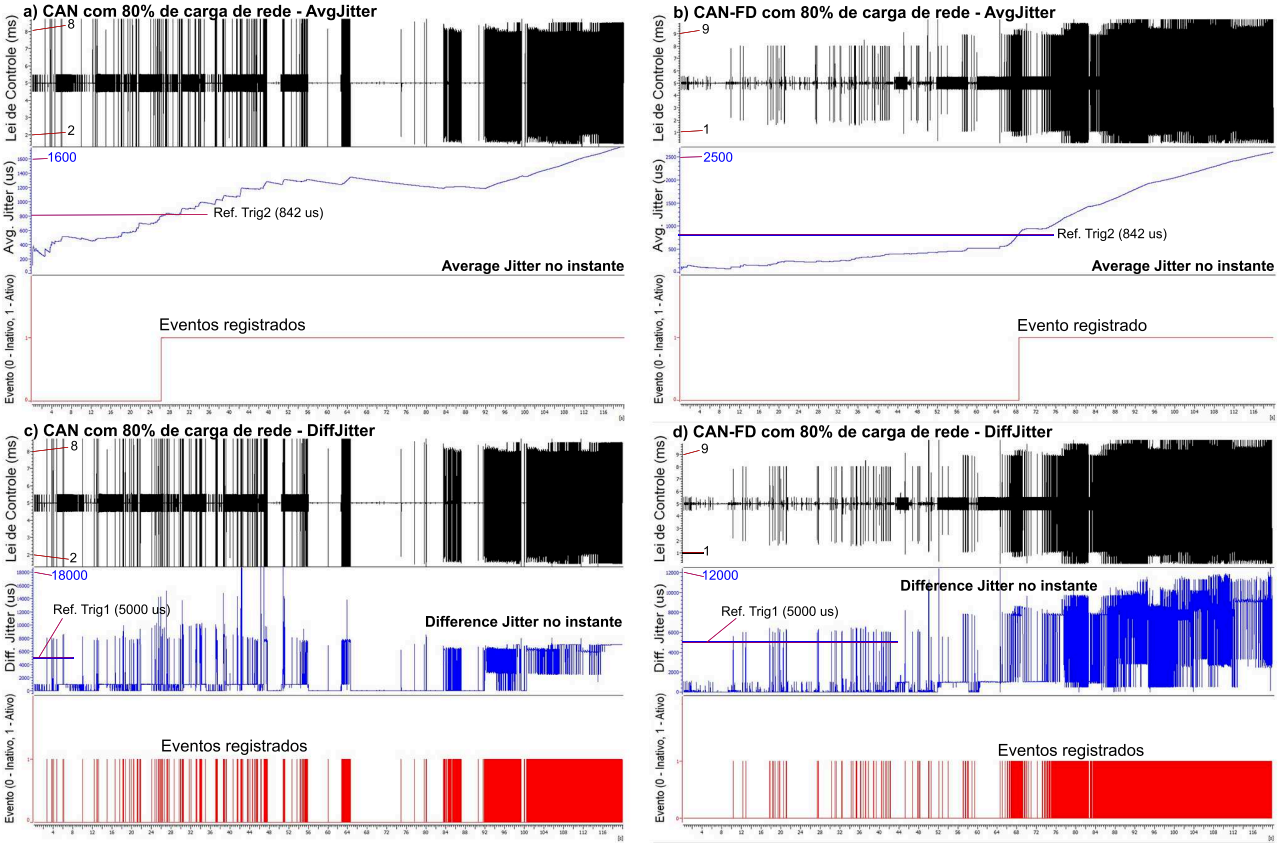


Figura 70 – Gráficos dos testes com carga de 80% - métricas average e difference jitter.



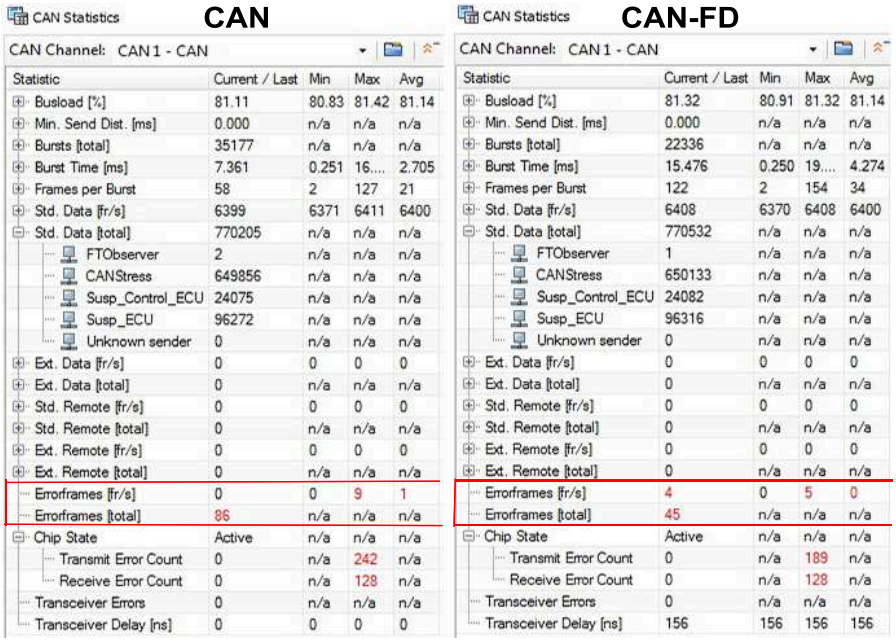
injeções de falhas, é possível observar na Figura 69 b) e d), que o sistema de controle operando no protocolo CAN-FD tolera as injeções de falhas. Do mesmo modo, nos testes apresentados na Figura 70 é possível observar o registro das oscilações da rede durante as injeções de falhas, destacando neste ponto, que nos experimentos com 80% de ocupação da rede, os parâmetros de repetições de falhas no protocolo CAN foram reduzidos pela metade. Essa redução permite a conclusão do experimento na rede CAN convencional, pois com 50% de carga de ocupação, a rede CAN já entrou em estado ‘‘*BussOff*” não concluindo os experimentos.

A Figura 71 destaca as estatísticas geradas pelo software CANoe durante a análise com 80% de carga da rede. É possível observar a maior taxa de erros (86 a 45) entre CAN e CAN-FD, tendo os erros de transmissão em 189 e 242 respectivamente, chegando perto do limite de 256, o que geraria novamente o estado “*BussOff*”.

Em ambos os experimentos com alta taxa de ocupação da rede, o mecanismo de diagnóstico também consegue registrar todos os eventos de anomalias no desempenho. Essas informações mostram, além da capacidade do mecanismo de diagnóstico, o melhor desempenho e tolerância a distúrbios do novo protocolo CAN-FD.

O hardware Vector VH6501 se mostrou eficiente na geração dos distúrbios, fato que destaca a sua adoção na indústria automotiva, na geração automática de disrupções que possibilitam a análise do comportamento de um sistema de controle perante problemas

Figura 71 – Estatísticas geradas pelo software CANoe com 80% de carga de rede.



CAN

CAN Statistics

CAN Channel: CAN 1 - CAN

Statistic	Current / Last	Min	Max	Avg
Busload [%]	81.11	80.83	81.42	81.14
Min. Send Dist. [ms]	0.000	n/a	n/a	n/a
Bursts [total]	35177	n/a	n/a	n/a
Burst Time [ms]	7.361	0.251	16....	2.705
Frames per Burst	58	2	127	21
Std. Data [fr/s]	6399	6371	6411	6400
Std. Data [total]	770205	n/a	n/a	n/a
FTObserver	2	n/a	n/a	n/a
CANStress	649856	n/a	n/a	n/a
Susp_Control_ECU	24075	n/a	n/a	n/a
Susp_ECU	96272	n/a	n/a	n/a
Unknown sender	0	n/a	n/a	n/a
Ext. Data [fr/s]	0	0	0	0
Ext. Data [total]	0	n/a	n/a	n/a
Std. Remote [fr/s]	0	0	0	0
Std. Remote [total]	0	n/a	n/a	n/a
Ext. Remote [fr/s]	0	0	0	0
Ext. Remote [total]	0	n/a	n/a	n/a
Errorframes [fr/s]	0	0	9	1
Errorframes [total]	86	n/a	n/a	n/a
Chip State	Active	n/a	n/a	n/a
Transmit Error Count	0	n/a	242	n/a
Receive Error Count	0	n/a	128	n/a
Transceiver Errors	0	n/a	n/a	n/a
Transceiver Delay [ns]	0	0	0	0

CAN-FD

CAN Statistics

CAN Channel: CAN 1 - CAN

Statistic	Current / Last	Min	Max	Avg
Busload [%]	81.32	80.91	81.32	81.14
Min. Send Dist. [ms]	0.000	n/a	n/a	n/a
Bursts [total]	22336	n/a	n/a	n/a
Burst Time [ms]	15.476	0.250	19....	4.274
Frames per Burst	122	2	154	34
Std. Data [fr/s]	6408	6370	6408	6400
Std. Data [total]	770532	n/a	n/a	n/a
FTObserver	1	n/a	n/a	n/a
CANStress	650133	n/a	n/a	n/a
Susp_Control_ECU	24082	n/a	n/a	n/a
Susp_ECU	96316	n/a	n/a	n/a
Unknown sender	0	n/a	n/a	n/a
Ext. Data [fr/s]	0	0	0	0
Ext. Data [total]	0	n/a	n/a	n/a
Std. Remote [fr/s]	0	0	0	0
Std. Remote [total]	0	n/a	n/a	n/a
Ext. Remote [fr/s]	0	0	0	0
Ext. Remote [total]	0	n/a	n/a	n/a
Errorframes [fr/s]	4	0	5	0
Errorframes [total]	45	n/a	n/a	n/a
Chip State	Active	n/a	n/a	n/a
Transmit Error Count	0	n/a	189	n/a
Receive Error Count	0	n/a	128	n/a
Transceiver Errors	0	n/a	n/a	n/a
Transceiver Delay [ns]	156	156	156	156

Fonte: Autor.

na rede intra-veicular, observando limites de operação e degradações que podem ser críticas para o correto funcionamento. Por outro lado, o hardware injeta distúrbios lógicos que afetam os quadros de mensagens específicas, pré-definidas durante a análise e programação do sistema de injeção dos distúrbios, não verificando falhas físicas reais em componentes da rede. O próximo cenário de estudo de caso preenche essa lacuna, avaliando o sistema de diagnóstico, em conjunto com o sistema de controle, com a injeção de falhas físicas em uma rede real.

5.3 Cenário de Teste 2 - Análise do mecanismo de diagnóstico com distúrbios gerados por um hardware de Injeção EFT

O segundo cenário de teste avalia a metodologia de teste proposta e o mecanismo de diagnóstico modelado para estudo de caso (Capítulo 4), utilizando um hardware de injeção de transientes elétrios rápidos (EFT), originado do primeiro estudo sobre susceptibilidade a EFT (ROQUE *et al.*, 2017), detalhado e apresentado na seção 4.1.2 (testes de susceptibilidade no protocolo CAN-FD). Maiores detalhes deste projeto de hardware podem ser encontrados na dissertação de mestrado de Daniel Pohren (POHREN, 2020).

O objetivo deste estudo é demonstrar que o mecanismo é capaz de registrar as anomalias de desempenho que ocorrem nos três principais protocolos de comunicação usados em redes intra-veiculares, tendo como base os parâmetros obtidos das análises de susceptibilidade aos transientes EFT e a modelagem do mecanismo seguindo a metodologia proposta nesta Tese.

Uma sequencia de experimentos com o sistema de controle de suspensão ativa foi

realizada tendo como base as análises prévias apresentadas nas tabelas 11, 12 e 13, nos protocolos CAN, CAN-FD e FlexRay. A injeção de transientes EFT ocorreu no observando a rede de comunicação operando somente com mensagens do sistema de controle (com carga de 2% e 12%) e com 30% de carga, com mensagens de tráfego geradas por dois nós (TG1 e TG2) programados dentro do software Vector CANoe. A sequencia de injeção de falhas foi a seguinte:

- EFT 47 vp com 687 us de rajada.
- EFT 57 vp com 1200 us de rajada.
- EFT 63 vp com 500 us de rajada.

Os scripts de teste que foram desenvolvidos na linguagem CAPL podem ser visualizados nos Apêndices C e D, bem como também no Anexo A. A seguir são detalhados os resultados destas sequencias de experimentos enfatizando a obtenção dos dados e a capacidade de detecção do sistema de diagnóstico.

5.3.1 Injeção dos transientes EFT no protocolo CAN

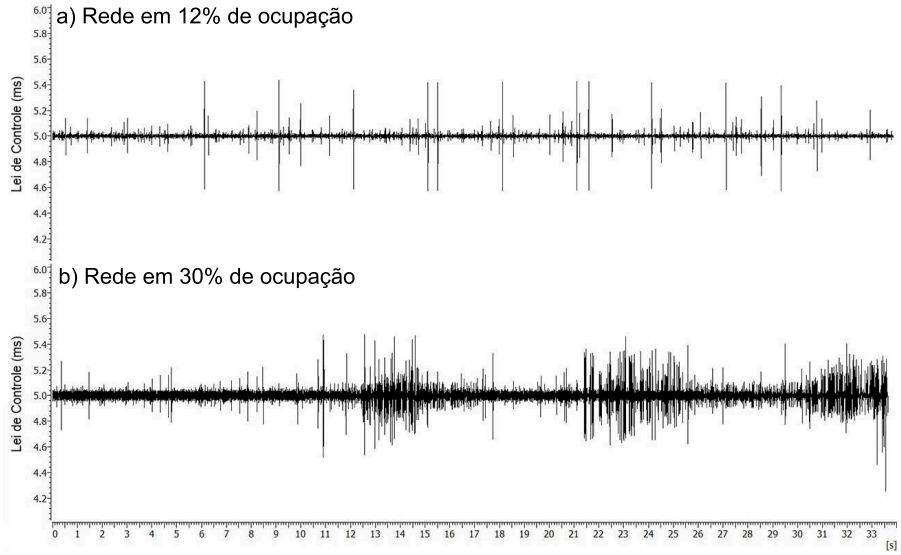
A referência deste estudo de caso para exemplificar um sistema de controle crítico, é um sistema de controle de suspensão ativa com a Lei de Controle enviada ciclicamente em 5ms. Os transientes EFT injetados geram distúrbios na modulação de sinais dos transceptores, resultando em um tratamento excessivo de erros e perda de pacotes. O mecanismo de diagnóstico incorporado ao sistema de controle, procura gerar informações a respeito dos efeitos de degradação, causados por diferentes tipos de falhas, neste caso, especificamente os transientes EFT, que são destacados em pesquisas da literatura recente (FONTANA; HUBING, 2015) (BAUER; DEUTSCHMANN; WINKLER, 2015) (ROQUE *et al.*, 2017) (STRICKER *et al.*, 2018) (MATSUSHIMA *et al.*, 2018).

Os testes realizados seguem um período de amostragem de aproximadamente 30 segundos, com injeções de distúrbios em intervalos de 1 ms (com EFT de rajada de 500 us e 687us) e 2 ms (com EFT de rajada de 1,2 ms). Por se tratar de um teste de susceptibilidade com detecção e diagnóstico das anomalias, o tempo de amostragem é considerado suficiente, pois nesse tempo há uma grande quantidade de ciclos da malha de controle. Outro fator que limita o tempo de amostragem é a característica e versão da ferramenta utilizada, que possui *buffers* de memória interna pequenos, que impossibilita o armazenamento da grande quantidade de eventos que esse tipo de falha gera. Nesse caso, versões mais recentes das ferramentas Vector CANoe e hardware VN8900, com *buffers* de memória maiores e possibilidade de adição de memória para armazenamento externo, que possibilitam o armazenamento e registro de logs de comunicação por longos períodos.

A primeira sequencia de injeções foi realizada no protocolo CAN padrão com taxa de comunicação de 1 Mbps. Uma medição inicial da rede de comunicação, com carga de

ocupação em 12% e 30%, foi realizada para fins de verificação do ciclo médio da Lei de Controle sem a injeção dos pulsos de EFT. A Figura 72 apresenta a medição inicial.

Figura 72 – Medição prévia da rede CAN 1Mbps antes da sequencia de injeções EFT.



Fonte: Autor.

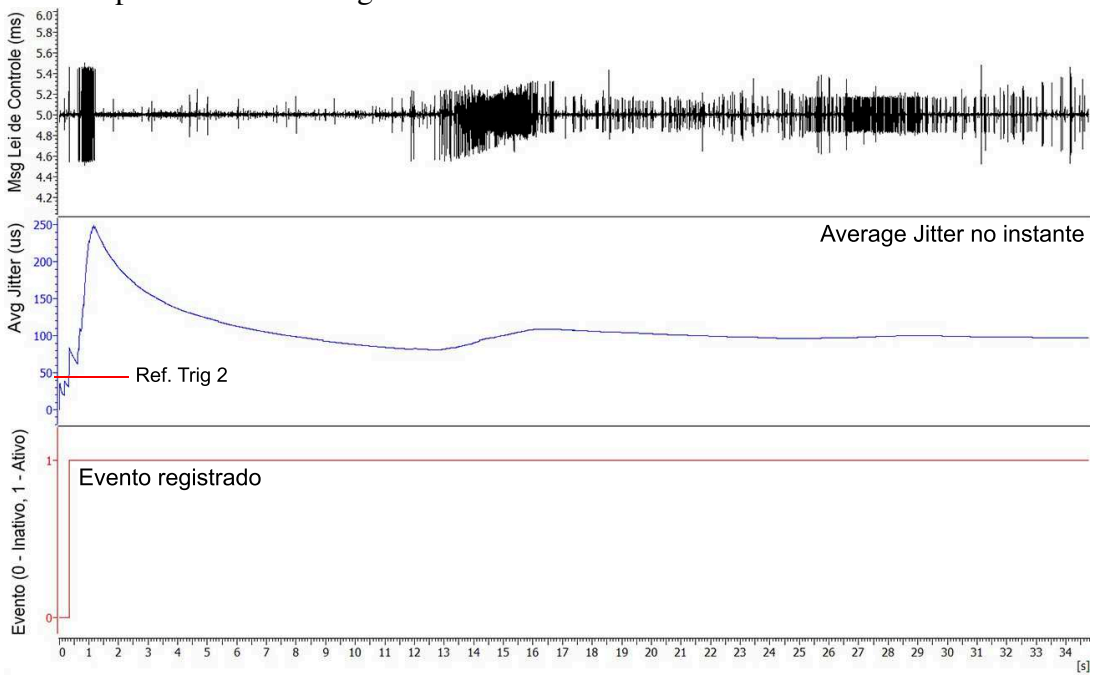
A medição inicial nos permite observar baixos níveis de atraso na rede de comunicação tendo tráfego somente de mensagens do sistema de controle (Figura 72 a.) e também com mensagens de tráfego (Figura 72 b.) que elevam a carga de ocupação para 30%, um nível considerado o limite em redes que possuam mensagens críticas. As figuras 73 e 74 apresentam os gráficos registrados com a primeira sequencia de injeção EFT (47 volts e 687us de rajada), com destaque a métrica *Average Jitter* em conjunto com o sistema de diagnóstico e detecção de eventos.

Por meio dos gráficos apresentados nas figuras 73 e 74 é possível observar a efetividade do sistema de diagnóstico no registro das anomalias, destacando os efeitos dos transientes EFT com 47 volts e 687 us, que geram distúrbios acentuados no ciclo de comunicação. A parte central das figuras ilustra o registro instantâneo da variação no jitter ao longo da amostragem, enquanto a parte inferior destaca o momento em que um evento anômalo foi registrado com base na métrica e nos gatilhos de referência adotados (42 e 90 us) em cada uma das variações de carga da rede. As figuras 75 e 76 destacam os resultados das injeções de EFT com 57 volts e 1200 us de rajada.

Este segundo experimento de injeção de falhas destaca que o intervalo maior de ocorrência dos distúrbios, gerou um efeito menor na variação média do ciclo de controle (em comparação com o primeiro experimento), sendo nítido que com 30% de carga de ocupação da rede os efeitos de degradação são mais acentuados.

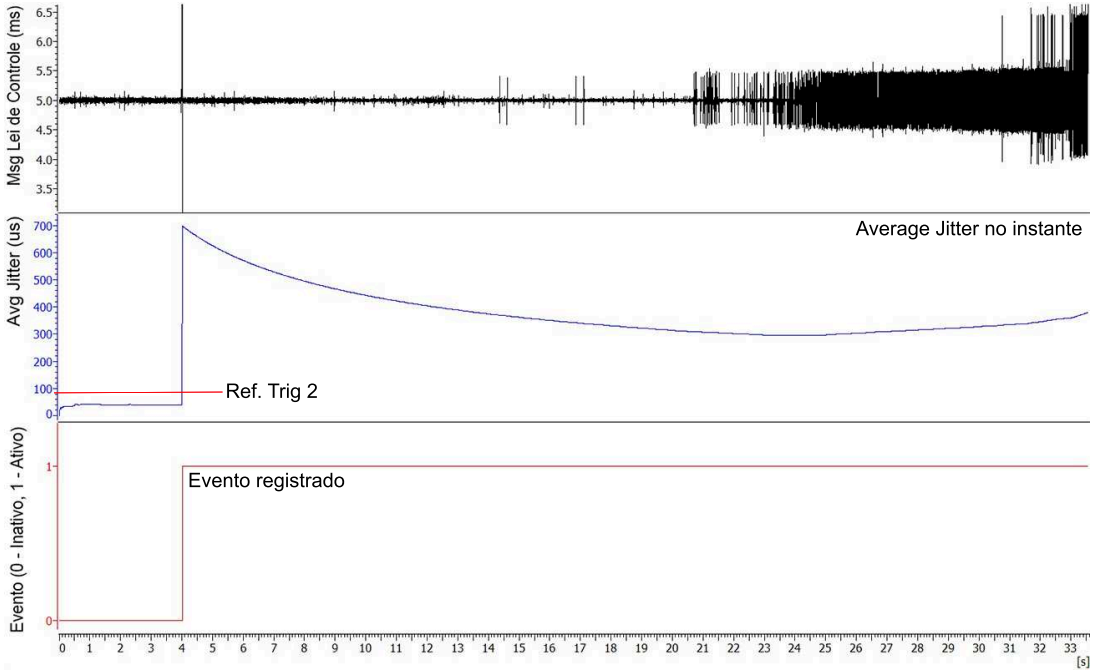
No primeiro teste, com 12% de carga de ocupação, (Figura 75) dois eventos anômalos foram detectados, registrando os momentos em que os gatilhos são acionados. No segundo teste, com 30% de carga de ocupação da rede, um evento anômalo é registrado

Figura 73 – Teste do sistema de diagnóstico com Injeção de EFT de 47vp e 687us na rede CAN 1Mbps com 12% de carga.



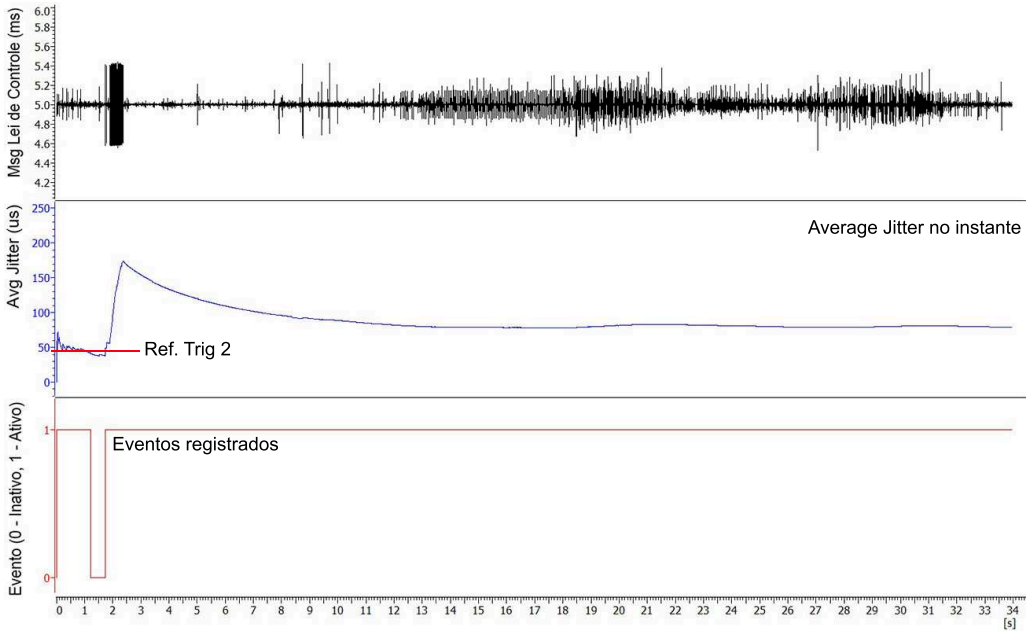
Fonte: Autor.

Figura 74 – Teste do sistema de diagnóstico com Injeção de EFT de 47vp e 687us na rede CAN 1Mbps com 30% de carga.



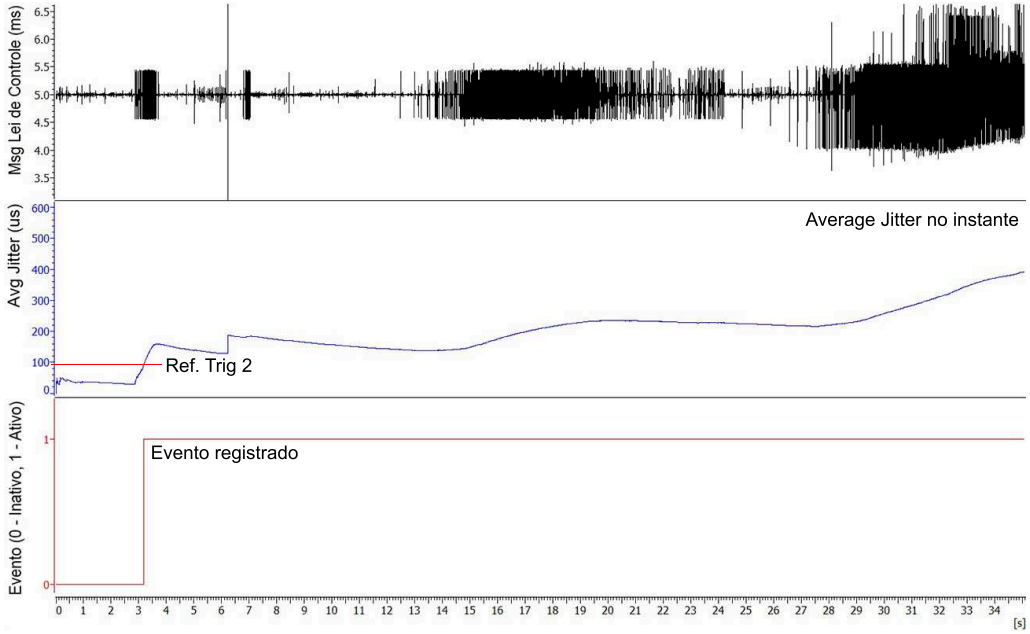
Fonte: Autor.

Figura 75 – Teste do sistema de diagnóstico com Injeção de EFT de 57vp e 1200us na rede CAN 1Mbps com 12% de carga.



Fonte: Autor.

Figura 76 – Teste do sistema de diagnóstico com Injeção de EFT de 57vp e 1200us na rede CAN 1Mbps com 30% de carga.

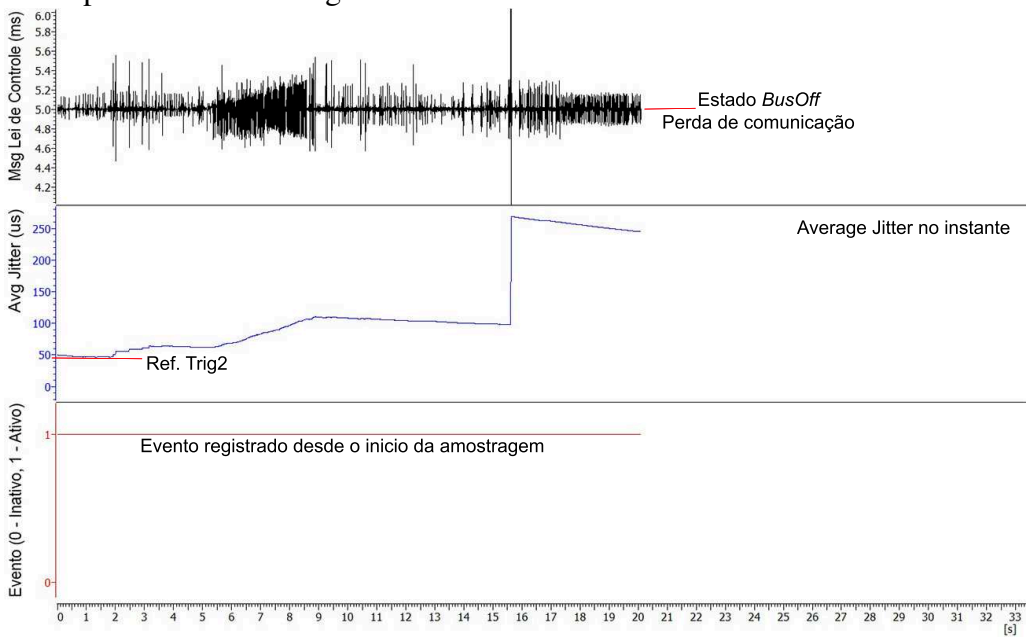


Fonte: Autor.

após o terceiro segundo da amostragem, elevando o ciclo médio para um patamar superior a referência de desempenho, e devido ao impacto dos transientes na rede os valores não retornaram aos parâmetros de referência.

À medida que a amplitude de tensão dos transientes aumenta, a degradação em mensagens cíclicas de controle também é acentuada, sendo observado que o intervalo de ocorrência destes distúrbios é um fator ainda mais importante. Por fim, uma ultima sequencia de injeções EFT foi realizada com amplitude de 63 volts e 500 us de rajada. As figuras 77 e 78 apresentam os gráficos destes experimentos.

Figura 77 – Teste do sistema de diagnóstico com Injeção de EFT de 63vp e 500us na rede CAN 1Mbps com 12% de carga.

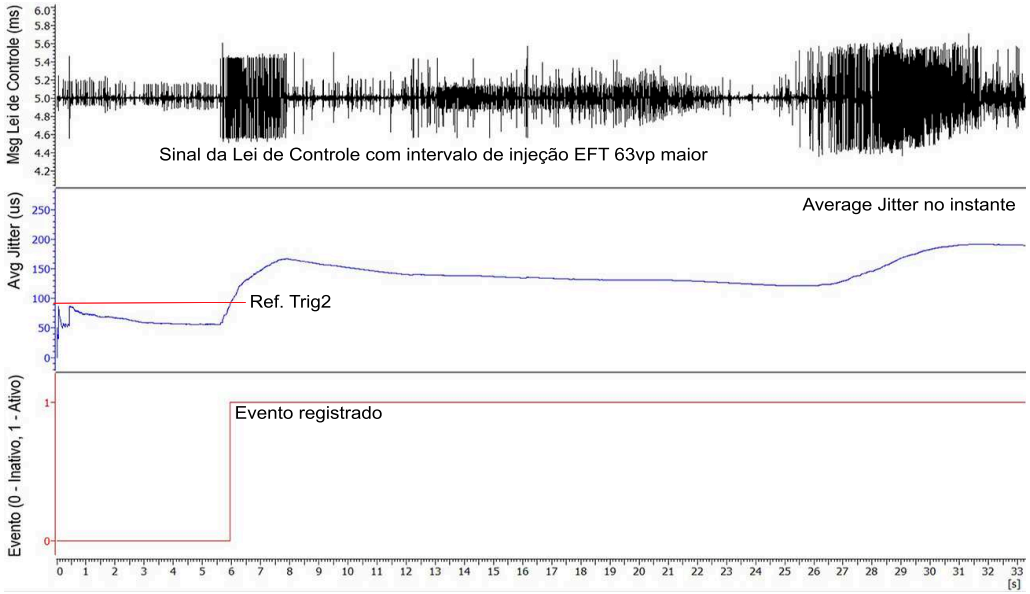


Fonte: Autor.

Essa observação permite avaliar a degradação causada com intervalos menores na ocorrência das disrupções de sinal do protocolo CAN. Nesta etapa, a Figura 77 destaca a ocorrência do estado de "*BusOff*" durante a injeção de EFT de 63 vp e 500 us, com 12% de carga de ocupação da rede, causando a interrupção de toda a comunicação na rede CAN. Essa situação ocorreu devido à alta incidência de transientes gerando disrupções de sinal, que consequentemente elevaram o contador de erros de transmissão a valores acima de 255, que é o limite tratado pelo protocolo.

A Figura 79 apresenta as estatísticas geradas pelo software CANoe durante a injeção de EFT com 63 volts, enfatizando o estado ‘‘*BusOff*" e a quantidade de quadros de erro durante os dois experimentos realizados.

Figura 78 – Teste do sistema de diagnóstico com Injeção de EFT de 63vp e 500us na rede CAN 1Mbps com 30% de carga.



Fonte: Autor.

Figura 79 – Estatísticas durante a Injeção de EFT com 63vp e 500us na rede CAN 1Mbps.

CAN Statistics — CAN Channel: CAN 1 - CAN

Statistic	Current...	Min	M...	Avg
Busload [%]	0.00	0....	1...	8....
Min. Send Dist. [...	0.000	n/a	n/a	n/a
Bursts [total]	4714	n/a	n/a	n/a
Burst Time [ms]	0.503	0....	0....	0....
Frames per Burst	0	2	4	4
Std. Data [fr/s]	0	0	1...	631
Std. Data [total]	23354	n/a	n/a	n/a
FTObserver	5	n/a	n/a	n/a
CANStress	0	n/a	n/a	n/a
Susp_Con...	4669	n/a	n/a	n/a
Susp_ECU	18680	n/a	n/a	n/a
Unknown s...	0	n/a	n/a	n/a
Ext. Data [fr/s]	0	0	0	0
Ext. Data [total]	0	n/a	n/a	n/a
Std. Remote [fr/s]	0	0	0	0
Std. Remote [tot...	0	n/a	n/a	n/a
Ext. Remote [fr/s]	0	0	0	0
Ext. Remote [total]	0	n/a	n/a	n/a
Errorframes [fr/s]	0	0	36	8
Errorframes [total]	304	n/a	n/a	n/a
Chip State	Off	n/a	n/a	n/a
Transmit Error...	256	n/a	256	n/a
Receive Error ...	0	n/a	9	n/a
Transceiver Errors	0	n/a	n/a	n/a
Transceiver Dela...	0	0	0	0

CAN Statistics — CAN Channel: CAN 1 - CAN

Statistic	Current / ...	Min	Max	Avg
Busload [%]	30.75	30.44	30....	30.71
Min. Send Dist. [ms]	0.000	n/a	n/a	n/a
Bursts [total]	12684	n/a	n/a	n/a
Burst Time [ms]	1.517	0.251	1.5...	0.736
Frames per Burst	12	2	12	6
Std. Data [fr/s]	2400	2396	2401	2400
Std. Data [total]	79871	n/a	n/a	n/a
FTObserver	1	n/a	n/a	n/a
CANStress	46586	n/a	n/a	n/a
Susp_Control_ECU	6656	n/a	n/a	n/a
Susp_ECU	26628	n/a	n/a	n/a
Unknown sender	0	n/a	n/a	n/a
Ext. Data [fr/s]	0	0	0	0
Ext. Data [total]	0	n/a	n/a	n/a
Std. Remote [fr/s]	0	0	0	0
Std. Remote [total]	0	n/a	n/a	n/a
Ext. Remote [fr/s]	0	0	0	0
Ext. Remote [total]	0	n/a	n/a	n/a
Errorframes [fr/s]	32	9	45	29
Errorframes [total]	957	n/a	n/a	n/a
Chip State	Active	n/a	n/a	n/a
Transmit Error Count	0	n/a	13	n/a
Receive Error Count	0	n/a	10	n/a
Transceiver Errors	0	n/a	n/a	n/a
Transceiver Delay [ns]	0	0	0	0

Fonte: Autor.

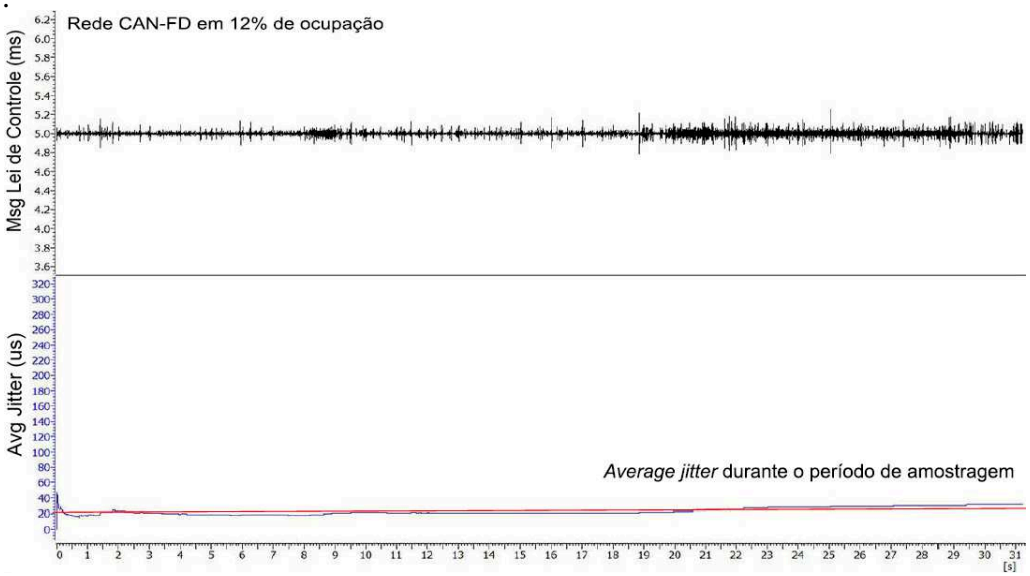
Com o intuito de concluir uma análise do sistema de diagnóstico na rede CAN, com a referida amplitude de EFT, o intervalo de injeção dos transientes foi elevado de 1 ms para 3 ms, permitindo a conclusão do experimento. Assim, a Figura 78 destaca a análise da rede de comunicação CAN com 30% de carga e a respectiva detecção dos eventos de degradação de desempenho. Observa-se também na primeira parte do gráfico o impacto acentuado nos atrasos do ciclo de comunicação, fato que evidencia o efeito de degradação com maior amplitude de tensão e menor frequência das injeções de EFT.

5.3.2 Injeção dos transientes EFT no protocolo CAN-FD

De acordo com os experimentos de susceptibilidade realizados (subseção 4.1.2 e seção 5.2) e com recentes resultados destacados em (POHREN *et al.*, 2019), o protocolo CAN-FD obteve melhores resultados durante as injeções de transientes EFT, tolerando uma quantidade maior de distúrbios. Devido a atualizações nas especificações do hardware/-camada física (clock de operação, bit time, transceptores), com novas especificações na camada de enlace do protocolo (ex: Bit EDL - reservado para especificação do formato extendido do frame e BRS - especifica a taxa de bit no campo de dados), o protocolo consegue trabalhar com taxas de transmissão de dados de forma flexível (variando de 1 a 8 Mbps), com campos de dados entre 1 e 64 bytes.

Essa características agregaram uma capacidade maior para o desenvolvimento de aplicações de diagnóstico e detecção de falhas, permitindo o uso de parte da largura de banda para mensagens e registro de eventos de status de diferentes componentes do veículo. Desta forma, a verificação do mecanismo de diagnóstico com a sequencia dos experimentos de injeção de falhas EFT, procura evidenciar e confirmar as informações obtidas em estudos anteriores com o CAN-FD, verificando ainda se o protocolo pode atender as demandas de aplicações futuras. A Figura 80 apresenta uma análise da rede CAN-FD antes das injeções EFT, para fins de verificação do atraso médio da Lei de Controle do sistema de suspensão ativa que é objeto deste estudo de caso.

Figura 80 – Medição prévia da rede CAN-FD 1 a 4 Mbps antes da sequencia de injeções EFT.



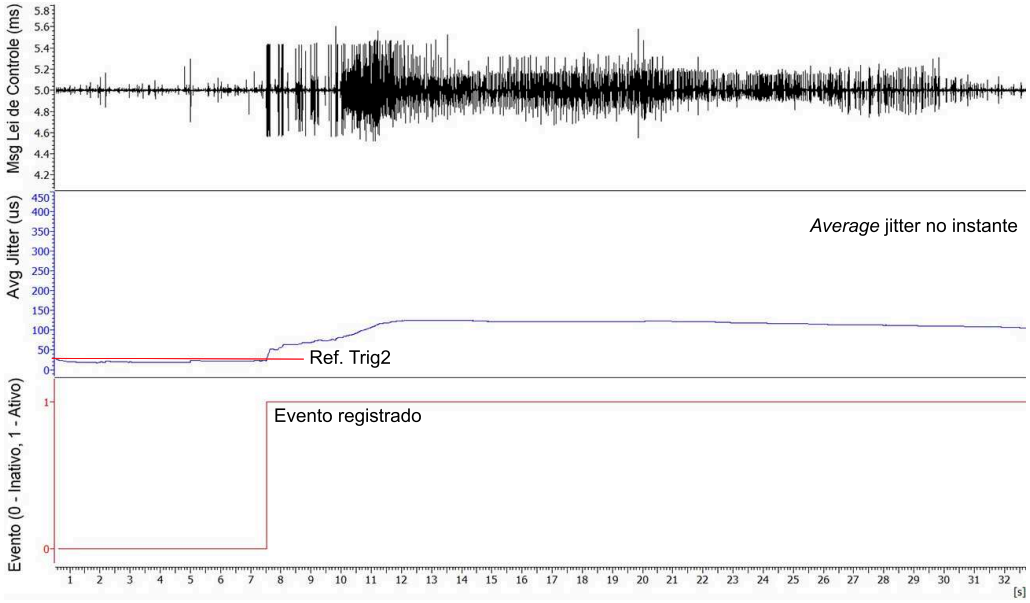
Fonte: Autor.

Observa-se que a variação média de atraso segue os padrões registrados anteriormente, com aproximadamente 20 us com a métrica *average jitter*, e picos de até 180 us com a métrica *difference jitter*, com barramento em 12% de ocupação (somente sistema de controle comunicando), valores inferiores aos registrados com o protocolo CAN padrão.

Com o barramento em 30% os valores registrados com as respectivas métricas ficaram em torno de 290 e 48 microssegundos (*difference jitter* e *average jitter*). Esses valores sequem os padrões de referência destacados na Tabela 12, definindo assim o gatilho 1 (*Diference Jitter* Trig1) em 190 e 300 us, e o gatilho 2 (*Average Jitter* Trig2) para 25 e 62 us, respectivamente nos testes com 12% e 30% de carga de ocupação da rede CAN-FD.

As figuras 81 e 82 apresentam os resultados da primeira sequencia de injeções EFT, com 47 volts e 687 us de rajada.

Figura 81 – Teste do sistema de diagnóstico com Injeção EFT de 47vp e 687us na rede CAN-FD 1 a 4 Mbps com 12% de carga.



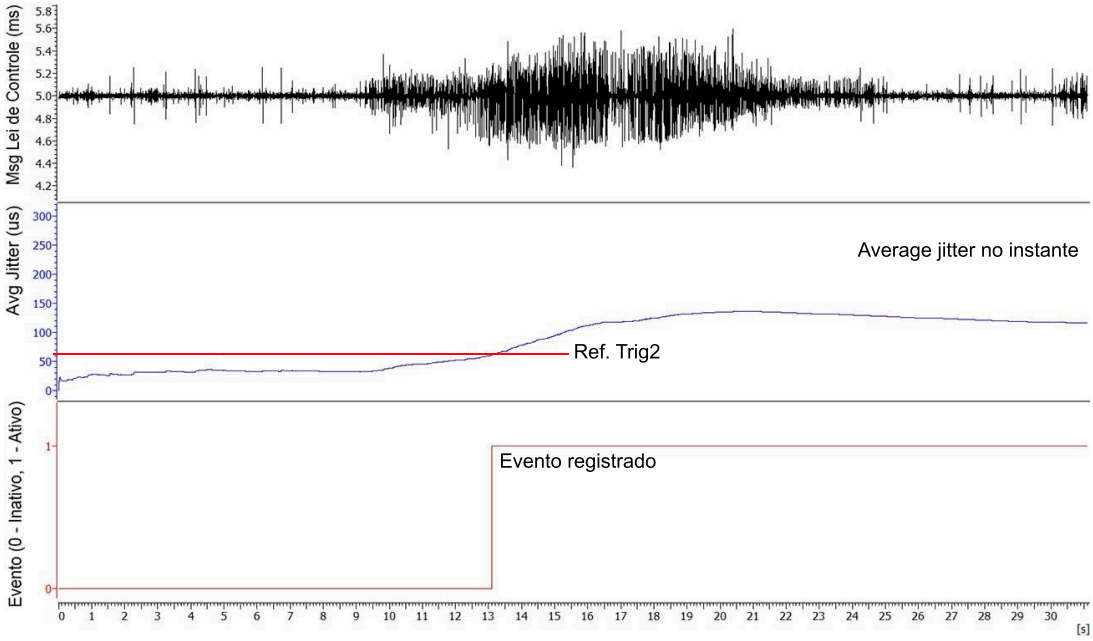
Fonte: Autor.

É possível observar nos gráficos que em ambas as situações de carga da rede são gerados eventos relativos ao gatilho 2 (Trig2), que ilustra a forma de detecção das anomalias de desempenho da rede CAN-FD. Em ambos os gráficos são destacados os pontos em que ocorre o registro dos eventos (Ref. Trig2), com atualização constante da métrica durante o período de análise. Os dados analisados destacam o aumento do ciclo médio de comunicação, evidenciado pelo fato de a média não retornar ao patamar de referência. Neste primeiro experimento com o CAN-FD, o intervalo de injeção de transientes EFT é de 1 ms, gerando a análise de estresse necessária para medir os efeitos de degradação na rede de comunicação.

Dando sequencia ao experimentos, nas figuras 83 e 84 são apresentados os gráficos gerados durante os as injeções de EFT com 57 volts e 1,2 milissegundos de rajada.

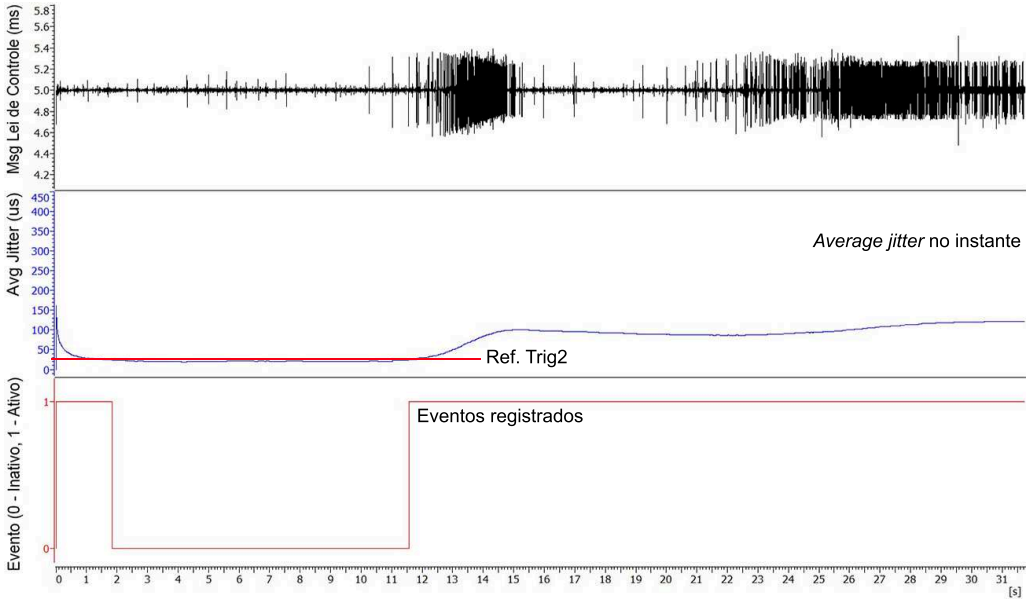
Como já observado nos testes de susceptibilidade aos transientes com o protocolo CAN, as injeções de EFT com 1,2 ms de rajada geraram efeitos similares, elevando o atraso na Lei de Controle para valores acima de 100 microssegundos. Tais valores são menores em relação aos efeitos gerados no protocolo CAN padrão, mas mostram que a

Figura 82 – Teste do sistema de diagnóstico com Injeção EFT de 47vp e 687us na rede CAN-FD 1 a 4 Mbps com 30% de carga.



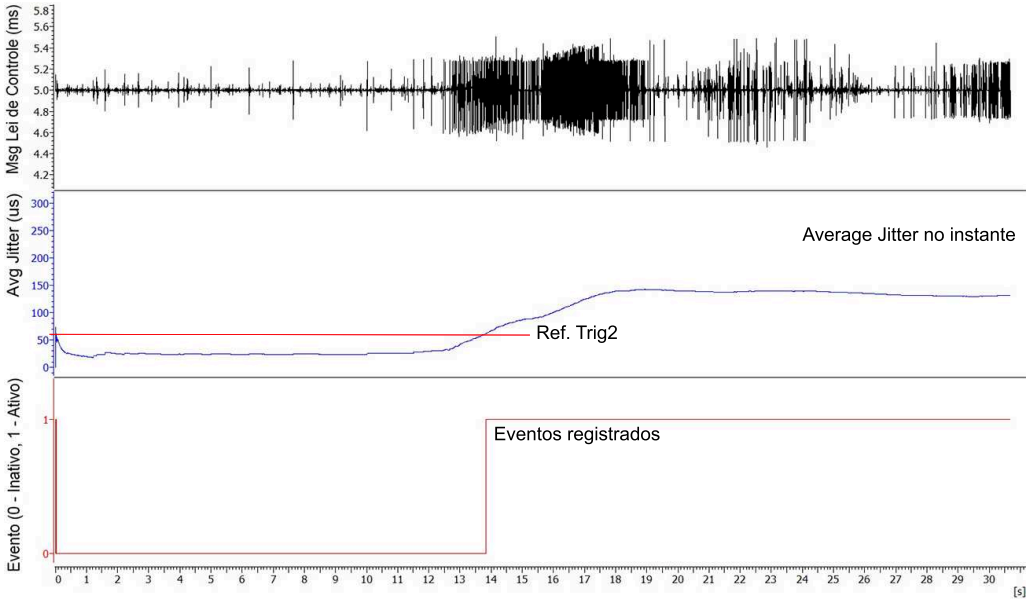
Fonte: Autor.

Figura 83 – Teste do sistema de diagnóstico com Injeção EFT de 57vp e 1200us na rede CAN-FD 1 a 4 Mbps com 12% de carga.



Fonte: Autor.

Figura 84 – Teste do sistema de diagnóstico com Injeção EFT de 57vp e 1200us na rede CAN-FD 1 a 4 Mbps com 30% de carga.



Fonte: Autor.

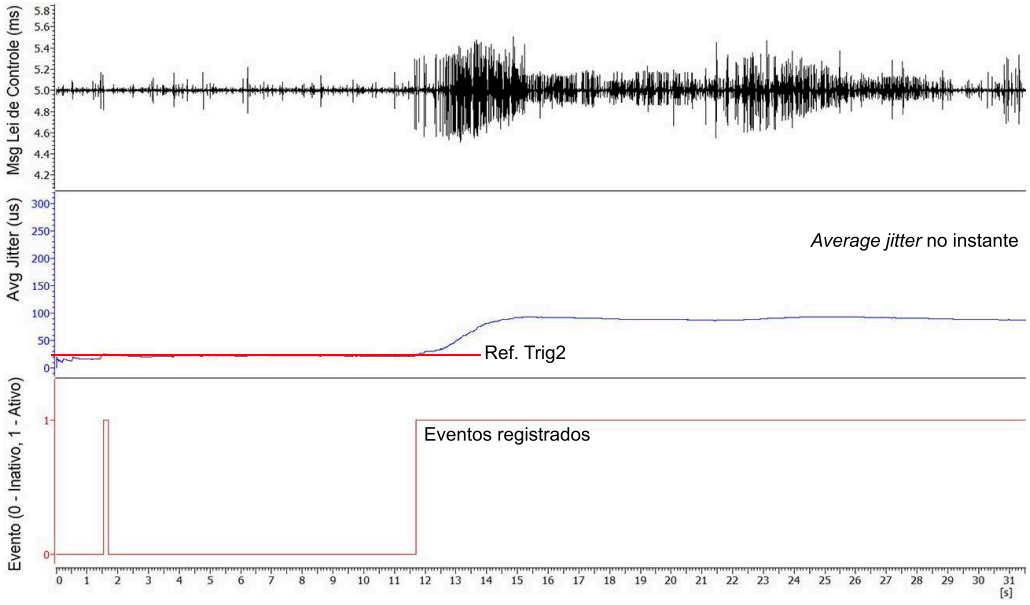
incidência de distúrbios pode também degradar a rede de comunicação CAN-FD e que o mecanismo é capaz de detectar tais variações. Durante os testes as variações se tornaram permanentes após 10 segundos de injeção de falhas, mantendo os valores de atraso acima dos parâmetros de referência.

Por fim, as figuras 85 e 86 destacam os gráficos obtidos pelo software CANoe durante a terceira sequencia de injeção de falhas, com EFT de 63 volts e 500 microssegundos.

Os experimentos realizados com maior amplitude de tensão destacam uma degradação de desempenho ainda mais acentuada, observados pela maior incidência de picos de atraso com a análise da rede em 30% de carga de ocupação (Figura 86). O algoritmo registra a oscilação de acordo com as métricas, onde o *average jitter* do período se mantém acima dos parâmetros e em crescimento constante. Devido ao período de amostragem e também ao fato de o protocolo CAN-FD tolerar uma quantidade de tratamento de erros maior, o estado "*BusOff*" não foi registrado.

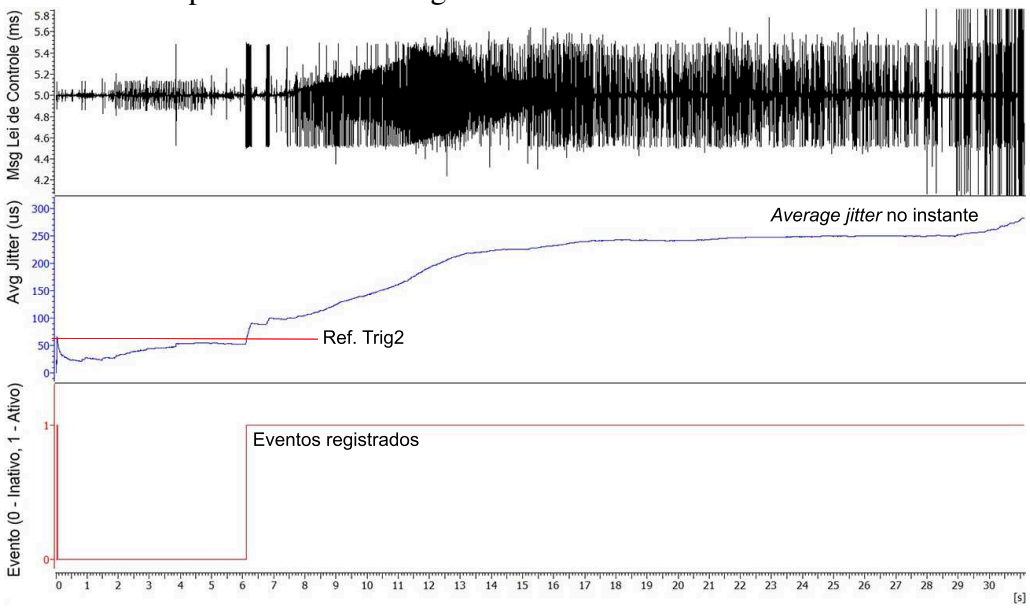
Adicionalmente, a Figura 87 destaca as incidência de quadros de erros durante esta última sequencia de injeção de EFT. É possível observar em destaque os registros da ECU "FTObserver", responsável por fazer o diagnóstico em tempo de execução das oscilações de desempenho no sistema de controle. A ferramenta de análise registrou um aumento na taxa de quadros de erros, entre 20 e 24 quadros por segundo em média.

Figura 85 – Teste do sistema de diagnóstico com Injeção EFT de 63vp e 500us na rede CAN-FD 1 a 4 Mbps com 12% de carga.



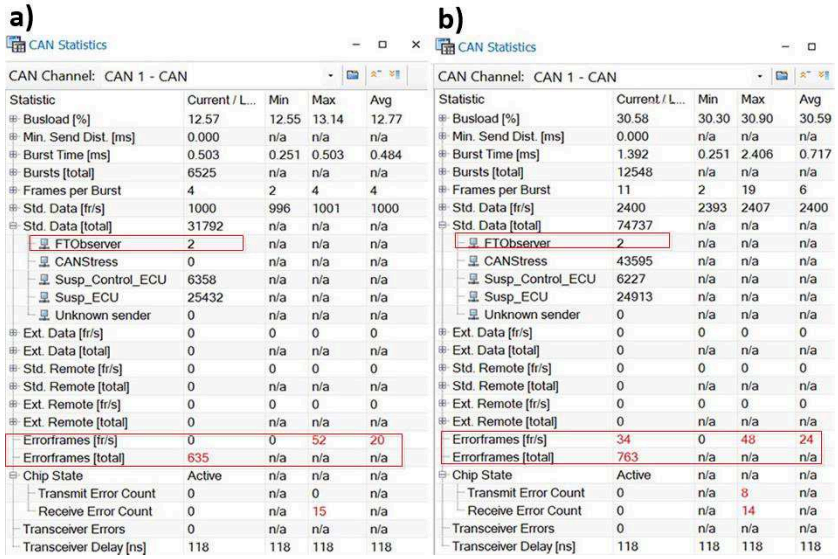
Fonte: Autor.

Figura 86 – Teste do sistema de diagnóstico com Injeção EFT de 63vp e 500us na rede CAN-FD 1 a 4 Mbps com 30% de carga.



Fonte: Autor.

Figura 87 – Estatísticas durante a Injeção de EFT de 63vp e 500us na rede CAN-FD 1 a 4 Mbps. a) Bus em 12% e b) Bus em 30%.



Statistic	Current / L...	Min	Max	Avg
Busload [%]	12.57	12.55	13.14	12.77
Min. Send Dist. [ms]	0.000	n/a	n/a	n/a
Burst Time [ms]	0.503	0.251	0.503	0.484
Bursts [total]	6525	n/a	n/a	n/a
Frames per Burst	4	2	4	4
Std. Data [fr/s]	1000	996	1001	1000
Std. Data [total]	31792	n/a	n/a	n/a
FTObserver	2	n/a	n/a	n/a
CANStress	0	n/a	n/a	n/a
Susp_Control_ECU	6358	n/a	n/a	n/a
Susp_ECU	25432	n/a	n/a	n/a
Unknown sender	0	n/a	n/a	n/a
Ext. Data [fr/s]	0	0	0	0
Ext. Data [total]	0	n/a	n/a	n/a
Std. Remote [fr/s]	0	0	0	0
Std. Remote [total]	0	n/a	n/a	n/a
Ext. Remote [fr/s]	0	0	0	0
Ext. Remote [total]	0	n/a	n/a	n/a
Errorframes [fr/s]	0	0	52	20
Errorframes [total]	635	n/a	n/a	n/a
Chip State	Active	n/a	n/a	n/a
Transmit Error Count	0	n/a	0	n/a
Receive Error Count	0	n/a	15	n/a
Transceiver Errors	0	n/a	n/a	n/a
Transceiver Delay [ns]	118	118	118	118

Statistic	Current / L...	Min	Max	Avg
Busload [%]	30.58	30.30	30.90	30.59
Min. Send Dist. [ms]	0.000	n/a	n/a	n/a
Burst Time [ms]	1.392	0.251	2.406	0.717
Bursts [total]	12548	n/a	n/a	n/a
Frames per Burst	11	2	19	6
Std. Data [fr/s]	2400	2393	2407	2400
Std. Data [total]	74737	n/a	n/a	n/a
FTObserver	2	n/a	n/a	n/a
CANStress	43595	n/a	n/a	n/a
Susp_Control_ECU	6227	n/a	n/a	n/a
Susp_ECU	24913	n/a	n/a	n/a
Unknown sender	0	n/a	n/a	n/a
Ext. Data [fr/s]	0	0	0	0
Ext. Data [total]	0	n/a	n/a	n/a
Std. Remote [fr/s]	0	0	0	0
Std. Remote [total]	0	n/a	n/a	n/a
Ext. Remote [fr/s]	0	0	0	0
Ext. Remote [total]	0	n/a	n/a	n/a
Errorframes [fr/s]	34	0	48	24
Errorframes [total]	763	n/a	n/a	n/a
Chip State	Active	n/a	n/a	n/a
Transmit Error Count	0	n/a	8	n/a
Receive Error Count	0	n/a	14	n/a
Transceiver Errors	0	n/a	n/a	n/a
Transceiver Delay [ns]	118	118	118	118

Fonte: Autor.

As informações apresentadas nas estatísticas da rede corroboram que o método de injeção de EFT funcionou adequadamente e que falhas na modulação de sinais resultou em grande quantidade de tratamento de erros. Esta terceira análise da rede CAN-FD confirma o impacto negativo de transientes de curta duração de rajada e com mais frequência de ocorrência. Assim, enfatiza-se que os testes realizados demonstram situações de falhas provocadas, em uma rede veicular real, as quais são importantes para testes de conformidade e definição dos limites de operação de vários sistemas de controle críticos.

5.3.3 Injeção dos transientes EFT no protocolo FlexRay

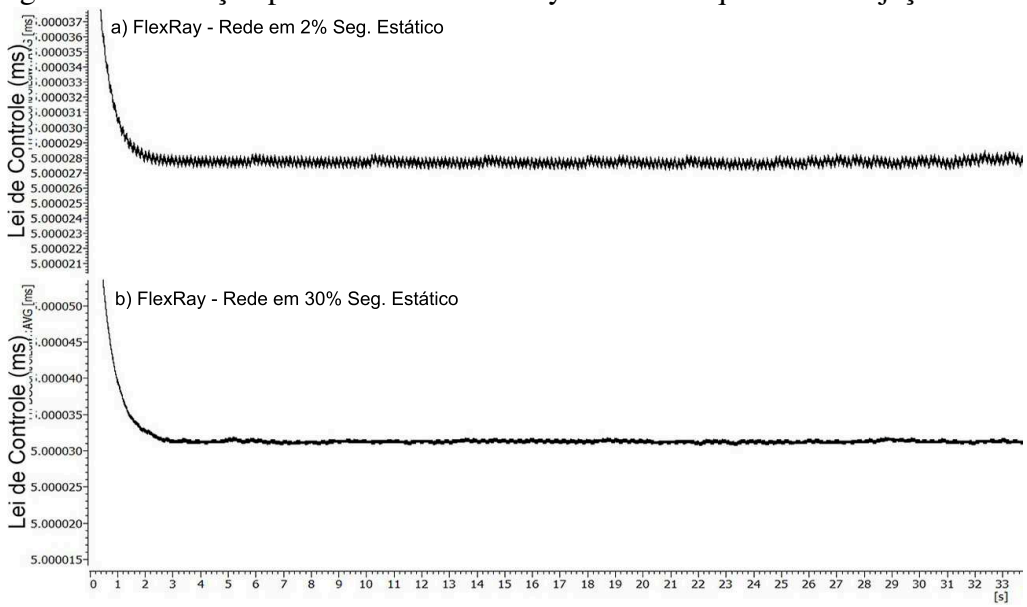
A terceira sequencia de injeções foi realizada no protocolo FlexRay com taxa de comunicação de 10 Mbps no Canal A. A sequencia de injeções de transientes EFT apresentada nesta seção, segue o mesmo método e sequencia apresentados nos testes de susceptibilidade da subseção 4.1.3. A diferença neste caso é a utilização de um nó extra na rede FlexRay (“FTObserver”), onde está implementado o mecanismo de diagnóstico que realiza o processamento das métricas e registro dos eventos de acordo com as referências definidas na Tabela 13.

Estes testes tem o intuito de verificar as possibilidades de detecção de anomalias de desempenho geradas por falhas ou interferências na rede de comunicação FlexRay, tendo como foco as mensagens cíclicas do sistema de controle de suspensão ativa. A implementação do sistema de controle segue a modelagem do controlador 3 apresentado em (MICHELIN, 2014), apresentado no Apêndice A. O protocolo FlexRay tem na sua origem de criação, a robustez e confiabilidade como premissas, fato comprovado pelos baixos níveis de interferências verificados nos testes de susceptibilidade. Mesmo durante os

testes de injeção de falhas o protocolo apresentou no pior caso (EFT de 63 vp e 500us, com 80% seg. estático e 100% seg. dinâmico) uma variação média máxima de apenas 94 microssegundos.

Desta forma, as mesmas sequencias de injeções de transientes EFT realizadas no protocolo CAN e CAN-FD também foram realizadas com o protocolo FlexRay (EFT de 47, 57 e 63 volts). Como parâmetro inicial a Figura 88 apresenta o gráfico da rede FlexRay antes das injeções EFT, verificando novamente os baixos atrasos médios (*average jitter*) nas mensagens da lei de controle do sistema de suspensão ativa.

Figura 88 – Medição prévia da rede FlexRay antes da sequencia de injeções EFT.

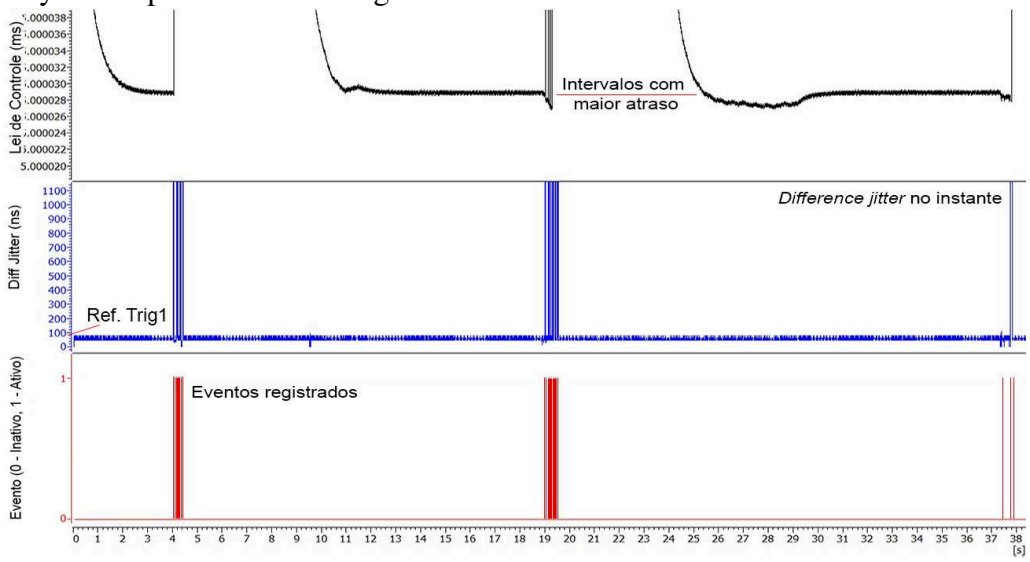


Fonte: Autor.

A medição inicial foi realizada com 2% e 30% de carga de ocupação do segmento estático, Figura 88 a) e b), confirmando os valores para a definição dos gatilhos do sistema de diagnóstico. Apesar do tempo da Lei de Controle ser de 5 ms, é possível observar na Figura 88 a) que a variação é de poucos nanosegundos (linha central em 5.000028). Assim como nos experimentos anteriores, ambas as métricas *average* e *difference jitter* são usadas para os registro de eventos, mas como a variação média é menos significativa e os picos de atrasos são mais acentuados e perceptíveis, nos gráficos optou-se por mostrar as detecções realizadas com a métrica *difference jitter*. A compilação de dados completa com ambas as métricas é apresentada no Capítulo 6, durante a análise dos resultados. As primeiras sequencias de injeção de transientes EFT com 47 volts e 687 us de rajada, na rede FlexRay com 2% e 30% de carga, são apresentados nas figuras 89 e 90.

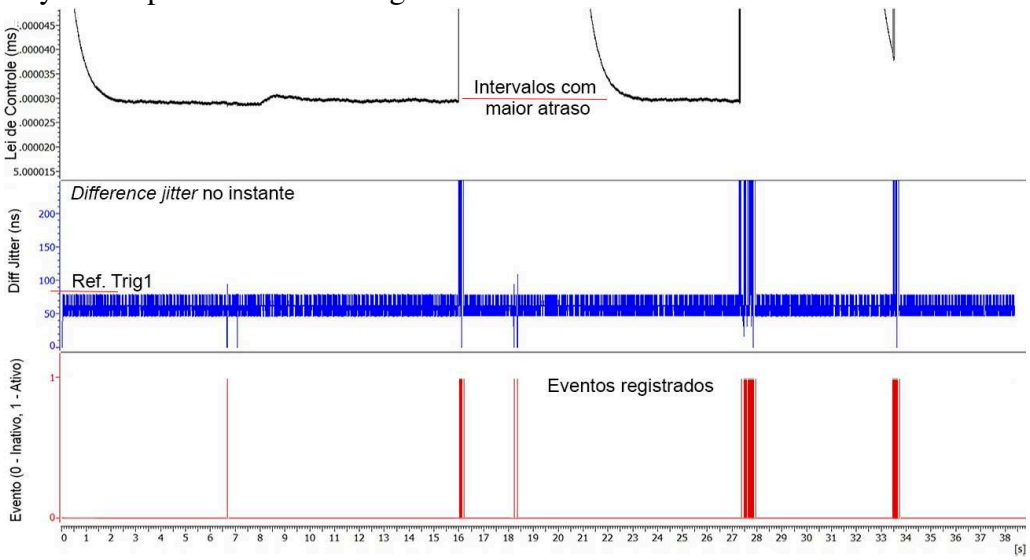
Os valores de tempo dos picos de atrasos durante a apresentação do *difference jitter* medido (dados da linha do meio nas figuras 89 e 90) estão em nanossegundos, lembrando que a referência limite para o gatilho é de 80 ns. Devido ao fato de os atrasos no protocolo FlexRay serem menores, para melhor destacar os pontos onde há impacto dos transientes

Figura 89 – Teste do sistema de diagnóstico com Injeção EFT de 47vp e 687us na rede FlexRay 10 Mbps com 2% de carga.



Fonte: Autor.

Figura 90 – Teste do sistema de diagnóstico com Injeção EFT de 47vp e 687us na rede FlexRay 10 Mbps com 30% de carga.

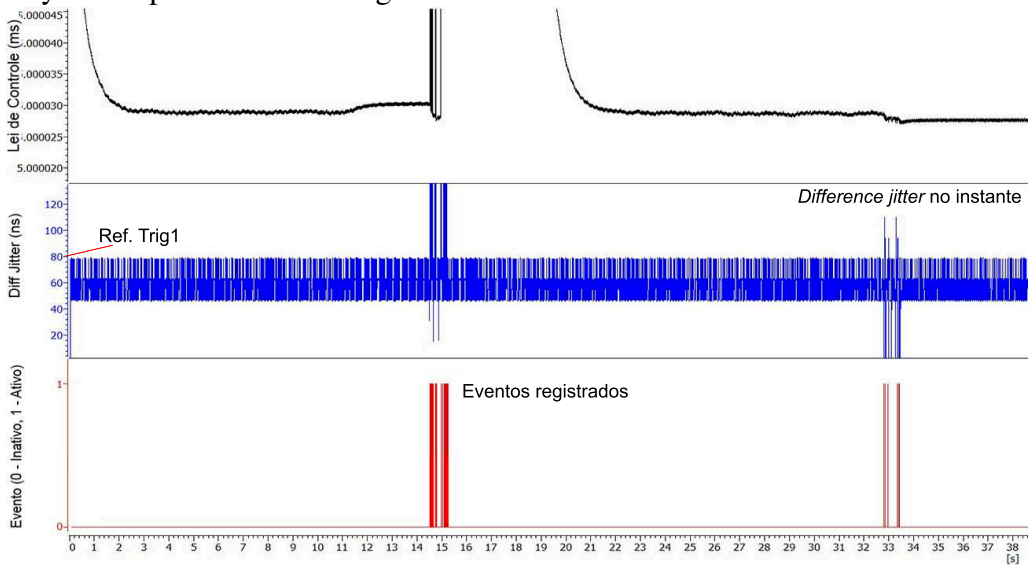


Fonte: Autor.

EFT, o gráfico do ciclo médio da lei de controle esta mais aproximado (*zoom in*), sendo notável os pontos onde ocorrerem os elevados distúrbios. Na parte inferior das figuras, o gráfico mostra no formato "ativo/inativo" quando um evento é gerado e registrado no arquivo de log. Devido ao nível acentuado dos distúrbios em relação ao ciclo médio normal, os tempos de comunicação demoram alguns segundos para retornar aos parâmetros usuais.

É importante destacar que nesta parte do estudo o foco é a aplicação do sistema de diagnóstico modelado de acordo os hardwares de injeção de falhas, não sendo necessário todos os testes com diferentes taxas de ocupação da rede FlexRay, os quais foram trabalhados na análise de susceptibilidade a falhas. O detalhamento qualitativo e quantitativo dos dados obtidos nestes experimentos serão apresentados na seção 6.2, quando da compilação e discussão de todos os resultados obtidos nos testes com os três protocolos de comunicação. A segunda sequencia de injeções EFT na rede FlexRay, com 57 volts e 1,2 ms de rajada é apresentada nas figuras 91 e 92.

Figura 91 – Teste do sistema de diagnóstico com Injeção EFT de 57vp e 1,2 ms na rede FlexRay 10 Mbps com 2% de carga.

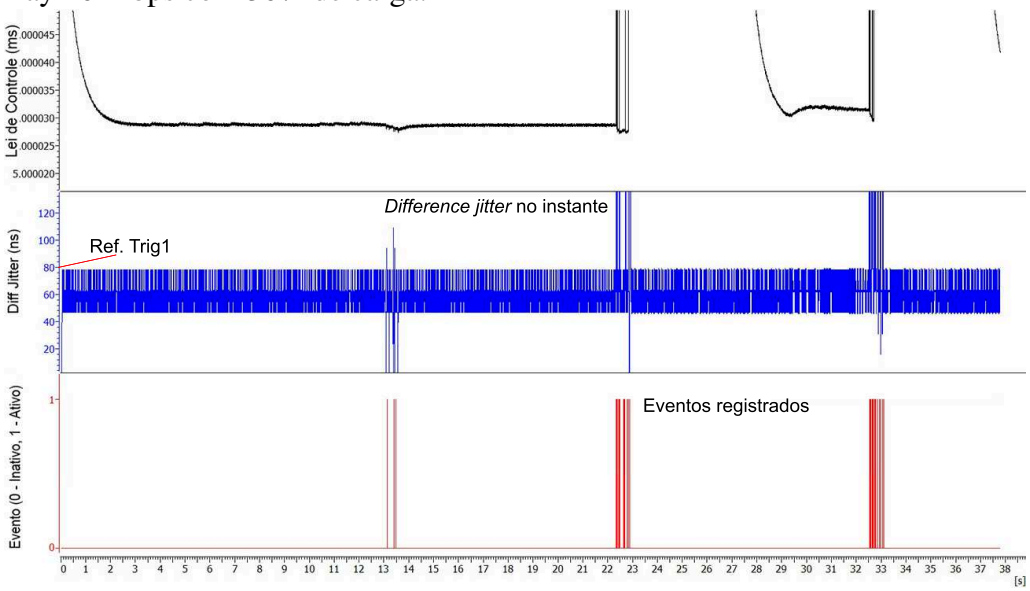


Fonte: Autor.

No gráfico da Figura 91, com carga de ocupação de 2% (somente o sistema de controle comunicando) é possível observar um distúrbio acentuado aos 15 segundos, o qual eleva o ciclo médio para a escala de microssegundos (em torno de 10 us), o que faz com que essa média demore aproximadamente 5 segundos para voltar ao patamar normal da rede. Por outro lado, na Figura 92, com carga de ocupação de 30%, é possível observar que essa degradação acentuada ocorre após 23 segundos de monitoramento, devido ao maior tráfego de mensagens no barramento.

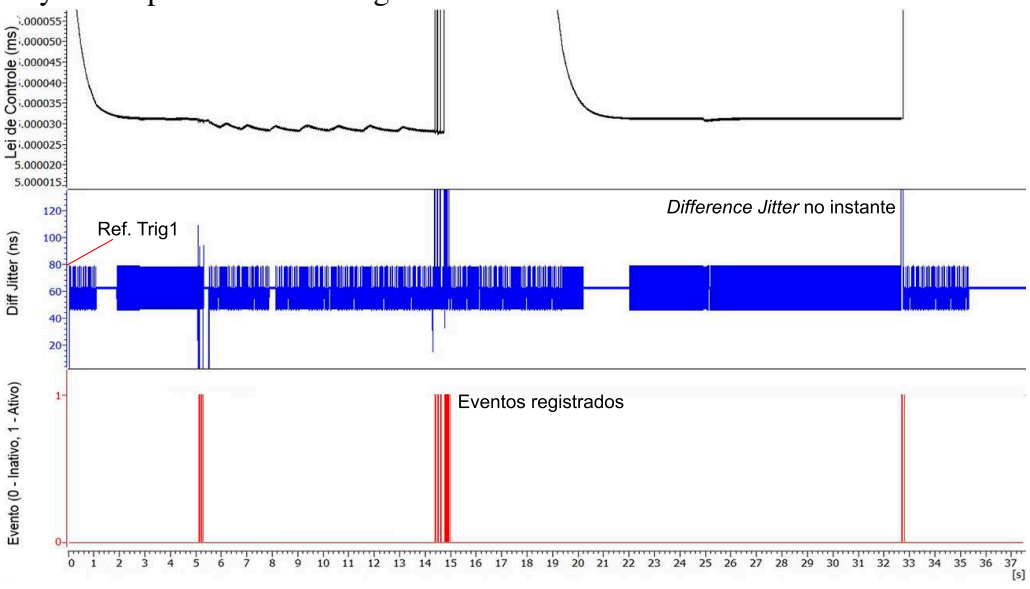
Por fim, a terceira sequencia de injeções EFT na rede FlexRay, com 63 volts e 500 us de rajada é apresentada nas figuras 93 e 94.

Figura 92 – Teste do sistema de diagnóstico com Injeção EFT de 57vp e 1,2 ms na rede FlexRay 10 Mbps com 30% de carga.



Fonte: Autor.

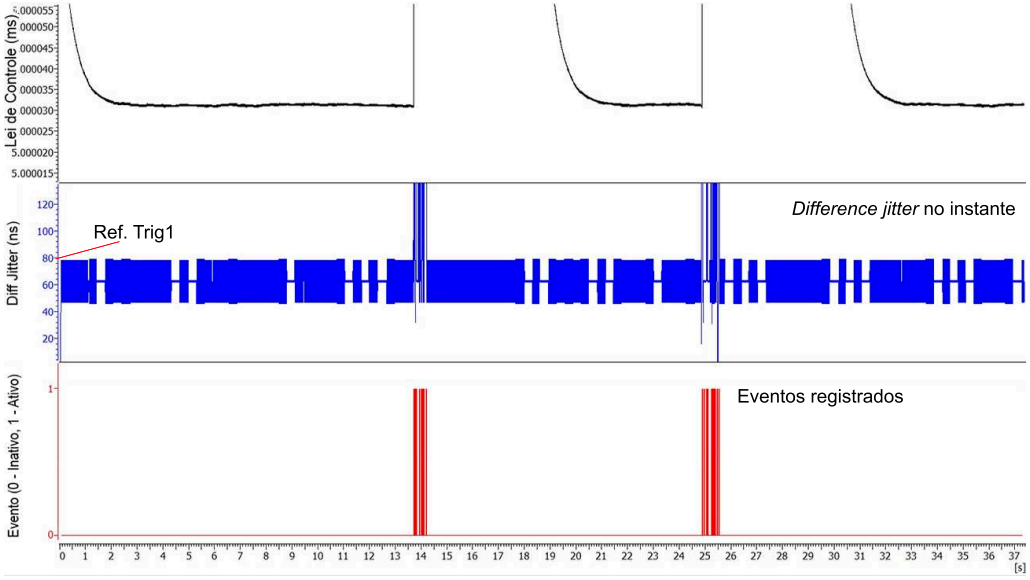
Figura 93 – Teste do sistema de diagnóstico com Injeção EFT de 63vp e 500 us na rede FlexRay 10 Mbps com 2% de carga.



Fonte: Autor.

Neste último experimento (figuras 93 e 94) observa-se uma ocorrência de pelo menos dois distúrbios acentuados durante o período de amostragem, devido a tensão maior dos transientes e também a influência do tráfego na rede. A quantidade de quadros de erros no período foi de 1855 (EFT de 47v e 687us, primeiro experimento), 824 (EFT de 57v e 1200us, segundo experimento) e 1029 (com EFT de 63 v e 500us, terceiro experimento), destacando da mesma forma que nos estudos anteriores, o maior efeito de degradação da rede com rajadas de EFT menores e mais frequentes. A principal diferença é realmente

Figura 94 – Teste do sistema de diagnóstico com Injeção EFT de 63vp e 500 ms na rede FlexRay 10 Mbps com 30% de carga.



Fonte: Autor.

a amplitude dos distúrbios no primeiro e terceiro experimento, que elevam as médias dos ciclos de comunicação para valores acima de 10 microssegundos, muito além da média normal do protocolo. Durante o período amostrado um número menor de eventos foi registrado com o mecanismo de diagnóstico, mostrando na prática o menor efeito de degradação no protocolo FlexRay.

6 ANÁLISE DOS RESULTADOS

6.1 Análise de desempenho da rede e do sistema de controle veicular no cenário 1 - hardware Vector VH6501

O estudo de caso representou uma aplicação prática do mecanismo de diagnóstico de anomalias de desempenho que foi modelado, sendo este, agregado a um sistema de controle de suspensão ativa em uma rede veicular, simulando situações típicas de interferências. A avaliação da capacidade de detecção dos distúrbios foi realizada em diferentes situações de carga da rede, e de acordo com estudos recentes destacados na literatura. O mecanismo é resultado de análises de degradação causada por falhas em redes veiculares, baseado em estudos dos efeitos de degradação causados por EFTs. O estudo dos EFTs em diferentes protocolos de comunicação, serviu para exemplificar as variações que os sistemas de controle críticos podem sofrer, o que possibilitou a especificação de requisitos que foram integrados ao sistema de controle. Essas informações sobre requisitos relacionados aos efeitos de falhas, destacam a importância da abordagem apresentada nesta tese, que mantém uma ligação entre as fases iniciais de projeto com as fases de teste e conformidade das funções do sistema de controle.

Todos os experimentos foram conduzidos de acordo com os passos apresentados na Figura 64, seguindo as tabelas de referência (11 e 12) desenvolvidas após a especificação de requisitos, focado nos problemas de degradação de desempenho constatados nos estudos sobre o impacto de EFTs.

O estudo usou como base as métricas *Average Jitter* e *Difference Jitter*, amplamente usadas em trabalhos da literatura no contexto de redes veiculares, para o acionamento de gatilhos e registro de eventos associados a degradação de performance do sistema. Este tipo de abordagem contribui também para a verificação do impacto das falhas nos protocolos de comunicação, permitindo o registro de quando e como os distúrbios gerados por falhas podem degradar os sistemas de controle críticos (restrições temporais rígidas).

É importante destacar que o mecanismo de diagnóstico desenvolvido pode gerar um grande volume de dados referentes as variações de desempenho da rede de comunicação. Neste primeiro cenário, esse fato também foi um dos motivadores da adoção de um tempo

de amostragem máximo de 120 segundos (em função da grande quantidade de eventos gerados e armazenados na memória interna), o qual permite a verificação da aplicabilidade do presente trabalho.

A seguir as tabelas 16 e 17 apresentam os resultados que foram compilados após a realização de todos os experimentos, na rede veicular com os protocolos CAN e CAN-FD. Os dados de tempo amostrados e relacionados as métricas estão em microsegundos e os eventos gerados são especificados de acordo com a quantidade de ocorrências.

Tabela 16 – Resultados obtidos com o protocolo CAN durante os experimentos.

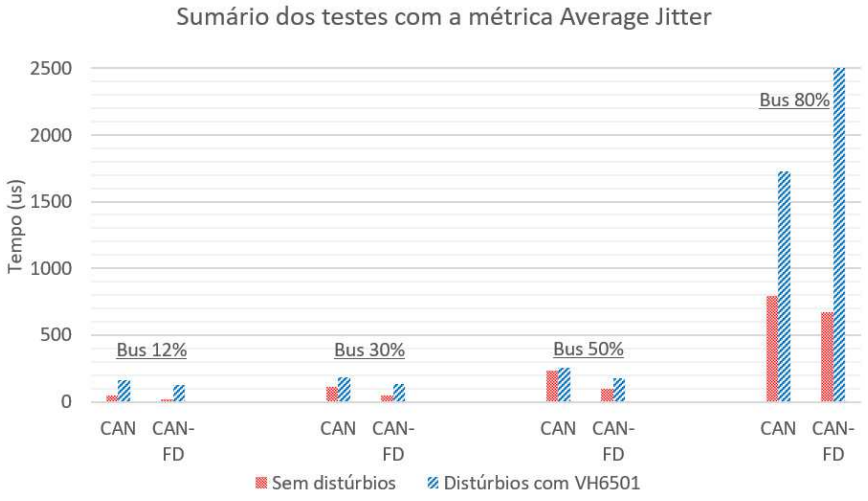
Métrica/Eventos	Rede 12%	Rede 30%	Rede 50%	Rede 80%
Difference Jitter (us)	5011	5656	16271	19091
Eventos Trig1	35	74	46	3019
Average Jitter (us)	159,91	182,66	256,17	1728,93
Eventos Trig2	2	3	5	2

Tabela 17 – Resultados obtidos com o protocolo CAN-FD durante os experimentos.

Métrica/Eventos	Rede 12%	Rede 30%	Rede 50%	Rede 80%
Difference Jitter (us)	4993	5890	6383	21665
Eventos Trig1	44	56	135	3196
Average Jitter (us)	123,79	131,76	174,97	2613,47
Eventos Trig2	1	1	1	1

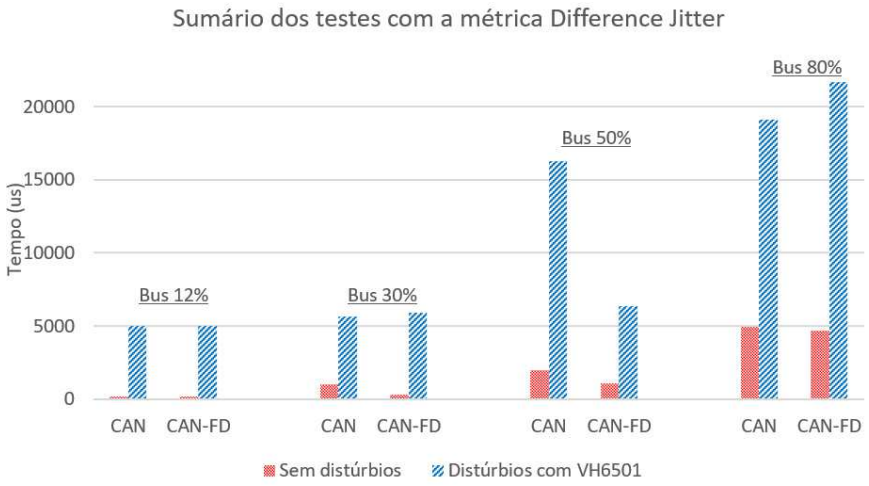
De forma complementar às tabelas, as figuras 95 e 96 sumarizam os dados de performance obtidos nos experimentos resultantes da aplicação do mecanismo de diagnóstico.

Figura 95 – Sumário dos testes com o hardware VH6501 - métrica average jitter.



Fonte: Autor.

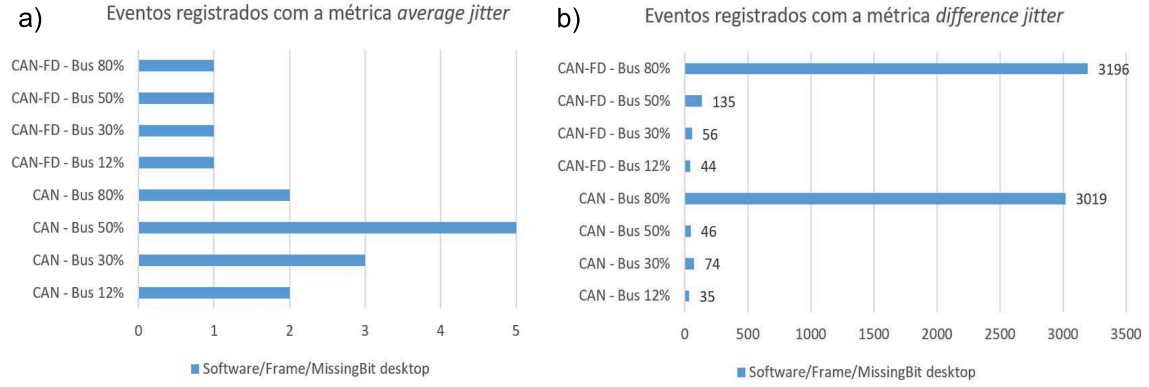
Figura 96 – Sumário dos testes com o hardware VH6501 - métrica difference jitter.



Fonte: Autor.

Estes gráficos destacam o melhor desempenho do protocolo CAN-FD, evidenciados pelos picos menores de atrasos registrados no tempo de amostragem, com base nas duas métricas de desempenho. Durante essa análise o hardware VH6501 se mostrou eficiente na geração dos distúrbios com os desktops (*Software trigger* e *Frame/Missing Bit trigger*) nos testes com o protocolo CAN e CAN-FD, destacando a aplicabilidade da ferramenta na obtenção de dados de redes intra-veiculares. Sumarizando os dados de atrasos registrados e apresentados nas tabelas 16 e 17, é possível evidenciar a aplicabilidade prática de um mecanismo de diagnóstico, com o registro dos eventos que podem ser críticos ao desempenho do sistema de controle. A Figura 97 a) e b) apresenta todos os eventos registrados durante os testes realizados com o hardware VH6501, considerando todas as cargas de ocupação da rede.

Figura 97 – Eventos registrados pelo mecanismo de diagnóstico durante todos os testes com o hardware VH6501.



Fonte: Autor.

Os valores especificados no gráfico da Figura 97 indicam a quantidade de eventos registrados nos arquivos de log, destacando que apesar da quantidade de eventos registrados ser maior no CAN-FD com 80%, esses eventos não tem a mesma amplitude e frequência

de ocorrência do CAN padrão.

Durante a análise do período de amostragem da rede, o gatilho 1 (relacionado a métrica *difference jitter*) gera um maior número de eventos porque está associado a picos de atraso entre as mensagens do sistema de controle. Por outro lado, o gatilho 2 (relacionado à métrica *average jitter*) está associado ao atraso médio entre mensagens de controle e o registro de eventos depende da amostra de tempo analisada, do protocolo e da quantidade de injeção de perturbação. Estes fatores podem ser modificados de acordo com os tipos de análise de estresse que o engenheiro pretende fazer em diferentes tipos de sistemas de controle.

O gatilho 1 indica distúrbios repentinos de desempenho no sistema de controle, permitindo a verificação e detecção de possíveis situações de falhas. Em aplicações futuras essas informações podem ser a base para a aplicação de algoritmos de Inteligência Artificial, pois com cada evento registrado, é possível realizar a correlação entre outros eventos, entre os nós (ECUs) e as mensagens de comunicação que foram afetadas. O gatilho 2 indica uma sequência de perturbações mais duradoura que gera degradação de desempenho, e que normalmente indica uma situação de falha iminente, pois uma ECU e suas mensagens estão sendo afetadas continuamente. Este fato foi evidenciado durante os testes, pois, durante a injeção de perturbação no protocolo CAN com 50% de carga de rede, após o registro de uma sequência de eventos, o barramento passou para o estado *offline* "*busoff-state*", necessitando a reinicialização da rede CAN, conforme destacado na Figura 69 a) e c). Neste ponto, é importante destacar a necessidade de ajuste na sequência de distúrbios usando o hardware VH6501, reduzindo as repetições de 20 para 10, a fim de concluir os testes. Assim, os resultados apresentados nas tabelas 16 e 17, levam em conta esta alteração. Durante os testes com o protocolo CAN-FD, todos os experimentos foram completados com o parâmetro interno de 20 repetições, no hardware VH6501.

Os dados apresentados nos resultados destacam que os picos de desempenho registrados usando a métrica difference jitter são sempre maiores no protocolo CAN. A principal diferença é que no protocolo CAN a frequência desses picos é maior, fato que aumenta o *average jitter*. Desta forma, é possível observar o melhor desempenho do CAN-FD com a métrica *average jitter*. Com 12%, 30% e 50% de carga de ocupação da rede, a variação média foi de 22,58%, 27,86% e 31,69% maior no protocolo CAN. No teste com 80% de carga, o *average jitter* foi 33,84% maior no CAN-FD, mas destacando que com 50% e 80% de carga de ocupação, a injeção de distúrbios com o hardware VH6501 foi dobrada.

Esses resultados mostram as possibilidades de detecção de distúrbios de desempenho em tempo de execução com o mecanismo desenvolvido e que o protocolo CAN-FD é mais tolerante às variações de desempenho quando comparado ao CAN tradicional. Em todos os cenários analisados o *average jitter* no CAN-FD foi menor, com exceção dos testes com 80% de carga, mas justificado pelo fato de a injeção de falhas ter sido o dobro. Outro ponto importante, é o registro de todos os eventos com o apoio de arquivos de log, que

contribui para análises posteriores e manutenção do histórico de desempenho do sistema que esta sob análise.

Por fim, o estudo efetuado evidenciou a aplicabilidade e integração do mecanismo de diagnóstico modelado em um sistema de controle veicular crítico, com um conjunto de mensagens que representam as funcionalidades de um sistema de controle de suspensão ativa. As ferramentas usadas possibilitaram a obtenção de todos os dados da rede e a realização dos cálculos das métricas de desempenho, bem como a inserção dos códigos CAPL para o sistema de controle, mecanismo de diagnóstico e códigos para injeção de falhas e distúrbios na rede veicular.

6.2 Análise de desempenho da rede e do sistema de controle veicular no cenário 2 - hardware de injeção EFT

O segundo cenário de estudo caracteriza outra aplicação prática do mecanismo de diagnóstico modelado, baseado na proposta da presente tese. Neste experimento o sistema de diagnóstico é avaliado com injeções de falhas reais na rede de comunicação, de acordo com o desenvolvimento de um novo hardware para injeção de falhas EFT. O estudo avalia a detecção das anomalias de desempenho nos três principais protocolos de comunicação usados em redes intra-veiculares (CAN, CAN-FD e FlexRay), seguindo as especificações de mensagens para um sistema de controle de suspensão ativa.

A avaliação da capacidade de detecção dos distúrbios foi realizada em duas situações de carga da rede, tendo somente o sistema de controle comunicando (2% em FlexRay e 12% em CAN/CAN-FD) e com mensagens de tráfego (30% em todos os protocolos). O protocolo FlexRay foi avaliado com a particularidade de o impacto na variação de carga ocorrer no segmento estático (mensagens periódicas), sendo o segmento dinâmico (de mensagens eventuais) mantido em 100%, mantendo a coerência com os estudos realizados anteriormente com o protocolo FlexRay (MICHELIN, 2014). Nestes estudos o foco é a detecção dos eventos de degradação de desempenho e não a susceptibilidade a falhas, por isso, não é necessário estender as análises para as outras variações de carga de rede.

Os estudos apresentados nesta seção são importantes para verificação dos efeitos de degradação em um sistema de controle crítico, enfatizando a possibilidade de analisar quando o sistema não atenderá os requisitos de um sistema de tempo real. A avaliação sistematizada dos principais protocolos usados em redes intra-veiculares fornece informações importantes para estabelecer os limites de operação destes sistemas, sendo estas informações cruciais para a definição e especificação de requisitos que podem ser incorporados ao projeto dos sistemas de controle. Estas questões são a base da metodologia proposta, onde a falha estudada de forma extensiva e a implementação do mecanismo de diagnóstico caracterizam exemplos da sua aplicação efetiva.

Os experimentos foram conduzidos de acordo com o método apresentado na Figura

64, conforme os parâmetros de referência apresentados nas tabelas 11, 12 e 13. As métricas *Average Jitter* e *Difference Jitter* são a base para a definição dos gatilhos, caracterizando uma forma de analisar o desempenho da rede veicular, efetuando o registro de eventos sobre a degradação de desempenho do sistema.

A implementação destas rotinas dentro da plataforma de software Vector CANoe, em linguagem CAPL, permite a geração de uma grande quantidade de informações (gráficos e registros de logs de toda a comunicação) sobre o funcionamento das ECUs. Neste segundo cenário de estudo foi adotado um tempo aproximado de 30 a 38 segundos de monitoramento da rede, devido justamente ao fato da geração de uma grande quantidade de eventos e registro de informações da rede que levam o hardware de análise aos limites de sua memória interna. Para a realização do monitoramento da rede veicular durante muitas horas ou dias, comumente seria necessário uma unidade de armazenamento externo conectado ao sistema de diagnóstico.

Uma compilação quantitativa dos dados obtidos durante os experimentos são apresentados nas tabelas 18 e 19 com a métrica *average jitter* e nas tabelas 20 e 21 com a métrica difference jitter, em ambas as cargas de rede analisadas. Essa organização visa contribuir para as análises apresentadas e também para a observação de todos os protocolos de forma conjunta. Os dados de tempo amostrados estão em microssegundos e todos os eventos são lançados de acordo com a quantidade de ocorrências durante o período de análise.

Tabela 18 – Resultados obtidos com a aplicação do mecanismo de diagnóstico em todos os protocolos, com barramento em 12% e 2% (FlexRay), com a métrica *average jitter*.

Protocolos	Sem distúrbios	EFT 47v 687us	EFT 57v 1,2ms	EFT 63v 500us
CAN	33,11	97,41	79,19	245,74
Eventos Trig2	-	1	2	1
CAN-FD	20,34	106,64	102,44	97,81
Eventos Trig2	-	2	2	2
FlexRay	0,0254	5,61	2,98	10,12
Eventos Trig2	-	2	3	6

Por meio das tabelas 18, 19, 20 e 21 é possível observar os valores obtidos durante cada tipo de injeção de EFT, destacando os picos de atraso maiores registrados nos testes com menor tempo de rajada (EFT de 47 e 63 volts), com pico de atraso entre as mensagens chegando a atingir 12 e 17 milissegundos. Cabe ressaltar que no terceiro experimento com CAN (Tabela 19, coluna 5), os valores obtidos com a carga de 30% na métrica *difference jitter* foram menores devido ao fato da realização de um ajuste no intervalo de injeção dos pulsos EFT, de 1 ms para 3 ms. Esse ajuste foi necessário para concluir o experimento, pois com intervalos de 1 ms (usado nos experimentos anteriores), a rede CAN entra em modo *offline* e não é possível concluir as análises. Esse fato ocorre devido

Tabela 19 – Resultados obtidos com a aplicação do mecanismo de diagnóstico em todos os protocolos, com barramento em 30%, com a métrica *average jitter*.

Protocolos	Sem distúrbios	EFT 47v 687us	EFT 57v 1,2ms	EFT 63v 500us
CAN	72,45	380,85	392,57	189,79*
Eventos Trig2	-	1	1	1
CAN-FD	49,51	116,28	132,23	284,42
Eventos Trig2	-	1	2	2
FlexRay	0,0257	13,01	11,64	19,95
Eventos Trig2	-	4	4	4

* Valor menor com EFT de 63 volts devido ao ajuste no intervalo de injeção EFT.

Tabela 20 – Resultados obtidos com a aplicação do mecanismo de diagnóstico em todos os protocolos, com barramento em 12% e 2% (FlexRay), com a métrica *difference jitter*.

Protocolos	Sem distúrbios	EFT 47v 687us	EFT 57v 1,2ms	EFT 63v 500us
CAN	240	998	908	17754
Eventos Trig1	-	611	364	332
CAN-FD	185	1082	1030	1067
Eventos Trig1	-	735	901	644
FlexRay	0,0771	209	37	325
Eventos Trig1	-	30	25	32

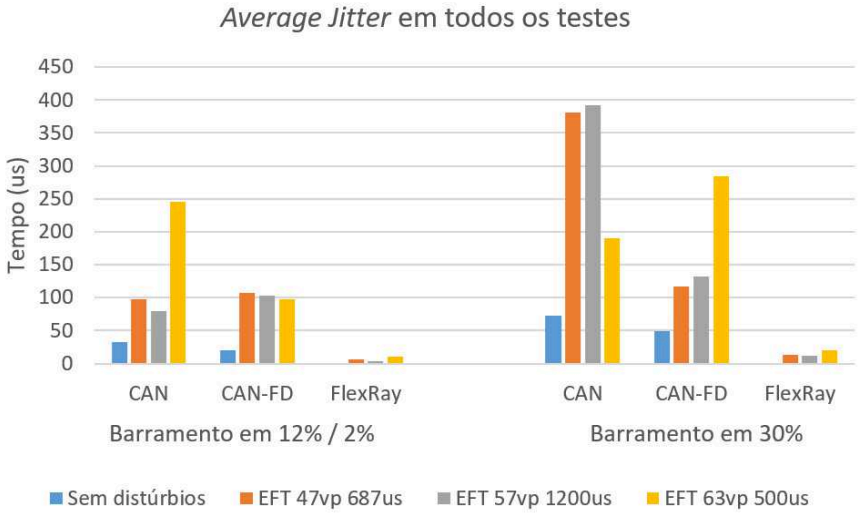
Tabela 21 – Resultados obtidos com a aplicação do mecanismo de diagnóstico em todos os protocolos, com barramento em 30%, com a métrica *difference jitter*.

Protocolos	Sem distúrbios	EFT 47v 687us	EFT 57v 1,2ms	EFT 63v 500us
CAN	790	12106	6804	1352
Eventos Trig1	-	808	1017	430
CAN-FD	297	1235	1044	2790
Eventos Trig1	-	444	579	1602
FlexRay	0,0775	492	188	372
Eventos Trig1	-	33	28	39

a alta susceptibilidade do protocolo CAN padrão aos transientes de maior tensão e menor tempo de rajada.

Em todas as tabelas é possível verificar os valores obtidos nos experimentos com o protocolo CAN-FD, destacando o baixo *average jitter*, que com exceção do ultimo experimento (média de 284 us), manteve o ciclo da lei de controle entre 97 e 132 us de atraso. Os dados apresentados com estes números evidenciam o ótimo desempenho do protocolo FlexRay, pois mesmo no pior caso de injeção EFT, o atraso médio chegou a apenas 19 microssegundos. Para consolidar as informações obtidas nos experimentos realizados, as figuras 98 e 99 sumarizam os dados de performance obtidos durante todas as injeções de falhas nos protocolos de comunicação estudados.

Figura 98 – Resumo dos testes com o mecanismo de diagnóstico - EFT hardware e métrica *average jitter*.



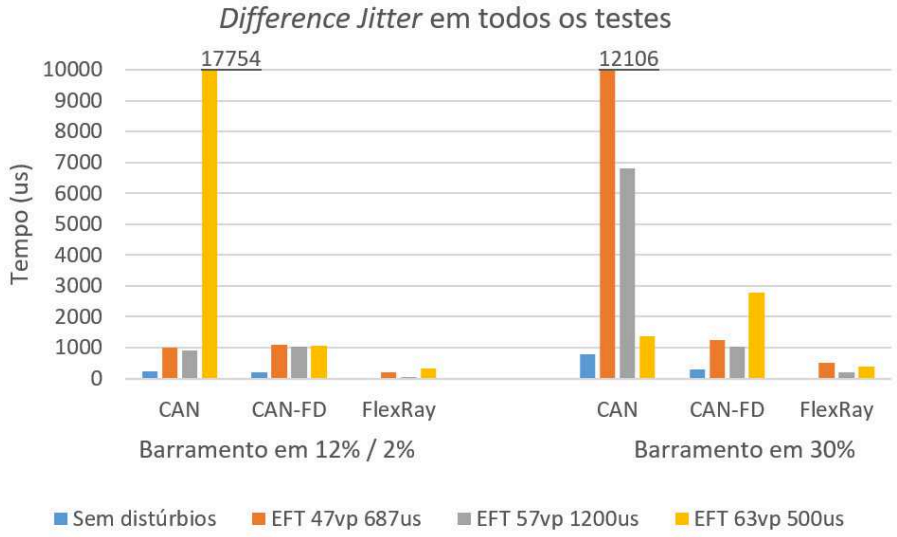
Fonte: Autor.

Com relação aos dados obtidos dos testes com o mecanismo de diagnóstico e a métrica *average jitter* (Figura 98) a susceptibilidade aos transientes EFT fica evidente, podendo neste caso avaliar o protocolo e os atrasos na lei de controle sob três óticas diferentes. A primeira observando os valores máximos de média atingidos durante as injeções EFT, a segunda observando o quanto o protocolo foi afetado em relação ao seu padrão normal de comunicação, e a terceira observando a influência da carga de ocupação da rede (*bus-load*) nos atrasos gerados na lei de controle e na capacidade de detecção das variações de desempenho. Vejamos essa análise:

- Primeira ótica:

  Observando as figuras 98 e 99 em conjunto com as Tabelas 18 e 19, verifica-se que os transientes EFT de 47 e 63 volts com menor tempo de rajada geram uma degradação maior em relação aos transientes de 57 volts. Fato justificado pela maior chance de incidência de erros, com uma quantidade maior de transientes dentro de um ciclo da lei de controle.

Figura 99 – Resumo dos testes com o mecanismo de diagnóstico - EFT hardware - métrica *difference jitter*.



Fonte: Autor.

Outro ponto destacado é que os atrasos resultantes das injeções de falhas, são sempre maiores nos protocolos CAN e CAN-FD em relação aos registrados no protocolo FlexRay. Por exemplo, com EFT de 47 volts, o atraso máximo no FlexRay foi de 5,61 microssegundos, mas nos protocolos CAN e CAN-FD esses valores foram respectivamente 19 e 17 vezes maiores (97,41 us e 106,64 us).

- Segunda ótica:

  Por outro lado, se observarmos o efeito de degradação em relação ao padrão normal de comunicação do protocolo, o protocolo FlexRay é o que mais sofre com os efeitos das injeções de falhas. O atraso médio da lei de controle com o FlexRay é de aproximadamente 25 nanossegundos, subindo para 5,61 microssegundos com EFT de 47 volts, o que gera um aumento de 220 vezes. Na mesma comparação o procolo CAN-FD tem um aumento de 5,24 vezes e o protocolo CAN de 2,94 vezes. Enfatiza-se que esses dados são especificamente para este experimento de estresse de falhas, com a avaliação e detecção das anomalias de desempenho.

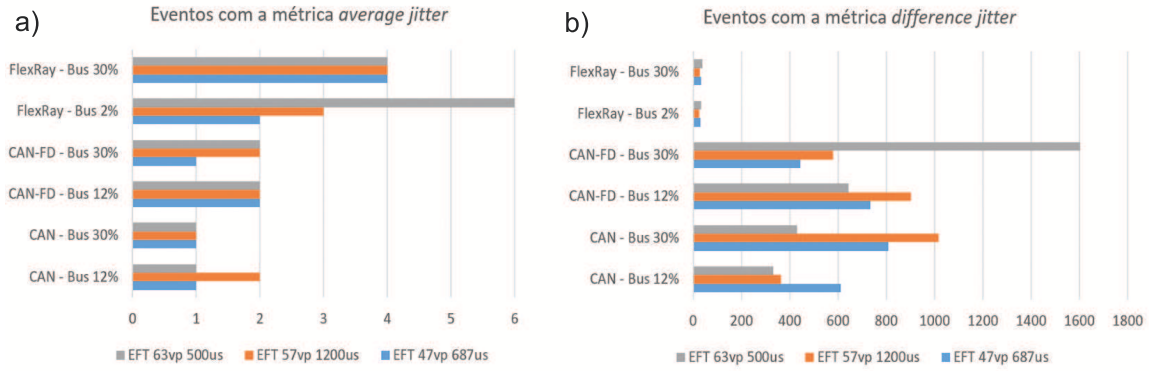
- Terceira ótica:

  A carga de ocupação da rede influência diretamente na capacidade de atendimento a requisitos temporais dos sistemas de controle. Na indústria tipicamente se usa como referência um limite de 30% de ocupação da rede para que protocolos como o CAN possam prover tais garantias. De tal modo, as variações da carga de ocupação da rede durante os experimentos foram importantes para a verificação do impacto das falhas EFT na lei de controle. Foi verificado que os picos de atraso e também a média de atraso no período de amostragem foram maiores em todos

os casos de teste. A detecção de eventos pelo sistema de diagnóstico também foi maior com o aumento da carga de rede para 30%, tendo um resultado diferente apenas no protocolo CAN-FD, onde o registro de picos de atraso foram menores na comparação com 12% de ocupação da rede. Essa redução no CAN-FD deve-se principalmente ao menor impacto dos eventos EFT na comunicação durante os testes, com consequente menor tratamento de erros.

Todos os dados computados e apresentados nesta análise comparativa são oriundos da aplicação de um método de injeção de falhas, com hardware desenvolvido seguindo padrões da indústria, para assim, especificar e modelar requisitos não funcionais relacionados a falhas, implementados e validados no mecanismo de diagnóstico desenvolvido. A computação das métricas, o registro de eventos sobre as variações de desempenho, bem como as inferências sobre o comportamento da lei de controle com injeções de falhas, só foram possíveis devido a modelagem do mecanismo de diagnóstico. Outro ponto fundamental deste estudo de caso é a geração dos eventos relacionados às anomalias de desempenho no sistema de controle em estudo. A Figura 100 a) e b) destaca todos os eventos registrados durante os testes realizados.

Figura 100 – Eventos registrados pelo mecanismo de diagnóstico durante todos os testes e nos três protocolos estudados.



Fonte: Autor.

Durante o estudo realizado, todas as variações de desempenho observadas com o software CANoe foram registradas pelo mecanismo de diagnóstico desenvolvido, detectando os efeitos destas falhas no sistema de controle veicular. Tais informações são a base para novas soluções de tolerância a falhas, bem como também, para definir notificações baseadas nos limites de operação de um sistema de controle crítico.

Por fim, este segundo estudo mostrou a possibilidade de integração do mecanismo de diagnóstico modelado em um sistema de controle veicular, neste caso, baseado nas mensagens cíclicas de um sistema de suspensão ativa. Os resultados apresentados evidenciam dois pontos importantes, a susceptibilidade dos protocolos de comunicação a transientes elétricos e também a possibilidade de integrar mecanismos inteligentes para a detecção e diagnóstico de situações críticas ao sistema de controle.

6.3 Discussão dos estudos e experimentos realizados

Os estudos de caso realizados destacam os principais protocolos de comunicação usados em sistemas de controle críticos nas redes intra-veiculares, os protocolos CAN, FlexRay e o mais recente CAN-FD. Para verificar a susceptibilidade destes protocolos a falhas e interferências, testes específicos de estresse foram realizados. Apesar de existirem diferentes tipos de falhas e interferências que afetam e degradam o desempenho dos protocolos, um estudo pontual e aprofundado foi realizado sobre um tipo específico de falha, os transientes elétricos rápidos (EFT). A escolha deste tipo de falha originou de pesquisas recentes na literatura, que indica lacunas sobre a verificação dos efeitos reais desta falha em transceptores e consequentemente nos sistemas de controle. Tal estudo é suportado pelas especificações de teste detalhadas na norma IEC 62228:2007, norma que também foi recentemente revisada sob a série IEC 62228 (2018 e 2019, partes 1 a 8), onde até o momento somente às três primeiras partes foram publicadas. Ao mesmo tempo, a escolha deste tipo de falha serviu de base e exemplo para o estudo de aplicabilidade da metodologia apresentada na presente pesquisa de doutorado.

Todos os testes de susceptibilidade consideraram diferentes amplitudes de tensão (de 19 a 63 volts) e duração de rajada (188 a 1200 us) dos transientes EFT. Entretanto, apesar de protocolos como o FlexRay e CAN-FD possuírem maior desempenho, os mesmos apresentaram degradações acentuadas em relação ao atraso médio de comunicação sem injeção de falhas. Os resultados indicam que transientes elétricos com menor tempo de rajada e maior amplitude de tensão tendem a gerar mais falhas na modulação de sinais, fato observado em todos os experimentos. O protocolo CAN padrão sofreu os maiores picos de atrasos e degradação com a métrica *average jitter*, entrando no estado *busoff* nos testes com largura de banda acima de 50%. O protocolo CAN-FD conseguiu tolerar as injeções de falhas, tendo o menor impacto no atraso médio do ciclo de controle. O protocolo FlexRay, por sua vez, foi o que mais sofreu com as injeções de falhas, pois os picos de atrasos gerados são muito acima da média sem injeções de falhas, saltando de poucos nanosegundos para picos de até 94 microssegundos. Essa ocorrência deve-se ao fato do protocolo trabalhar com tempos de bit menores e ser mais susceptível aos transientes, fato que pode impor restrições de uso ao protocolo em ambientes com muita susceptibilidade a esse tipo de interferência.

Com base nestes estudos a metodologia desenvolvida aplica o framework RT-FRIDA para especificar requisitos relacionados aos efeitos de degradação dos transientes EFT, modelando um mecanismo de diagnóstico (nomeado de “FaultObserver”) das degradações de desempenho. O dados das análises de susceptibilidade dos protocolos serviram para a definição de limites de operação, os quais foram utilizados como referência para gatilhos de diagnóstico de anomalias na rede. Estes gatilhos permitem ao sistema de controle registrar eventos de anomalias no desempenho em tempo de execução. Os gatilhos

permitem registrar eventos definidos de acordo com métricas de desempenho estudadas (*Average Jitter* e *Difference Jitter*), as quais possibilitam a observação de atrasos gerados em ciclos de comunicação entre ECUs e respectivas mensagens de controle críticas.

O mecanismo de diagnóstico modelado foi desenvolvido na linguagem CAPL (baseada na linguagem padrão C++), dentro do software CANoe, permitindo assim que este possa ser parametrizado e reutilizado para testes em diferentes tipos de sistemas de controle intra-veiculares. No estudo de caso, o mecanismo foi incorporado a uma rede intra-veicular, junto a um sistema de controle de suspensão ativa, com mensagens cíclicas de 5 ms, representando um cenário real de aplicação e validação. Experimentos foram conduzidos nos três protocolos de comunicação estudados, com dois tipos de hardwares para injeção de falhas.

No primeiro estudo de caso, os experimentos foram conduzidos com o hardware Vector VH6501 (*CAN disturbance interface*), mostrando a efetividade de aplicação do mecanismo de diagnóstico em redes CAN e CAN-FD (até o momento o hardware não provê suporte ao protocolo FlexRay). Os experimentos foram conduzidos injetando distúrbios lógicos que geram tratamentos típicos de falhas na rede de comunicação. Para fins de verificação do desempenho do sistema de diagnóstico e comportamento dos protocolos de comunicação perante falhas, os testes foram realizados com diferentes cargas de ocupação da rede. Os picos de atrasos registrados mostraram o melhor desempenho do protocolo CAN-FD, especificamente com a métrica *average jitter*. Em todas as situações de carga de ocupação da rede (12% - somente sistema de controle, 30%, 50% e 80%), foi possível observar a efetiva detecção de eventos pelo mecanismo de diagnóstico.

No segundo estudo de caso, o mecanismo de diagnóstico foi avaliado com injeções de transientes EFT, de acordo com hardware originado dos primeiros estudos de susceptibilidade a transientes em CAN (ROQUE *et al.*, 2017), o qual foi melhorado no escopo da dissertação de mestrado de Daniel Pohren (POHREN, 2020). Os experimentos de injeção de falhas seguiram o método de teste desenvolvido no âmbito desta tese, com amplitude de tensão e tempos de duração da rajada de acordo com pesquisas da literatura recente. Além da variação dos tipos de EFT, os experimentos foram conduzidos com dois tipos de carga de ocupação da rede, verificando os distúrbios gerados quando somente o sistema de controle estava comunicando na rede (12% em CAN e CAN-FD e 2% em FlexRay), e também com mensagens apena para aumentar o tráfego e elevar a carga de ocupação para 30%. A adoção destas duas cargas de ocupação se justifica pelo fato de ser um teste para validação do mecanismo de diagnóstico e não mais uma análise de susceptibilidade, por mais que tais efeitos tenham sido confirmados novamente. Do ponto de vista de desempenho dos protocolos, por se tratar de um protocolo mais robusto e com maior largura de banda, o FlexRay obteve os melhores resultados, tendo atingido nos testes de estresse de pior caso, um atraso médio com a métrica *average jitter* de até 19 us. Com a mesma métrica o protocolo CAN padrão chegou a registrar atraso médio de 392.57 us e o pro-

tocolo CAN-FD no máximo 284.42 us, ambos os resultados com carga de ocupação de 30%. Por outro lado, se considerada a degradação em relação ao período de comunicação normal, o protocolo FlexRay obteve o maior impacto negativo, pois o atraso médio sobe de aproximadamente 25 ns para picos de 5, 10 e 19 us.

Em ambos os estudos de caso foram registrados arquivos com todos os eventos de anomalias de desempenho, contribuindo também para a análise futura dos eventos por algoritmos especializados. Essa análise permitiu observar que durante 30 dias de monitoramento em tempo de execução, aproximadamente 8 GB de dados podem ser gerados, por sistema de controle crítico monitorado. Essa questão é importante, pois um dos fatores críticos das atuais ECU é o espaço de armazenamento, levantando a discussão a respeito da criação de espaços maiores de armazenamento local, conexões com sistemas em nuvem, e levando em consideração técnicas de segurança destas informações. Esses resultados são importantes para a criação de sistemas de monitoramento e diagnóstico que visam o tratamento de situações críticas em tempo de execução, as quais são fundamentais para as novas tecnologias aplicadas na indústria automotiva.

7 CONCLUSÕES E TRABALHOS FUTUROS

7.1 Conclusões

A indústria automotiva representa um cenário que vem passando por evoluções tecnológicas constantes, devido à quantidade crescente de ECU interconectadas, formando redes intra-veiculares complexas e heterogêneas. No caminho dessa evolução, esta pesquisa discute a criticidade da comunicação nestas redes, com destaque a susceptibilidade a falhas transientes nos protocolos de comunicação, fato que motiva o estudo de novas técnicas de modelagem, teste e diagnóstico de falhas, para assim, contribuir com o projeto de sistemas de controle mais robustos e confiáveis.

A presente tese contribui apresentando uma nova metodologia para teste e modelagem de falhas que afetam e degradam protocolos de comunicação intra-veiculares, agregando estas questões às fases de especificação de requisitos. Assim, durante o projeto é possível adicionar funcionalidades para detectar ou mitigar os efeitos negativos de diferentes falhas, como por exemplo, diagnosticando e prevenindo situações críticas causadas pelo não atendimento de restrições temporais rígidas. Neste contexto, foi identificada a possibilidade de reutilizar o *framework* RT-FRIDA, pois este foi desenvolvido para a modelagem e especificação de requisitos de sistemas de tempo real, tendo como base a AOM, mas possuindo lacuna referente a especificação de requisitos relacionados a falhas. As falhas possuem efeitos transversais, assim, com base nos conceitos de orientação a aspectos, o *framework* RT-FRIDA foi estendido, preenchendo esta lacuna, a qual possibilitou a especificação de falhas que afetam sistemas de controle de tempo real no contexto da indústria automotiva. O estudo resultou no detalhamento de documentos representados por *templates* e *checklists* específicos para a modelagem e especificação de requisitos não funcionais, relacionados a falhas em redes intra-veiculares.

A aplicabilidade da metodologia teve como referência os três estudos de susceptibilidade realizados em CAN, CAN-FD e FlexRay, que seguiram as normas IEC 62228 e ISO 26262. Assim, a pesquisa apresentou o desenvolvimento de um novo método de teste e análise de degradação de desempenho, apoiado por hardwares para injeção de falhas EFT em redes intra-veiculares. Os testes de susceptibilidade a EFT apresentados evidenciaram

maior degradação do tradicional protocolo CAN em relação aos protocolos CAN-FD e FlexRay, gerando maiores atrasos no ciclo de comunicação.

Os resultados apresentados nos experimentos de susceptibilidade a falhas EFT, demostraram os riscos associados aos sistemas de controle veiculares críticos, especialmente com o crescente aumento da complexidade das redes intra-veiculares. A metodologia desenvolvida representa uma alternativa que contribui para a modelagem de falhas, evidenciadas por um mecanismo de diagnóstico desenvolvido e testado em uma rede real. As contribuições apresentadas neste estudo são de suma importância para o processo de teste e desenvolvimento de novas tecnologias embarcadas, cada vez mais presentes na indústria automotiva com adoção crescente de carros elétricos e autônomos.

7.2 Trabalhos futuros

As contribuições apresentadas nesta tese, destacam possibilidades de trabalhos futuros, os quais são destacados e relacionados a seguir:

- Realizar testes de susceptibilidade a outros tipos de problemas, como por exemplo falhas geradas por interferências eletromagnéticas irradiadas e ataques maliciosos a rede, com o intuito realizar novas aplicações da metodologia, considerando também cenários de outras indústrias (Manufatura, Aviônica, entre outros).

- Realizar testes considerando o uso de um armazenamento dedicado em veículo de teste, considerando o envio periódico dos dados de desempenho e diagnóstico para um sistema de computação em nuvem.

- Obter um *dataset* (conjunto de dados e mensagens) de um veículo real e realizar o monitoramento em tempo de execução de diferentes mensagens de controle críticas, observando assim, os efeitos durante um período maior.

- Aplicar algoritmos de Inteligência Artificial na base de dados gerada, com o intuito de realizar predições de possíveis falhas ou situações de manutenção em ECU.

7.3 Publicações

Durante o desenvolvimento da presente Tese de Doutorado diversas publicações foram realizadas, as quais consolidam a relevância dos resultados alcançados.

Artigos em periódicos publicados:

- Roque, A. S., Pohren, D., Freitas, E. P. e Pereira, C. E. An Approach to Address Safety as Non-Functional Requirements in Distributed Vehicular Control Systems. **Journal of Control, Automation and Electrical Systems**, vol 30, Issue 5, p.700-715, October, 2019. (Qualis A4).

- Pohren, D. H., Roque, A. S., Kranz, T. I., Freitas, E. P. P. e Pereira, C. E. An Analysis of the Impact of Transient Faults on the Performance of the CAN-FD Protocol. **IEEE Transactions on Industrial Electronics**, 2019. (Qualis A1).

- Roque, A. S., Pohren, D., Freitas, E. P. e Pereira, C. E. ROQUE, Alexandre S. et al. EFT fault impact analysis on performance of critical tasks in intravehicular networks. **IEEE Transactions on Electromagnetic Compatibility**, vol 59, Issue 5, p.1415-1423, October, 2017. (Qualis A2).

Artigos em periódicos submetidos:

- "A fault modeling based runtime diagnostic mechanism for vehicular distributed control systems- **IEEE Trans Intelligent Transportation Systems**. (Qualis A1).

- "Impact Analysis of Electrical Fast Transients on FlexRay protocol according to IEC 62228- **IEEE Transactions on Electromagnetic Compatibility**. (Qualis A2).

Conferências Internacionais - Aceitos e apresentados:

- Roque, A. S., Freitas, E. P., Jazdi, N. e Pereira, C. E. *An approach of fault modeling in communication protocols supported by NFR and fault tree analysis* **5th IFAC Symposium on Telematics Appications, IFAC-TA-2019.** , Chengdu, China, 2019.

- Roque, A. S., Nunes, G. L., Freitas, E. P. e Pereira, C. E. *Requirements specification and evaluation for transient faults in communication protocols based on RT-FRIDA framework.* **IFAC Conf. on Embedded Systems, Computational Intelligence and Telematics in Control, CESCIT.** , Portugal, 2018.

- Nunes, G. L., Roque, A. S., Araujo, S. R. e Pereira, C. E. *Downlink cyclic resources scheduling algorithm for industrial wireless M2M communication.* **IFAC Conference on Embedded Systems, Computational Intelligence and Telematics in Control, CESCIT.** IFAC-PapersOnLine, 51(10), Portugal, 2018.

- Roque, A. S., Steinmetz, Charles, Freitas, E. P., Pereira, C. E. Modeling faults in communication protocols based on an aspect-oriented method. In: **IEEE 15th Int. Conf. of Industrial Informatics INDIN'2017**, Emden, Germany, 2017.

- Steinmetz, Charles, Schroeder, Greyce, Roque, A. S., Pereira, C. E., Wagner, Carolin, Saalmann, Philipp, Hellingrath, Bernd. *Ontology-driven IoT code generation for FIWARE.* In: **IEEE 15th Int. Conf. of Industrial Informatics INDIN'2017**, Emden, Germany, 2017.

- Roque, A. S., Pohren, D., Pereira, C. E. e Freitas, E. P. *Communication analysis in CAN networks under EFT injection.* In: **IEEE International Conference on Automatica (ICA-ACCA)**, IEEE Chile, 19-21 Oct, 2016.

REFERÊNCIAS

ADEDJOUMA, M.; YAKYMETS, N. A framework for model-based dependability analysis of cyber-physical systems. *In*: IEEE INTERNATIONAL SYMPOSIUM ON HIGH ASSURANCE SYSTEMS ENGINEERING (HASE), 19., 2019, Hangzhou, China. **Proceedings...** IEEE, 2019. p. 82–89.

AGRAWAL, M. *et al.* CAN-FD-Sec: improving security of can-fd protocol. *In*: SECURITY AND SAFETY INTERPLAY OF INTELLIGENT SOFTWARE SYSTEMS: ESORICS 2018 INTERNATIONAL WORKSHOPS, ISSA 2018 AND CSITS 2018, BOOK OF REVISED SELECTED PAPERS, 2019, Barcelona, Spain. **Proceedings...** Springer, 2019. v. 11552, p. 77–93.

AKKAYA, I. *et al.* Systems engineering for industrial cyber–physical systems using aspects. **Proceedings of the IEEE**, New York, NY, v. 104, n. 5, p. 997–1012, 2016.

ALI, S.; BRIAND, L. C.; HEMMATI, H. Modeling robustness behavior using aspect-oriented modeling to support robustness testing of industrial systems. **Software & Systems Modeling**, Switzerland, v. 11, n. 4, p. 633–670, 2012.

ALMEIDA, L.; PEDREIRAS, P.; FONSECA, J. A. G. The FTT-CAN protocol: why and how. **IEEE transactions on industrial electronics**, New York, NY, v. 49, n. 6, p. 1189–1201, 2002.

AN, H. S.; JEON, J. W. Analysis of CAN-FD to CAN message routing method for CAN-FD and CAN gateway. *In*: INTERNATIONAL CONFERENCE ON CONTROL, AUTOMATION AND SYSTEMS (ICCAS), 17., 2017, Jeju, South Korea. **Proceedings...** IEEE, 2017. p. 528–533.

ANICULAESEI, A. *et al.* Automated generation of requirements-based test cases for an adaptive cruise control system. *In*: IEEE WORKSHOP ON VALIDATION, ANALYSIS AND EVOLUTION OF SOFTWARE TESTS (VST), 2018, Campobasso, Italy. **Proceedings...** IEEE, 2018. p. 11–15.

ANKARSON, P. *et al.* Impact of different interference types on an LTE communication link using conducted measurements. *In*: IEEE INTERNATIONAL SYMPOSIUM ON ELECTROMAGNETIC COMPATIBILITY (EMC), 2015, Dresden, Germany. **Proceedings...** IEEE, 2015. p. 177–182.

ASSIS, A. C. d. **Implementação e avaliação do protocolo FTT-CAN sobre o sistema AUTOSAR**. 2011. Dissertação (Mestrado em Engenharia Elétrica) — Universidade Federal do Rio Grande do Sul, Porto Alegre, 2011.

ATAIDE, F. H. **Proposta de melhoria de tempo de resposta para o protocolo FTT-CAN**: estudo de caso em aplicação automotiva. 2010. Dissertação (Mestrado em Engenharia Elétrica) — Universidade Federal do Rio Grande do Sul, Porto Alegre, 2010.

ATAIDE, F. H.; PEREIRA, C. E. A new approach to improve the response time of the FFT-CAN protocol. **Sba Controle & Automação**, Campinas, SP, Brasil, v. 23, n. 5, p. 621–635, 2012.

ATZORI, L.; IERA, A.; MORABITO, G. The internet of things: a survey. **Computer networks**, Amsterdam, v. 54, n. 15, p. 2787–2805, 2010.

AUTOSAR, G. **Automotive open system architecture - AUTOSAR.** Disponível em: <https://www.autosar.org/>. Acesso em: 2019.

AVIZIENIS, A. *et al.* Basic concepts and taxonomy of dependable and secure computing. **IEEE transactions on dependable and secure computing**, New York, NY, v. 1, n. 1, p. 11–33, 2004.

AZIMI, S.; MORAMARCO, A.; STERPONE, L. Reliability evaluation of heterogeneous systems-on-chip for automotive ECUs. *In*: IEEE INTERNATIONAL SYMPOSIUM ON INDUSTRIAL ELECTRONICS (ISIE), 26., 2017, Edinburgh, UK. **Proceedings...** IEEE, 2017. p. 1291–1296.

BARONTI, F. *et al.* Design and verification of hardware building blocks for high-speed and fault-tolerant in-vehicle networks. **IEEE transactions on industrial electronics**, New York, NY, v. 58, n. 3, p. 792–801, 2011.

BARRA, E.; MORATO, J. Early knowledge organization assisted by aspects. **Science of Computer Programming**, Amsterdam, v. 121, p. 34–54, 2016.

BAUER, S.; DEUTSCHMANN, B.; WINKLER, G. Prediction of the robustness of integrated circuits against EFT/BURST. *In*: IEEE INTERNATIONAL SYMPOSIUM ON ELECTROMAGNETIC COMPATIBILITY (EMC), 2015, Dresden, Germany. **Proceedings...** IEEE, 2015. p. 45–49.

BECKER, K.; VOSS, S.; SCHÄTZ, B. Formal analysis of feature degradation in fault-tolerant automotive systems. **Science of Computer Programming**, Amsterdam, v. 154, p. 89–133, 2018.

BERBER, S. M. An automated method for BER characteristics measurement. **IEEE transactions on instrumentation and measurement**, New York, NY, v. 53, n. 2, p. 575–580, 2004.

BERTAGNOLLI, S. d. C. **FRIDA**: um método para elicitação e modelagem de rnfs. 2009. Tese (Doutorado em Ciência da Computação) — Universidade Federal do Rio Grande do Sul, Porto Alegre, 2009.

BIRCH, J. *et al.* Safety cases and their role in ISO 26262 functional safety assessment. *In*: INTERNATIONAL CONFERENCE ON COMPUTER SAFETY, RELIABILITY, AND SECURITY - SAFECOMP, 2013, Toulouse, France. **Proceedings...** Springer, 2013. p. 154–165.

BORTH, T. F. **Analisando os impactos do uso do protocolo CAN-FD em aplicações automotivas**: estudo de caso. 2016. Dissertação (Mestrado em Engenharia Elétrica) — Universidade Federal do Rio Grande do Sul, Porto Alegre, 2016.

BREED, G. Bit error rate: fundamental concepts and measurement issues. **High Frequency Electronics**, Mount Horeb, Wisconsin, USA, v. 2, n. 1, p. 46–47, 2003.

BROEKMAN, B.; NOTENBOOM, E. **Testing embedded software**. London, UK: Addison-Wesley:Pearson Education, 2003.

BROSTER, I.; BURNS, A.; RODRIGUEZ-NAVAS, G. Timing analysis of real-time communication under electromagnetic interference. **Real-Time Systems**, Berlin, v. 30, n. 1-2, p. 55–81, 2005.

BURNS, A.; DAVIS, R. **Mixed criticality systems – a review**. York, UK: Department of Computer Science, University of York, 2013. 1–69 p. (Tech Report).

CENA, G.; VALENZANO, A. Overclocking of controller area networks. **Electronics Letters**, New York, NY, v. 35, n. 22, p. 1923–1925, 1999.

CHITCHYAN, R. *et al.* **Survey of aspect-oriented analysis and design approaches**. Lancaster, UK: AOSD-Europe-ULANC-9, 2015.

CHUNG, L. *et al.* **Non-functional requirements in software engineering**. New York, USA: Springer Science & Business Media, 2012. v. 5.

CLARKE, S.; BANIASSAD, E. **Aspect-oriented analysis and design**. Boston, Massachusetts: Addison-Wesley Professional, 2005.

COLNARIC, M.; VERBER, D. **Distributed embedded control systems**: improving dependability with coherent design. London, UK: Springer Science & Business Media, 2007.

DARDAR, R. *et al.* Industrial experiences of building a safety case in compliance with ISO 26262. *In*: IEEE INTERNATIONAL SYMPOSIUM ON SOFTWARE RELIABILITY ENGINEERING WORKSHOPS (ISSREW), 23., 2012, Dallas, TX, USA. **Proceedings...** IEEE, 2012. p. 349–354.

DE ANDRADE, R. *et al.* Analytical and experimental performance evaluations of CAN-FD bus. **IEEE Access**, New York, NY, v. 6, p. 21287–21295, 2018.

DICK, J.; HULL, E.; JACKSON, K. **Requirements engineering**. 4th. ed. London, UK: Springer, 2017.

DO SOUTO, P. F.; PORTUGAL, P.; VASQUES, F. Reliability evaluation of broadcast protocols for flexRay. **IEEE Transactions on Vehicular Technology**, New York, NY, v. 65, n. 2, p. 525–541, 2016.

ELATTAR, Y.; METWALLI, S.; RABIE, M. PDF versus PID controller for active vehicle suspension. *In*: INTERNATIONAL AMME CONFERENCE, 17., 2016, Cairo, Egypt. **Proceedings...** AMME, 2016. v. 19, p. 21.

FLEXRAY CONSORTIUM, I. . **FlexRay communication systems protocol specification, version 3.0.1 - ISO 17458**. UK, 2010. 341 p.

FONTANA, M.; HUBING, T. H. Characterization of CAN network susceptibility to EFT transient noise. **IEEE Transactions on Electromagnetic Compatibility**, New York, NY, v. 57, n. 2, p. 188–194, 2015.

FREITAS, E. P. d. **Metodologia orientada a aspectos para a especificação de sistemas tempo-real embarcados distribuídos**. 2007. Dissertação (Mestrado em Ciência da Computação) — Universidade Federal do Rio Grande do Sul, Porto Alegre, 2007.

FREITAS, E. P. *et al.* Using aspect-oriented concepts in the requirements analysis of distributed real-time embedded systems. In: **Embedded system design**: topics, techniques and trends. Berlin: Springer, 2007. p. 221–230.

GALLOWAY, B.; HANCKE, G. P. Introduction to industrial control networks. **IEEE Communications surveys & tutorials**, New York, NY, v. 15, n. 2, p. 860–880, 2013.

GAO, Z.; CECATI, C.; DING, S. X. A survey of fault diagnosis and fault-tolerant techniques part i: fault diagnosis with model-based and signal-based approaches. **IEEE Transactions on Industrial Electronics**, New York, NY, v. 62, n. 6, p. 3757–3767, 2015.

GE, X.; YANG, F.; HAN, Q.-L. Distributed networked control systems: a brief overview. **Information Sciences**, Amsterdam, v. 380, p. 117–131, 2017.

GESSNER, D. *et al.* SfiCAN: a star-based physical fault-injection infrastructure for can networks. **IEEE Transactions on Vehicular Technology**, New York, NY, v. 63, n. 3, p. 1335–1349, 2014.

GIES, D. Transients and surge protection considerations in electrical equipment-offense and defense. *In*: IEEE SYMPOSIUM ON PRODUCT COMPLIANCE ENGINEERING (ISPCE), 2013, Austin, TX, USA. **Proceedings...** IEEE, 2013. p. 1–6.

GROOVER, M. P. **Automation, production systems, and computer-integrated manufacturing**. 3rd. ed. Upper Saddle River, NJ: Prentice Hall Press, 2007.

GUNES, V. *et al.* A survey on concepts, applications, and challenges in cyber-physical systems. **KSII Transactions on Internet & Information Systems**, USA, v. 8, n. 12, 2014.

HARTWICH, F. CAN with flexible data-rate. *In*: INTERNATIONAL CAN CONFERENCE, 13., 2012, Reutlingen, Germany. **Proceedings...** Citeseer - Robert Bosch GmbH, 2012. p. 1–9.

HERNANDEZ-ALCANTARA, D. *et al.* Modeling, diagnosis and estimation of actuator faults in vehicle suspensions. **Control Engineering Practice**, Amsterdam, v. 49, p. 173–186, 2016.

HOFFMANN, M. *et al.* Effectiveness of fault detection mechanisms in static and dynamic operating system designs. *In*: IEEE INTERNATIONAL SYMPOSIUM ON OBJECT/COMPONENT/SERVICE-ORIENTED REAL-TIME DISTRIBUTED COMPUTING (ISORC), 17., 2014, Reno, NV, USA. **Proceedings...** IEEE, 2014. p. 230–237.

HUANG, C. *et al.* Fault tolerant steer-by-wire systems: an overview. **Annual Reviews in Control**, Amsterdam, v. 47, p. 98–111, 2019.

HUANG, H.; LEU, K.-L.; CHEN, Y.-Y. An effective multiple-level fault-tolerant framework for flexRay network systems. *In*: INTERNATIONAL CONFERENCE ON CONNECTED VEHICLES AND EXPO (ICCVE), 2015, Shenzhen, China. **Proceedings...** IEEE, 2015. p. 54–55.

HUANG, S. *et al.* Transient fault tolerant control for vehicle brake-by-wire systems. **Reliability Engineering & System Safety**, Amsterdam, v. 149, p. 148–163, 2016.

HWANG, I. *et al.* A survey of fault detection, isolation, and reconfiguration methods. **IEEE transactions on control systems technology**, New York, NY, v. 18, n. 3, p. 636–653, 2010.

IEC-61000. **Electromagnetic compatibility (EMC) part 4-4, testing and measurement techniques – electrical fast transient, burst immunity test**. Geneva, CH: IEC – International Electrotechnical Commission, 2012. (Standard).

IEC-61025. **Fault tree analysis (FTA)**. Geneva, CH: IEC – International Electrotechnical Commission, 2006. (Standard).

IEC-TS-62228. **Integrated circuits – EMC evaluation of CAN transceivers**. Geneva, CH: IEC – International Electrotechnical Commission, 2007. (Standard).

IFAC-CONTROL, O. **International federation of automatic control**. Disponível em: <https://www.ifac-control.org/>. Acesso em: 2019.

IQBAL, M. Z. *et al.* Experiences of applying UML/MARTE on three industrial projects. *In*: INTERNATIONAL CONFERENCE ON MODEL DRIVEN ENGINEERING LANGUAGES AND SYSTEMS, 2012, Innsbruck, Austria. **Proceedings...** Springer, 2012. p. 642–658.

ISO-11898. **Road vehicles – Controller area network (CAN) – Part 1**: data link layer and physical signalling. Geneva, CH: ISO – International Organization for Standardization, 2015. (Standard).

ISO-17458. **Road vehicles – flexRay communications system**. Geneva, CH: ISO – International Organization for Standardization, 2013. (Standard).

ISO-26262. **Road vehicles – functional safety**. Geneva, CH: ISO – International Organization for Standardization, 2011. (Standard).

JIANG, J.; YU, X. Fault-tolerant control systems: a comparative study between active and passive approaches. **Annual Reviews in control**, Amsterdam, v. 36, n. 1, p. 60–72, 2012.

JUAN, R. O. S.; JEONG, M.-W.; KIM, H.-S. Development of burst error effect reduction algorithm for CAN using interleaver method. *In*: INTERNATIONAL SOC DESIGN CONFERENCE (ISOCC), 2016, Jeju, South Korea. **Proceedings...** IEEE, 2016. p. 165–166.

KABIR, S. An overview of fault tree analysis and its application in model based dependability analysis. **Expert Systems with Applications**, Amsterdam, v. 77, p. 114–135, 2017.

KATERIS, D. *et al.* A machine learning approach for the condition monitoring of rotating machinery. **Journal of Mechanical Science and Technology**, Berlin, v. 28, n. 1, p. 61–71, 2014.

KHAITAN, S. K.; MCCALLEY, J. D. Design techniques and applications of cyberphysical systems: a survey. **IEEE Systems Journal**, New York, NY, v. 9, n. 2, p. 350–365, 2015.

KICZALES, G. *et al.* Aspect-oriented programming. *In*: EUROPEAN CONFERENCE ON OBJECT-ORIENTED PROGRAMMING, 1997, Finland. **Proceedings...** Springer, 1997. p. 220–242.

KIENZLE, J. *et al.* Aspect-oriented design with reusable aspect models. In: **Transactions on aspect-oriented software development VII**. Berlin: Springer, 2010. p. 272–320.

KOBAYASHI, N. *et al.* Quantitative non functional requirements evaluation using softgoal weight. **J. Internet Serv. Inf. Secur.**, Republic of Korea, v. 6, n. 1, p. 37–46, 2016.

KOPETZ, H. **Real-time systems, design principles for distributed embedded applications**. New York: KLUWER Academic Publishers, 2002.

KOPETZ, H. **Real-time systems**: design principles for distributed embedded applications. Austria: Springer Science & Business Media, 2011.

LANGE, R. **Métodos de escalonamento de mensagens para o sistema de comunicação FlexRay**. 2015. Tese (Doutorado em Engenharia de Automação e Sistemas) — Universidade Federal de Santa Catarina, Florianópolis, 2015.

LEE, T.-Y. *et al.* A reliability scheduling algorithm for the static segment of flexRay on vehicle networks. **Sensors**, [S.l.], v. 18, n. 11, p. 3783, 2018.

LEE, Y. S.; KIM, J. H.; JEON, J. W. FlexRay and ethernet AVB synchronization for high QoS Automotive Gateway. **IEEE Transactions on Vehicular Technology**, New York, NY, v. 66, n. 7, p. 5737–5751, 2017.

LEEN, G.; HEFFERNAN, D. TTCAN: a new time-triggered controller area network. **Microprocessors and Microsystems**, Amsterdam, v. 26, n. 2, p. 77–94, 2002.

LIM, H. *et al.* Quantitative analysis of ringing in a controller area network with flexible data rate for reliable physical layer designs. **IEEE Transactions on Vehicular Technology**, New York, NY, p. 1–1, 2019.

LIN, C.-W. *et al.* Security-aware design methodology and optimization for automotive systems. **ACM Transactions on Design Automation of Electronic Systems (TODAES)**, New York, NY, v. 21, n. 1, p. 18, 2015.

LIU, B.; BAI, W.; ZHEN, G. A prompt retransmission method for in-vehicle network flexRay. *In*: CHINESE CONTROL CONFERENCE (CCC), 36., 2017, Dalian, China. **Proceedings...** IEEE, 2017. p. 7841–7846.

LU, K.-L.; CHEN, Y.-Y.; HUANG, L.-R. FMEDA-Based fault injection and data analysis in compliance with ISO-26262. *In*: ANNUAL IEEE/IFIP INTERNATIONAL CONFERENCE ON DEPENDABLE SYSTEMS AND NETWORKS WORKSHOPS (DSN-W), 48., 2018, Luxembourg City, Luxembourg. **Proceedings...** IEEE, 2018. p. 275–278.

LU, W. *et al.* Early fault detection approach with deep architectures. **IEEE Transactions on Instrumentation and Measurement**, New York, NY, v. 67, n. 7, p. 1679–1689, 2018.

MACHER, G. *et al.* SAHARA: a security-aware hazard and risk analysis method. *In*: DESIGN, AUTOMATION & TEST IN EUROPE CONFERENCE & EXHIBITION, 2015, Grenoble, France. **Proceedings...** EDA Consortium, 2015. p. 621–624.

MAHAPATRO, A.; KHILAR, P. M. Fault diagnosis in wireless sensor networks: a survey. **IEEE Communications Surveys & Tutorials**, New York, NY, v. 15, n. 4, p. 2000–2026, 2013.

MARQUES, L. *et al.* Error recovery in time-triggered communication systems using servers. *In*: IEEE INTERNATIONAL SYMPOSIUM ONINDUSTRIAL EMBEDDED SYSTEMS (SIES), 8., 2013, Porto, Portugal. **Proceedings...** IEEE, 2013. p. 205–212.

MARQUES, L. *et al.* Comparing scheduling policies for a message transient error recovery server in a time-triggered setting. *In*: IEEE EMERGING TECHNOLOGY AND FACTORY AUTOMATION (ETFA), 2014, Barcelona, Spain. **Proceedings...** IEEE, 2014. p. 1–6.

MARQUES, L. *et al.* Efficient transient error recovery in flexRay using the dynamic segment. *In*: IEEE EMERGING TECHNOLOGY AND FACTORY AUTOMATION (ETFA), 2014, Barcelona, Spain. **Proceedings...** IEEE, 2014. p. 1–4.

MARY, G. I.; ALEX, Z. C.; JENKINS, L. Response time analysis of messages in controller area network: a review. **Journal of Computer Networks and Communications**, London, v. 2013, 2013.

MATSUSHIMA, S. *et al.* Trends of EMC standards for automotive network devices and communication quality of ethernet in relation to parameters of pulse disturbances. **IEEE Electromagnetic Compatibility Magazine**, New York, NY, v. 7, n. 1, p. 46–50, 2018.

MICHELIN, T. J. **Análise do impacto da comunicação via rede flexray em sistemas de controle**. 2014. Dissertação (Mestrado em em Engenharia Elétrica) — Universidade Federal do Rio Grande do Sul, Porto Alegre, 2014.

MO, H. *et al.* Modeling and analysis of the reliability of digital networked control systems considering networked degradations. **IEEE Transactions on Automation Science and Engineering**, New York, NY, v. 14, n. 3, p. 1491–1503, 2017.

NA, W.; DAO, N.-N.; CHO, S. Mitigating WiFi interference to improve throughput for in-vehicle infotainment networks. **IEEE Wireless Communications**, New York, NY, v. 23, n. 1, p. 22–28, 2016.

NAHAS, M.; PONT, M. J.; SHORT, M. Reducing message length variations in resource constrained embedded systems implemented using the controller area network (CAN) protocol. **Journal of Systems Architecture**, Amsterdam, v. 55, n. 6, p. 344–354, 2009.

NAKAMURA, M. *et al.* Testbeds of a hybrid-ARQ-based reliable communication for CAN in highly electromagnetic environments. *In*: IEEE INTERNATIONAL FUTURE ENERGY ELECTRONICS CONFERENCE (IFEEC), 2., 2015, Taipei, Taiwan. **Proceedings...** IEEE, 2015. p. 1–6.

NASRI, O. *et al.* Automotive decentralized diagnosis based on CAN real-time analysis. **Journal of Systems Architecture**, Amsterdam, v. 98, p. 249–258, 2019.

NAVET, N.; SIMONOT-LION, F. **In-vehicle communication networks – a historical perspective and review**. Luxembourg: University of Luxembourg, 2013.

NGUYEN, P. H.; KLEIN, J.; LE TRAON, Y. Model-driven security with a system of aspect-oriented security design patterns. *In*: WORKSHOP ON VIEW-BASED, ASPECT-ORIENTED AND ORTHOGRAPHIC SOFTWARE MODELLING, 2., 2014, York, United Kingdom. **Proceedings...** ACM, 2014. p. 51.

NGUYEN, T.-H.; CHEON, B. M.; JEON, J. W. CAN FD performance analysis for ECU re-programming using the CANoe. *In*: IEEE INTERNATIONAL SYMPOSIUM ON CONSUMER ELECTRONICS (ISCE 2014), 18., 2014, JeJu Island, South Korea. **Proceedings...** IEEE, 2014. p. 1–4.

NXP SEMICONDUCTORS, I. **FlexRay tranceiver TJA1080A**. Disponível em: <https://www.nxp.com/docs/en/data-sheet/TJA1080A.pdf>. Acesso em: 2019.

OETJENS, J.-H. *et al.* Safety evaluation of automotive electronics using virtual prototypes: state of the art and research challenges. *In*: ANNUAL DESIGN AUTOMATION CONFERENCE (DAC), 51., 2014, San Francisco, CA, USA. **Proceedings...** ACM, 2014. p. 1–6.

OMG, O. M. G. **OMG.**: the uml profile for modeling and analysis of real time and embedded system (marte). Disponível em: <http://www.omgmarte.org/>. Acesso em: 2019.

PANNILA, E.; EDIRISINGHE, M. Power system switching transients in passenger automobiles. *In*: INTERNATIONAL CONFERENCE ON INFORMATION AND AUTOMATION FOR SUSTAINABILITY (ICIAFS), 7., 2014, Colombo, Sri Lanka. **Proceedings...** IEEE, 2014. p. 1–6.

PATTANAIK, B.; CHANDRASEKARAN, S. Recovery and reliability prediction in fault tolerant automotive embedded system. *In*: INTERNATIONAL CONFERENCE ON EMERGING TRENDS IN ELECTRICAL ENGINEERING AND ENERGY MANAGEMENT (ICETEEEM), 2012, Chennai, India. **Proceedings...** IEEE, 2012. p. 257–262.

PATTANAIK, B.; CHANDRASEKARAN, S. Safety reliability enhancement in fault tolerant automotive embedded system. **International Journal of Innovative Technology and Exploring Engineering (IJITEE)**, Damkheda, India, v. 2, n. 2, p. 63–68, 2013.

PEREIRA, C. E.; CARRO, L. Distributed real-time embedded systems: recent advances, future trends and their impact on manufacturing plant control. **Annual Reviews in Control**, Amsterdam, v. 31, n. 1, p. 81–92, 2007.

PEREIRA, C. E.; NEUMANN, P. Industrial communication protocols. In: **Springer Handbook of Automation**. Berlin: Springer, 2009. p. 981–999.

PIPER, T. *et al.* Mitigating timing error propagation in mixed-criticality automotive systems. *In*: IEEE INTERNATIONAL SYMPOSIUM ON REAL-TIME DISTRIBUTED COMPUTING (ISORC), 18., 2015, Auckland, New Zealand. **Proceedings...** IEEE, 2015. p. 102–109.

POHREN, D. **Estudo do impacto de transientes elétricos em protocolos de comunicação em sistemas embarcados**. 2020. Dissertação (Mestrado em em Engenharia Elétrica) — Universidade Federal do Rio Grande do Sul, Porto Alegre, 2020.

POHREN, D. H. *et al.* An analysis of the impact of transient faults on the performance of the CAN-FD protocol. **IEEE Transactions on Industrial Electronics**, New York, NY, v. 1, p. 1–1, 2019.

POUSSOT-VASSAL, C. **Robust LPV multivariable automotive global chassis control**. 2008. Tese (Doutorado em Eletrônica, Eletrotécnica, Automação e Processamento de Sinais) — Institut National Polytechnique de Grenoble-INPG, Grenoble, 2008.

PRADHAN, D. K. **Fault-tolerant computer system design**. Upper Saddle River, NJ: Prentice-Hall, 1996.

PTOLEMAEUS, C. **System design, modeling, and simulation**: using ptolemy ii. USA: Ptolemy.org Berkeley, 2014. v. 1.

RIBEIRO, F. G. C. *et al.* **Modelagem de requisitos de software de tempo-real usando SysML e MARTE**. 2013. Dissertação (Mestrado em Ciência da Computação) — Universidade Federal de Uberlândia, Uberlândia, MG, 2013.

RODRIGUEZ-SÁNCHEZ, M. C. *et al.* An embedded systems course for engineering students using open-source platforms in wireless scenarios. **IEEE Transactions on Education**, New York, NY, v. 59, n. 4, p. 248–254, 2016.

ROQUE, A. d. S. *et al.* An approach to address safety as non-functional requirements in distributed vehicular control systems. **Journal of Control, Automation and Electrical Systems**, Berlin, p. 1–16, 2019.

ROQUE, A. S. *et al.* Communication analysis in CAN networks under EFT injection. *In*: IEEE INTERNATIONAL CONFERENCE ON AUTOMATICA (ICA-ACCA), 2016, Curico, Chile. **Proceedings...** IEEE, 2016. p. 1–6.

ROQUE, A. S. *et al.* EFT fault Impact analysis on performance of critical tasks in intravehicular networks. **IEEE Transactions on Electromagnetic Compatibility**, New York, NY, v. 59, n. 5, p. 1415–1423, 2017.

RUIJTERS, E.; STOELINGA, M. Fault tree analysis: a survey of the state-of-the-art in modeling, analysis and tools. **Computer science review**, Amsterdam, v. 15, p. 29–62, 2015.

SAHA, I.; ROY, S.; RAMESH, S. Formal verification of fault-tolerant startup algorithms for time-triggered architectures: a survey. **Proceedings of the IEEE**, New York, NY, v. 104, n. 5, p. 904–922, 2016.

SALDIVAR, A. A. F. *et al.* Industry 4.0 with cyber-physical integration: a design and manufacture perspective. *In*: INTERNATIONAL CONFERENCE ON AUTOMATION AND COMPUTING (ICAC), 21., 2015, Glasgow, UK. **Proceedings...** IEEE, 2015. p. 1–6.

SÁNCHEZ, P. *et al.* Model-driven development for early aspects. **Information and Software Technology**, Amsterdam, v. 52, n. 3, p. 249–273, 2010.

SEDAGHAT, Y.; MIREMADI, S. G. Classification of activated faults in the flexRay-based networks. **Journal of Electronic Testing**, Berlin, v. 26, n. 5, p. 535–547, 2010.

SHAH, M. B. N. *et al.* Error handling algorithm and probabilistic analysis under fault for CAN-based steer-by-wire system. **IEEE Transactions on Industrial Informatics**, New York, NY, v. 12, n. 3, p. 1017–1034, 2016.

SHIRAI, R.; SHIMIZU, T. Failure protection for controller area network against EMI emitted by buck converter. *In*: ANNUAL IEEE APPLIED POWER ELECTRONICS CONFERENCE AND EXPOSITION (APEC), 2019, Anaheim, CA, USA. **Proceedings...** IEEE, 2019. p. 644–649.

SOMMERVILLE, I. **Software engineering**. 9th. ed. Boston, Massachusetts: Addison-Wesley, Pearson Education, 2011.

SPRIGGS, J. **GSN-The goal structuring notation**: a structured approach to presenting arguments. London, UK: Springer Science & Business Media, 2012.

STALLINGS, W. **Data and computer communications**. New Jersey: Prentice Hall, 2007.

STANKOVIC, J. A. Real-time and embedded systems. **ACM Computing Surveys (CSUR)**, New York, NY, USA, v. 28, n. 1, p. 205–208, 1996.

STARON, M. Requirements engineering for automotive embedded systems. In: **Automotive Systems and Software Engineering**. Berlin: Springer, 2019. p. 11–28.

STRICKER, J. *et al.* Fault impact assessment for automotive smart power products in an electric power steering application. *In*: INTERNATIONAL SEMICONDUCTOR CONFERENCE (CAS), 2018, Sinaia, Romania. **Proceedings...** IEEE, 2018. p. 201–204.

SUBRAMANIAN, N.; ZALEWSKI, J. Quantitative assessment of safety and security of system architectures for cyberphysical systems using the NFR approach. **IEEE Systems Journal**, New York, NY, v. 10, n. 2, p. 397–409, 2016.

TAKARABE, E. W. **Sistemas de controle distribuídos em redes de comunicação.** 2009. Tese (Doutorado em Engenharia Mecânica) — POLI-Universidade de São Paulo, São Paulo, 2009.

TALBOT, S. C.; REN, S. Comparision of fieldbus systems can, ttcan, flexray and lin in passenger vehicles. *In*: IEEE INTERNATIONAL CONFERENCE ON DISTRIBUTED COMPUTING SYSTEMS WORKSHOPS, ICDCS WORKSHOPS, 29., 2009, Montreal, QC, Canada. **Proceedings...** IEEE, 2009. p. 26–31.

TANENBAUM, A. S.; VAN STEEN, M. **Distributed systems**: principles and paradigms. 2rd. ed. Upper Saddle River, NJ: Prentice-Hall, 2007.

TENRUH, M. *et al.* **Modelling, simulation, and performance analysis of a CAN FD system with SAE benchmark based message set**. Nuremberg, Germany: CAN-CIA.org, 2015.

TEXAS INSTRUMENTS, I. **Product information TMS570LS3137**. Disponível em: <http://www.ti.com/product/TMS570LS3137-EP>. Acesso em: 2019.

THEISSLER, A. Detecting known and unknown faults in automotive systems using ensemble-based anomaly detection. **Knowledge-Based Systems**, Amsterdam, v. 123, p. 163–173, 2017.

TUOHY, S. *et al.* Intra-vehicle networks: a review. **IEEE Transactions on Intelligent Transportation Systems**, New York, NY, v. 16, n. 2, p. 534–545, 2015.

URSACHI, C.; HELEREA, E. Immunity to electrical fast transient pulses of computer systems. *In*: INTERNATIONAL CONFERENCE ON APPLIED AND THEORETICAL ELECTRICITY (ICATE), 2014, Craiova, Romania. **Proceedings...** IEEE, 2014. p. 1–4.

VECTOR INFORMATIK, G. **Product information CANoe/CANAnalyser**. Disponível em: <https://www.vector.com/int/en/products/software/canoe/>. Acesso em: 2018.

VYAS, V.; VISHWAKARMA, R. G.; JHA, C. Integrate aspects with UML: aspect oriented use case model. *In*: INTERNATIONAL CONFERENCE ON PARALLEL, DISTRIBUTED AND GRID COMPUTING (PDGC), 4., 2016, Waknaghat, India. **Proceedings...** IEEE, 2016. p. 134–138.

VYATKIN, V. Software engineering in industrial automation: state-of-the-art review. **IEEE Transactions on Industrial Informatics**, New York, NY, v. 9, n. 3, p. 1234–1249, 2013.

WANG, H. *et al.* Sliding mode control for steer-by-wire systems with AC motors in road vehicles. **IEEE transactions on Industrial Electronics**, New York, NY, v. 61, n. 3, p. 1596–1611, 2014.

WASICEK, A.; DERLER, P.; LEE, E. A. Aspect-oriented modeling of attacks in automotive cyber-physical systems. *In*: ACM/EDAC/IEEE DESIGN AUTOMATION CONFERENCE (DAC), 51., 2014, San Francisco, CA, USA. **Proceedings...** IEEE, 2014. p. 1–6.

WEHRMEISTER, M. A. **An aspect-oriented model-driven engineering approach for distributed embedded real-time systems**. 2009. Tese (Doutorado em Ciência da Computação) — Universidade Federal do Rio Grande do Sul, Porto Alegre, 2009.

WEHRMEISTER, M. A.; PEREIRA, C. E.; RAMMIG, F. J. Aspect-oriented model-driven engineering for embedded systems applied to automation systems. **IEEE Transactions on Industrial Informatics**, New York, NY, v. 9, n. 4, p. 2373–2386, 2013.

WHITMORE, A.; AGARWAL, A.; DA XU, L. The internet of things: a survey of topics and trends. **Information Systems Frontiers**, Berlin, v. 17, n. 2, p. 261–274, 2015.

WIMMER, M. *et al.* A survey on UML-based aspect-oriented design modeling. **ACM Computing Surveys (CSUR)**, New York, NY, v. 43, n. 4, p. 28, 2011.

WOLF, W. **Computers as components - principles of embedded computing system design**. 2rd. ed. USA: Elsevier, 2008.

WOO, S. *et al.* A practical security architecture for in-vehicle CAN-FD. **IEEE Transactions on Intelligent Transportation Systems**, New York, NY, v. 17, n. 8, p. 2248–2261, 2016.

XIA, J. *et al.* Real-time and reliability analysis of time-triggered CAN-bus. **Chinese Journal of Aeronautics**, Amsterdam, v. 26, n. 1, p. 171–178, 2013.

YAMAMOTO, S. An approach for evaluating softgoals using weight. In: **Information and Communication Technology**. Daejeon, Korea: Springer, 2015. p. 203–212.

YU, P. *et al.* Application mobility in pervasive computing: a survey. **Pervasive and Mobile Computing**, Amsterdam, v. 9, n. 1, p. 2–17, 2013.

ZENG, W.; KHALID, M. A.; CHOWDHURY, S. In-vehicle networks outlook: achievements and challenges. **IEEE Communications Surveys & Tutorials**, New York, NY, v. 18, n. 3, p. 1552–1571, 2016.

ZENG, W.; KHALID, M.; CHOWDHURY, S. A qualitative comparison of flexRay and ethernet in vehicle networks. *In*: IEEE CANADIAN CONFERENCE ON ELECTRICAL AND COMPUTER ENGINEERING (CCECE), 28., 2015, Halifax, NS, Canada. **Proceedings...** IEEE, 2015. p. 571–576.

ZHANG, J. *et al.* Modeling injection of electrical fast transients into power and IO pins of ICs. **IEEE Transactions on Electromagnetic Compatibility**, New York, NY, v. 56, n. 6, p. 1576–1584, 2014.

APÊNDICE A CONJUNTO DE CHECKLISTS: NFR DO SISTEMA DE CONTROLE DE SUSPENSÃO ATIVA

Tabela A.1 – Checklist para os requisitos de Embarcados do sistema de controle de suspensão ativa. Legenda: R - Relevante, P - Prioridade

	R	P	Restrições/Descrição
Embarcados			
Área			
Existe restrição quanto à área (em silício ou placa de circuito integrado) ocupada por algum componente do sistema?			
Consumo de Potência			
Existe restrição quanto ao consumo de potência de algum elemento do sistema?			
Quanto ao consumo do sistema, existe necessidade de se inserir monitoramento ou controle?	X	Média	Alguns tipos de falhas podem gerar aumento no consumo de energia de componentes.
Energia Total			
Existe restrição quanto à energia disponibilizada ao sistema?	X	Média	Deve-se seguir as especificações de alimentação da industria.
Existe estimativa de quanto deve durar a energia disponibilizada ao sistema?			
Existem fontes alternativas de energia no sistema?			
Existe restrição referente a geração de calor pelo sistema (silhueta térmica)?			
Memória			
Existe restrição quanto ao uso da memória de armazenamento do sistema? (seja para dados ou programas)			
Existe limitação quanto ao uso da memória de execução no sistema?			

Tabela A.2 – Checklist para os requisitos de Tempo do sistema de controle de suspensão ativa. Legenda: R - Relevante, P - Prioridade

	R	P	Restrições/Descrição
Tempo			
Temporização			
Existem atividades ou amostragens periódicas?	X	Alta	Leitura dos sensores em períodos de 5 ms e processamento da Lei de Controle em seguida.
Existem atividades esporádicas?			
Existem atividades aperiódicas?			
Existe restrição quanto a latência para se efetivar o início da execução de alguma atividade no sistema?			
Existem instantes específicos de início/-fim para execução de atividades do sistema?			
Foi especificado algum tempo de pior caso para a execução de atividades do sistema? (ou ao menos existe preocupação com relação a esta propriedade?)	X	Alta	Foi definido uma tolerância para este projeto de 7 ms.
Precisão			
Existem atividades com flexibilidade no atendimento de seus requisitos temporais?	X	Alta	Não há flexibilidade.
Caso exista flexibilidade no atendimento de requisitos temporais, o sistema suporta retardo em alguma atividade temporizada?			
O sistema suporta variações no atendimento de requisitos temporais?			
Em uma situação de uso degradado do sistema, existe a possibilidade de se utilizar dados antigos?			
Existe a necessidade de algum tipo de controle sobre a validade dos dados utilizados em algum processo do sistema?	X	Alta	Obtenção dos dados sensoriais da planta dentro do prazo de processamente da Lei de Controle, caso contrário o sistema instabiliza.
Existe limite quanto a diferença entre o tempo lógico utilizado pelo sistema e o tempo físico gerado por componentes do sistema?			

Tabela A.3 – Checklist para os requisitos de Desempenho do sistema de controle de suspensão ativa. Legenda: R - Relevante, P - Prioridade

	R	P	Restrições/Descrição
Desempenho			
Vazão			
Existe limite quanto a ocupação da rede que o sistema pode usar para prover garantias de execução?	X	Alta	Limite de referência segue a relação da vazão com a carga da rede de acordo com o protocolo de comunicação utilizado. 30 % em CAN tradicional e 60% CAN-FD e FlexRay.
Existe alguma restrição quanto à captura ou armazenamento de dados?			
Existem pontos de convergência de dados?	X	Alta	Mensagens de controle (sensor e atuador) são enviadas e processadas na ECU Controlador.
Existem restrições importantes quanto à utilização da banda no sistema?			
Tempo de Resposta			
Existem limitações temporais para o retorno de respostas finais do sistema?	X	Alta	A função da suspensão ativa depende do acionamento de atuadores no tempo definido. Desta forma os limites temporais de resposta devem seguir os 5 ms previamente definidos.
Em caso de desempenho degradado (sobrecarga do sistema), existem respostas finais do sistema que podem ser penalizadas em detrimento de outras?	X	Alta	Sim. Deve-se definir limites de degradação com bases em estudos de falhas.
Existe a possibilidade de se distribuir tarefas para garantir o tempo de resposta de uma atividade do sistema?			

Tabela A.4 – Checklist para os requisitos de Distribuição do sistema de controle de suspensão ativa. Legenda: R - Relevante, P - Prioridade

	R	P	Restrições/Descrição
Distribuição			
Alocação de Tarefas			
Existem critérios espaciais que determinem à distribuição das tarefas do sistema?	X	Baixa	Critérios seguem as normas de instalação físicas.
Existindo pontos de convergência de dados, há possibilidade de redistribuição do tratamento destes dados?			
Estações Participantes			
As Estações Participantes tem processamento dedicado a uma determinada atividade?	X	Alta	Sim, cada ECU executa sua função especifica.
Em caso de sobrecarga do sistema, alguma estação pode substituir ou auxiliar no processamento de outra?	X	Alta	Não. Em caso de sobrecarga seguem-se os requisitos de falhas.
Existe restrição quanto a capacidade de processamento das Estações Participantes?	X	Alta	Devem seguir especificações técnicas da indústria automotiva.
Existe restrição espacial que empeça a instalação de alguma estação em algum ponto de atuação ou leitura do sistema?			
Comunicação			
Existe restrição quanto ao uso de alguma tecnologia de comunicação?	X	Alta	Protocolos certificados pela indústria automotiva.
A comunicação exige garantia de entrega de mensagem?	X	Alta	Deve possuir garantia de entrega das mensagens perióticas.
Existe restrição quanto ao tamanho dos dados transmitidos ou recebidos?			
Existe diferença (tamanho, prioridade, ...) entre tráfego de controle e de dados?			
É necessário tratamento que garanta a integridade dos dados que trafegam pelo sistema?			
Sincronização			
Existem recursos ou dados compartilhados?			
É necessário controle de concorrência sobre os dados compartilhados?			
Existe alguma política pré-estabelecida para acesso de recursos ou dados compartilhados?			
Existe hierarquia quanto ao acesso aos recursos ou dados compartilhados?			

Tabela A.5 – Checklist para os requisitos de Falhas do sistema de controle de suspensão ativa. Legenda: R - Relevante, P - Prioridade

	R	**P**	**Restrições/Descrição**
Falhas			
Tipo			
Existe um tipo de falha específico que pode degradar o sistema?	X	Alta	Transientes elétricos rápidos - EFT
Caso exista um tipo de falha conhecido, existem normas para testes e verificação dos distúrbios causados por tal falha?	X	Média	Sim. Testes seguindo o padrão de injeção de falhas definido nas normas IEC 62228, ISO 7637
Caso não exista uma norma especificada, existem métodos ou técnicas de stress na rede que pode levar a uma análise aproximada?	X	Média	Existem ferramentas de stress que aproximam o efeito, mas é recomendado seguir as normas.
Origem e Causas			
Existe origem ou causa conhecida do tipo de falha?	X	Média	Transientes gerados por comutação de energia em diferentes componentes espalhados pelo veículo. *Power System Switching Transients*
Existem formas possíveis de detecção deste tipo de falha?	X	Baixa	A adição de sensores ou chips com capacidade de tolerância e detecção é possível, mas os custos podem ser proibitivos.
Efeitos			
Existem rotinas de registro de degradação de desempenho da rede de comunicação?	X	Alta	Registro de de logs para análise posterior.
Existem métricas especificas de desempenho monitoradas em tempo real?	X	Alta	Sim. Carga de ocupação, Jitter e Jitter de pior caso.
Existem limites de tolerância definidos para estas métricas?	X	Alta	Sim. Com base em testes anteriores foi definida uma tolerância de 25% no jitter médio.
Existem ajustes na periodicidade de msg em caso de degradação de desempenho?			
Existem gatilhos de notificação em caso de distúrbios frequentes na comunicação?	X	Média	Sim. Mediante análise de *overhead* na comunicação.
No uso de mecanismos de hardware para detecção, existem rotinas de tratamento?			
Transversalidade			
Existe unidades de controle especificas afetadas?	X	Média	Não. Todas podem ser afetadas.
Existem mensagens críticas especificas afetadas?	X	Alta	Sim. LeiDeControle.

APÊNDICE B CONJUNTO DE TEMPLATES: NFR DO SISTEMA DE CONTROLE DE SUSPENSÃO ATIVA

Tabela B.1 – Template do NFR1 do Sistema de Controle de Suspensão Ativa

	Item	
Identificação	**Identificador**	NFR1
	Nome	Leitura Periódica de Mensagens de Sensoriamento
	Autor	Alexandre dos Santos Roque
Especificação	**Classificação**	Tempo/Temporização/Período
	Descrição	O sistema deve ser capaz de ler mensagens periódicas enviadas pela ECU Planta em amostragens de 5 ms.
	Casos de Uso Afetados	SuspensionSensing, ComputingControl
	Contexto	O correto processamento da Lei de Controle depende da leitura de dados periódicos dos sensores da planta, sendo essa amostragem um requisito crítico do sistema.
	Escopo	Parcial
Decisão/Evolução	**Prioridade**	Alta
	Situação	Finalizado

Tabela B.2 – Template do NFR2 do Sistema de Controle de Suspensão Ativa

	Item	
Identificação	**Identificador**	NFR2
	Nome	Envio periódico de mensagens de controle
	Autor	Alexandre dos Santos Roque
Especificação	**Classificação**	Tempo/Temporização/Período
	Descrição	O sistema deve ser capaz de processar as mensagens periódicas da ECU Planta de modo a evitar a instabilidade do sistema.
	Casos de Uso Afetados	SuspensionActuation, ComputingControl
	Contexto	Na ECU Controlador o sistema efetua a computação da Lei de Controle e envia a mensagem de ajuste dos atuadores assim que receber os conjunto de mensagens da planta, dentro do período de amostragem definido. Esse processamento deve seguir os limites da dinâmica física da tecnologia de atuação aplicada.
	Escopo	Parcial
Decisão/Evolução	**Prioridade**	Alta
	Situação	Finalizado

Tabela B.3 – Template do NFR3 do Sistema de Controle de Suspensão Ativa

	Item	
Identificação	**Identificador**	NFR3
	Nome	Tempo de leitura de mensagem de pior caso.
	Autor	Alexandre dos Santos Roque
Especificação	**Classificação**	Tempo/Temporização/WCET
	Descrição	O valor de tolerância definido é um exemplo para o limite de tempo de calculo de ajustes nos parâmetros de atuação. O valor de 7 ms foi definido com base em testes de atrasos gerados e em projetos de referência da área.
	Casos de Uso Afetados	SuspensionSensing, ComputingControl
	Contexto	O critério de pior caso definido leva em consideração projetos de controle anteriores. Os ajustes dessa temporização de atuação deve levar em consideração a tecnologia e dinâmica de atuação aplicada, devido a sua natureza multidisciplinar.
	Escopo	Parcial
Decisão/Evolução	**Prioridade**	Alta
	Situação	Finalizado

Tabela B.4 – Template do NFR4 do Sistema de Controle de Suspensão Ativa

	Item	
Identificação	**Identificador**	NFR4
	Nome	Prazo limite para a execução da malha de controle.
	Autor	Alexandre dos Santos Roque
Especificação	**Classificação**	Tempo/Temporização/Deadline
	Descrição	O funcionamento adequado do sistema de controle depende do cumprimento da periodicidade especificada.
	Casos de Uso Afetados	SuspensionSensing, SuspensionActuation, ComputingControl
	Contexto	Parâmetros das mensagens são especificados de acordo com a modelagem da dinâmica do sistema de controle. Caracterizado por ser um sistema crítico que influência a segurança e conforto do automóvel.
	Escopo	Global
Decisão/Evolução	**Prioridade**	Alta
	Situação	Finalizado

Tabela B.5 – Template do NFR5 do Sistema de Controle de Suspensão Ativa

	Item	
Identificação	**Identificador**	NFR5
	Nome	Validade dos dados.
	Autor	Alexandre dos Santos Roque
Especificação	**Classificação**	Tempo/Precisão/Utilidade
	Descrição	A validade dos dados obtidos das mensagens depende do correto funcionamento dos sensores e dos processos de verificação de erros usados pelo protocolo de comunicação.
	Casos de Uso Afetados	SuspensionSensing, SuspensionActuation, ComputingControl
	Contexto	A validação temporal e de integridade segue as políticas do protocolo de comunicação, auxiliada pelo mecanismo de diagnóstico de degradações agregado ao sistema de controle.
	Escopo	Global
Decisão/Evolução	**Prioridade**	Alta
	Situação	Finalizado

Tabela B.6 – Template do NFR6 do Sistema de Controle de Suspensão Ativa

	Item	Conflito com NFR14
Identificação	**Identificador**	NFR6
	Nome	Limitações e convergência de dados
	Autor	Alexandre dos Santos Roque
Especificação	**Classificação**	Desempenho/Vazão e Carga
	Descrição	Para prover as garantias temporais de aplicações críticas, limitações são impostas por meio do monitorando das variações de carga do barramento, levando em conta a quantidade de mensagens que trafegam no barramento. A convergência ocorre com a centralização do controle.
	Casos de Uso Afetados	SuspensionSensing, SuspensionActuation, ComputingControl
	Contexto	A vazão efetiva é calculada em função dos pacotes transmitidos com os pacotes recebidos sem erros. Adicionalmente a literatura de referência sugere a limitação de 30% de carga para CAN, podendo usar 60% para CAN-FD e FlexRay. / Seguir o NFR14
	Escopo	Global
Decisão/Evolução	**Prioridade**	Alta
	Situação	Revisado após conflito / Finalizado

Tabela B.7 – Template do NFR7 do Sistema de Controle de Suspensão Ativa

	Item	Conflito com NFR15
Identificação	**Identificador**	NFR7
	Nome	Tempo de resposta do sistema
	Autor	Alexandre dos Santos Roque
Especificação	**Classificação**	Desempenho/Tempo de Resposta
	Descrição	Para a obtenção o efeito de ajuste esperado na suspensão (isolar o chassi do veículo das perturbações do terreno), a computação da lei de controle e a atuação devem acontecer nos períodos definidos.
	Casos de Uso Afetados	SuspensionActuation, ComputingControl
	Contexto	Devem ser definidos parâmetros de tolerância para o tempo de resposta do sistema de controle. / Parâmetros devem seguir o NFR15.
	Escopo	Parcial
Decisão/Evolução	**Prioridade**	Alta
	Situação	Revisado após conflito / Finalizado

Tabela B.8 – Template do NFR8 do Sistema de Controle de Suspensão Ativa

	Item	
Identificação	**Identificador**	NFR8
	Nome	Acesso aos recursos do sistema
	Autor	Alexandre dos Santos Roque
Especificação	**Classificação**	Distribuição/Sincronização
	Descrição	O acesso aos dados é orientado pelos nós participantes da malha de controle. O sincronismo é realizado na ECU da planta, responsável por montar os pacotes de mensagens periódicas.
	Casos de Uso Afetados	SuspensionSensing, SuspensionActuation, ComputingControl
	Contexto	O acesso e controle das mensagens é feito por identificação dos nós participantes e identificação das mensagens.
	Escopo	Global
Decisão/Evolução	**Prioridade**	Alta
	Situação	Finalizado

Tabela B.9 – Template do NFR9 do Sistema de Controle de Suspensão Ativa

	Item	
Identificação	**Identificador**	NFR9
	Nome	Acesso a rede de comunicação
	Autor	Alexandre dos Santos Roque
Especificação	**Classificação**	Distribuição/Comunicação
	Descrição	Pela natureza das redes intra-veiculares o acesso a rede depende da topologia de interconexão dos nós definido no projeto. Neste estudo o acesso a rede é realizado na topologia de barramento e é especifico ao nós que compõem a malha de controle.
	Casos de Uso Afetados	SuspensionSensing, SuspensionActuation, ComputingControl
	Contexto	O protocolo usado para efetivar a comunicação do sistema deve garantir a integridade das mensagens e operar em limites que não interfiram nas garantias temporais requeridas pelo sistema de controle.
	Escopo	Global
Decisão/Evolução	**Prioridade**	Alta
	Situação	Finalizado

Tabela B.10 – Template do NFR10 do Sistema de Controle de Suspensão Ativa

	Item	
Identificação	**Identificador**	NFR10
	Nome	Acesso a memória interna
	Autor	Alexandre dos Santos Roque
Especificação	**Classificação**	Embarcados/Memória
	Descrição	O sistema de controle faz uso de buffers de memória interna restritos ao uso do protocolo de comunicação. O sistema deve usar memória restrita para registro de logs de comunicação, usadas para fins de diagnóstico e detecção de degradação de desempenho.
	Casos de Uso Afetados	ComputingControl, DiagnosticFaultObserver
	Contexto	O sistema de diagnóstico deve prever os limites de memória atuais das ECUs, ou usar recursos de memória externos dedicados.
	Escopo	Parcial
Decisão/Evolução	**Prioridade**	Alta
	Situação	Finalizado

Tabela B.11 – Template do NFR11 do Sistema de Controle de Suspensão Ativa

	Item	
Identificação	**Identificador**	NFR11
	Nome	Tipos de falhas e interferências
	Autor	Alexandre dos Santos Roque
Especificação	**Classificação**	Falhas/Tipo
	Descrição	A prevenção de efeitos de degradação causados por falhas transientes é tratada. Diferentes tipos de falhas pode afetar o desempenho do sistema de controle, neste estudo os EFTs são tratados.
	Casos de Uso Afetados	SuspensionSensing, SuspensionActuation, ComputingControl
	Contexto	O sistema de controle foi testado de acordo com as normas IEC 62228 e ISO 7637, que especificam o comportamento do tipo de falha.
	Escopo	Global
Decisão/Evolução	**Prioridade**	Alta
	Situação	Finalizado

Tabela B.12 – Template do NFR12 do Sistema de Controle de Suspensão Ativa

	Item	
Identificação	**Identificador**	NFR12
	Nome	Origem das falhas
	Autor	Alexandre dos Santos Roque
Especificação	**Classificação**	Falhas/Origem e Causas
	Descrição	O sistema de diagnóstico pode tratar causas especificas das falhas, pata tal a verificação da origem e possíveis causas são importantes.
	Casos de Uso Afetados	SuspensionSensing, SuspensionActuation, ComputingControl
	Contexto	No contexto deste estudo, a origem de falhas EFT é conhecida e estudada (*Power System Switching Transients*), porém não é a única fonte. Esta informação é apenas considerada para fins de estudo, não tendo um tratamento especifico no momento.
	Escopo	Global
Decisão/Evolução	**Prioridade**	Baixa
	Situação	Finalizado

Tabela B.13 – Template do NFR13 do Sistema de Controle de Suspensão Ativa

	Item	
Identificação	**Identificador**	NFR13
	Nome	Efeitos de degradação
	Autor	Alexandre dos Santos Roque
Especificação	**Classificação**	Falhas/Efeitos/Logs
	Descrição	Após análises de testes de injeção de falhas seguindo normas, o sistema de diagnóstico registra logs de variações na rede de comunicação de acordo parâmetros e métricas de desempenho definidas.
	Casos de Uso Afetados	SuspensionSensing, SuspensionActuation, ComputingControl
	Contexto	Neste estudo o sistema de diagnóstico agregado a rede de comunicação registra logs de um período especifico no tempo para fins de diagnóstico e notificação em tempo real.
	Escopo	Global
Decisão/Evolução	**Prioridade**	Alta
	Situação	Finalizado

Tabela B.14 – Template do NFR14 do Sistema de Controle de Suspensão Ativa

	Item	
Identificação	**Identificador**	NFR14
	Nome	Métricas para análise
	Autor	Alexandre dos Santos Roque
Especificação	**Classificação**	Falhas/Efeitos/Métricas
	Descrição	Especificação de métricas que podem mensurar o impacto de uma falha especifica, em diferentes níveis ou camadas.
	Casos de Uso Afetados	SuspensionSensing, SuspensionActuation, ComputingControl
	Contexto	Neste estudo o sistema de diagnóstico agregado a rede de comunicação adota como padrão as variações na vazão/carga da rede, bem como também o Jitter e Jitter de pior caso observados em testes prévios.
	Escopo	Global
Decisão/Evolução	**Prioridade**	Alta
	Situação	Finalizado

Tabela B.15 – Template do NFR15 do Sistema de Controle de Suspensão Ativa

	Item	
Identificação	**Identificador**	NFR15
	Nome	Limites de tolerância
	Autor	Alexandre dos Santos Roque
Especificação	**Classificação**	Falhas/Efeitos/Limites
	Descrição	Especificação de limites de tolerância nas métricas analisadas, para fins de disparo de gatilhos de detecção e diagnóstico, ou de registro de logs para diagnóstico.
	Casos de Uso Afetados	ComputingControl, DiagnosticFaultObserver
	Contexto	Neste estudo o sistema de diagnóstico monitora as métricas e registra logs. Limites de 10% na carga e 25% nas variações de Jitter são definidos com base nos testes. Podem ser enviadas mensagens específicas sobre esse diagnóstico pela rede, mas o projeto deve prever o *overhead* de comunicação e se existirá uma ECU dedicada a notificações de alto nível.
	Escopo	Global
Decisão/Evolução	**Prioridade**	Alta
	Situação	Finalizado

APÊNDICE C CÓDIGO FONTE CAPL DA IMPLEMENTAÇÃO DO SISTEMA DE CONTROLE DE SUSPENSÃO ATIVA (CAN E CAN-FD)

Neste apêndice são apresentados os códigos desenvolvidos na linguagem CAPL, dentro do software CANoe, especificamente para simular o comportamento da planta de controle do sistema de suspensão ativa. Esse código foi utilizado para a realização dos experimentos usando o hardware VN8910A, com os protocolos CAN e CAN-FD.

C.0.1 Código CAPL - Planta

```
1  // ——— Active Suspension – Plant Behavior ———
2  includes
3  {
4  }
5
6  variables
7  {
8          // Timers' intervals
9          // Interval for plant simulation
10         long timerSimInterval = 500000; // 0.5 CAN/CAN–FD
11         // Interval for message output
12         long timerMsgOutputInterval = 5000000; // 5 ms
13
14         // Future State x(k+1)
15         float fst0 = 0;
16         float fst1 = 0;
17         float fst2 = 0;
18         float fst3 = 0;
19
20         // Plant States x(k)
21         float st0 = 0;
22         float st1 = 0;
23         float st2 = 0;
24         float st3 = 0;
25
```

```
26          // Plant Inputs
27          float u0 = 0;    // Disturbance
28          float u1 = 0;    // Control Input
29
30          // Messages
31          message Msg_Susp_Deflection suspension_deflection;
32          message Msg_Body_Vert_Speed body_vert_speed;
33          message Msg_Tire_Deflection tire_deflection;
34          message Msg_Susp_Set_Vert_Speed suspension_set_vert_speed;
35
36          // Timer for Simulation
37          timer t_plant;
38          // Timer for Message output
39          timer t_output;
40  }
41
42  on start
43  {
44          setTimer(t_plant, 0, timerSimInterval);
45          setTimer(t_output,0, timerMsgOutputInterval); // 5 ms
46  }
47
48  on timer t_plant
49  {
50          u0 = getValue(env_ext_disturbance_part);
51          st0 = fst0;
52          st1 = fst1;
53          st2 = fst2;
54          st3 = fst3;
55
56          fst0 = (0.999996)*st0 + (9.98508e-05)*st1 + (2.79578e-05)*st2 +
57          (-9.98494e-05)*st3 + (-1.39789e-09)*u0 + (1.48982e-10)*u1;
58          fst1 = (-0.00935111)*st0 + (0.999683)*st1 + (-8.88859e-05)*st2 +
59          (0.00031745)*st3 + (4.4443e-09)*u0 + (3.16987e-07)*u1;
60          fst2 = (3.92741e-06)*st0 + (1.33329e-07)*st1 + (0.999972)*st2 +
61          (9.98653e-05)*st3 + (-9.99986e-05)*u0 + (-1.33133e-10)*u1;
62          fst3 = (0.0785482)*st0 + (0.00266658)*st1 + (-0.559246)*st2 +
63          (0.997305)*st3 + (2.79623e-05)*u0 + (-2.66265e-06)*u1;
64
65          setTimer(t_plant, 0, timerSimInterval);
66          putValue(env_suspension_deflection, st0);
67          putValue(env_vertical_speed, st1);
68          putValue(env_tire_deflection, st2);
69          putValue(env_susp_set_vert_speed, st3);
70          putValue(env_ext_disturbance, u0);
71  }
72
```

```
73 on timer t_output
74 {
75         // Defleccao da suspensao
76         suspension_deflection.Susp_Deflection = st0;
77         // Velocidade vertical do corpo do veiculo
78         body_vert_speed.Body_Vert_Speed = st1;
79         // defleccao do pneu
80         tire_deflection.Tire_Deflection = st2;
81         // velocidade vertical da roda do veiculo
82         suspension_set_vert_speed.Susp_Set_Vert_Speed = st3;
83 
84         output(suspension_deflection);
85         output(body_vert_speed);
86         output(tire_deflection);
87         output(suspension_set_vert_speed);
88         setTimer(t_output, 0, timerMsgOutputInterval);
89 }
90 
91 on message Msg_Control_Law
92 {
93         u1 = this.Control_Law;
94 }
```

C.0.2 Código CAPL - Controle

```
1 
2 // ---- Active Suspension - Control Law -----
3 includes
4 {
5 }
6 
7 variables
8 {
9         // Controlador
10        // Realimentação de estados baseada na técnica de dados amostrados
11        float k0 = 2612.11380774278;
12        float k1 = -2124.70716737742;
13        float k2 = 1017.44264662319;
14        float k3 = -4.58946148568921;
15 
16        float st0 = 0;
17        float st1 = 0;
18        float st2 = 0;
19        float st3 = 0;
20 
21        message Msg_Control_Law control_law;
22 }
23 
```

```
24 on message Msg_Susp_Deflection
25 {
26         st0 = this.Susp_Deflection;
27 }
28
29 on message Msg_Body_Vert_Speed
30 {
31         st1 = this.Body_Vert_Speed;
32 }
33
34 on message Msg_Tire_Deflection
35 {
36         st2 = this.Tire_Deflection;
37 }
38
39 on message Msg_Susp_Set_Vert_Speed
40 {
41         st3 = this.Susp_Set_Vert_Speed;
42         control_law.Control_Law = k0*st0 + k1*st1 + k2*st2 + k3*st3;
43         output(control_law);
44 }
```

APÊNDICE D CÓDIGO FONTE CAPL DA IMPLEMENTAÇÃO DO MECANISMO DE DIAGNÓSTICO

```
1  // ECU FaultObserver implementation ----------------------
2  // Alexandre Roque-2019 ----------------------------------
3  // University of Stuttgart and Vector Informatik GmbH ---
4
5  includes
6  {
7  }
8
9  variables
10 {
11         double timer_A, timer_B, timer_S, timer_E, timer_el, difA, difT=0;
12         double arrayDif[900000], arrayAvgSimple[900000];
13         double avgSimple,sumJ=0;
14         // average simple and sum of time between messages
15         double avgJit, avgJitAnt, tempDouble;
16         double difJit, difJitAnt, bt=10000000,wt=0;
17         double tolerance1,tolerance2,WCET;
18         long nJit;     // number of jitter instant calculed
19         long nMsg=1;  // number of messages
20         long busl;
21         int i,tempInt, tempTime, counterTrig1, counterTrig2;
22
23         // ALARM MessagesFTObserver
24         message Msg_Jitter msg_jit;
25
26         float cl;      // Control Input
27         double pf, ptx, PLR=0, thtotal;
28         char logAvg[30];
29         dword handle, handle1;
30         message Msg_Control_Law control_law;
31         message Msg_Trigger_AvgJit trigAvgJit;
32 }
33
```

```
34 on stopMeasurement
35 {
36         // Measurement Time
37         timer_E = timeNowNS();
38         timer_el = (timer_E - timer_S)/1000000000; // in seconds
39
40         // Throughput at the enf of the measurement
41         thtotal = canGetStdData(1); // Channel CAN 1
42         thtotal = ((thtotal * 127) / timer_el) / 1000;
43         tempInt = thtotal;
44         filePutString("\nThroughput: ",14,handle);
45         ltoa(tempInt,logAvg,10);
46         filePutString(logAvg,elcount(logAvg),handle);
47         filePutString(".",elcount(logAvg),handle);
48         tempDouble = (thtotal - tempInt)*1000;
49         ltoa(tempDouble,logAvg,10);
50         filePutString(logAvg,elcount(logAvg),handle);
51         filePutString("Kbps ",10,handle);
52
53         // BUSLOAD
54         filePutString("\n - Busload: ",1,handle);
55         busl = canGetBusLoad(1);
56         ltoa(busl,logAvg,10);
57         filePutString(logAvg,elcount(logAvg),handle);
58         filePutString(" %",5,handle);
59
60         // Packet loss rate - PLR = Pf / Ptx
61         pf = canGetErrorCount(1);  // number of CAN-FRAME Errors
62         ptx = canGetStdData(1);    // number of total CAN frames
63         if(pf>0)
64                 PLR = (pf / ptx)*1000; // Repres 10^-3
65         filePutString("\nPf: ",7,handle);
66         ltoa(pf,logAvg,10);
67         filePutString(logAvg,elcount(logAvg),handle);
68         filePutString(" - Ptx: ",7,handle);
69         ltoa(ptx,logAvg,10);
70         filePutString(logAvg,elcount(logAvg),handle);
71
72         filePutString("\nPLR: ",7,handle);
73         PLR = PLR + 1; // part int
74         filePutString("0.",2,handle);
75         tempDouble = (1 - PLR); // part flt
76         ltoa(tempDouble,logAvg,10);
77         filePutString(logAvg,elcount(logAvg),handle);
78         // Qtd Events Trigger 1 - Dif. Jitter Oscilation
79         filePutString("\nTrig1: ",9,handle);
80         ltoa(counterTrig1,logAvg,10);
```

```
81          filePutString(logAvg,elcount(logAvg),handle);
82          // Qtd Events Trigger 2 - Avg. Jitter Oscilation
83          filePutString("\nTrig2:␣",9,handle);
84          ltoa(counterTrig2,logAvg,10);
85          filePutString(logAvg,elcount(logAvg),handle);
86  }
87
88  on Start
89  {
90          // Log Files according to triggers, messages and ECUs
91          timer_S = timeNowNS();
92          handle = openFileWrite("log-jitter.asc",0);
93          handle1 = openFileWrite("log-trigger1.asc",0);
94  }
95
96  on message Msg_Control_Law
97  {
98          cl= this.Control_Law;
99          timer_A = timeNowNS();  // time in nanoseconds
100
101 // --- CALCULO EM TEMPO REAL DAS METRICAS----------------------
102 //------------------------------------------------------------
103         if(timer_B==0){
104                 timer_B = timer_A - 5000000 ;   // 5 ms start value
105                 arrayDif[0] = 5000;
106         }
107         arrayDif[nMsg] = timer_A - timer_B; // dif time between messages
108
109         ltoa(arrayDif[nMsg],logAvg,10);
110         filePutString(logAvg,elcount(logAvg),handle);
111         filePutString("␣;␣",elcount(logAvg),handle);
112         filePutString(logAvg,elcount(logAvg),handle1);
113         filePutString("␣;␣",elcount(logAvg),handle1);
114
115         arrayDif[nMsg] = arrayDif[nMsg] / 1000;  // reduce the range to microseconds
116         difT = difT+arrayDif[nMsg];              // sum of these dif time
117
118         if(arrayDif[nMsg] > arrayDif[nMsg-1])
119                 difJit = arrayDif[nMsg] - arrayDif[nMsg-1];
120         else
121                 difJit = arrayDif[nMsg-1] - arrayDif[nMsg]; // Difference Jitter
122         @sysvar::FTObserver::sysDifJitter = difJit;
123
124         avgSimple = difT/nMsg;       // instant/window simple cycle average
125         if(arrayDif[nMsg] > wt)
126                 wt = arrayDif[nMsg];
127         if(arrayDif[nMsg] < bt)
```

```
128                 bt = arrayDif[nMsg];
129         WCET = wt - bt;
130
131         sumJ = 0;
132         for(i=1;i<=nMsg;i++)
133                 sumJ = sumJ + _pow((arrayDif[i]-avgSimple),2);
134         avgJit = sqrt(sumJ/(nMsg)); // instant average Jitter
135         @sysvar::FTObserver::sysAvgJitter = avgJit;
136 //-------------------------------------------------------------
137         ltoa(difJit,logAvg,10);
138         filePutString(logAvg,elcount(logAvg),handle);
139         filePutString("␣;␣",elcount(logAvg),handle);
140
141         tempInt = avgJit; // part int
142         ltoa(tempInt,logAvg,10);
143         filePutString(logAvg,elcount(logAvg),handle);
144         filePutString(".",elcount(logAvg),handle);
145         tempDouble = (avgJit - tempInt)*1000; // part flt
146         ltoa(tempDouble,logAvg,10);
147         filePutString(logAvg,elcount(logAvg),handle);
148
149         filePutString("␣;␣",elcount(logAvg),handle);
150         ltoa(WCET,logAvg,10);
151         filePutString(logAvg,elcount(logAvg),handle);
152
153 // Trigger 1 ------------------------------------------------
154         tolerance1 = 250; // Parameters for analysis
155         // DifJit  - bus 12% - CAN: 250 us / CAN-FD: 190 / FlexRay: 80
156         //         - bus 30% - CAN: 800 us / CAN-FD: 300 / FlexRay: 80
157         //         - bus 50% - CAN: 2000us / CAN-FD: 1060/ FlexRay: -
158         //         - bus 80% - CAN: 5000us / CAN-FD: 5000/ FlexRay: -
159         if(difJit > tolerance1) {
160                 filePutString("␣;␣1",4,handle);
161                 if(@sysvar::FTObserver::sysTriggerDifJit==0){
162                         counterTrig1++;
163                         @sysvar::FTObserver::sysTriggerDifJit = 1;
164                         //output(trigAvgJit);
165                         // Message for diagnostic
166                 }else
167                         @sysvar::FTObserver::sysTriggerDifJit = 0;
168                 difJitAnt = difJit;
169         }else{
170                 filePutString("␣;␣0",4,handle);
171                 difJitAnt = difJit;
172                 @sysvar::FTObserver::sysTriggerDifJit = 0;
173         }
174 //-------------------------------------------------------------
```

```
175 // Trigger 2 ------------------------------------------------------
176         tolerance2 = 42;
177         // Avg  - bus 12% - CAN: 42 us  / CAN-FD: 25  / FlexRay: 32
178         //      - bus 30% - CAN: 90 us  / CAN-FD: 62  / FlexRay: 32
179         //      - bus 50% - CAN: 294 us / CAN-FD: 120 / FlexRay: -
180         //      - bus 80% - CAN: 850 us / CAN-FD: 842 / FlexRay: -
181         if(avgJit > tolerance2){
182                 filePutString("␣;␣1",4,handle);
183                 if(@sysvar::FTObserver::sysTriggerAvgJit==0){
184                         counterTrig2++;
185                         output(trigAvgJit);    // Message for diagnostic
186                 }
187                 @sysvar::FTObserver::sysTriggerAvgJit = 1;
188                 avgJitAnt = avgJit;
189         }else{
190                 filePutString("␣;␣0",4,handle);
191                 avgJitAnt = avgJit;
192                 @sysvar::FTObserver::sysTriggerAvgJit = 0;
193         }
194
195         filePutString("\n",1,handle);
196 //-------------------------------------------------------------------
197         // sample of calculation - time window - FIFO array
198         if(nMsg==10000){
199                 difT = difT - arrayDif[1];
200                 for(i=1 ; i < 10000 ; i++)
201                         arrayDif[i] = arrayDif[i+1];
202         }else
203                 nMsg++;
204         timer_B = timer_A;
205         // update timer_B with the time of previous message
206 }
```

APÊNDICE E CÓDIGO FONTE CAPL DA IMPLEMENTAÇÃO COM A FERRAMENTA VECTOR VH6501

Este apêndice apresenta os códigos desenvolvidos para a realização dos distúrbios na rede CAN e CAN-FD com a ferramenta VH6501, durante o estágio realizado na Vector e Universidade de Stuttgart, Alemanha. Distúrbios de acordo com os desktops software trigger, frame trigger e missing bit trigger.

E.0.1 Código desenvolvido para a primeira sequência de distúrbios

```
1  //--- SCRIPT CAPL Test 1 - Disturbance Software Trigger
2  //--- Desenvolvido por Alexandre Roque
3  //--- At Vector Informatik GmbH - 2019
4  includes
5  {
6  }
7
8  variables
9  {
10         long deviceID, indexTrig=0;
11         long timer_Interval = 5;
12         timer t_TriggerInterval;
13 }
14
15 On start
16 {
17         setTimer(t_TriggerInterval,timer_Interval);
18 }
19
20 on preStart
21 {
22         deviceID = @sysvar::CANDisturbanceInterface1::DeviceNo;
23 }
24
25 On timer t_TriggerInterval
26 {
```

```
27          CanDisturbanceSequence              sequence;
28          CanDisturbanceTriggerRepetitions    repetitions;
29          canDisturbanceFrameSequence         frameSequence;
30          // Disturb Msg Control Law
31          message 0xA                         msgFrameSeq;
32          long                                result;
33
34          switch(indexTrig){
35
36          case (0):
37  //-------------------------------------------------------------------
38  // First Trigger - Leads to a StuffBitError
39  //Clear sequence
40          sequence.Clear();
41
42  //configure a sequence one bit long and send a dominant bit
43          // 320 = 2 us - 160 = 1 us
44          result = sequence.AppendToSequence(160, 'd');
45          // 1 tick = 6.25 ns
46          if(result == 1)                                 // clock 160 Mhz
47          {
48                  result= canDisturbanceTriggerNow(deviceID, sequence);
49
50          if(result == 1)
51                  write("Software␣Trigger␣1␣(StuffBitError)␣is␣configured.");
52          else
53                  write("Error␣by␣configuration␣of␣trigger␣now␣%d", result);
54          }
55          indexTrig++;
56          break;
57  //-------------------------------------------------------------------
58          case (1):
59  //-------------------------------------------------------------------
60  // Second Trigger - Leads to a Multiple StuffBitErrors
61
62  //Clear sequence
63          sequence.Clear();
64
65          //configure a sequence one bit long and send a dominant bit
66          result = sequence.AppendToSequence(160, 'd'); // //# before 320
67
68          if(result == 1)
69          {
70          //Define two repetitions and one cycle with a 1 ms hold off time
71          repetitions.Cycles = 1;
72          repetitions.HoldOffCycles = 1;
73          repetitions.HoldOffRepetitions = 1;
```

```
74          repetitions.Repetitions = 5; //# before 2
75
76          result= canDisturbanceTriggerNow(deviceID, sequence, repetitions);
77
78          if(result == 1)
79                  write("Software Trigger 2 (Multiple StuffBitErrors) is configured.");
80          else
81                  write("Error by configuration of trigger now %d", result);
82          }
83          indexTrig++;
84          break;
85  //-------------------------------------------------------------------
86          case (2):
87  //-------------------------------------------------------------------
88  // Third Trigger
89  //set message to sequence
90          frameSequence.SetMessage(deviceID, msgFrameSeq);
91          result = canDisturbanceTriggerNow(deviceID, frameSequence); //output the seque
92          if(result == 1)
93                  write("Software Trigger 3 (Frame DLC=0) is configured.");
94          else
95                  write("Error by configuration of trigger now %d", result);
96
97          indexTrig++;
98          break;
99  //-------------------------------------------------------------------
100         case (3):
101 //-------------------------------------------------------------------
102 // Fourth Trigger
103 //set message to sequence
104         frameSequence.SetMessage(deviceID, msgFrameSeq);
105
106         //Define two repetitions and one cycle with a 1 ms hold off time
107         repetitions.Cycles = 1;
108         repetitions.HoldOffCycles = 1;
109         repetitions.HoldOffRepetitions = 1;
110         repetitions.Repetitions = 5; // # Before 2
111
112         //output the sequence twice
113         result = canDisturbanceTriggerNow(deviceID, frameSequence, repetitions);
114
115         if(result == 1)
116                 write("Software Trigger 4 (Frame DLC=0 Multiple) is configured.");
117         else
118                 write("Error by configuration of trigger now %d", result);
119
120         indexTrig=0;
```

```
121         break;
122 //-------------------------------------------------------------------------
123 }
124         setTimer(t_TriggerInterval,timer_Interval);
125 }
```

E.0.2 Código desenvolvido para a segunda sequência de distúrbios

```
1  //--- SCRIPT CAPL Test 2
2  //--- Disturbance Frame Trigger and Missing Bit Trigger
3  //--- Desenvolvido por Alexandre Roque
4  //--- At Vector Informatik GmbH - 2019
5  includes
6  {
7  }
8
9  variables
10 {
11         long deviceID, indexTrig=0;
12         long timer_Interval = 5;
13         timer t_TriggerInterval;
14 }
15
16 On start
17 {
18         setTimer(t_TriggerInterval,timer_Interval);
19 }
20
21 on preStart
22 {
23         deviceID = @sysvar::CANDisturbanceInterface1::DeviceNo;
24 }
25
26 On timer t_TriggerInterval
27 {
28         CanDisturbanceFrameTrigger        frameTrigger;
29         CanDisturbanceFrameSequence       frameSequence;
30         CanDisturbanceSequence            sequence;
31         CanDisturbanceTriggerRepetitions  repetitions;
32         long                              result;
33         message 0xA                       triggerMessage; // Msg Control Law
34         long                              validityMask;
35         message 0x200                     messageForSequence; // Bad Message
36         long flags;
37         flags = 0;
38
39         switch(indexTrig){
40
```

```
41          case (0):
42  //-------------------------------------------------------------------
43  // First Trigger - Leads to a Ack Slot Error
44  //clear the sequence
45          sequence.Clear();
46
47          //ID must standard ID and a CAN message must on the bus
48          validityMask = @sysvar::CanDisturbance::Enums::ValidityMaskFlags::IDBase
49          | @sysvar::CanDisturbance::Enums::ValidityMaskFlags::IDE
50          | @sysvar::CanDisturbance::Enums::ValidityMaskFlags::FDF ;
51
52          frameTrigger.SetMessage(triggerMessage, deviceID, validityMask);
53          //trigger position is the CRC delimiter
54          frameTrigger.TriggerFieldType =
55          @sysvar::CanDisturbance::Enums::FieldType::CRCDel;
56          frameTrigger.TriggerFieldOffset = 0;
57
58          //configure a sequence 160 FPGA ticks long and send a recessive bit
59          //at the Ack slot bit on the bus.
60          result = sequence.AppendToSequence(160, 'R');
61
62          //Configure the frame trigger and the sequence to the CANstress device
63          if(result == 1)
64          {
65                  result = canDisturbanceTriggerEnable(deviceID,
66                  frameTrigger, sequence);
67                  if(result == 1)
68                          write("Frame␣Trigger␣1␣(Ack␣Slot)␣is␣enabled");
69                  else
70                          write("Enable␣trigger␣error␣Result␣=%d", result);
71          }
72          indexTrig++;
73          break;
74  //-------------------------------------------------------------------
75          case (1):
76  //-------------------------------------------------------------------
77  // Second Trigger - Leads to a Ack Delimiter Error
78  //clear the sequence
79          sequence.Clear();
80
81          //configure the message should be triggered
82          //ID must extended ID and a CAN message must on the bus
83          validityMask = @sysvar::CanDisturbance::Enums::ValidityMaskFlags::IDBase
84          | @sysvar::CanDisturbance::Enums::ValidityMaskFlags::IDExtended
85          | @sysvar::CanDisturbance::Enums::ValidityMaskFlags::IDE
86          | @sysvar::CanDisturbance::Enums::ValidityMaskFlags::FDF;
87
```

```
88          frameTrigger.SetMessage(triggerMessage, deviceID, validityMask);
89          //trigger position is the CRC delimiter
90          frameTrigger.TriggerFieldType =
91          @sysvar::CanDisturbance::Enums::FieldType::AckSlot;
92          frameTrigger.TriggerFieldOffset = 0;
93
94          //configure a sequence 160 FPGA ticks long and send a dominant bit
95          //at the Ack Delimiter bit on the bus.
96          result = sequence.AppendToSequence(160, 'd');
97
98          //Define two repetitions and one cycle with a 1 ms hold off time
99          repetitions.Cycles = 1;
100         repetitions.HoldOffCycles = 1;
101         repetitions.HoldOffRepetitions = 0;
102         repetitions.Repetitions = 2;
103
104         //Configure the frame trigger and the sequence to the CANstress device
105         if(result == 1)
106         {
107         result = canDisturbanceTriggerEnable(deviceID,frameTrigger,sequence,repetition
108
109         if(result == 1)
110                 write("Frame␣Trigger␣2␣(Ack␣Delimiter)␣is␣enabled");
111         else
112                 write("Enable␣trigger␣error␣Result␣=%d", result);
113         }
114
115         indexTrig++;
116         break;
117 //-------------------------------------------------------------------
118         case (2):
119 //-------------------------------------------------------------------
120 // Third Trigger - Leads to a IFS Error
121 //clear the sequence
122         sequence.Clear();
123
124         //configure the message should be triggered
125         //ID must standard ID and a CAN message must on the bus
126         validityMask = @sysvar::CanDisturbance::Enums::ValidityMaskFlags::IDBase
127         | @sysvar::CanDisturbance::Enums::ValidityMaskFlags::IDE
128         | @sysvar::CanDisturbance::Enums::ValidityMaskFlags::FDF;
129
130         frameTrigger.SetMessage(triggerMessage, deviceID, validityMask);
131
132         //trigger position is the 2nd IFS bit
133         frameTrigger.TriggerFieldType =
134         @sysvar::CanDisturbance::Enums::FieldType::EndOfFrame;
```

```
135         frameTrigger.TriggerFieldOffset = 9;
136
137         //configure the frame sequence with the a message
138         frameSequence.SetMessage(deviceID, messageForSequence);
139
140         //Configure the frame trigger and the frame sequence to the CANstress device
141         result = canDisturbanceTriggerEnable(deviceID, frameTrigger, frameSequence);
142
143         if(result == 1)
144                 write("Frame␣Trigger␣3␣(IFS␣Once)␣is␣enabled");
145         else
146                 write("Trigger␣enabling␣error␣result␣=%d", result);
147
148         indexTrig++;
149         break;
150 //-------------------------------------------------------------------
151         case (3):
152 //-------------------------------------------------------------------
153 // Fourth Trigger - Leads to a IFS Error Multiple
154 //clear the sequence
155         sequence.Clear();
156
157         //configure the message should be triggered
158         //ID must extended ID and a CAN message must on the bus
159         validityMask = @sysvar::CanDisturbance::Enums::ValidityMaskFlags::IDBase
160         | @sysvar::CanDisturbance::Enums::ValidityMaskFlags::IDExtended
161         | @sysvar::CanDisturbance::Enums::ValidityMaskFlags::IDE
162         | @sysvar::CanDisturbance::Enums::ValidityMaskFlags::FDF;
163
164         frameTrigger.SetMessage(triggerMessage, deviceID, validityMask);
165
166         //trigger position is the 2nd IFS bit
167         frameTrigger.TriggerFieldType =
168         @sysvar::CanDisturbance::Enums::FieldType::EndOfFrame;
169         frameTrigger.TriggerFieldOffset = 9;
170
171         //configure the frame sequence with the a message
172         frameSequence.SetMessage(deviceID, messageForSequence);
173
174         //Define two repetitions and one cycle with a 1 ms hold off time
175         repetitions.Cycles = 1;
176         repetitions.HoldOffCycles = 1;
177         repetitions.HoldOffRepetitions = 1;
178         repetitions.Repetitions = 3;
179
180         //Configure the frame trigger and the sequence to the CANstress device
181         result = canDisturbanceTriggerEnable(deviceID, frameTrigger,
```

```
182         frameSequence, repetitions);
183         if(result == 1)
184                 write("Frame Trigger 4 (IFS Multiple) is enabled");
185         else
186                 write("Trigger enabling error result =%d", result);
187
188         indexTrig=0;
189         break;
190 //-------------------------------------------------------------------
191         }
192         setTimer(t_TriggerInterval,timer_Interval);
193 }
194
195
196 void ConfigureFrameTriggerForm3(long flags, long fieldType, long offset)
197 {
198         CanDisturbanceFrameTrigger        frameTrigger;
199         CanDisturbanceSequence            sequence;
200         CanDisturbanceTriggerRepetitions  repetitions;
201         long                              result;
202         long                              validityMask;
203         message 0xA                       triggerMessage;
204
205         //clear the sequence
206         sequence.Clear();
207         //configure the message should be triggered
208
209         //ID must extended ID and a CAN message must on the bus
210         validityMask = @sysvar::CanDisturbance::Enums::ValidityMaskFlags::IDBase
211         | @sysvar::CanDisturbance::Enums::ValidityMaskFlags::IDExtended
212         | @sysvar::CanDisturbance::Enums::ValidityMaskFlags::IDE
213         | @sysvar::CanDisturbance::Enums::ValidityMaskFlags::FDF;
214
215         frameTrigger.SetMessage(triggerMessage, deviceID, validityMask);
216         //trigger position is the CRC delimiter
217         frameTrigger.TriggerFieldType = fieldType;
218         frameTrigger.TriggerFieldOffset = offset;
219
220         //configure a sequence one bit long and send a dominant bit at the RTR
221         //bit on the bus.
222         result = sequence.AppendToSequence(160, 'd');
223
224         //Define two repetitions and one cycle with a 1 ms hold off time
225         repetitions.Cycles = 1;
226         repetitions.HoldOffCycles = 1;
227         repetitions.HoldOffRepetitions = 0;
228         repetitions.Repetitions = 3;
```

```
229
230         //Configure the frame trigger and the sequence to the CANstress device
231         if(result == 1)
232         {
233                 result = canDisturbanceTriggerEnable(deviceID, frameTrigger,
234                 sequence, repetitions, flags);
235                 if(result == 1)
236                         write("Missing Bit - Trigger 5 (CRC Delimiter) is enabled");
237                 else
238                         write("Enable trigger error Result =%d", result);
239         }
240 }
```

ANEXO A CÓDIGO FONTE DA IMPLEMENTAÇÃO FLEX-RAY

Este anexo apresenta o código fonte da implementação CAPL do sistema de controle de suspensão ativa, baseado na pesquisa de (Michelin, 2014).

A.0.1 Código CAPL - Planta

```
1  includes
2  {
3  }
4
5  variables
6  {
7          // Future State x(k+1)
8          float fst0 = 0;
9          float fst1 = 0;
10         float fst2 = 0;
11         float fst3 = 0;
12
13         // Plant States x(k)
14         float st0 = 0;
15         float st1 = 0;
16         float st2 = 0;
17         float st3 = 0;
18
19         // Plant Inputs
20         float u0 = 0;    // Disturbance
21         float u1 = 1;    // Control Input
22
23         // Output
24         float accel = 0;
25
26         // Protocol PDUs
27         frPDU FrPDUState1 pdustate1;
28         frPDU FrPDUState2 pdustate2;
29         frPDU FrPDUState3 pdustate3;
```

```
30          frPDU FrPDUState4 pdustate4;
31
32          // Timer for Simulation
33          timer t_plant;
34  }
35
36  on start
37  {
38          setTimer(t_plant,0,100000);
39  }
40
41  on preStart
42  {
43          frSetSendPDU(pdustate1);
44          frSetSendPDU(pdustate2);
45          frSetSendPDU(pdustate3);
46          frSetSendPDU(pdustate4);
47  }
48
49  on timer t_plant
50  {
51          u0 = getValue(dist_part1) + 0.000001;
52          st0 = fst0;
53          st1 = fst1;
54          st2 = fst2;
55          st3 = fst3;
56
57          fst0 = (0.999996)*st0 + (9.98508e-05)*st1 + (2.79578e-05)*st2 +
58                 (-9.98494e-05)*st3 + (-1.39789e-09)*u0 + (1.48982e-10)*u1;
59          fst1 = (-0.00935111)*st0 + (0.999683)*st1 + (-8.88859e-05)*st2 +
60                 (0.00031745)*st3 + (4.4443e-09)*u0 + (3.16987e-07)*u1;
61          fst2 = (3.92741e-06)*st0 + (1.33329e-07)*st1 + (0.999972)*st2 +
62                 (9.98653e-05)*st3 + (-9.99986e-05)*u0 + (-1.33133e-10)*u1;
63          fst3 = (0.0785482)*st0 + (0.00266658)*st1 + (-0.559246)*st2 +
64                 (0.997305)*st3 + (2.79623e-05)*u0 + (-2.66265e-06)*u1;
65          accel = (-93.6508)*st0 + (-3.1746)*st1 + (0)*st2 + (3.1746)*st3 +
66                  (0.0031746)*u1;
67
68          setTimer(t_plant,0,100000);
69
70          putValue(state1,st0);
71          putValue(state2,st1);
72          putValue(state3,st2);
73          putValue(state4,st3);
74          putValue(acc,accel);
75          putValue(disturbance,u0);
76  }
```

```
77
78 on frStartCycle *
79 {
80         pdustate1.xb = st0;
81         frUpdatePDU(pdustate1,1,-1);
82
83         pdustate2.dxb = st1;
84         frUpdatePDU(pdustate2,1,-1);
85
86         pdustate3.xw = st2;
87         frUpdatePDU(pdustate3,1,-1);
88
89         pdustate4.dxw = st3;
90         frUpdatePDU(pdustate4,1,-1);
91
92         putValue(tps, timeNowNS());
93 }
94
95 on frFrame FrControlLaw
96 {
97         u1 = this.cl;
98         putValue(tua, timeNowNS());
99         putValue(ctrl_msg_time, messageTimeNS(this));
100 }
```

A.0.2 Código CAPL - Controle

```
1  includes
2  {
3  }
4
5  variables
6  {
7          float k0 = 2612.11380774278;
8          float k1 = -2124.70716737742;
9          float k2 = 1017.44264662319;
10         float k3 = -4.58946148568921;
11
12         float st0 = 0;
13         float st1 = 0;
14         float st2 = 0;
15         float st3 = 0;
16
17         frPDU FrPDUControlLaw resp;
18 }
19
20 on preStart
21 {
```

```
22          frSetSendPDU(resp);
23  }
24
25  on frPDU FrPDUState1
26  {
27          st0 = this.xb;
28  }
29
30  on frPDU FrPDUState2
31  {
32          st1 = this.dxb;
33  }
34
35  on frPDU FrPDUState3
36  {
37          st2 = this.xw;
38  }
39
40  on frPDU FrPDUState4
41  {
42          st3 = this.dxw;
43          resp.c1 = k0*st0 + k1*st1 + k2*st2 + k3*st3;
44          frUpdatePDU(resp,1,-1);
45  }
```