UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE TECNOLOGIA E GEOCIÊNCIAS
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
CURSO DE GRADUAÇÃO EM ENGENHARIA ELÉTRICA

ALEXANDRE EINSTEIN FERREIRA DA SILVA

INTEGRAÇÃO DE SOFTWARE PARA TESTE E VALIDAÇÃO DE FUNÇÕES VEICULARES EM AMBIENTE HIL

Recife
2025

ALEXANDRE EINSTEIN FERREIRA DA SILVA

INTEGRAÇÃO DE SOFTWARE PARA TESTE E VALIDAÇÃO DE FUNÇÕES VEICULARES EM AMBIENTE HIL

Trabalho de Conclusão de Curso apresentado ao Departamento de Engenharia Elétrica da Universidade Federal de Pernambuco, como requisito parcial para obtenção do grau de Engenheiro Eletricista.

Orientador(a): Prof. Dr. Rafael Cavalcanti Neto

Coorientador: Prof. MSc. Valdemar Moreira Cavalcante Junior

Recife
2025

Ficha de identificação da obra elaborada pelo autor,
através do programa de geração automática do SIB/UFPE

Ferreira da Silva, Alexandre Einstein .
Integração de Software para Teste e Validação de Funções Veiculares em Ambiente HIL / Alexandre Einstein Ferreira da Silva. - Recife, 2025.
58 p. : il., tab.

Orientador(a): Rafael Cavalcanti Neto
Cooorientador(a): Valdemar Moreira Cavalcante Junior
Trabalho de Conclusão de Curso (Graduação) - Universidade Federal de Pernambuco, Centro de Tecnologia e Geociências, Engenharia Elétrica - Bacharelado, 2025.

1. Hardware-in-the-loop. 2. Validação de software. 3. Simulação em tempo Real. 4. CANalyzer. 5. Provetech:TA. 6. Motor de combustão interna. I. Cavalcanti Neto , Rafael . (Orientação). II. Cavalcante Junior , Valdemar Moreira . (Coorientação). IV. Título.

620 CDD (22.ed.)

ALEXANDRE EINSTEIN FERREIRA DA SILVA

Integração de Software para Teste e Validação de Funções Veiculares em Ambiente HIL

Trabalho de Conclusão de Curso apresentado ao Departamento de Engenharia Elétrica da Universidade Federal de Pernambuco, como requisito parcial para obtenção do grau de Engenheiro Eletricista.

Aprovado em: 08/04/2025

BANCA EXAMINADORA

_____
Prof. Dr. Rafael Cavalcanti Neto (Orientador)
Universidade Federal de Pernambuco

_____
Prof. MSc. Valdemar Moreira Cavalcante Junior (Coorientador)
Universidade Federal de Pernambuco

_____
Prof. Dr. Eduardo José Barbosa (Examinador Interno)
Universidade Federal de Pernambuco

_____
Prof. MSc. Ericles Mauricio Barbosa (Examinador Externo)
Universidade Federal de Pernambuco

“Toda a nossa ciência, comparada com a realidade, é primitiva e infantil – e, no entanto, é a coisa mais preciosa que temos.”
— Albert Einstein

AGRADECIMENTOS

Primeiramente, gostaria de dedicar meus mais sinceros agradecimentos ao meu falecido irmão Paulo Jefferson, que, mesmo não estando mais fisicamente presente, sempre foi uma fonte constante de apoio e inspiração. Em meu coração, sempre serás o melhor.

Agradeço também à minha família e à minha noiva Joyce, que sempre estiveram ao meu lado, me oferecendo amor, paciência e apoio incondicional durante toda essa jornada. Sem o suporte e a confiança de todos vocês, este trabalho e essa graduação como um todo não teria sido possível.

Ao professor aposentado Luiz Antônio Magnata da fonte, que abriu a minha mente durante a graduação e propiciou que eu não tivesse outros grandes obstáculos na graduação, devido ao nível de maturidade que obtive durante suas disciplinas.

Agradeço meu coorientador, Valdemar Cavalcante, que me guiou incessantemente por todo o decorrer desse trabalho, e à toda a banca de avaliação, Professores Ericles, Rafael e Eduardo.

E, por fim, a todos os amigos que fiz ao longo do curso de Engenharia Elétrica, como Pedro Paulo, Abelardo, Danilo Matheus e Hector Luís. Cada um de vocês contribuiu de forma única para minha formação e desenvolvimento não só como Engenheiro e sou grato pelas experiências compartilhadas, pelas risadas, pela colaboração e pelo apoio que obtive.

RESUMO

A crescente complexidade de sistemas embarcados veiculares exige métodos de validação rigorosos para garantir seu funcionamento seguro e eficiente. O *Hardware-in-the-Loop* (HIL) é uma abordagem amplamente utilizada na indústria automotiva, permitindo a simulação e teste de algoritmos de controle em tempo real sem a necessidade de um veículo físico. Este trabalho apresenta a aplicação da metodologia HIL para a validação de funções veiculares, utilizando um modelo de motor a combustão interna (Internal Combustion Engine – ICE) no software Provetech:TA, com análise de variáveis realizada no CANalyzer. O uso do HIL possibilitou a avaliação detalhada do desempenho de algoritmos de funções veiculares sob diferentes condições operacionais, permitindo a replicação de cenários complexos e a análise de respostas do sistema em tempo real. Os resultados demonstraram a confiabilidade da metodologia empregada, evidenciando que a abordagem HIL proporciona um ambiente seguro e eficiente para o desenvolvimento e validação de software automotivo, reduzindo custos e riscos associados a testes em veículos reais.

Palavras-chave: *Hardware-in-the-Loop*, validação de software, simulação em tempo real, CANalyzer, Provetech:TA, motor de combustão interna.

ABSTRACT

The increasing complexity of vehicular embedded systems requires rigorous validation methods to ensure their safe and efficient operation. Hardware-in-the-Loop (HIL) is an approach widely used in the automotive industry, allowing control algorithms to be simulated and tested in real time without the need for a physical vehicle. This work presents the application of the HIL methodology for validating vehicle functions, using an internal combustion engine (ICE) model in Provetech:TA software, with variable analysis carried out in CANalyzer. The use of HIL enabled detailed performance evaluation of vehicle function algorithms under different operating conditions, allowing complex scenarios to be replicated and system responses to be analyzed in real time. The results demonstrated the reliability of the methodology employed, showing that the HIL approach provides a safe and efficient environment for the development and validation of automotive software, reducing the costs and risks associated with testing on real vehicles.

Keywords: Hardware-in-the-Loop, software validation, real-time simulation, CANalyzer, Provetech:TA, Internal Combustion Engine.

LISTA DE ILUSTRAÇÕES

Figura 1 - Ilustração da divisão de papéis e artefatos utilizados na metodologia Scrum. 22
Figura 2 – Ilustração do ciclo V, juntando a divisão em cascata com a validação *bottom-up*. 24
Figura 3 – Ilustração da modelagem dos componentes em ambiente virtual para a realização do MIL. 28
Figura 4 – A montagem do SIL através dos modelos montados na etapa anterior. 29
Figura 5 – A realização da etapa do PIL, onde os dados modelados nas etapas anteriores são testados em performance. 30
Figura 6 – Ilustração de como é realizado um teste em HIL. 31
Figura 7 – Diversos ambientes de simulação dSPACE. 33
Figura 8 – Interface do software Provetech:TA. 39
Figura 9 – Interface gráfica do *software* CANalyzer. 40
Figura 10 – Hardware VN1640 para Leitura de CAN/LIN. 42
Figura 11 – Hardware ETAS/INCA para calibração e carregamento dos Sofwares na Controladora. 42
Figura 12 – Leitura CAN no Simulador HIL. 43
Figura 13 – Integração entre Provetech e CANalyzer por meio do VN1640. 43
Figura 14 – Configurações para a leitura de sinais da rede CAN no *software* CANAlyzer. 44
Figura 15 – Escolha do canal para a aquisição de sinais da rede CAN no software CANAlyzer 44
Figura 16 – Comportamento da Função Cruise Control desativada, vista pelo software CANalyzer. 46
Figura 17 – Comportamento da Função Cruise Control ativada, vista pelo software CANalyzer. 46
Figura 18 – Momento do teste realizado no software Provetech:TA no momento t2 (injeção nula). 47
Figura 19 – Interface de testes do software Provetech:TA realizando a função de *Cruise Control.* 47
Figura 20 – Realização do teste da função *Speed Limit* feita no Provetech:TA. 49

Figura 21 – Comportamento da função *Speed Limit* analizada pelo CANalyzer. 50
Figura 22 – Realização do teste da função *Start-Stop* feita no Provetech:TA. 51
Figura 23 – Comportamento da função Start-Stop analizada pelo CANalyzer. 52
Figura 24 – Realização do teste da função *Limp-Home* feita no Provetech:TA. 53
Figura 25 – Comportamento da função *Limp-Home* analizada pelo CANalyzer. 54

LISTA DE TABELAS

Tabela 1 – Comparativo das técnicas de desenvolvimentos de projetos. 27
Tabela 2 – Comparativo Detalhado das técnicas de validação *in-the-loop*. 32

LISTA DE ABREVIATURAS E SIGLAS

ADAS	*Advanced Driver Assistance Systems*
CAN	*Controller Area Network*
ECU	*Electronic Control Unit*
HIL	*Hardware-in-the-loop*
ICE	*Internal Combustion Engine*
MIL	*Model-in-the-loop*
PID	*Proportional-Integral-Derivative*
PIL	*Processor-in-the-loop*
PO	*Product Owner*
SIL	*Software-in-the-loop*
GPS	*Global Positioning System*

SUMÁRIO

1 INTRODUÇÃO 15
1.1 OBJETIVO GERAL 16
1.2 OBJETIVOS ESPECÍFICOS 16
1.3 ORGANIZAÇÃO DO TRABALHO 16
2 FUNDAMENTAÇÃO TEÓRICA 18
2.1 METODOLOGIAS DE DESENVOLVIMENTO DE PROJETOS UTILIZADOS NO SETOR AUTOMOTIVO 18
2.1.1 Metodologias Ágeis 19
2.1.2 Metodologia Scrum 20
2.1.3 O Ciclo V em Desenvolvimento de Projetos 23
2.2 FERRAMENTAS "*IN-THE-LOOP*" PARA TESTES DE VALIDAÇÃO 27
2.2.1 Model-in-the-Loop (MIL) 28
2.2.2 Software-in-the-Loop (SIL) 29
2.2.3 Processor-in-the-Loop (PIL) 30
2.2.4 Hardware-in-the-Loop (HIL) 31
2.3 DESENVOLVIMENTO E TESTE DE FUNÇÕES VEICULARES UTILIZANDO *HARDWARE-IN-THE-LOOP* 32
2.3.1 Cruise Control (Controle de Cruzeiro) 33
2.3.1.1 Testes em HIL do Sistema de Controle de Cruzeiro *34*
2.3.2 Sistema Start-Stop 34
2.3.2.1 Testes em HIL do Sistema Start-Stop *35*
2.3.3 Função Speed-Limit (Limitador de velocidade) 35
2.3.3.1 Testes em HIL do Sistema Limitador de Velocidade *36*
2.3.4 Função Limp-Home (Modo de Emergência) 36
2.3.4.1 Testes em HIL do Sistema de Modo de Emergência *37*
3 DESENVOLVIMENTO E RESULTADOS 39
3.1 INTEGRAÇÃO DE *SOFTWARES* AUTOMOTIVOS PARA VALIDAÇÃO DE CENÁRIOS 39
3.1.1 Uso do Modelo ICE no software Provetech:TA 40
3.1.2 Monitoramento e Análise com o CANalyzer 41
3.1.3 Integração e Análise de Resultados 41
3.2 RESULTADOS 44
3.2.1 Função de Controle de Velocidade (Cruise Control) 45
3.2.2 Função Limitadora de Velocidade (Speed Limit Control) 48
3.2.3 Função Start-Stop 50
3.2.4 Função Limp-Home 52
3.3 CONCLUSÕES PARCIAIS 54
4 CONCLUSÕES E PROPOSTAS DE CONTINUIDADE 56
4.1 TRABALHOS FUTUROS 57

REFERÊNCIAS **58**

1 INTRODUÇÃO

Os avanços contínuos na indústria automobilística têm impulsionado o desenvolvimento de sistemas cada vez mais complexos, resultando em melhorias significativas na eficiência, segurança e conforto dos veículos modernos. Para garantir a qualidade e a confiabilidade desses aspectos, torna-se essencial a adoção de métodos rigorosos de validação, garantindo conformidade com as normas de segurança vigentes e eficiência na implementação de novas tecnologias (GIETELINK, 2006; SCHUETTE, 2020).

Um dos métodos de validação mais amplamente utilizados na engenharia automotiva é o *Hardware-in-the-Loop* (HIL). Esse método permite a realização de testes de sistemas automotivos por meio de simulações em um ambiente controlado, recriando condições operacionais virtuais que refletem com alta fidelidade os cenários reais (ISERMANN, 2012). No HIL, os componentes físicos do sistema são interligados a modelos matemáticos detalhados, possibilitando a interação entre o hardware real e o ambiente simulado. Dessa forma, controladores, sensores e atuadores de um veículo podem ser avaliados sob condições similares às reais, sem a necessidade de um protótipo físico completo, reduzindo custos e tempo de desenvolvimento (GAO, 2019).

A crescente demanda por processos de desenvolvimento mais precisos, eficientes e ágeis impulsiona a adoção de métodos como o HIL, que se mostra fundamental frente aos desafios impostos por tendências emergentes, como a eletrificação da mobilidade e o desenvolvimento de veículos autônomos (RAJAMANI, 2011). A utilização dessa abordagem não apenas melhora a segurança e a confiabilidade dos sistemas automotivos, mas também acelera a inovação e a competitividade da indústria automobilística global. Diante desse contexto, este trabalho explora a aplicação do HIL no desenvolvimento de veículos, com ênfase na validação de seus sistemas, destacando suas vantagens e desafios na engenharia automotiva.

1.1 Objetivo Geral

Validar quatro funções automotivas críticas, sendo elas Start-Stop, Controle de Cruzeiro Adaptativo, Limitador de Velocidade e Modo de Emergência, utilizando a metodologia *Hardware-in-the-Loop* (HIL), com foco na análise de desempenho e confiabilidade.

1.2 Objetivos Específicos

- Implementar e configurar o ambiente de teste HIL, integrando os softwares utilizados para a simulação e validação de funções automotivas, com monitoramento e análise de variáveis;
- Realizar testes funcionais das funções automotivas, avaliando seu desempenho em diferentes cenários;
- Coletar e analisar dados das variáveis relevantes durante os testes no ambiente HIL;

1.3 Organização do Trabalho

Este trabalho está estruturado de forma a abordar desde os conceitos teóricos e metodológicos até os resultados obtidos e as conclusões finais. A organização é apresentada da seguinte forma:

No Capítulo 2 são abordadas as metodologias de desenvolvimento de projetos, com foco em suas aplicações no setor automotivo, e são introduzidos os conceitos de testes de validação, com destaque para o HIL e, sua relevância no desenvolvimento de sistemas automotivos.

No Capítulo 3, é descrita a metodologia adotada no trabalho, sendo apresentado em detalhes o ambiente de simulação HIL, o modelo ICE desenvolvido, e a integração com o software para monitoramento de variáveis. Ainda neste capítulo, são apresentados os resultados obtidos nos testes realizados utilizando o ambiente HIL. Cada função automotiva é analisada com base nas variáveis monitoradas e nos critérios estabelecidos. Os resultados são discutidos em termos de desempenho e confiabilidade.

No Capítulo 4, são apresentadas as conclusões do trabalho, com ênfase nos resultados obtidos e nas contribuições proporcionadas à área de estudo. Também são discutidas as limitações do estudo e propostas de sugestões para trabalhos futuros que visem ampliar a aplicação da metodologia HIL em sistemas automotivos.

2 FUNDAMENTAÇÃO TEÓRICA

No contexto do desenvolvimento de projetos, a escolha da metodologia adequada desempenha um papel fundamental para o sucesso do produto final. Entre as diversas abordagens existentes, as Metodologias Ágeis, o Scrum e o Ciclo V se destacam por suas características distintas e pela forma como estruturam os processos de criação, validação e entrega. Cada uma delas foi concebida para atender a diferentes demandas, sendo utilizadas em variados setores da indústria, desde o desenvolvimento de softwares até projetos de engenharia complexos.

No âmbito da engenharia automotiva, essas metodologias são frequentemente complementadas por tecnologias in-the-loop, como MIL (Model-in-the-Loop), SIL (Software-in-the-Loop) e HIL (Hardware-in-the-Loop), que possibilitam a validação progressiva de sistemas embarcados de forma segura, eficiente e escalável. Tais abordagens permitem que as funcionalidades veiculares — como controle de cruzeiro, sistemas de partida automática (start-stop), limitadores de velocidade e modos de emergência (limp-home) — sejam testadas de maneira antecipada e realista, contribuindo significativamente para a robustez e confiabilidade dos produtos desenvolvidos.

2.1 Metodologias de desenvolvimento de projetos utilizados no setor automotivo

No setor automotivo, o desenvolvimento de projetos exige metodologias que garantam eficiência, qualidade e capacidade de adaptação frente à complexidade dos sistemas embarcados. Dentre as abordagens mais utilizadas, destacam-se as Metodologias Ágeis, o Scrum e o Ciclo V de desenvolvimento, cada uma com suas particularidades em termos de organização, validação e entrega. A escolha entre essas metodologias depende do tipo de projeto, dos requisitos do cliente e do nível de controle necessário ao longo das etapas de desenvolvimento.

2.1.1 Metodologias Ágeis

O surgimento da ferramenta organizacional conhecida como Metodologia Ágil veio como uma resposta aos desafios frequentemente enfrentados por equipes de desenvolvimento de software, mais especificamente as que possuíam dificuldade em seguir o modelo em cascata (*waterfall)*, principalmente por conta da rigidez nas fases de desenvolvimento, a dificuldade de adaptação a mudanças de requisitos ao longo do projeto e o tempo prolongado até a entrega de resultados visíveis. Por exemplo, em projetos complexos, falhas detectadas apenas nas etapas finais comprometem significativamente prazos e custos, o que evidencia a limitação do modelo em contextos que exigem flexibilidade e ciclos de feedback mais curtos.

O Manifesto Ágil, publicado em 2001, trouxe às equipes de desenvolvimento de software em todo o mundo novos valores e princípios, que priorizam indivíduos e interações no lugar de processos e ferramentas, softwares funcionando sobre documentação abrangente, colaboração com clientes sobre negociação de contratos, e respostas a mudanças sobre um plano em andamento (HIGHSMITH, 2009). Esses princípios fortaleceram o desenvolvimento baseado em flexibilidade a ambientes dinâmicos, ao promover ciclos curtos de desenvolvimento, a possibilidade de realizar incrementos na entrega, e um grande foco na comunicação contínua.

A metodologia Ágil possui quatro principais características para o desenvolvimento de projetos, sendo elas (BECK, 2001):

- **Ciclos Iterativos e Incrementais**: Na metodologia, o trabalho é dividido em pequenas iterações, denominadas “*sprints*”, nas quais cada pedaço entrega uma funcionalidade do produto. Esta ação possibilita com que os ajustes no projeto sejam contínuos, até a entrega final do mesmo, e que esses ajustes sejam sempre baseados no feedback recebido.
- **Colaboração com o Cliente**: Com esta metodologia, os clientes participam ativamente do projeto, revisam entregas e fornecem as devidas orientações para as etapas seguintes do projeto.
- **Flexibilidade e Adaptação**: Quando ocorrem mudanças no projeto realizado usando a Metodologia Ágil, as mudanças a serem realizadas nos projetos são esperadas, e assim, incorporadas rapidamente, portanto garantindo que as reais necessidades do cliente sejam sanadas pelo produto final.

- **Auto-Organização da equipe**: A Metodologia Ágil promove a formação de equipes multidisciplinares que assumem responsabilidades por todas as etapas do desenvolvimento do projeto, desde o planejamento até a entrega. Essa auto-organização permite maior autonomia e engajamento dos membros, otimizando o fluxo de trabalho.

Essa metodologia pode ser utilizada para priorizar funções críticas do sistema a ser desenvolvido, entregando essas partes em *sprints* iniciais, e baseando-se no *feedback* de clientes, a equipe pode realizar ajustes antes de incorporar funcionalidades mais avançadas.

Em suma, essa metodologia traz benefícios como a redução de riscos, maior satisfação dos clientes e usuários, e um compromisso constante com a melhoria contínua. Por meio de entregas frequentes, os problemas são identificados e solucionados rapidamente, enquanto o envolvimento contínuo do cliente assegura que o produto final esteja alinhado às expectativas. Cada iteração representa uma oportunidade para aprendizado e refinamento do projeto, resultando em soluções mais eficazes e adaptadas às necessidades do mercado.

2.1.2 Metodologia Scrum

Outra metodologia amplamente conhecida e utilizada em todo o mundo, dentro das metodologias de desenvolvimento, é o Scrum. Ele pode ser definido como um *framework* leve, e adaptável para o desenvolvimento de produtos com maior complexidade. Ele se baseia em equipes auto-organizadas, e nos conceitos de inspeção e de adaptação (SCHWABER, 2017).

No trabalho, há uma divisão em *sprints*, que normalmente duram de 2 a 4 semanas, culminando na entrega de um incremento funcional para o projeto. Durante cada *sprint*, os papéis e eventos envolvidos são bem definidos na equipe, garantindo que o processo seja guiado de maneira clara e estruturada.

Segundo Schwaber, nessa metodologia, os papéis a ser assumidos por diversas partes da equipe podem ser listados por:

- ***Product Owner* (PO):** É o responsável por representar os interesses do cliente ou usuário, além de priorizar os requisitos no *Backlog* do produto.

- ***Scrum Master:*** Este papel é desempenhado por quem facilita o processo, gerencia a equipe, resolve impedimentos e garante que os princípios ágeis sejam seguidos.
- **Equipe de Desenvolvimento:** Se tornam responsáveis por desenvolver as entregas, incorporar *feedbacks* e fornecer novos incrementos de valor ao projeto durante cada *sprint*.

Ainda no Scrum, há uma divisão bem definida nos eventos realizados por todas as partes, sendo eles (SCHWABER, 2017):

- ***Sprint Planning*****:** Neste evento, os objetivos do *sprint* são estabelecidos pela equipe, que define a tarefa e os prazos para a conclusão de cada objetivo.
- ***Daily Scrum*****:** Consiste em uma reunião diária curta, de cerca de 15 minutos, para que o progresso de todas as partes seja alinhado, e possíveis impedimentos possam ser detectados previamente.
- ***Sprint Review*****:** Também consiste em uma reunião, contudo, esta é realizada ao fim do sprint, com o objetivo de apresentar ao cliente o que foi desenvolvido e coletar o feedback necessário para ajustes e melhorias.
- ***Sprint Retrospective***: É uma avaliação interna do desempenho da equipe no sprint. Neste evento, são listados os aspectos do projeto que funcionaram bem, assim como as dificuldades e desafios enfrentados, para uma melhora nos próximos sprints.

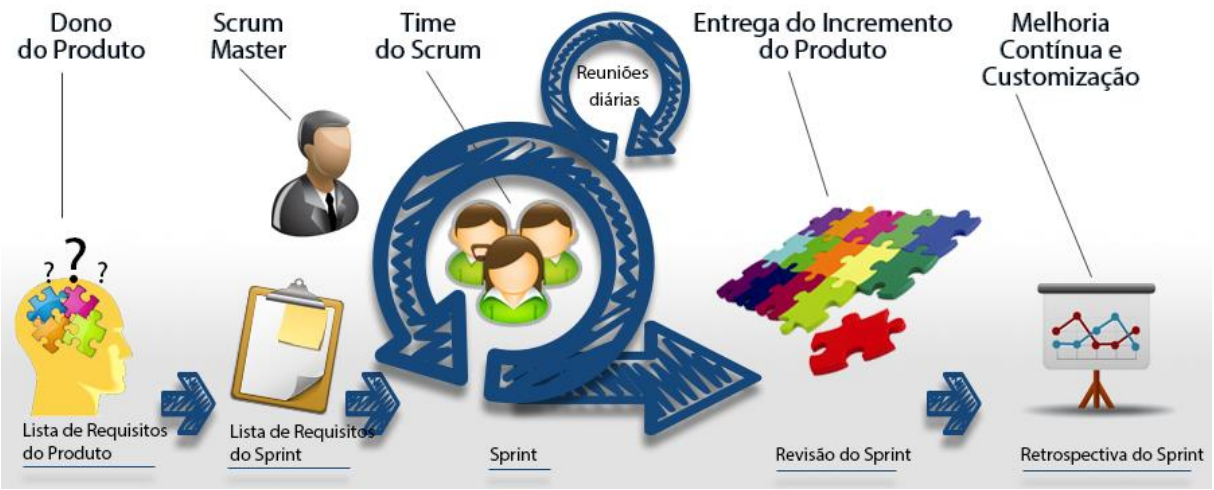
O Scrum também utiliza artefatos, definidos como documentos e informações que auxiliam as equipes de desenvolvimento, proporcionando uma visão mais clara do trabalho, melhor organização das prioridades, e maior qualidade do produto final. Os principais artefatos da metodologia são (SCHWABER, 2017):

- ***Product Backlog***: Uma lista dinâmica de requisitos priorizados pelo *Product Owner* (PO);
- ***Sprint Backlog*****:** Um conjunto de tarefas selecionadas para o atual *sprint*;
- ***Increment*****:** Um produto funcional que é entregue ao fim de cada *sprint*.

Basicamente, em uma equipe que adota o Scrum como metodologia para o desenvolvimento de seus produtos, o *Product Owner* é responsável por priorizar as funcionalidades e repassá-las à equipe de desenvolvimento por meio do *Product Backlog*. Nesse período, as tarefas definidas tornam-se prioritárias, com todo o time trabalhando exclusivamente nelas. Ao final do ciclo, é realizado um *feedback* com o

cliente, assim redefinindo requisitos, e retomando a dinâmica de priorizar funcionalidades, até a conclusão do projeto. O processo do Scrum pode ser ilustrado como visto na Figura 1.

Figura 1 - Ilustração da divisão de papéis e artefatos utilizados na metodologia Scrum.



Fonte: Adaptado de Schwaber (2025).

Entre os principais benefícios da adoção do Scrum, destacam-se:

- **Visibilidade Contínua:** A alta frequência de reuniões e entregas incrementais proporciona transparência sobre o progresso do projeto para todos os envolvidos.
- **Flexibilidade:** Ao longo do desenvolvimento, as prioridades podem ser ajustadas rapidamente, permitindo uma resposta eficaz a mudanças no mercado ou nos requisitos do cliente.
- **Foco no Valor:** A organização de prioridades garante que os itens de maior valor para o cliente sejam entregues primeiro.

2.1.3 O Ciclo V em Desenvolvimento de Projetos

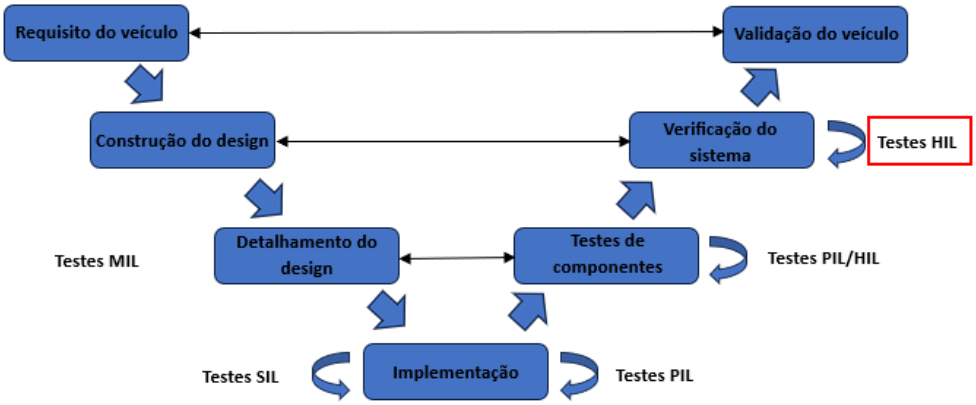
Por fim, outra metodologia amplamente utilizada no desenvolvimento de sistemas complexos, especialmente em áreas industriais, como a automotiva, ferroviária e aeroespacial, é o Ciclo V. Essa metodologia consiste em descrever as etapas do desenvolvimento e da validação do projeto através de etapas sequenciais, proporcionando maior clareza de informações sobre as inter-relações entre as fases de concepção, verificação e validação.

O modelo em V enfatiza a necessidade de uma validação contínua, garantindo que cada etapa de desenvolvimento seja acompanhada por atividades de verificação e testes correspondentes às mesmas (FORSBERG; MOOZ, 1991). O formato em V reflete a descida em cascata durante as etapas de design, e após chegar ao ponto mais baixo, segue um modelo *bottom-up* (subida) em suas etapas de validação.

A divisão do ciclo V consiste principalmente de duas partes. O lado esquerdo, que funciona de maneira em cascata, destinado ao desenvolvimento e especificação dos requisitos do projeto. Esta primeira fase é responsável por definir as análises a serem realizadas, e a realização da modelagem do design do projeto. Já o lado direito consiste num modelo em escalada, e é destinado às etapas de validação e verificação dos requisitos definidos na fase em cascata, concluindo o ciclo com a fase de testes e integração do projeto.

Cada etapa existente no lado esquerdo possui uma contraparte no lado direito, para assim garantir que todos os requisitos sejam atendidos. O centro do “V” é o momento em que o devido projeto migre de uma fase teórica (especificações), para uma fase prática (implementação). Uma representação gráfica das etapas do Ciclo V pode ser observada na Figura 2.

Figura 2 – Ilustração do ciclo V, juntando a divisão em cascata com a validação *bottom-up*.



Fonte: Adaptado de Santos (2018).

A estruturação em etapas, no Ciclo V, pode ser dividida da seguinte maneira:

- **Levantamento de requisitos do sistema:** Esta etapa corresponde ao topo do lado esquerdo, e consiste em definir o que será realizado no projeto. Os requisitos são discutidos em conjunto com as partes interessadas, procurando o maior nível de clareza e abrangência possível, para possuir o máximo de informações possíveis, com o objetivo de definir objetivamente quais serão os próximos passos. Em projetos automotivos, por exemplo, podem ser especificados aspectos como consumo de combustível, segurança do veículo, ou conectividade.
- **Requisitos do subsistema:** Nesta etapa, os requisitos do sistema principal são decompostos em subsistemas menores, reduzindo a complexidade geral do projeto. Essa abordagem facilita o desenvolvimento de soluções específicas para cada subsistema, que posteriormente serão integrados para formar o sistema completo.
- **Design do sistema:** Nesta etapa, é desenvolvida a arquitetura, definindo como cada subsistema, identificado na etapa anterior, irá interagir, deixando claro como será feita a integração de cada parte do projeto ao final da validação.
- **Design detalhado:** Cada subsistema que foi dividido do sistema principal então é projetado em detalhes, desenvolvendo-se em modelos matemáticos como a dinâmica entre componentes do sistema irá interagir com algoritmos desenvolvidos para os fins definidos na etapa de requisitos.

- **Implementação:** Este ponto representa o nível mais baixo do Ciclo V, onde a construção do sistema é iniciada. Nessa fase, são desenvolvidos protótipos e realizada a integração entre software e hardware. Com o modelo do projeto já definido, inicia-se então a fase de validação.
- **Testes de unidade:** Nesta etapa, os componentes são testados individualmente, como por exemplo, um teste de algoritmo, para a correta validação dos mesmos.
- **Testes de integração:** Após a realização da validação de cada unidade, realizada na etapa anterior, os subsistemas são integrados, como definido na etapa de design, e então são testados novamente, como um sistema completo dessa vez.
- **Validação do sistema:** Nessa etapa, o sistema é testado de forma integral, considerando todos os requisitos estabelecidos nas fases iniciais do projeto. Protótipos completos são produzidos e submetidos a testes rigorosos, em ambientes simulados a princípio, e também em situações reais, visto que fatores como variabilidades do ambiente, interferências físicas e comportamentos imprevisíveis de componentes só podem ser plenamente observados em testes práticos.
- **Entrega ao cliente:** Representando a etapa final e o ponto superior direito do Ciclo V, o produto final é entregue às partes interessadas, acompanhado de documentação detalhada e suporte técnico.

Uma série de benefícios pode ser notada com a adoção do Ciclo V em desenvolvimento de sistemas, sendo eles principalmente relacionados à:

- **Clareza e rastreabilidade:** Cada requisito possui um teste correspondente, garantindo uma completa cobertura durante todo o projeto.
- **Mitigação de riscos:** Os problemas no desenvolvimento do projeto são detectados e corrigidos em etapas iniciais, reduzindo custos relacionados a retrabalho.
- **Adaptação a regulamentações:** Facilidade do projeto se adequar a conformidades de normas de segurança, como o exemplo de adaptação à ISO 26262, no desenvolvimento de veículos automotivos.

O Ciclo V é amplamente adotado no setor automotivo devido à sua rigorosa abordagem e estruturação detalhada, sendo essencial para sistemas críticos de

segurança veicular. Um exemplo prático de sua aplicação é o desenvolvimento de veículos autônomos. O processo começa com o levantamento de requisitos, como a detecção de pedestres, a capacidade de manter a faixa e o sistema de frenagem automática (ISO 26262). Em seguida, o sistema de assistência à condução (ADAS) é decomposto em subsistemas menores, como visão computacional (câmeras), sensores e o controle eletrônico desses componentes. O design detalhado envolve, por exemplo, a integração de algoritmos de reconhecimento de objetos com os sensores do veículo.

Após a definição do design, os testes de unidade são realizados, como a validação dos algoritmos de detecção de pedestres utilizando bancos de dados simulados de imagens urbanas. Em seguida, os subsistemas, como sensores e câmeras, são integrados e testados em simulações virtuais, garantindo que suas funções combinadas atendam a todos os requisitos. Finalmente, na etapa de validação do sistema, o veículo protótipo é testado em condições reais, como a condução em cidades movimentadas e sob diferentes condições climáticas, para validar que o sistema cumpra os requisitos de segurança e desempenho.

É possível notar que, ao contrário de metodologias como a Ágil, e o Scrum, o Ciclo V apresenta limitações quando os requisitos de projeto necessitam de mudanças recorrentes, já que o mesmo trabalha em um formato de cascata, com definições fixas, e caso uma mudança súbita seja necessária, o ciclo V inteiro necessitaria ser refeito, para a redefinição dos sistemas a serem testados, e a integração das mudanças no sistema como um inteiro. Porém, isso não exclui a integração de metologias ágeis e Scrum integrados à metodologia de desenvolvimento em V.

Na Tabela 1 é possível observar uma comparação de cada técnica de desenvolvimento, listando particularidades de cada método apresentado.

Tabela 1 – Comparativo das técnicas de desenvolvimentos de projetos.

Aspecto	Metodologias Ágeis	Scrum	Ciclo V
Foco principal	Entrega contínua de valor, adaptação às mudanças.	Divisão do trabalho em Sprints curtos com entregas incrementais.	Estruturação e validação de todas as etapas do ciclo de vida de um sistema.
Iteratividade	Altamente iterativo, com ciclos curtos de feedback.	Totalmente iterativo dentro de Sprints.	Menos iterativo, segue um fluxo linear com validação em etapas definidas.
Planejamento	Adaptativo, planejamento contínuo ao longo do projeto.	Planejamento inicial com detalhamento em cada Sprint.	Planejamento rigoroso no início, com menos flexibilidade para mudanças.
Flexibilidade	Alta, com mudanças integradas facilmente.	Alta, mas limitada ao escopo definido no Sprint atual.	Baixa, mudanças são difíceis de acomodar após a definição inicial.
Ponto forte	Flexibilidade para mudanças e foco no cliente.	Agilidade no desenvolvimento e entregas constantes.	Clareza e controle sobre todo o processo de desenvolvimento.
Ponto fraco	Pode carecer de estrutura em projetos maiores e complexos.	Requer equipes maduras para boa execução.	Pouca adaptabilidade às mudanças durante o ciclo.
Áreas de aplicação recomendadas	Projetos que precisam de flexibilidade e adaptação rápida.	Projetos de desenvolvimento incremental com equipes organizadas.	Projetos complexos e críticos que exigem alta confiabilidade.

Fonte: (SMARTSHEET, 2025).

2.2 Ferramentas "*In-the-Loop*" para Testes de Validação

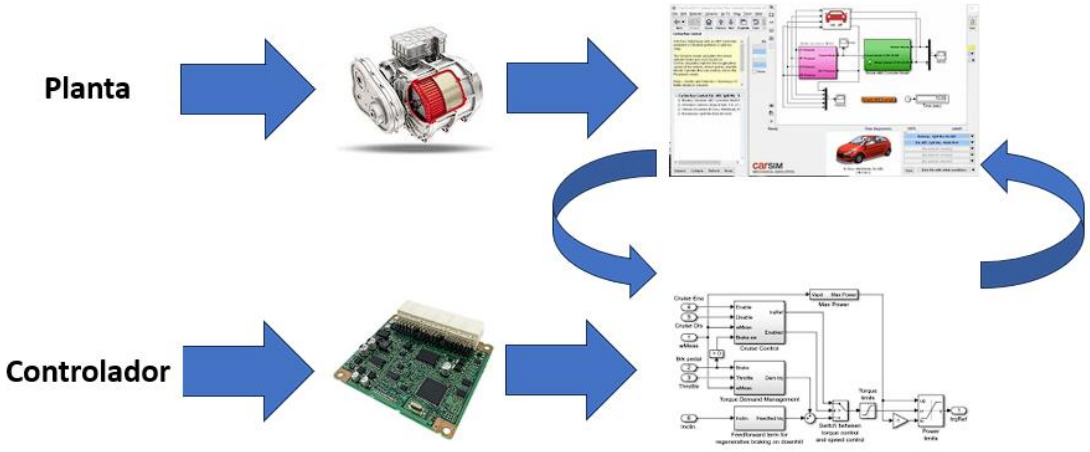
Os testes *in-the-loop* são etapas essenciais no ciclo de desenvolvimento de sistemas, principalmente de controle, permitindo validações de forma progressiva, desde o modelo conceitual até o hardware real. Essas ferramentas auxiliam na identificação de falhas e na otimização de processos, garantindo a confiabilidade de sistemas com alta complexidade, como em veículos autônomos, sistemas de energia, e automação industrial.

2.2.1 Model-in-the-Loop (MIL)

O *Model-in-the-Loop* (MIL) é a etapa inicial no processo de validação, onde um modelo matemático completo e detalhado do sistema em questão é desenvolvido, e testado em um ambiente de simulação. Essa abordagem garante que os controladores projetados atendam aos requisitos de desempenho antes mesmo da geração de códigos (AZEVEDO, 2020).

Nesta etapa, todas as partes, sejam sensores, atuadores, controladores, etc, são simulados em ambientes virtuais. O modelo pode incluir dinâmicas de veículos, condições ambientais, lógicas de controle, entre outros, assim dando assistência para os executores dos testes ajustarem os parâmetros do sistema e testar cenários extremos, sem risco de danos a um hardware real. Na Figura 3 é possível observar uma ilustração dos passos do MIL.

Figura 3 – Ilustração da modelagem dos componentes em ambiente virtual para a realização do MIL.



Fonte: O próprio autor.

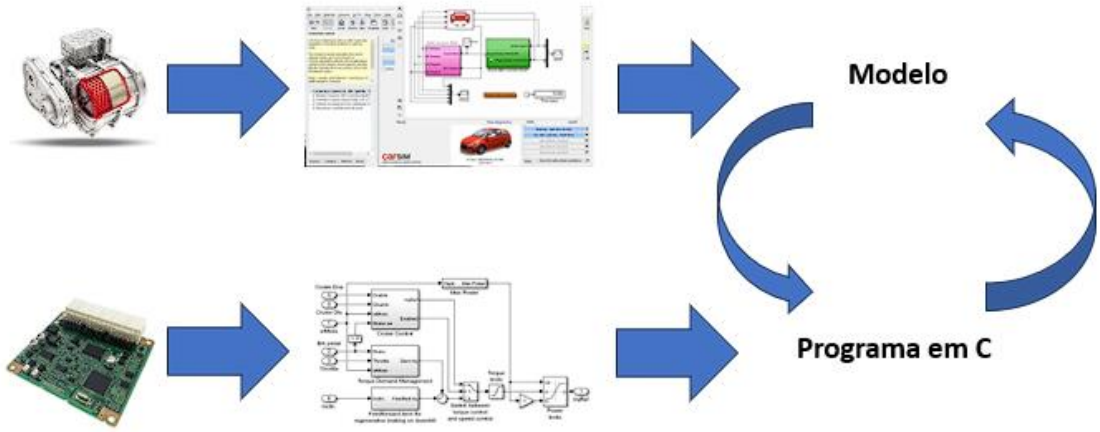
O *Model-in-the-Loop* é particularmente eficaz na identificação de inconsistências de projeto, e na realização de testes em estágios iniciais de desenvolvimento. Essa abordagem reduz significativamente custos associados, seja monetário, ou de tempo, de etapas posteriores.

2.2.2 Software-in-the-Loop (SIL)

Já o *Software-in-the-Loop* é a etapa seguinte no processo de validação, onde os códigos gerados no MIL são testados a partir do modelo em um ambiente simulado. Contudo, diferente do MIL, o SIL possui outro foco, sendo este o software de controle, o que permite a verificação da funcionalidade esperada do código automaticamente gerado, ou escrito manualmente.

Essa etapa reduz a lacuna entre a modelagem e a implementação, assim fornecendo um ambiente seguro para a identificação dos erros que poderiam surgir durante a fase de transição entre o design do modelo, e o código gerado (SCHÖFFMAN, 2017). Durante a etapa de testes em software, o mesmo é conectado ao modelo matemático, onde ambos são executados em simulação. Na Figura 4 é possível observar uma ilustração dos passos do SIL.

Figura 4 – A montagem do SIL através dos modelos montados na etapa anterior.



Fonte: O próprio autor.

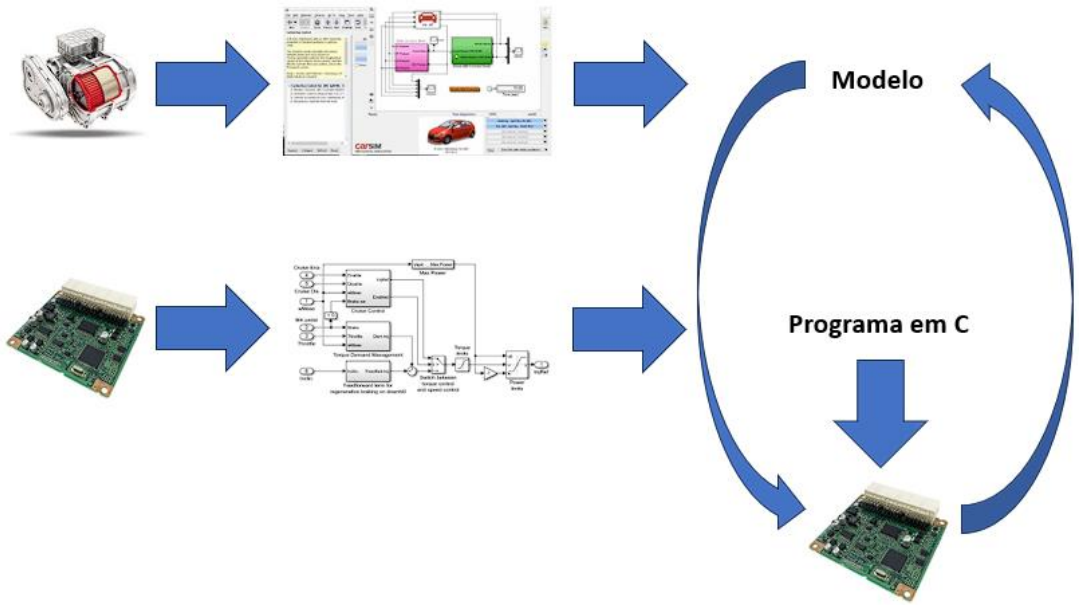
Em suma, o SIL pode ser utilizado para testar o software de projetos, antes de integrá-lo ao hardware real, assim garantindo que o código se comporte conforme o esperado em condições simuladas. Portanto, o SIL possui como vantagens principais, a detecção de erros em lógica ou sintaxe nos códigos, verificação de algoritmos antes da implantação no hardware, e redução de custos de depuração em hardware real.

2.2.3 Processor-in-the-Loop (PIL)

A etapa denominada *Processor-in-the-Loop* possui como ênfase a conexão do software de controle ao processador final, ou um emulador do mesmo, possuindo a finalidade de validação do desempenho do mesmo em condições muito semelhantes à realidade. O PIL mede o impacto do hardware no desempenho do sistema, e avalia obstáculos e restrições, como o uso de memória, o consumo de energia, e o tempo de execução (GOHARI, 2019).

A utilidade do PIL advém particularmente dos trabalhos utilizando sistemas embarcados, pois ele permite a identificação de problemas relacionados ao desempenho do sistema devido a recursos de processamento limitados, lentidões do sistema, ou falhas de integração. Na Figura 5 é possível observar uma ilustração dos passos do PIL.

Figura 5 – A realização da etapa do PIL, onde os dados modelados nas etapas anteriores são testados em performance.



Fonte: O próprio autor.

Os principais benefícios do PIL incluem a identificação de gargalos de desempenho nos processadores testados, a validação da integração entre software e hardware, e a possibilidade de simular condições reais sem a necessidade de utilizar o hardware completo. Essa etapa assegura a estabilidade do sistema, garantindo um

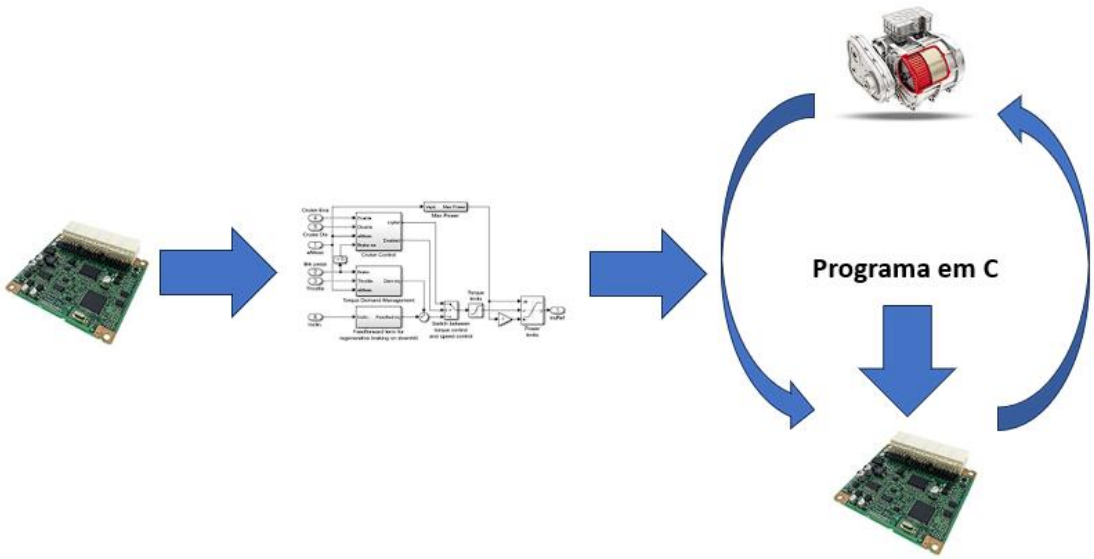
funcionamento eficiente em tempo real e o cumprimento de restrições de recursos estabelecidas.

2.2.4 Hardware-in-the-Loop (HIL)

Sendo a etapa final antes da implementação real, o HIL consiste na integração do hardware de controle a um modelo simulado do ambiente operacional. O HIL cria a ponte entre a validação virtual e prática, assim permitindo que os sistemas sejam testados em condições muito próximas da real, porém, sem os riscos associados a falhas catastróficas (ISERMANN, 2014).

Nesta etapa, os sensores e atuadores do sistema real são substituídos por simulações, todavia, o controlador opera como se fosse um sistema real, através de centrais sofisticadas que fornecem um ambiente virtual para tal. Em setores como o automotivo, aeroespacial, e de energia, essa abordagem é amplamente utilizada, principalmente pela ausência de riscos desastrosos associados, o que acaba otimizando custos e permitindo uma repetibilidade nas ações de testes destrutivos. Na Figura 6 é possível observar uma ilustração dos passos do HIL.

Figura 6 – Ilustração de como é realizado um teste em HIL.



Fonte: O próprio autor.

As principais vantagens do *Hardware-in-the-Loop* consistem na execução de testes seguros em cenários de falha crítica, sem acarretar desastres, avaliação

precisa do desempenho em tempo real dos sistemas testados, e a redução de custos com protótipos físicos, por razões já apresentadas anteriormente.

É possível observar em detalhes na Tabela 2 um comparativo das técnicas de validação *in-the-loop*, levando em consideração o foco da aplicação, o ambiente de testes, e aplicações comuns de cada etapa de validação.

Tabela 2 – Comparativo Detalhado das técnicas de validação *in-the-loop*.

Técnica	Foco Principal	Ambiente de Teste	Aplicação Comum
Model-In-the-Loop	Modelagem matemática inicial	Simulação	Verificar lógica de controle
Software-In-the-Loop	Código de controle gerado	Simulação	Testar funcionalidade do código
Processor-In-the-Loop	Software em processador real/emulado	Processador	Validar desempenho e restrições do hardware
Hardware-In-the-Loop	Integração hardware e modelo simulado	Controlador Real	Testar condições próximas da realidade

Fonte: O próprio autor.

2.3 Desenvolvimento e Teste de Funções Veiculares utilizando *Hardware-in-the-Loop*

As soluções mais atuais, quando se trata de aplicação em tecnologias automotivas, exigem a utilização de sensores, atuadores e algoritmos de controle sofisticados e bem desenvolvidos. Sendo a etapa final, o uso do HIL permite a simulação de componentes em condições reais, através da integração entre hardware e software, realizado em um ambiente seguro e eficiente.

Usualmente, ambientes simuladores como dSPACE (como apresentado na Figura 7) ou Typhoon HIL são utilizados para a reprodução de algoritmos, e para simular precisamente as condições em tempo real, e os mesmos são capazes de serem integrados aos sensores a ser testados.

Figura 7 – Diversos ambientes de simulação dSPACE.

Fonte: dSPACE (2025).

A seguir algumas funções veiculares podem ser explicadas em detalhes, juntamente com o desenvolvimento de testes de validação através do Ciclo V.

2.3.1 Cruise Control (Controle de Cruzeiro)

O sistema de controle de cruzeiro permite manter a velocidade do veículo constante, sem a necessidade de intervenção contínua do motorista. Com base no Ciclo V, as etapas anteriores à validação por *Hardware-in-the-Loop* incluem (MATHWORKS, 2025):

- **Requisitos do sistema:** O sistema deve manter uma velocidade que será definida pelo motorista, inclusive que será ajustada em subidas e descidas.
- **Design detalhado:** Os algoritmos são desenvolvidos pela equipe, com o objetivo do sistema de controle interpretar comandos do motorista, e controlar a injeção de combustível, ou a frenagem elétrica.
- **Implementação:** São desenvolvidos controladores do tipo proporcional-integral-derivativo (PID), e os mesmos são integrados ao controlador eletrônico do veículo.

- **Testes unitários e de integração:** Componentes individuais, como sensores de velocidade e aceleração, são testados isoladamente. Em seguida, realiza-se a validação das interações entre sensores e atuadores para garantir que o sistema funcione de forma coesa e atenda aos requisitos definidos.

2.3.1.1 Testes em HIL do Sistema de Controle de Cruzeiro

Para a realização dos testes em HIL desse sistema, dentro dos equipamentos de teste, são simuladas as condições da estrada, incluindo inclinação, atrito da via, tráfego, entre outros. O simulador é integrado aos sensores do veículo, como sensores de velocidade, e de posição do acelerador. A resposta do sistema a cenários adversos, como acelerações súbitas, ou descidas íngremes, é testada e avaliada. Então a partir dessas ações, falhas no software são identificadas, sem o risco de acidentes em testes reais. Então, após a detecção e correção de falhas, pode ser avaliado se o veículo simulado é capaz de manter a velocidade alvo, sem oscilações excessivas ou atrasos (MATHWORKS, 2023).

2.3.2 Sistema Start-Stop

Para o sistema *Start-Stop*, que desliga o motor do veículo ao parar e o religa com base na interação do motorista, os testes HIL validam as funções desenvolvidas. Antes de chegar a essa etapa, o ciclo de desenvolvimento segue os seguintes passos (SANTOS, 2018):

- **Requisitos do sistema:** O objetivo é reduzir o consumo de combustível e minimizar emissões de gases, sem comprometer o conforto do motorista, especialmente durante operações repetitivas em ambientes urbanos.
- **Design detalhado:** São definidos os sensores necessários, como os de velocidade e posição da embreagem e/ou freio, e desenvolvidos algoritmos que determinam quando o motor deve ser desligado ou religado.
- **Testes unitários e de integração:** Essa etapa valida a comunicação entre sensores, a central eletrônica do veículo e o motor de partida, assegurando que os sinais são interpretados corretamente e que o sistema reage de forma confiável.

2.3.2.1 Testes em HIL do Sistema Start-Stop

No simulador, é possível criar um cenário urbano virtual, onde a resposta do sistema a um tráfego com paradas frequentes é testada.

Dessa forma, é possível traçar o tempo de resposta do sistema, ao reiniciar o motor, os impactos na bateria e no alternador, quando submetido a ciclos repetidos, e se a função será suspensa após um determinado número de ciclos (MATHWORKS, 2023).

O sistema é então testado para garantir que o motor ligue novamente o mais rápido possível, quando o motorista pressionar o acelerador, mesmo com adversidades como temperaturas extremas

2.3.3 Função Speed-Limit (Limitador de velocidade)

O fim da funcionalidade conhecida como *Speed Limit* é a de garantir que limites de velocidade, juntamente com diversas outras regras de trânsito, sejam cumpridos, podendo realizar avisos sonoros para chamar a atenção do condutor, até ativamente realizar a ação de reduzir a velocidade do veículo.

Baseando-se no Ciclo V, as etapas que precedem os testes HIL são (SANTOS, 2018):

- **Requisitos do sistema:** Deverão ser identificadas sinalizações no trânsito (reconhecimento visual), ou integrar-se a mapas digitais e utilizar conectividade GPS.
- **Design detalhado:** Algoritmos são criados para interpretar dados do sistema de visão computacional, que capturarão através de câmeras as informações, e ajustará a velocidade do veículo.
- **Implementação:** É feita a integração com o controle eletrônico do sistema de frenagem e também com o sistema do acelerador do veículo.
- **Testes unitários de integração:** São realizados testes de validação com os sensores para garantir o reconhecimento de sinais e precisão da resposta.

2.3.3.1 Testes em HIL do Sistema Limitador de Velocidade

O simulador cria um cenário para a realização de testes dos sensores de visão (câmeras), simulando sinais de trânsito em diferentes condições, seja com alterações na visibilidade, ou em vários pontos de distâncias diferentes das sinalizações de trânsito (MATHWORKS, 2023).

Exemplificando, o HIL simulará um trecho de estrada com limites de velocidade variados, permitido testar o comportamento do veículo ao exceder esses limites. Verifica-se então se o sistema ajusta automaticamente a aceleração para entrar dentro dos limites permitidos. Também são testadas as interações com o motorista, se o sistema irá emitir alertas sonoros e/ou visuais nas situações onde os sensores reconheçam os sinais.

Com os testes mencionados, é possível avaliar benefícios provenientes da realização de testes utilizando *Hardware-in-the-Loop*. É possível observar como os testes foram conduzidos com segurança, sem a necessidade de por nenhum ser humano ou animal em risco, e nem destruir protótipos veiculares como etapas para validação dos sistemas, isso em funções que utilizam muitas informações advindas do tráfego. Dessa forma, evitam-se acidentes, protegendo tanto bens materiais quanto vidas humanas.

2.3.4 Função Limp-Home (Modo de Emergência)

A função *Limp Home* consiste em um modo de emergência que, ao detectar falhas críticas no veículo, reduz a potência do motor e limita a velocidade, dessa forma, permitindo que o condutor continue dirigindo até um local seguro sem comprometer a integridade do veículo, reduzindo o risco de danos adicionais ou falhas mais graves.

Baseando-se no Ciclo V, as etapas que precedem os testes HIL são (KALINOWSKI, 2019):

- **Requisitos do sistema**: O modo *Limp-Home* deve atuar na redução de potência e na limitação da velocidade do veículo em caso de falhas críticas, garantindo que o motorista consiga conduzir até um local seguro sem comprometer a segurança e a integridade do veículo.

- **Design detalhado**: São desenvolvidos algoritmos de monitoramento contínuo dos sensores do motor, transmissão e acelerador, permitindo a detecção de falhas e a ativação do modo de emergência conforme a gravidade do problema identificado.
- **Implementação**: O sistema é integrado à Unidade Central Eletrônica (ECU) do veículo, conectando-se aos módulos de controle do motor e transmissão para restringir o desempenho, garantindo que a resposta do acelerador e da transmissão siga os parâmetros de segurança predefinidos.
- **Testes unitários de integração:** São realizados testes com sensores e módulos eletrônicos para validar a correta ativação do modo Limp-Home, assegurando que as restrições de potência e velocidade ocorram conforme esperado e que o veículo permaneça operacional para permitir um deslocamento seguro.

2.3.4.1 Testes em HIL do Sistema de Modo de Emergência

Para a validação dos testes em HIL da função *Limp-Home*, o simulador submete o veículo a diferentes condições de falha, como perda de comunicação com sensores, superaquecimento do motor ou falhas na transmissão. São testadas diversas situações para verificar se o sistema responde corretamente, reduzindo a potência do motor e limitando a velocidade do veículo (MATHWORKS, 2023).

Exemplificando, o HIL irá simular falhas progressivas no sistema de injeção ou na transmissão, analisando se o veículo entra no modo de emergência e se mantém operacional o suficiente para permitir que o condutor alcance um local seguro. Além disso, são testadas interações com o motorista, como alertas visuais no painel e mensagens informando a necessidade de manutenção.

Por se tratar de um ambiente virtual, as simulações possibilitam uma ampla variedade de cenários para estressar o sistema ao máximo. Com todas as validações realizadas, é possível identificar falhas antes da integração completa ao veículo, permitindo ajustes no software sem custos adicionais com componentes físicos, apenas refinando os algoritmos de controle.

Além da segurança, esses testes oferecem vantagens como a redução de custos e o aumento da eficiência em comparação à utilização de protótipos reais.

Outro benefício importante é a eliminação de limitações presentes em testes de pista, como aclives e declives acentuados. No ambiente HIL, essas restrições são facilmente superadas, proporcionando maior flexibilidade na avaliação do desempenho do veículo.

Como as simulações são ambientes virtuais, é possível ter uma ampla gama de cenários para estressar os sistemas ao máximo. Com todas as validações realizadas, torna-se possível identificar falhas antes da integração completa dos sistemas ao veículo, permitindo sua correção sem custos adicionais com materiais, apenas ajustando o algoritmo em teste.

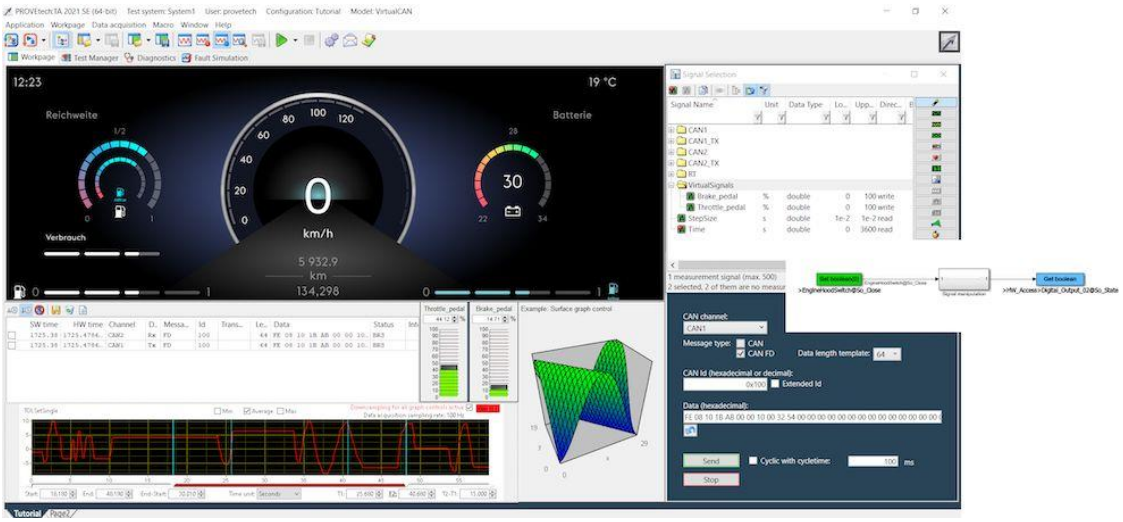
3 DESENVOLVIMENTO E RESULTADOS

Neste capítulo são apresentados os procedimentos realizados para a execução do trabalho, detalhando as etapas práticas e a estrutura adotada para alcançar os objetivos propostos.

3.1 Integração de *Softwares* Automotivos para Validação de Cenários

Para a realização deste trabalho, são utilizadas ferramentas e procedimentos que avaliam e verificam com precisão e eficiência o comportamento de sistemas automotivos. Para tanto, a abordagem é construída sobre um modelo de motor de combustão interna (ICE), configurado e carregado no *software* Provetech:TA, que executa a simulação e verificação do sistema. Na Figura 8 é possível observar um exemplo de interface de usuário do *software*, onde vários elementos podem ser arranjados dentro da interface do mesmo e o simulador irá simulá-los como num veículo real.

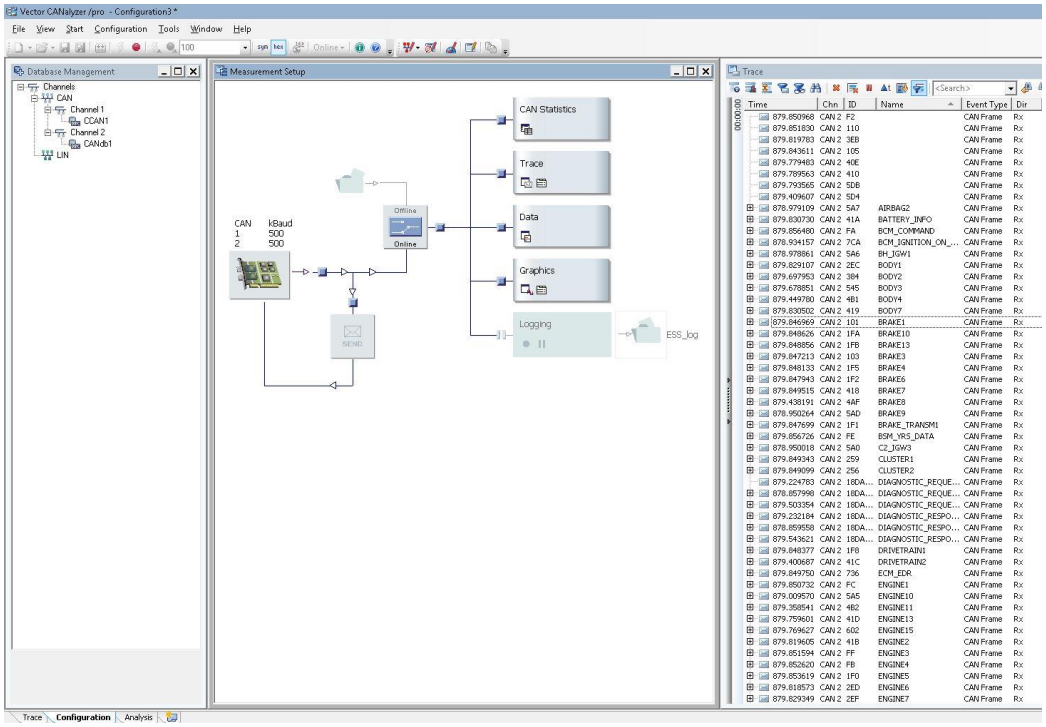
Figura 8 – Interface do software Provetech:TA.



Fonte: MATHWORKS (2025).

Enquanto isso, o *CANalyzer*, ferramenta amplamente utilizada na área automotiva, é utilizado para a aquisição de dados em tempo real (*online*) e para a análise aprofundada de gráficos e resultados no modo *offline*. Na Figura 9 é possível observar a interface gráfica do *software,* onde é possível selecionar os sinais a serem analisados.

Figura 9 – Interface gráfica do *software* CANalyzer.



Fonte: O próprio autor.

3.1.1 Uso do Modelo ICE no software Provetech:TA

O modelo ICE, que representa virtualmente o funcionamento de um motor de combustão interna, foi configurado e carregado no Provetech:TA, um *software* conhecido pela sua robustez na simulação de sistemas complexos. Esse modelo permite simular em tempo real do comportamento dinâmico do motor sob diferentes condições de operação, fornecendo uma base confiável para validações e testes. Além disso, o Provetech:TA possibilita a integração com outros sistemas e ferramentas, promovendo um ambiente de testes flexível e adaptável.

No contexto deste trabalho, o Provetech:TA foi configurado para executar cenários específicos de funcionamento do motor, replicando condições reais de operação com alta fidelidade. Entre os parâmetros simulados estavam as curvas de torque, rotação e consumo de combustível, elementos necessários para a análise de desempenho do motor e dos subsistemas automotivos relacionados.

Contudo, devido à alta complexidade do modelo ICE utilizado, bem como por questões de confidencialidade técnica, não será possível apresentar em detalhes a estrutura interna ou os componentes específicos que compõem esse modelo.

3.1.2 Monitoramento e Análise com o CANalyzer

O CANalyzer é um *software* amplamente utilizado na indústria automotiva para testar comunicações em redes CAN (*Controller Area Network*) e LIN (*Local Interconnect Network*), permitindo capturar e analisar mensagens de comunicação em tempo real. Durante o teste, o CANalyzer é configurado para monitorar sinais de comunicação entre o modelo ICE e outros componentes virtuais ou físicos integrados ao ambiente de teste. Os dados coletados em tempo real incluem informações como velocidade, parâmetros de controle, mensagens de diagnóstico e comandos trocados no barramento CAN. O software também conta com recursos gráficos avançados, que permitem registrar essas informações e analisá-las posteriormente no modo *offline*, possibilitando uma avaliação detalhada do comportamento do sistema.

3.1.3 Integração e Análise de Resultados

A utilização paralela do CANalyzer com o Provetech:TA proporciona uma visão detalhada do comportamento do motor e da comunicação entre os sistemas, possibilitando identificar possíveis discrepâncias ou falhas no fluxo de informações. Os gráficos gerados pelo CANalyzer em modo offline foram analisados, com foco na consistência dos sinais capturados, no tempo de resposta do sistema e em possíveis falhas de comunicação. Esta análise possibilitou a avaliação da eficiência do sistema proposto e a validação do modelo implementado no Provetech:TA.

Para a realização da integração entre os dois softwares, foi necessário a utilização de interligações físicas de condutores entre um simulador, e outra máquina, que realizará a análise das mensagens CAN. Na Figura 10 e na Figura 11 é possível observar os hardwares utilizados para a realizar a integração do simulador com a controladora que irá rodar o software CANalyzer. Na Figura 12 e na Figura 13 é possível observar a ligação física entre o simulador onde o software Provetech é executado, e a controladora que realizará a leitura dos dados da rede CAN do veículo simulado. Na Figura 14 e na Figura 15 é possível observar as configurações para a leitura dos dados da rede CAN, que será aquisitado do simulador de um veículo, rodando o software Provetech:TA.

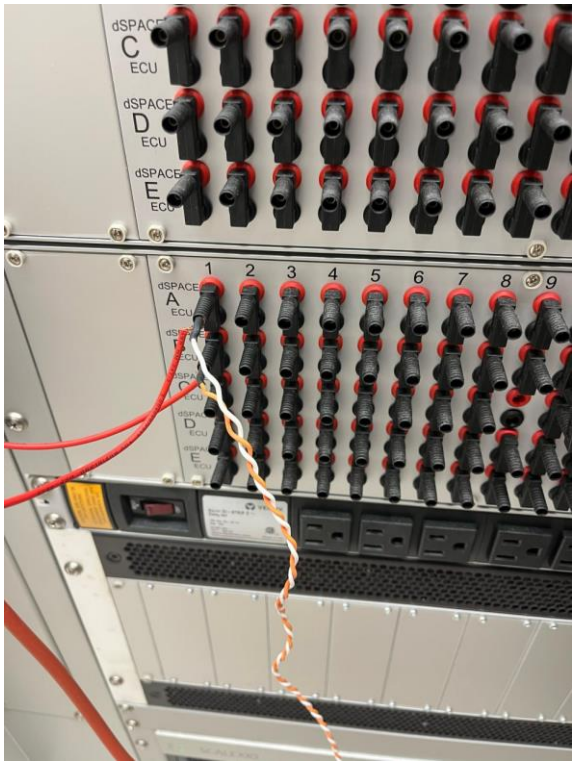
Figura 10 – Hardware VN1640 para Leitura de CAN/LIN.

Fonte: VECTOR (2025).

Figura 11 – Hardware ETAS/INCA para calibração e carregamento dos Sofwares na Controladora.

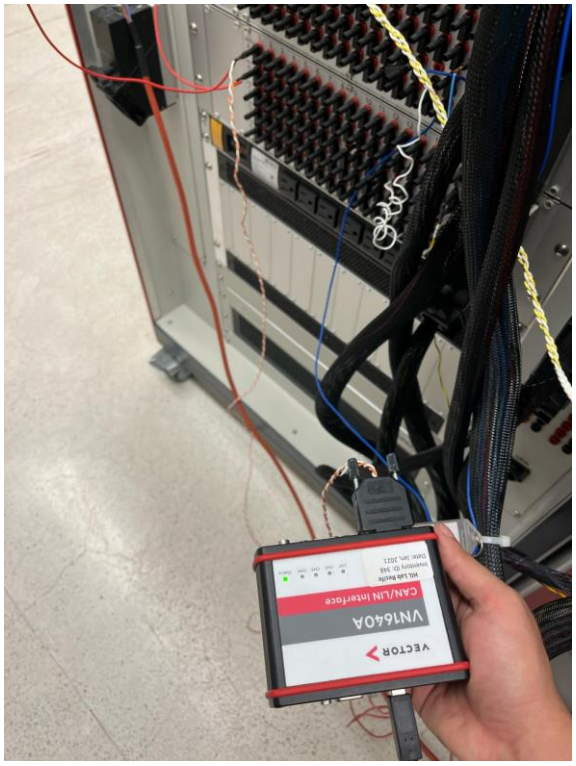
Fonte: ETAS (2025).

Figura 12 – Leitura CAN no Simulador HIL.



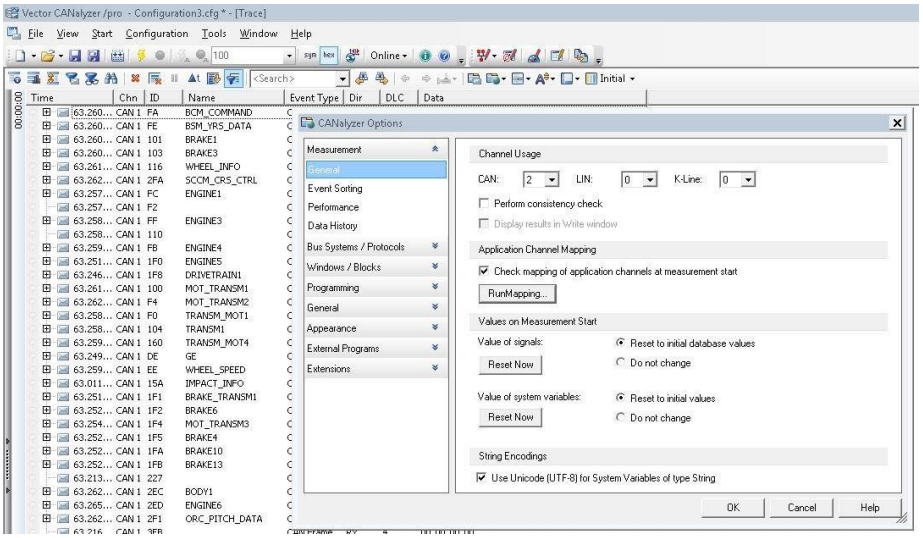
Fonte: O próprio autor.

Figura 13 – Integração entre Provetech e CANalyzer por meio do VN1640.



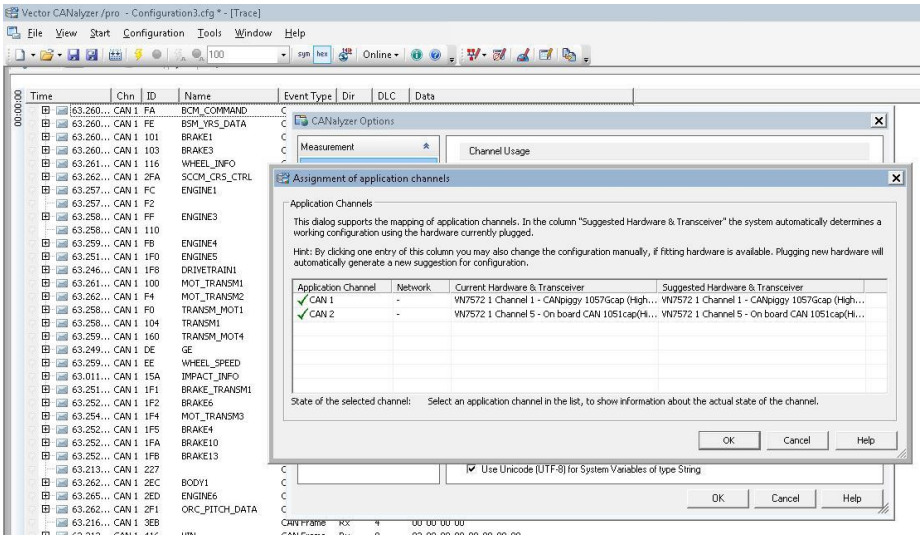
Fonte: O próprio autor.

Figura 14 – Configurações para a leitura de sinais da rede CAN no *software* CANAlyzer.



Fonte: O próprio autor.

Figura 15 – Escolha do canal para a aquisição de sinais da rede CAN no software CANAlyzer



Fonte: O próprio autor.

3.2 Resultados

Nos tópicos a seguir, serão apresentados os resultados das aplicações, incluindo os cenários de validação definidos e os parâmetros utilizados para análise.

3.2.1 Função de Controle de Velocidade (Cruise Control)

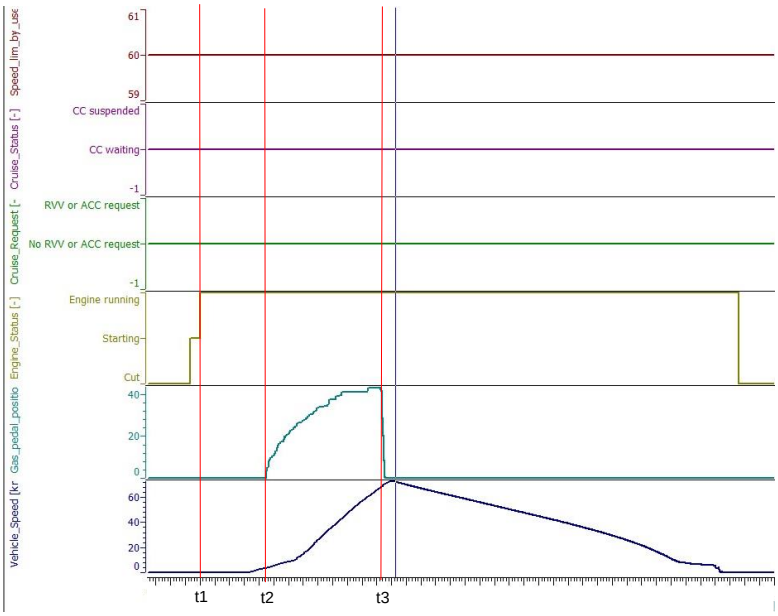
A análise dos testes de validação realizados pelo *software* CANalyzer foi baseada no monitoramento de seis variáveis principais que descrevem o estado do sistema e do veículo durante a operação da função.

As variáveis monitoradas durante os testes podem ser listadas a seguir:

- ***Speed_lim_by_user [km/h]:*** Representa a velocidade definida pelo usuário, que o veículo deve manter automaticamente quando a função de *Cruise Control* é ativada.
- ***Cruise_Status:*** Indica o estado de ativação da função, que pode ser ativada ou desativada pelo usuário. No gráfico, pode apresentar três estados diferentes, sendo eles: ativo (*CC active*), inativo (*CC waiting*) e suspenso (*CC suspended*).
- ***Cruise_Request:*** Refere-se ao status da função de controle de cruzeiro, que pode estar ativa (*RVV or ACC Request*) ou inativa (*No RVV or ACC Request*).
- ***Engine_Status:*** Relacionado ao estado do motor durante os testes, podendo assumir três condições possíveis: *engine running* (Motor funcionando normalmente), *starting* (Motor em partida) e *cut* (Motor desativado).
- ***Gas_pedal_position [%]:*** Indica a porcentagem de abertura do pedal do acelerador, refletindo a injeção de combustível no motor.
- ***Vehicle_Speed [km/h]:*** Representa a velocidade do veículo durante o teste.

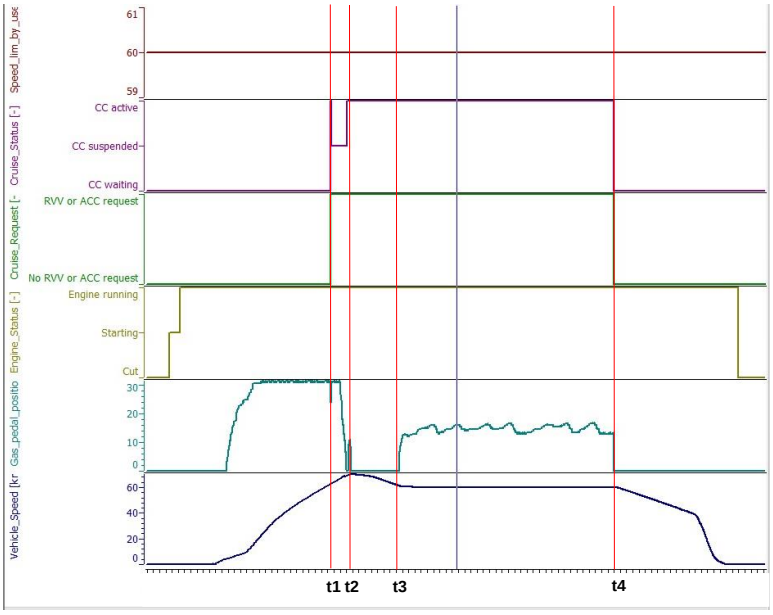
Para um entendimento completo do comportamento da função e das variáveis monitoradas, foram realizadas duas análises, onde em uma a função está inativa, e em outra, a função está ativa. As duas podem ser observadas respectivamente na Figura 16 e na Figura 17.

Figura 16 – Comportamento da Função Cruise Control desativada, vista pelo software CANalyzer.



Fonte: O próprio autor.

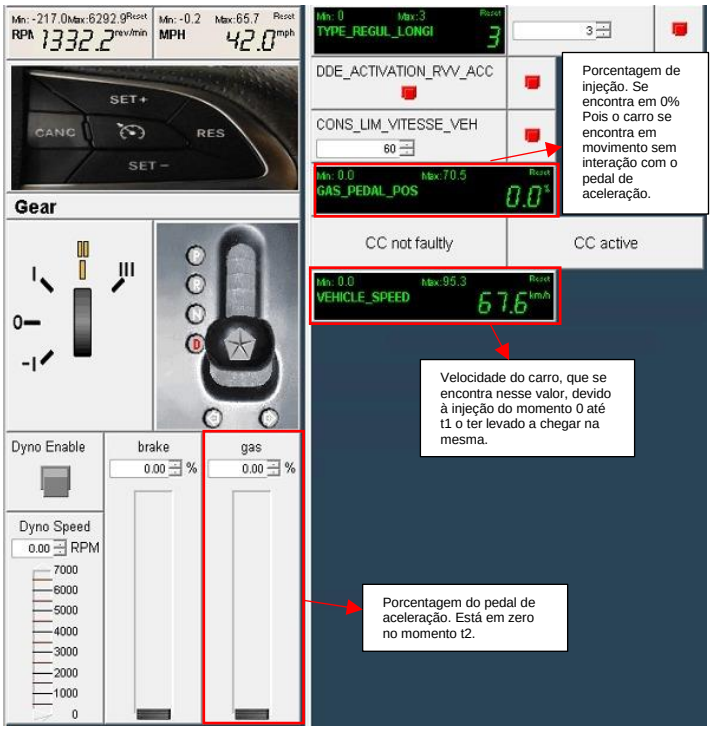
Figura 17 – Comportamento da Função Cruise Control ativada, vista pelo software CANalyzer.



Fonte: O próprio autor.

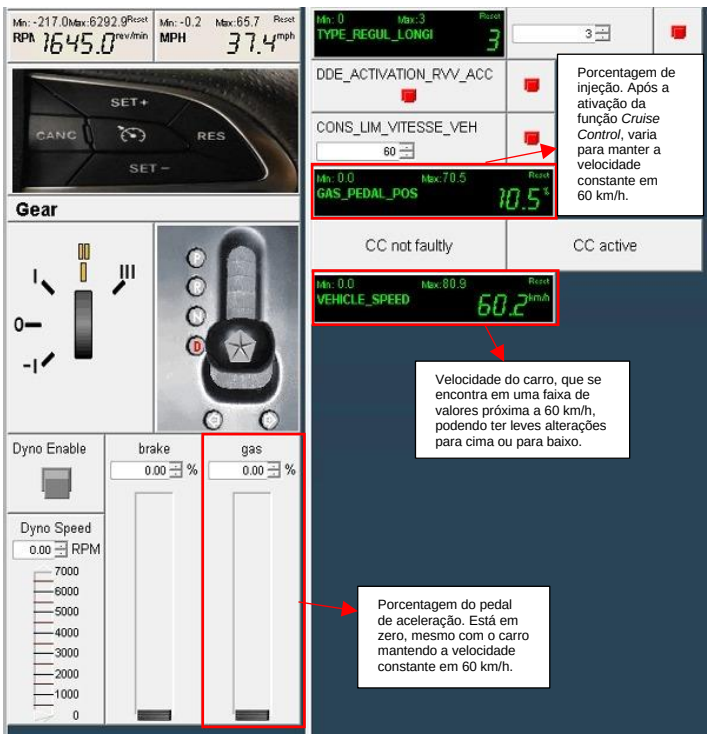
Também é possível observar os resultados obtidos no software Provetech:TA, na Figura 18 e na Figura 19.

Figura 18 – Momento do teste realizado no software Provetech:TA no momento t2 (injeção nula).



Fonte: O próprio autor.

Figura 19 – Interface de testes do software Provetech:TA realizando a função de *Cruise Control.*



Fonte: O próprio autor.

Em questões de análise, é possível observar que, na Figura 16, através da resposta obtida no CANalyzer, a função *Cruise Control* se encontra desativada. No momento t2, a injeção do veículo é alterada manualmente, elevando-se o parâmetro *Gas_pedal_position* para cerca de 40%, e no momento t3, esse parâmetro é zerado. Em resposta a essa ação, a velocidade subiu gradualmente até cerca de 70km/h, e, após isso, foi gradualmente descendo até a sua parada.

Já, na Figura 19, é possível observar as respostas da função ativada. Inicialmente, neste teste, a função está inativa, e a injeção é elevada manualmente até o momento t1, onde o carro atinge uma velocidade um pouco acima de 60km/h. No momento t1, a função então é ativada, com um ajuste de velocidade de 60km/h, e é possível ver a resposta do veículo após esta ação. É possível ver que entre o momento t1 e t2, a função se encontra suspensa, contudo isso se dá ao fato do pisal do acelerador não estar zerado. No momento t2, é possível então ver a função ser ativada novamente, por não haver mais interação do usuário com o acelerador. Após esse momento, a injeção é zerada automaticamente até o momento t2, onde a velocidade é reduzida para cerca de 60km/h. Esta resposta pode ser observada na interface do Provetech:TA, como visto na Figura 18, onde a injeção se encontra em zero. Após a velocidade chegar aos 60km/h, a injeção é reativada, porém, de forma irregular, numa margem entre 10 e 20%, assim mantendo a resposta da velocidade de uma maneira constante. Após o momento t4, a função é desativada, e então o veículo perde velocidade até a mesma chegar a zero.

3.2.2 Função Limitadora de Velocidade (Speed Limit Control)

A análise dos testes de validação realizados pelo *software* CANalyzer desta função, assim como no teste anterior, se baseou no monitoramento de seis variáveis principais que descrevem o estado do sistema e do veículo durante a operação da função. Quatro dessas variáveis possuem o mesmo propósito das mesmas analisadas no teste da função *Cruise Control*, sendo elas a “*Speed_lim_by_user*”, “*Engine Status*”, “*Gas_pedal_position*” e “*Vehicle Speed*”.

As outras variáveis monitoradas durante os testes podem ser listadas a seguir:

- ***SL_Status:*** Representa o estado de ativação da função, que pode ser ativada ou desativada pelo usuário. No gráfico, pode apresentar três estados

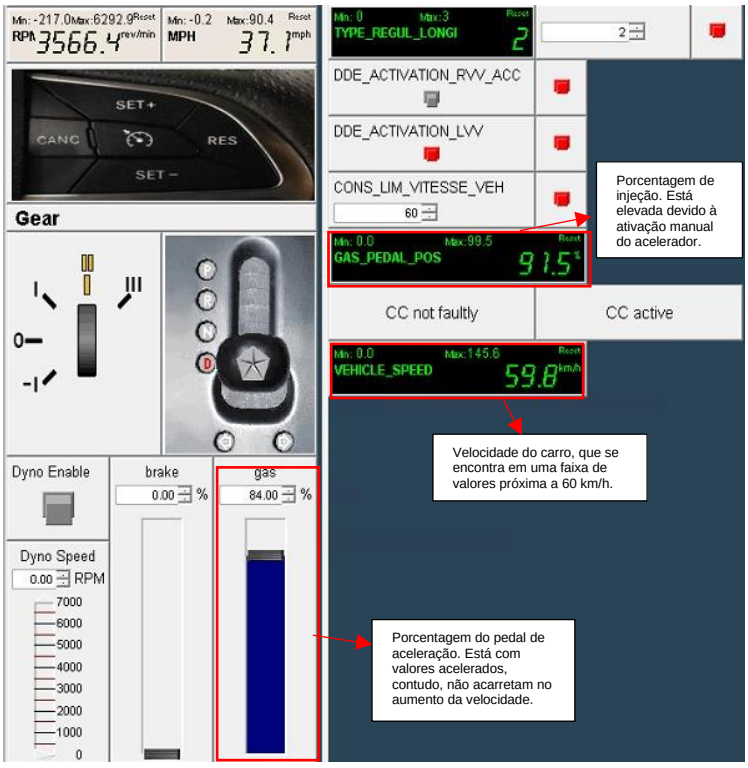
diferentes, sendo eles: ativo (*CC active*), inativo (*CC waiting*) e suspenso (*CC suspended*).

- ***SL_Request: I***ndica o status da função de controle de velocidade, que pode estar ativa (*LVV Request*) ou inativa (*No LVV Request*).

Para a realização do teste da função, foi necessário apenas aumentar a injeção através da elevação da porcentagem da injeção. Com o aumento gradual da injeção ao longo do tempo, o veículo deveria elevar sua velocidade, contudo, com a ativação da função o mesmo deve manter constante a velocidade máxima estabelecida pela função. Nesse teste, o limite de velocidade utilizado foi de 60km/h. A Figura 20 mostra a interface do *software* Provetech:TA para a função.

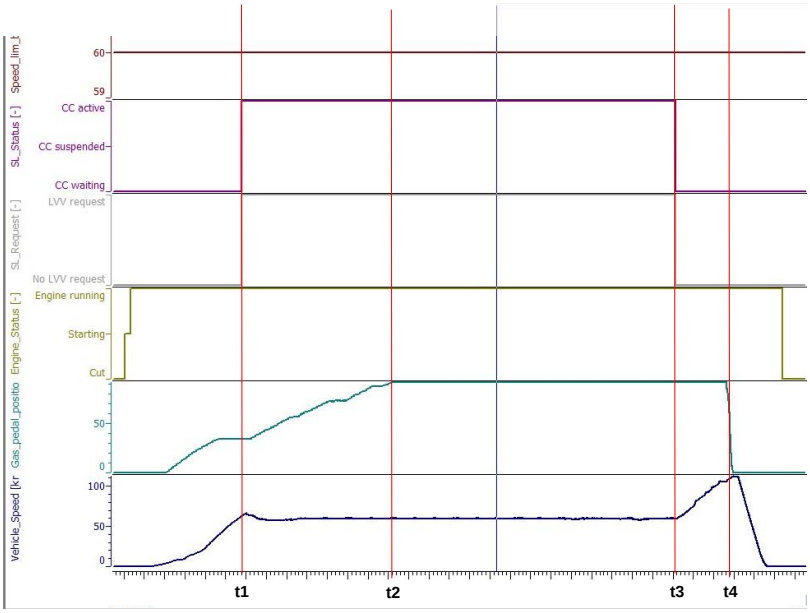
Na Figura 21 apresenta as respostas do CANalyzer. Observa-se que a função é ativada no momento t1, e após o momento t2, a velocidade se mantém constante, mesmo com o aumento da injeção. No entanto, após o momento t3, quando a função é desabilitada, a velocidade do veículo começa a aumentar novamente, até que a injeção seja zerada, no momento t4.

Figura 20 – Realização do teste da função *Speed Limit* feita no Provetech:TA.



Fonte: O próprio autor.

Figura 21 – Comportamento da função *Speed Limit* analizada pelo CANalyzer.



Fonte: O próprio autor.

3.2.3 Função Start-Stop

A análise dos testes de validação realizados pelo *software* CANalyzer desta função se baseou no monitoramento de seis variáveis principais que descrevem o estado do sistema e do veículo durante a operação da função. Quatro dessas variáveis possuem o mesmo propósito das mesmas analisadas no teste da função *Cruise Control*, sendo elas a "*Engine Status*", "*Gas_pedal_position*" e "*Vehicle Speed*".

As outras variáveis monitoradas durante os testes podem ser listadas a seguir:

- ***KeyPosition:*** Representa o estado da posição da chave, que pode assumir cinco estados diferentes, entre eles o "IGN_LK", "2", "ACC", "RUN", e "START". Para a análise, as variáveis mais importantes são a "IGN_LK", que representa que o motor do veículo está desligado, o "RUN", que representa que o motor do veículo está em funcionamento normal, e o "START", que representa a partida do motor.
- ***ESS_Status:*** Indica o status da função *Start-Stop*, que pode assumir os valores de ativo (D_AND_SB), inativo (NONE) ou falha no sistema (SYSFLT).
- ***BreakerPedalStatus:*** Representa o estado do pedal do freio do veículo, este assumindo apenas dois estados possíveis: ativo (Active) e inativo (Not_Active).

A Figura 22 mostra a interface do software Provetech:TA para a função. Na Figura 23 é possível ver as respostas do CANalyzer.

Para a realização dos testes desta função, primeiramente, o carro está parado, e com o pedal de freio ativo. Anterior ao primeiro momento t1, é possível perceber que o veículo automaticamente realiza a partida do motor, mesmo com o pedal de freio ativado.

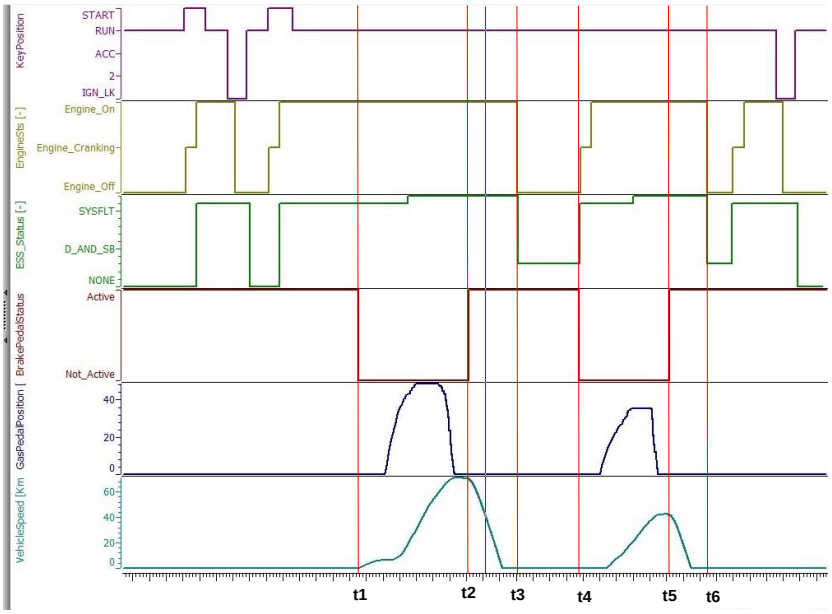
Após o momento t1, é necessário simular a movimentação do veículo, então o mesmo tem sua injeção aumentada, o que implica num movimento por alguns segundos, dessa forma deixando a função pronta para a ativação novamente. Para o prosseguimento dos testes, o veículo tem seu pedal de freio ativo por alguns segundos, e então após essa ação, o veículo então deve automaticamente realizar a partida do sistema, e entrar no seu regime normal de funcionamento em tempo hábil, para assim deixar o veículo pronto para se movimentar novamente.

É possível observar que no momento t1, o pedal de freio está acionado, e então após t1, o acelerador é acionado para o carro se mover. No momento t2, então o pedal é acionado, e é possível observar que no momento t3, o motor é desligado, permanecendo nesse estado até o momento t4, onde o pedal de freio é acionado. O processo é repetido, acionando-se o pedal de freio novamente em t5, e em t6 o motor novamente é desligado.

Figura 22 – Realização do teste da função *Start-Stop* feita no Provetech:TA.

Fonte: O próprio autor.

Figura 23 – Comportamento da função Start-Stop analizada pelo CANalyzer.



Fonte: O próprio autor.

3.2.4 Função Limp-Home

A análise dos testes de validação realizados pelo *software* CANalyzer desta função se baseou no monitoramento de cinco variáveis principais que descrevem o estado do sistema e do veículo durante a operação da função. Três dessas variáveis possuem o mesmo propósito das mesmas analisadas no teste da função *Start-Stop*, sendo elas a “*KeyPosition*”, “*Engine Status*” e “*Vehicle Speed*”.

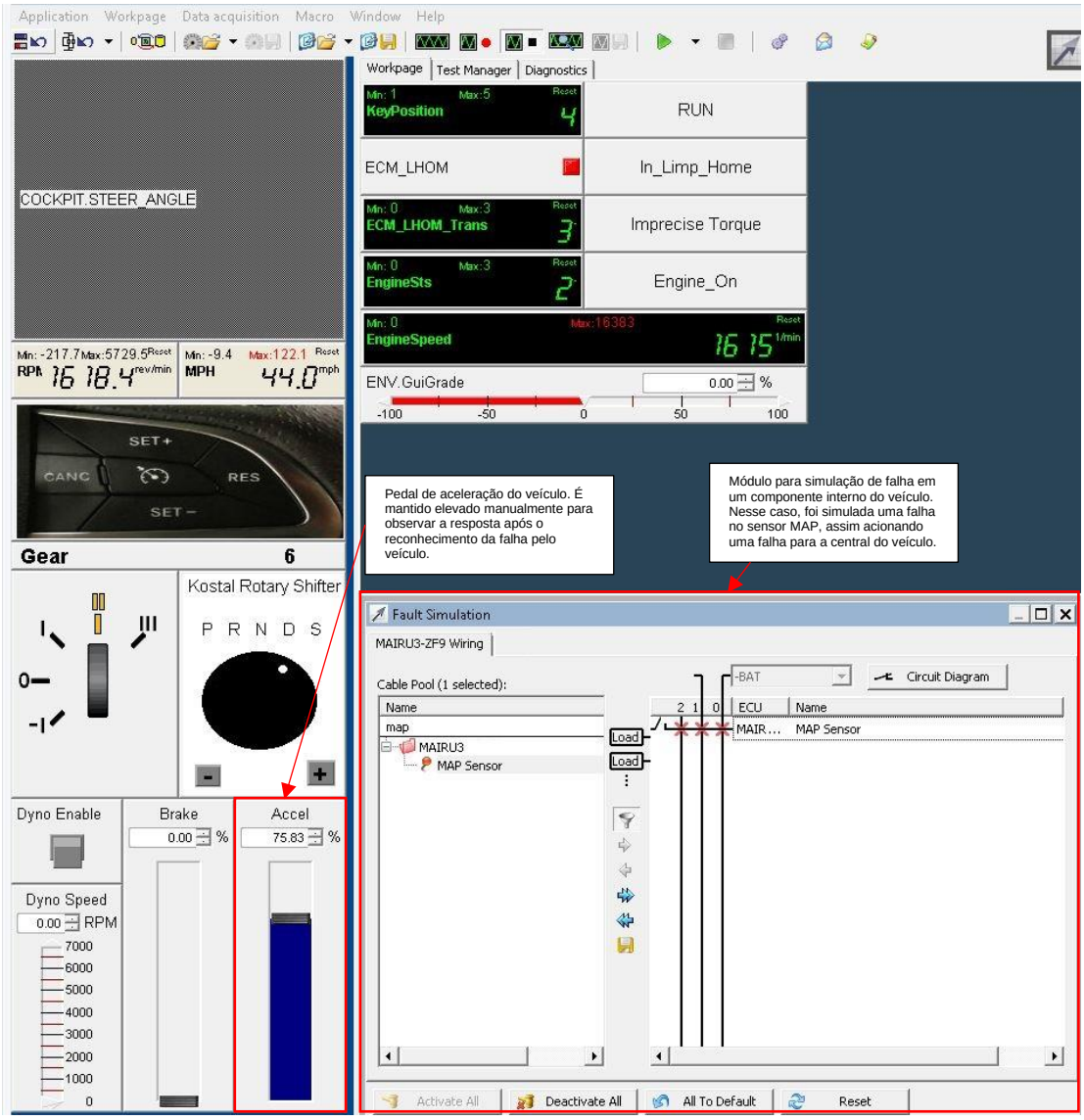
As outras variáveis monitoradas durante os testes podem ser listadas a seguir:

- ***ECM_LHOME:*** Representa o estado da função, podendo assumir os estados de ativa (*In_Limp_Home*) ou inativa (*Not_In_Limp_Home*).
- ***ECM_LHOM_Trans:*** Representa a detecção da falha, que neste caso, está relacionada ao torque do motor do veículo. Para a validação dos testes, assume que o torque está impreciso.

Para a ativação da função *Limp-Home* dentro do ambiente HIL, foi simulada uma falha no sensor MAP (*Manifold Absolute Pressure*), como visto na Figura 24, responsável por medir a pressão do coletor de admissão e fornecer dados da injeção de combustível e do desempenho do motor. A simulação dessa falha resultou na ativação da função, limitando a potência do motor e reduzindo progressivamente a velocidade do veículo para minimizar riscos e evitar danos mecânicos.

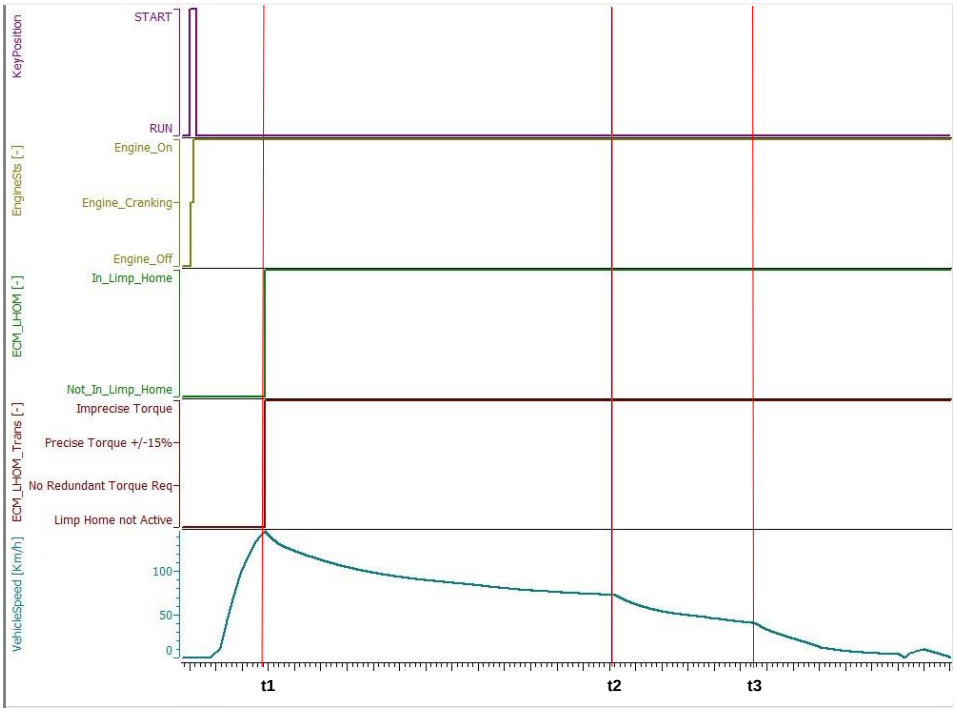
A análise dos dados registrados no CANalyzer, na Figura 25, mostrou que a velocidade do veículo apresentou uma queda exponencial após a ativação da função *Limp-Home,* ocorrida no momento t1 pela simulação de falha realizada pelo Provetech:TA. Nos momentos t2 e t3, houve alterações na simulação da inclinação de terreno ao qual o veículo estava submetido, sendo possível observar que a resposta do sistema foi um comportamento adaptativo devido a condições simuladas do terreno. Essa mudança influenciou diretamente a dinâmica da velocidade, que se ajustou à nova condição do terreno, resultando em pequenas variações na taxa de desaceleração.

Figura 24 – Realização do teste da função *Limp-Home* feita no Provetech:TA.



Fonte: O próprio autor.

Figura 25 – Comportamento da função *Limp-Home* analizada pelo CANalyzer.



Fonte: O próprio autor.

3.3 Conclusões Parciais

Com os resultados obtidos na simulação no software Provetech:TA, juntamente com a análise de dados no CANalyzer, foi possível observar o comportamento dinâmico dos sistemas testados no HIL, sob diferentes condições operacionais. Os resultados indicaram que as funções testadas, como *Cruise Control, Start-Stop, Speed Limit e Limp-Home*, responderam conforme o esperado, garantindo a estabilidade e previsibilidade das variáveis monitoradas. Além disso, a capacidade de replicar várias condições adversas, incluindo mudanças de velocidade, atuação do motor e variações no terreno simulado, possibilitou uma análise detalhada do desempenho dos sistemas de controle.

O teste da função *Cruise Control*, por exemplo, demonstrou a capacidade do sistema de manter a velocidade definida pelo usuário, ajustando-se conforme a necessidade e garantindo a estabilidade do controle. A análise dos resultados da simulação demonstrou que o sistema aciona e mantém a velocidade com precisão,

desativando-se corretamente em condições previstas, como a intervenção do motorista ou a necessidade de desaceleração.

Já a função *Start-Stop*, que tem por objetivo economizar combustível e reduzir emissões do veículo mediante o desligamento temporário do motor nas situações de parada, mostrou-se positivamente às condições simuladas, indicando o sistema ativar e desligar o motor nos momentos exatos e também promoverá uma transição suave dos estados de running e OFF.

A função *Speed Limit* mostrou-se eficiente na limitação da velocidade do veículo, respeitando o valor pré-definido pelo usuário. Durante os testes, foi verificado que o sistema atuava para restringir a velocidade, utilizando estratégias de controle que evitam oscilações bruscas e garantem a segurança do veículo.

Além dessas funções, a inclusão do *Limp-Home* foi um diferencial importante no estudo. Esta função é importante para manter a viabilidade do veículo em serviço em caso de falhas no sistema propulsor ou componentes vitais deste sistema. Durante os testes, foi verificado que, com falhas simuladas, o sistema Limp-Home respondia adequadamente, acionando e reduzindo a entrada de potência disponível para o motor e limitando a velocidade do veículo, de modo que, em um incidente no mundo real, o motorista pudesse continuar até um local seguro. A análise das variáveis relacionadas demonstrou que o controle de emergência funcionou conforme esperado, minimizando riscos e protegendo os componentes mecânicos de danos mais severos.

4 CONCLUSÕES E PROPOSTAS DE CONTINUIDADE

Este trabalho teve como objetivo validar funções automotivas utilizando a metodologia HIL, integrando os *softwares* Provetech:TA e CANalyzer para simulações de desempenho e análise dos testes de validação realizados. A abordagem adotada permitiu a criação de um ambiente simulado no qual as funções *Cruise Control, Start-Stop, Speed-Limit* e *Limp-Home* foram testadas sob diversas condições, visando garantir a confiabilidade e eficiência dos algoritmos de controle.

A implementação do ambiente de testes HIL demonstrou ser uma ferramenta eficaz para a verificação e validação de sistemas embarcados automotivos. Ao longo dos testes, foi possível observar o comportamento das variáveis monitoradas, analisando sua resposta a diferentes cenários. Os resultados confirmaram que as funções operaram conforme o esperado, respeitando os parâmetros estabelecidos e apresentando respostas coerentes com o comportamento dinâmico do veículo simulado.

Além de validar as funções individualmente, os experimentos ressaltaram a importância da metodologia HIL no desenvolvimento e teste de sistemas automotivos. A simulação em tempo real permitiu antecipar possíveis falhas e otimizar os parâmetros antes da implementação física, reduzir custos e aumentar a confiabilidade dos componentes embarcados. A utilização dos *softwares* Provetech:TA e CANalyzer foram fundamentais durante a simulação, visualização e detalhamento das análises dos comportamentos de cada um dos sistemas apresentados.

Diante dos resultados obtidos, conclui-se que a aplicação da metodologia HIL para a validação de funções automotivas foi bem-sucedida, permitindo avaliar de forma realista o desempenho dos sistemas em um ambiente seguro e controlado. Os testes realizados demonstraram que as funções implementadas atenderam aos requisitos projetados, reforçando a eficiência da abordagem adotada.

Dessa forma, o presente trabalho reafirma a importância da metodologia HIL no desenvolvimento de soluções automotivas, evidenciando a capacidade de redução de custos, aumento na segurança durante testes, e garantir uma confiabilidade dos sistemas embarcados antes da sua implementação no produto final.

4.1 Trabalhos Futuros

Como proposta de continuação dos trabalhos, é importante ampliar o escopo dos testes com a simulação de condições mais complexas, tais como mudanças bruscas na velocidade do veículo, falhas simultâneas em vários sistemas ou interações entre diferentes módulos de controle.

Outra proposta de exploração do tema envolve o confronto dos testes HIL com testes reais de validação automotiva, com comparação dos resultados obtidos em cada ambiente de testes.

REFERÊNCIAS

AZEVEDO, R. T.; LIMA, M. J.; ANDRADE, D. G. Uma abordagem baseada em Model-In-the-Loop (MIL) para validação de controladores. Revista Brasileira de Automação e Controle, 2020. p. 23-30.

BECK, K. Manifesto para Desenvolvimento Ágil de Software. Agile Manifesto, 2001. Disponível em: https://agilemanifesto.org/. Acesso em: 15 fev. 2025.

BREIVOLD, H. Using HIL for Testing Automotive Embedded Systems. IEEE Transactions on Industrial Electronics, 2015. p. 123-135.

dSPACE. Solutions for Simulation and Validation. Disponível em: https://www.dspace.com/. Acesso em: 15 fev. 2025.

ETAS. Cables and Connectors - Data Acquisition & Processing Tools. Disponível em: https://www.etas.com/ww/en/products-services/data-acquisition-processing-tools/hardware-products/cables-and-connectors/. Acesso em: 15 fev. 2025.

FORSBERG, K.; MOOZ, H. The Relationship of System Engineering to the Project Cycle. INCOSE International Symposium, 1991. p. 57-65.

GAO, Y.; CHEN, Y.; HUANG, D. Hardware-in-the-loop testing of automotive control systems. Annual Reviews in Control, 2019. p. 189-202.

GIETELINK, O. P. J.; BÜSKENS, J. D.; VAN DER VORST, M. Development of advanced driver assistance systems with vehicle hardware-in-the-loop simulations. Vehicle System Dynamics, 2006. p. 569-590.

GOHARE, M. Processor-In-the-Loop: A critical step in embedded system validation. International Journal of Embedded Systems, 2019. p. 157-169.

HIGHSMITH, J. Agile Project Management: Creating Innovative Products. 2ª ed. Addison-Wesley, 2009.

ISERMANN, R. Hardware-in-the-loop simulation for control and system testing. IEEE Control Systems Magazine, 2014. p. 76-89.

ISERMANN, R.; SCHMIDL, J.; SIEBER, S. Hardware-in-the-loop simulation for the design and testing of engine-control systems. Control Engineering Practice, 2012. p. 643-653.

ISO 26262. Road vehicles - Functional safety. International Standard, 2018.

KALINOWSKI, Fábio; MARTINS, Felipe Furtado; SILVA, Ivan Tavares de Lima. *Evaluation of system performance in automotive embedded systems using HIL simulation*. In: INTERNATIONAL AUTOMOTIVE TECHNOLOGY CONFERENCE,

2019, São Paulo. Anais... São Paulo: SAE Brasil, 2019. Disponível em: https://doi.org/10.4271/2019-36-0154. Acesso em: 12 abr. 2025.

KOENIGS, S.; LOHMEYER, A.; REUSS, H. Hardware-in-the-loop simulation for automotive development. Automotive Systems Engineering Review, 2019. p. 97-112.

MATHWORKS. Provetech TA - Model-Based Development for AUTOSAR. Disponível em: https://www.mathworks.com/products/connections/product_detail/provetech-ta.html. Acesso em: 15 fev. 2025.

MATHWORKS. *Daimler Designs Cruise Controller for Mercedes-Benz Trucks*. Disponível em: https://www.mathworks.com/company/user_stories/daimler-designs-cruise-controller-for-mercedes-benz-trucks.html. Acesso em: 12 abr. 2025.

MATHWORKS. *Hardware-in-the-Loop Simulation*. 2023. Disponível em: https://www.mathworks.com/discovery/hardware-in-the-loop.html. Acesso em: 12 abr. 2025.

PESARIN, F. The V-Model in Automotive Development. Automotive Systems Review, 2018. p. 103-120.

RAJAMANI, R. Vehicle Dynamics and Control. Springer, 2011.

SANTOS, Jeeves Lopes dos; VASCONCELOS, Felype Nery de Oliveira; SILVA, Elaine Cristina Guglielmoni; GUIMARÃES, André Rolim Almeida; SILVA, Alisson Sabarense da. *A method to calculate hardware in the loop applications representativeness*. SAE Technical Paper 2018-36-0171, 2018. Disponível em: https://doi.org/10.4271/2018-36-0171.

SCHUETTE, S.; THALER, F.; THOMA, M.; MÜLLER, M. Efficient validation of autonomous driving functions using virtual scenarios in a hardware-in-the-loop framework. IEEE Transactions on Intelligent Vehicles, 2020. p. 709-719.

SCHWABER, K.; SUTHERLAND, J. The Scrum Guide: The Definitive Guide to Scrum. Scrum Guides, 2017.

SMARTSHEET. Agile vs. Scrum vs. Waterfall vs. Kanban. Disponível em: https://www.smartsheet.com/agile-vs-scrum-vs-waterfall-vs-kanban. Acesso em: 15 fev. 2025.

VECTOR. VN16xx Network Interfaces - CAN, LIN, FlexRay, Ethernet. Disponível em: https://www.vector.com/br/pt/produtos/products-a-z/hardware/network-interfaces/vn16xx/. Acesso em: 15 fev. 2025.