Certified Tester

Automotive Software Tester (CT-AuT)

Syllabus

v2.1

International Software Testing Qualifications Board

BSTQB®
CT-AuT

Direitos autorais

Histórico da Revisão

Versão	Data	Observações
2.1	2025/04/30	Atualização menor: alterações no conteúdo existente afetadas por atualizações de padrões referenciados, glossário ISTQB e ISTQB CTFL v4. Redação aprimorada para melhor compreensão. Transferido para o modelo de syllabus atual. Consulte Apêndice C – Notas da Versão para obter detalhes.
2.0.2	2018/07/04	Aprovação da GA da versão em inglês (tradução em inglês da V2.0 DE)
2.0 DE	2017/03/31	Objetivos de aprendizado e conteúdo com base em V.1.1. Lançamento da edição (correspondente em alemão) por reunião do GTB WG de 31.03.2017 (Frankfurt a. M.).
1.1 DE	2015/06/14	Revisão do conteúdo e comparação com o syllabus alemão ISTQB® Certified Tester Foundation Level 2011 V1.0.1 e o Glossário ISTQB® V2.2 Liberação por reunião do grupo de trabalho GTB de 15.03.2015 (Munique).
1,0 DE	2011/01/19	Autor: Dr. Hendrik Dettmering, desenvolvido a pedido da gasq GmbH Os direitos autorais foram totalmente transferidos para a German Testing Board e. V.

Histórico da versão de tradução do BSTQB

Data	Observações
2025/08/04	Lançamento da versão na Língua Portuguesa

Índice

Agradecimentos

Este documento foi formalmente lançado pela Assembleia Geral do ISTQB® em 2018. A versão revisada 2.1 foi formalmente liberada pelo Product Owner CT-AuT, Horst Pohlmann, de acordo com o processo

Ele foi produzido por uma equipe do German Testing Board para o International Software Testing Qualifications Board: Ralf Bongard (líder), Klaudia Dussa-Zieger, Matthias Friedrich, Thorsten Geiselhart, Horst Pohlmann, Ralf Reißing, Alexander Schulz.

A equipe agradece a Michael Humm pelas revisões técnicas, à equipe de revisão e aos Conselhos de Administração pelas sugestões e contribuições.

As seguintes pessoas participaram da revisão, dos comentários e da votação deste syllabus:

Laura Albert, Ádám Bíró, Darvay Tamás Béla, Yuliia Fomuliaieva, Matthias Hamburg, Jarek Hryszko, Joanna Kazun, Imre Mészáros, Krisztián Miskó, Gary Mogyorodi, Ingvar Nordström, Mirosław Panek, Lukáš Piška, Meile Posthuma, Márton Siska, Klaus Skafte, Radosław Smilgin, Michael Stahl.

A equipe agradece a Gary Mogyorordi pela edição técnica.

Agradecimentos do BSTQB

O BSTQB® agradece à equipe do BSTQB WGT (Grupo de Trabalho de Traduções do BSTQB) pelo empenho em traduzir este material. Atuaram na tradução e revisão: George Fialkovitz Jr., Irene Nagase, Osmar Higashi, Paula Oliveira F., Rogério Athaide de Almeida, Stênio Viveiros, Thiago Cesar Andrade.

O BSTQB® também agradece aos ISTQB® Accredited Training Providers no Brasil que contribuíram na revisão da tradução com sugestões e questionamentos. No momento deste trabalho os seguintes provedores estavam credenciados: Conecteseaqui, Exseed, Iterasys.

O BSTQB® também agradece às empresas brasileiras credenciadas no ISTQB® Partner Program que contribuíram na revisão da tradução com sugestões e questionamentos. No momento deste trabalho as seguintes empresas estavam credenciadas: Deltapoint, Keeggo, Venturus.

0 Introdução

0.1 Objetivo deste Syllabus

Esse syllabus forma a base para a Qualificação Internacional de Teste de Software para o Testador de Software Automotivo. O ISTQB® fornece esse syllabus da seguinte forma:

1. Aos conselhos membros, para traduzir para seus idiomas locais e credenciar os provedores de treinamento. Os conselhos membros podem adaptar o syllabus às suas necessidades específicas de idioma e modificar as referências para adaptá-las às suas publicações locais.
2. Aos órgãos de certificação, para que elaborem questões de exame em seus idiomas locais, adaptadas aos objetivos de aprendizagem deste syllabus.
3. Aos provedores de treinamento, para produzir materiais de treinamento e determinar métodos de ensino adequados.
4. Para candidatos à certificação, para se preparar para o exame de certificação (como parte de um curso de treinamento ou de forma independente).
5. Para a comunidade internacional de engenharia de software e sistemas, para promover a profissão de teste de software e sistemas e como base para livros e artigos.

0.2 O Testador Certificado de Software Automotivo

A qualificação de Testador de Software Automotivo destina-se a qualquer pessoa envolvida em testes de software no domínio automotivo. Isso inclui funções como testadores, analistas de teste, engenheiros de teste, consultores de teste, gerentes de teste, testadores de aceite do usuário e desenvolvedores de software. A qualificação de Testador de Software Automotivo também é adequada para qualquer pessoa que busque uma compreensão básica dos testes de software automotivo, como gerentes de projeto, gerentes de segurança, gerentes de qualidade, gerentes de desenvolvimento de software, analistas de negócios, diretores de TI e consultores de gerenciamento.

0.3 Carreira para testadores

O esquema ISTQB® oferece suporte para profissionais de teste em todos os estágios de suas carreiras. As pessoas que obtiverem a certificação ISTQB® Certified Tester Automotive Software Tester também podem se interessar pelos níveis Core Advanced (Analista de Testes, Analista de Testes Técnicos e Gerente de Testes) e, posteriormente, pelo nível Expert (Gerenciamento de Testes ou Melhoria do Processo de Testes). O fluxo de Especialista oferece um mergulho profundo em áreas que têm abordagens de teste e atividades de teste específicas, por exemplo, em Teste Ágil, Teste de IA ou Teste de Aplicativo Móvel, ou que agrupam o know-how de teste para determinados domínios do setor, por exemplo, Jogos de Azar ou Jogos de Azar. Acesse www.istqb.org para obter as informações mais recentes sobre o ISTQB's Certified Tester Scheme.

0.4 Resultados de negócio

Esta seção lista os resultados de negócio esperados de um candidato que tenha obtido a certificação Automotive Software Tester.

Uma pessoa certificada como Testador de Software Automotivo pode...

AuT-BO1	Colaborar efetivamente em uma equipe de teste. ("colaborar")
AuT-BO2	Adaptar as técnicas de teste conhecidas do ISTQB® Certified Tester Foundation level (CTFL®) aos requisitos específicos do projeto. ("adaptar")

AuT-BO3	Considere os requisitos básicos dos padrões relevantes (ou seja, Automotive SPICE® e ISO 26262) para a seleção das técnicas de teste adequadas. ("selecionar")
AuT-BO4	Apoiar a equipe de teste no planejamento baseado em riscos das atividades de teste e aplicar elementos conhecidos de estruturação e priorização. ("support & apply")
AuT-BO5	Aplicar os ambientes de teste virtuais (ou seja, MiL, SiL e HiL) em ambientes de teste. ("aplicar")

0.5 Objetivos de aprendizado e nível cognitivo de conhecimento

Os objetivos de aprendizagem apoiam os resultados de negócio e são usados para criar os exames Certified Tester Automotive Software Tester.

Os níveis dos objetivos específicos de aprendizagem são mostrados no início de cada capítulo, e classificados da seguinte forma:

- K1: Lembre-se
- K2: Compreender
- K3: Aplicar

Mais detalhes e exemplos de objetivos de aprendizagem são fornecidos no Apêndice A.

Para todos os termos listados como palavras-chave logo abaixo dos títulos dos capítulos, o nome e a definição corretos do glossário do ISTQB® devem ser lembrados (K1), mesmo que não sejam explicitamente mencionados no objetivo de aprendizagem.

0.6 Exame de certificação do Certified Tester Automotive Software Tester

O exame do certificado Certified Tester Automotive Software Tester será baseado neste syllabus. As respostas às perguntas do exame podem exigir o uso de material baseado em mais de uma seção deste syllabus. Todas as seções do syllabus são passíveis de exame, exceto a Introdução, os Apêndices e as seções marcadas como apenas informativas. As perguntas do exame confirmarão o conhecimento das palavras-chave e dos objetivos de aprendizagem em todos os níveis K.

Padrões e livros são incluídos como referências, mas seu conteúdo não é passível de exame, além do que é resumido no próprio syllabus a partir desses padrões e livros.

Para obter mais detalhes, consulte o documento Exam Structures and Rules para o documento Certified Tester Automotive Software Tester Syllabus.

Para fazer o exame de Certified Automotive Software Tester, os candidatos devem ter o certificado ISTQB® Certified Tester - Foundation Level (CTFL®) e interesse em testes em projetos de desenvolvimento automotivo

No entanto, é altamente recomendável que os candidatos também o façam:

- ter pelo menos um conhecimento básico mínimo em desenvolvimento ou teste de software (p. ex., seis meses de experiência como testador de software ou desenvolvedor), ou
- ter feito um curso credenciado de acordo com o padrão ISTQB® (por um conselho de membros do ISTQB®) e/ou
- ter adquirido experiência inicial em testes de projetos de desenvolvimento de sistemas elétricos/eletrônicos (E/E) no setor automotivo

Nota sobre os requisitos de entrada: O certificado ISTQB® Foundation Level deve ser obtido antes de fazer o exame de certificação Automotive Software Tester

0.7 Credenciamento

Um Conselho Membro do ISTQB® pode credenciar provedores de treinamento cujo material do curso siga este syllabus. Os provedores de treinamento devem obter as diretrizes de credenciamento do Conselho de Membros ou do órgão que realiza o credenciamento. Um curso credenciado é reconhecido como estando em conformidade com este syllabus e pode ter um exame ISTQB® como parte do curso.

As diretrizes de credenciamento para este syllabus seguem as diretrizes gerais de credenciamento publicadas pelo Processes Management and Compliance Working Group.

0.8 Normas Técnicas

Organizações internacionais de padronização, como IEEE e ISO, emitiram normas associadas a características de qualidade e testes de software. Esses padrões são referenciados neste syllabus. O objetivo dessas referências é fornecer uma estrutura (como nas referências à ISO/IEC 25010 com relação às características de qualidade) ou fornecer uma fonte de informações adicionais, se o leitor desejar. Observe que os syllabus estão usando Normas como referência. As Normas não se destinam a exame. Consulte o capítulo 7 para obter mais informações sobre Normas.

0.9 Nível de detalhe

O nível de detalhamento desse syllabus permite cursos e exames consistentes em nível internacional. Para atingir esse objetivo, o syllabus consiste em:

- Objetivos gerais de instrução que descrevem a intenção do syllabus do Certified Tester Automotive Software Tester
- Uma lista de termos que os alunos devem ser capazes de lembrar
- Objetivos de aprendizagem para cada área de conhecimento, descrevendo o resultado cognitivo de aprendizagem a ser alcançado
- Uma descrição dos principais conceitos, incluindo referências a fontes como literatura ou padrões aceitos

O conteúdo do syllabus não é uma descrição de toda a área de conhecimento de teste de software automotivo; ele reflete o nível de detalhes a ser abordado nos cursos de treinamento do Certified Tester Automotive Software Tester. Ele se concentra em conceitos e técnicas de teste que se aplicam a projetos de software no domínio automotivo.

O syllabus utiliza a terminologia (ou seja, o nome e o significado) usada em testes de software e garantia de qualidade de acordo com o Glossário ISTQB®.

Para conhecer a terminologia das disciplinas relacionadas, consulte os respectivos glossários: ISO 26262, para engenharia de segurança funcional, e IREB® Glossary, para engenharia de requisitos.

0.10 Como este plano de estudos está organizado

Há quatro capítulos com conteúdo passível de exame. O título de nível superior de cada capítulo especifica o tempo para o capítulo; o tempo não é fornecido abaixo do nível do capítulo. *Para cursos de treinamento credenciados, o syllabus exige um mínimo de 12* horas de instrução, distribuídas pelos quatro capítulos da seguinte forma:

Capítulo 1: Introdução ao testador de software automotivo (30 minutos)

- Os testadores aprendem sobre os desafios do desenvolvimento de produtos automotivos, incluindo metas opostas, complexidade crescente e o impacto dos padrões sobre tempo, custo, qualidade e riscos.

- Eles também aprendem sobre as seis fases do ciclo de vida de um produto e sua função no lançamento de produtos.

Capítulo 2: Normas para o teste de sistemas elétricos/eletrônicos E/E (300 minutos)

- Os testadores aprendem sobre o ASPICE, abrangendo seus níveis de capacidade, processos relevantes para o teste, requisitos de documentação e a função de uma estratégia de teste, juntamente com as expectativas de rastreabilidade e estratégia de teste do ASPICE.
- Para a ISO 26262, os testadores se concentram na cultura de segurança, nos níveis de integridade da segurança automotiva, nas técnicas de teste e na compreensão de sua função no ciclo de vida da segurança.
- Os testadores também entendem a influência da AUTOSAR nos testes e comparam os objetivos da ASPICE, ISO 26262 e CTFL® e os respectivos níveis de teste.

Capítulo 3: Testes em um ambiente virtual (160 minutos)

- Os testadores aprendem sobre a finalidade, a estrutura e as funções essenciais dos ambientes de teste virtuais automotivos, incluindo sistemas de loop fechado e de loop aberto, e adquirem uma compreensão dos diferentes ambientes de teste XiL (ou seja, MiL, SiL e HiL), suas aplicações, vantagens e limitações.
- Eles também aprendem a atribuir escopos de teste apropriados a cada ambiente de teste.

Capítulo 4: Teste estático e teste dinâmico (230 minutos)

- Os testadores aprendem a explicar e aplicar diretrizes essenciais para códigos, como o MISRA-C, e requisitos, como o ISO/IEC/IEEE 29148.
- Os testadores criam casos de teste para cobertura de condições/decisões modificadas, compreendem a injeção de falhas e os testes back-to-back e selecionam técnicas e abordagens de teste adequadas com base no contexto do projeto.

1 Introdução ao Teste de Software Automotivo - 30 min.

Palavras-chave

nenhum

Objetivos de aprendizado para o Capítulo 1

1.1 Requisitos de objetivos de projeto divergentes e complexidade crescente do produto

AuT-1.1. (K2) Explique e dê exemplos dos desafios do desenvolvimento de produtos automotivos que surgem dos objetivos divergentes do projeto e da crescente complexidade do produto

1.2 Aspectos do projeto influenciados pelos padrões

AuT-1.2 (K1) Recordar os aspectos do projeto que são influenciados pelos padrões (p. ex., tempo, custo, qualidade, riscos do projeto e riscos do produto)

1.3 Os seis estágios genéricos do ciclo de vida do sistema

AuT-1.3 (K1) Relembrar os seis estágios genéricos do ciclo de vida do sistema de acordo com a ISO/IEC/IEEE 24748-1

1.4 A contribuição e a participação do testador no processo de liberação

AuT-1.4 (K1) Relembrar a função (ou seja, contribuição e colaboração) do testador no processo de liberação

Introdução

Um dos sete princípios do teste de software é "O teste depende do contexto". Este capítulo descreve o contexto de desenvolvimento do sistema de E/E no qual um "Testador de Software Automotivo"[1] opera. Por um lado, metas conflitantes, como eficiência de custo, requisitos de segurança e tempo de colocação no mercado rápido, complexidade crescente e demanda intensa por soluções inovadoras levam a desafios específicos. Por outro lado, os padrões e o ciclo de vida dos veículos formam a estrutura na qual o testador está trabalhando. No final, o testador está trabalhando para liberar o software e os sistemas.

1.1 Requisitos de objetivos de projeto divergentes e aumento da complexidade do produto

Os fabricantes de equipamentos originais (OEMs) e os fornecedores estão lançando novos modelos de automóveis[2] com mais frequência do que no passado, apesar de enfrentarem pressões de custo cada vez maiores. Os seguintes aspectos influenciam esse processo:

- Aumento do número de modelos e da complexidade:
  Para poder atender melhor às necessidades individuais do cliente final, os OEMs oferecem cada vez mais modelos de carros. No entanto, isso reduz as quantidades por modelo. Para cobrir os aumentos resultantes nos custos de desenvolvimento e produção, os produtores desenvolvem diversos modelos como variedades de uma plataforma comum. O desenvolvimento de uma plataforma, no entanto, é muito mais complexo do que o desenvolvimento de um único modelo, devido à necessidade de manter o controle sobre as muitas variações possíveis.
- Aumento da gama de funcionalidades:
  O cliente final solicita cada vez mais inovações enquanto mantém as funções existentes, o que faz com que a gama de funções aumente.
- Número cada vez maior de configurações:
  O cliente final quer ajustar seu modelo de carro de acordo com seus desejos individuais. Isso requer uma ampla gama de configurações para um único modelo de carro, inclusive em termos de funcionalidade.
- Aumento dos requisitos de qualidade:
  Apesar dos níveis crescentes de funcionalidade e complexidade, o cliente final espera a mesma qualidade do veículo e de suas funções, ou até mesmo uma qualidade superior.

Como os objetivos do projeto - tempo, custo e escopo - estão competindo (ou seja, conhecidos como o "triângulo do gerenciamento de projetos", sendo que a qualidade geralmente é considerada um aspecto do escopo), os OEMs e os fornecedores devem se esforçar para obter um desenvolvimento de sistema mais eficiente, que permita tempos de desenvolvimento mais curtos, apesar da complexidade crescente, dos requisitos de qualidade cada vez maiores e dos orçamentos menores.

1.2 Aspectos do projeto influenciados pelos padrões

Os padrões influenciam os principais aspectos do projeto, como tempo, custo, qualidade, riscos do projeto e riscos do produto:

[1] A seguir, usaremos apenas o termo "Testador": Ele deve ser entendido como a forma abreviada de "Testador de software automotivo".

[2] Exemplo de um estudo da consultoria de gestão Progenium: "Em 1990, apenas 101 modelos diferentes de carros eram oferecidos... em 2014, esse número aumentou para 453"

Além disso, "a próxima década de transformação da mobilidade coloca o consumidor em primeiro lugar" traz novos desafios

- Os padrões aumentam a eficiência dos processos (exemplo, reduzindo o tempo ou o custo de desenvolvimento e mantendo a qualidade estável):
  - nomeação de uniformes
  - maior transparência
  - colaboração mais fácil (ou seja, interna e externa)
  - maior capacidade de reutilização
  - experiência consolidada ("Melhores práticas")
- Com diretrizes tecnológicas bem estabelecidas, eles ajudam a descobrir riscos e defeitos antecipadamente e a resolver os riscos e defeitos identificados.
- As normas são a base para as auditorias. Portanto, um auditor pode avaliar a qualidade de um produto ou processo. Ao mesmo tempo, o auditor pode verificar se os padrões atendem aos requisitos.
- Os padrões fazem parte das disposições e diretrizes contratuais ou regulamentares.

Este syllabus faz referência a normas relevantes para o teste de software automotivo, incluindo:

- ISO 26262 e ASPICE, que fornecem processos e métodos padronizados
- AUTOSAR, que padroniza a arquitetura e os produtos de software

1.3 Os seis estágios genéricos do ciclo de vida do sistema

O ciclo de vida do sistema de um carro e seus componentes[3] começa com a ideia do produto e termina com a desativação. Ele envolve processos de desenvolvimento, negócios, logística e tecnologia de produção.

Os marcos com critérios de entrada e saída definidos garantem processos maduros e estruturam o ciclo de vida do sistema[4] em seis estágios com atividades de teste típicas:[5]

- Conceito: Planejamento de teste para definir a estratégia e os objetivos do teste
- Desenvolvimento: Análise de teste, projeto de teste, implementação de teste, execução de teste, monitoramento de teste, controle de teste e conclusão de teste garantem a qualidade.
- Produção: Teste de fim de linha para verificar a prontidão
- Utilização: Normalmente, não são realizadas atividades de teste.
- Suporte: Testes de manutenção para garantir a confiabilidade contínua
- Aposentadoria: O teste de migração valida o descomissionamento ou a transferência de dados.

O processo de desenvolvimento de produtos automotivos amplamente utilizado costuma condensá-los em três estágios principais: conceito, desenvolvimento e produção, onde ocorre a maioria das atividades de teste e lançamento.

1.4 A contribuição e a participação do testador no processo de liberação

No ambiente automotivo, um projeto atinge um marco ao declarar uma versão quando os objetivos são verificados como atingidos. A partir desse momento, considera-se que o item de liberação tem a maturidade necessária para

[3] ECUs (hardware e software), bem como componentes.

[4] O ciclo de vida da segurança da ISO 26262 consiste em fases semelhantes.

[5] Para atividades de teste, consulte também: processo de teste fundamental.

o uso pretendido. O item de liberação inclui a configuração do software, inclusive a parametrização e, se necessário, o hardware e mecânica, além da documentação de apoio.

O testador fornece informações importantes para o processo de liberação por meio do relatório de teste final:

- itens testados
- características de qualidade avaliadas (p. ex., tempo de resposta e uso de recursos)
- defeitos conhecidos
- métricas do produto
- informações para recomendação de liberação ao atingir os critérios de saída com base no nível de liberação, conforme definido nas Diretrizes de Melhores Práticas (p. ex., testes em terrenos fechados, testes em vias públicas e recomendações de instalação)

Além disso, o testador participa da criação de outros produtos relevantes para o lançamento:

- priorizar e participar da decisão sobre mudanças
- priorizar os recursos para a ordem de implementação

2 Normas para o Teste de Sistemas Elétricos/Eletrônicos (E/E) - 300 min.

Palavras-chave

segurança funcional, tabela de métodos

Palavras-chave específicas do domínio

nível de integridade de segurança automotiva (ASIL), verificação de software, sistema

Objetivos de aprendizado para o Capítulo 2

2.1 SPICE automotivo (ASPICE)

AuT-2.1.1.1 (K1) Recordar as duas dimensões do ASPICE

AuT-2.1.1.3 (K2) Explicar os níveis de capacidade de 0 a 3 do ASPICE

AuT-2.1.2.1 (K1) Recordar a finalidade dos processos específicos de teste do ASPICE

AuT-2.1.2.2 (K2) Explicar o significado dos quatro níveis de classificação e dos indicadores de capacidade do ASPICE a partir da perspectiva de testes

AuT-2.1.2.3 (K2) Explicar os requisitos do ASPICE para uma estratégia de teste, incluindo os critérios para verificação de regressão

AuT-2.1.2.4 (K1) Relembrar os requisitos do ASPICE para o testware

AuT-2.1.2.5 (K3) Usar medidas de verificação para verificação de unidades de software

AuT-2.1.2.6 (K2) Explicar os diferentes requisitos de rastreabilidade do ASPICE do ponto de vista dos testes

2.2 ISO 26262

AuT-2.2.1.1 (K2) Explicar o objetivo da segurança funcional para sistemas de E/E

AuT-2.2.1.2 (K1) Recordar a contribuição dos testadores para a cultura de segurança

AuT-2.2.2 (K2) Discuta a função do testador na estrutura do ciclo de vida da segurança de acordo com a ISO 26262

AuT-2.2.3.2 (K1) Relembrar as partes da ISO 26262 que são relevantes para o testador

AuT-2.2.4.1 (K1) Recordar os níveis de criticidade da ASIL

AuT-2.2.4.2 (K2) Explicar a influência da ASIL nas técnicas de teste e nos tipos de teste para testes estáticos e testes dinâmicos e o escopo de teste resultante

AuT-2.2.5 (K3) Usar as tabelas de métodos da ISO 26262

2.3 AUTOSAR

AuT-2.3.1 (K1) Recordar os objetivos do projeto AUTOSAR

AuT-2.3.3 (K1) Recordar a influência da AUTOSAR no trabalho do testador

2.4 Comparação entre ASPICE, ISO 26262 e CTFL®

AuT-2.4.1 (K1) Relembrar os diferentes objetivos da ASPICE e da ISO 26262

AuT-2.4.2 (K2) Explicar as diferenças entre ASPICE e ISO 26262 e CTFL® com relação aos níveis de teste

2.1 SPICE Automotivo (ASPICE)

Introdução

A melhoria de processos segue a abordagem de que a qualidade de um sistema é influenciada pela qualidade do processo de desenvolvimento. Os modelos de processo oferecem uma opção de melhoria por meio da medição da capacidade do processo de uma unidade organizacional ou projeto em comparação com o modelo de referência. Além disso, o modelo serve como estrutura para a melhoria dos processos de uma unidade organizacional ou projeto usando os resultados da avaliação.

Desde 2001, o Grupo de Usuários do SPICE[6] e o Automotive Special Interest Group (AUTOSIG) desenvolveram o ASPICE. Desde 2005, a norma está bem estabelecida no setor automotivo. Todas as declarações feitas neste parágrafo referem-se à versão 4.0 do ASPICE.

2.1.1 Projeto e estrutura da Norma

2.1.1.1 As duas dimensões do ASPICE

O ASPICE define um modelo de avaliação com duas dimensões principais de avaliação:

Na **dimensão do processo**, a ASPICE define o Modelo de Referência do Processo (PRM). O PRM serve como referência para comparar os processos da organização com ele, para que possam ser avaliados e melhorados. Para cada processo, a ASPICE define a finalidade, os resultados, as ações necessárias (ou seja, práticas básicas) e os itens de informação (ou seja, resultados e produtos do trabalho).

Na **dimensão de capacidade,** a ASPICE define vários atributos de processo (ou seja, características mensuráveis) que dividem o nível de capacidade. Para cada processo, há atributos genéricos e específicos do processo. A ISO/IEC 33020 serve como base para a avaliação da capacidade do processo.

Com a ajuda desse modelo, é possível avaliar os processos (ou seja, a dimensão do processo) em relação à sua capacidade (ou seja, a dimensão da capacidade).

2.1.1.2 Categorias de processo na dimensão do processo [informativo].

A ASPICE categoriza os processos em grupos, que por sua vez são classificados em categorias:

Os processos primários do ciclo de vida incluem todos os processos que servem como processos-chave:

- Grupo de processo de aquisição (ACQ)
- Grupo de processos de suprimentos (SPL)
- Grupo de processos de engenharia de sistemas (SYS)
- Grupo de processos de engenharia de software (SWE)
- Grupo de processos de engenharia de aprendizado de máquina (MLE)
- Grupo de processos de engenharia de hardware (HWE)
- Grupo de processo de validação (VAL)

Os processos de ciclo de vida de suporte incluem todos os processos que dão suporte a outros processos:

- Grupo de processo de suporte (SUP)

[6] Acrônimo de "Software Process Improvement and Capability Determination" (Melhoria do processo de software e determinação de capacidade)

Os processos do ciclo de vida organizacional incluem todos os processos que apoiam os objetivos da empresa:

- Grupo de processos gerenciais (MAN)
- Grupo de processos de melhoria de processos (PIM)
- Grupo de processo de reutilização (REU

Para o testador, os grupos de processos de engenharia de sistemas (SYS) e engenharia de software (SWE) são de interesse especial, possivelmente também o MLE, VAL e HWE

2.1.1.3 Níveis de capacidade (CL) na dimensão de capacidade

O avaliador avalia a capacidade do processo com a ajuda de um sistema de avaliação de seis níveis (ou seja, exibição dos níveis). A ASPICE define os níveis de capacidade de 0 a 3[7] da seguinte forma:

- Nível 0 (processo incompleto): O processo não é implementado ou não consegue atingir seu objetivo. Nesse nível, há pouca ou nenhuma evidência de qualquer realização sistemática do resultado do processo. O processo não existe ou não atinge o objetivo do processo. Exemplo: O testador verifica apenas uma pequena parte dos requisitos.
- Nível 1 (processo executado): O processo implementado atinge seu objetivo (mas pode ser executado de forma inconsistente). Exemplo: Não há um planejamento completo visível para um processo de teste. Entretanto, o testador pode mostrar o nível de cumprimento dos requisitos.
- Nível 2 (processo gerenciado): O projeto planeja e supervisiona o processo de teste em sua execução. Em determinadas circunstâncias, ele adapta o curso de ação durante a execução do teste para atender aos objetivos do teste. Os requisitos para os produtos de trabalho são definidos. Um membro do projeto analisa os produtos de trabalho e os aprova. Exemplo: O gerente de testes define os objetivos do teste, planeja as atividades de teste e gerencia o processo de teste. Isso inclui reagir adequadamente aos desvios.
- Nível 3 (processo estabelecido): O projeto usa um processo de teste padronizado, e os resultados são usados para aprimoramento constante. Exemplo: Existe uma estratégia de teste organizacional. Após a conclusão do teste, o gerente de testes trabalha para desenvolvê-la ainda mais.

2.1.2 Requisitos da Norma

2.1.2.1 Processos específicos de teste do ASPICE

A ASPICE define os processos de teste de acordo com todos os processos de desenvolvimento de software e sistemas:

- Verificação da unidade de software (SWE.4) requer testes estáticos e dinâmicos. Ela avalia os componentes do software com base em seu projeto detalhado (SWE.3).
- Verificação de componentes de software e verificação de integração (SWE.5) avaliam o software integrado com base no projeto arquitetônico do software (SWE.2).
- Verificação de software (SWE.6) avalia o software integrado com base na análise de requisitos de software (SWE.1).
- Integração do sistema e verificação da integração (SYS.4) avaliam o sistema integrado com base no projeto arquitetônico do sistema (SYS.3).
- Verificação do sistema (SYS.5) avalia o sistema integrado com base na análise dos requisitos do sistema (SYS.2).

[7] Atualmente, os níveis de capacidade 4 e 5 não estão no foco do setor automotivo.

2.1.2.2 Níveis de classificação e indicadores de capacidade

Um avaliador pode avaliar a capacidade do processo por meio de indicadores de capacidade. O ASPICE os define para 9 atributos de processo (PA). Para os níveis de capacidade 1 a 3, eles são definidos da seguinte forma (usando o exemplo do SWE.6 entre parênteses):

- PA 1.1: Atributo do processo de performance do processo (p. ex., o testador é orientado no processo de teste)
- PA 2.1: Atributo do processo de gerenciamento da performance do processo (p. ex., o testador planeja, supervisiona e controla as atividades de teste)
- PA 2.2: Atributo do processo de gerenciamento de produtos de trabalho[8] (p. ex., o testador verifica a qualidade do testware)
- PA 3.1: Atributo do processo de definição do processo (p. ex., a pessoa responsável pelo processo de teste define uma estratégia de teste organizacional)
- PA 3.2: Atributo do processo de implantação do processo (p. ex., o testador aplica a estratégia de teste definida no PA 3.1)

Para a execução do processo, o ASPICE define dois tipos de indicadores de capacidade: Práticas Básicas (BP) e itens de informação. Além disso, são definidas Práticas Genéricas (GP) e recursos. A avaliação dos atributos do processo é baseada no nível de implementação dos indicadores em quatro níveis de classificação:

- **N** (None): não preenchido (0% até ≤ 15%)
- **P** (Partly): parcialmente cumprido (> 15% até ≤ 50%)
- **L** (Largely): amplamente cumprido (> 50% até ≤ 85%)
- **F** (Fully): totalmente cumprido (> 85% até ≤ 100%)

Para que um processo atinja um determinado nível de capacidade, os indicadores de capacidade devem ser alcançados de forma amplamente atendida (L) ou totalmente atendida (F)

2.1.2.3 Estratégia de teste e critérios para verificação de regressão

O ASPICE exige uma estratégia de teste (ou seja, medidas de verificação) para cada processo de teste (consulte a seção 2.1.2.1) a partir do nível de capacidade 2. O gerente de teste a desenvolve durante o planejamento do teste. As diretrizes de teste, os objetivos do projeto e os requisitos contratuais e regulamentares formam a base para isso.

O testador está ciente dos testes antecipados como um princípio de teste. Isso também se aplica ao teste de software no ambiente automotivo. No entanto, outro aspecto entra em jogo aqui, porque os ambientes de teste em níveis de teste mais altos são significativamente mais caros. Por exemplo, para testar em níveis mais altos, é necessário um hardware especialmente desenvolvido e incorporado (p. ex., como um protótipo ou modelo exclusivo). A estratégia de teste define os ambientes de teste específicos do nível, mas também quais testes o testador deve executar em quais ambientes de teste.

Os critérios para a verificação de regressão são uma parte essencial da estratégia de teste. Eles especificam os detalhes da verificação de regressão[9] . O desafio está na escolha economicamente sensata dos casos de teste (ou seja, o valor agregado do teste). Os critérios para a verificação de regressão definem o objetivo do teste e a abordagem do teste para a escolha dos testes de regressão. Por exemplo, a escolha pode ser baseada em riscos. Uma análise de impacto ajuda a identificar as áreas nas quais o testador deve se concentrar com os testes de

[8] O gerenciamento de produtos de trabalho se aplica a todas as informações documentadas

[9] Semelhante à estratégia de teste avesso à regressão no ISTQB

regressão. No entanto, o gerente de testes também pode solicitar que o testador repita todos os casos de teste automatizados para cada versão.

2.1.2.4 Testware no ASPICE

O ASPICE 4.0 não impõe requisitos ao material de teste. Em vez disso, estabelece requisitos relativos à documentação de informações específicas que surgem durante o teste. Para as atividades de teste, o ASPICE exige os seguintes itens de informação (II):

- 08-60: Medidas de verificação (p. ex., simulações, testes dinâmicos e testes estáticos), normalmente documentadas em um plano de teste
- 13-25: Resultados da verificação (p. ex., registros de verificação documentados, relatórios de testes e relatórios de defeitos)

Para cada item de informação, o ASPICE define exemplos de características e conteúdo do item de informação (IIC). Eles servem como um indicador objetivo para a execução do processo.

A ISO/IEC/IEEE 29119-3 pode ser usada para estruturar ainda mais os itens de informação.

2.1.2.5 Medidas de verificação para verificação de unidades de software

O testador verifica a conformidade com o projeto detalhado do software e com os requisitos funcionais e não funcionais de acordo com as medidas de verificação. As medidas definem como o testador fornece as evidências. O testador pode usar combinações de técnicas de teste estáticas e dinâmicas para verificar as unidades de software.

No SWE.4. BP.1, o ASPICE exige a definição de medidas de verificação (consulte a seção 2.1.2.4) para a verificação de unidades de software. Isso permite que o testador avalie até que ponto a unidade de software atende aos seus requisitos. Os exemplos a seguir podem ser usados como medidas de verificação da unidade de software:

- Casos de teste de unidade de software, incluindo dados de teste
- Análise estática com suporte de ferramentas, que avalia a conformidade com as diretrizes de codificação (p. ex., MISRA-C, consulte a seção 4.1.1)
- Revisões de unidades de software ou partes de unidades de software que não podem ser avaliadas por análise estática com suporte de ferramentas

Se um desenvolvedor alterar uma unidade de software, o testador também deverá avaliar essa alteração. Portanto, as medidas de verificação das unidades de software também incluem os critérios para a verificação de regressão. Isso inclui a verificação do código alterado, o teste de confirmação e a verificação repetida das partes não alteradas (testes de regressão estáticos e dinâmicos).

2.1.2.6 Rastreabilidade no ASPICE

Como no CTFL® Core Syllabus [ISTQB_FL_SYL], o ASPICE também exige rastreabilidade bidirecional[10] . Isso permite que os testadores:

- Analisar o impacto, por exemplo, análise do impacto da mudança
- Avaliar a cobertura
- Status do rastreamento

[10] A seguir, o termo rastreabilidade sempre implicará a rastreabilidade bidirecional.

Além disso, isso permite que o(s) testador(es) garanta(m) a consistência entre os elementos vinculados, textual e semanticamente.

A ASPICE diferencia entre rastreabilidade vertical e horizontal:

Verticalmente: A ASPICE exige que os requisitos das partes interessadas sejam vinculados em todos os níveis às unidades de software. Ao fazer isso, o vínculo em todos os níveis de desenvolvimento garante a consistência entre os produtos de trabalho relacionados.

Horizontalmente: A ASPICE exige rastreabilidade e consistência entre os resultados do trabalho de desenvolvimento e as especificações e os resultados dos testes correspondentes.

Além disso, a prática básica SUP.10.BP4 exige rastreabilidade bidirecional entre as solicitações de mudança e os produtos de trabalho afetados pelas solicitações de mudança. Como as solicitações de mudança geralmente são iniciadas por um defeito, a rastreabilidade bidirecional é estabelecida entre as solicitações de mudança e os relatórios de defeitos correspondentes. Devido ao número ocasionalmente grande de links, uma cadeia consistente de ferramentas pode ser útil. Isso permite que o testador crie e gerencie as dependências de forma eficiente.

2.2 ISO 26262

2.2.1 Segurança funcional e Cultura de Segurança

2.2.1.1 Objetivo da segurança funcional para sistemas de E/E

A complexidade funcional e técnica dos sistemas incorporados está aumentando constantemente. Ao mesmo tempo, sistemas elétricos e eletrônicos poderosos baseados em software permitem novas funcionalidades complexas, como a automação das funções de direção de um carro.

Devido ao aumento da complexidade, o risco de uma ação errônea ocorrer durante o desenvolvimento também está aumentando. A consequência pode ser um defeito não reconhecido no sistema. Para sistemas com um potencial de risco inerente para a vida e a integridade física, o engenheiro de segurança ou o gerente de segurança responsável precisa analisar os riscos potenciais. Se houver um risco real, é necessário identificar medidas adequadas para reduzir o possível impacto do risco a um nível aceitável.

Os métodos para a execução de tais análises estão resumidos nos padrões de segurança funcional . A Norma básica é o IEC 61508. A International Organization for Standardization (ISO) adaptou a ISO 26262 a partir dessa norma, que está disponível em sua segunda edição a partir de 2018.

A segurança funcional envolve a garantia de que os sistemas de E/E operem sem causar riscos razoáveis devido a perigos decorrentes do comportamento de mau funcionamento. Nesse sentido, o termo deve ser diferenciado de outros termos relacionados, como segurança da informação (ou segurança cibernética), segurança do produto ou segurança ocupacional.

A segurança ocupacional e a segurança cibernética não estão no foco da ISO 26262. No entanto, a falta de segurança cibernética pode colocar em risco a segurança funcional. Tanto a segurança funcional quanto a segurança cibernética contribuem para a segurança do produto.

2.2.1.2 Contribuição do testador para a cultura de segurança

No desenvolvimento de produtos de acordo com a ISO 26262, monitorar apenas os processos de sua própria organização não é suficiente. Todos os participantes precisam seguir uma abordagem de processo cruzado. Todos devem entender seu impacto no processo de desenvolvimento e na segurança do produto. Isso inclui parceiros e fornecedores externos.

Os participantes devem entender que suas próprias ações não acontecem independentemente de outros processos. Cada etapa do desenvolvimento constitui uma contribuição essencial para a conformidade e a implementação dos requisitos relevantes para a segurança. Essa responsabilidade não termina com o lançamento no mercado. Ela continua até o final do ciclo de vida da segurança.

Os testadores contribuem para a cultura de segurança participando de forma responsável das fases do ciclo de vida do desenvolvimento de software e realizando seu trabalho com uma visão contínua do contexto geral do desenvolvimento do produto.

2.2.2 Integração do testador no ciclo de vida da segurança

O ciclo de vida da segurança descreve as atividades voltadas para a segurança durante o desenvolvimento do produto. Ele começa com a primeira ideia de produto e a identificação de possíveis riscos. Após a especificação dos requisitos de segurança resultantes, segue-se a implementação em um produto específico. O ciclo de vida da segurança termina com o descarte do produto no final de sua vida útil.

O ciclo de vida da segurança, de acordo com a ISO 26262, passa pelas seguintes fases:

- 1ª fase: Fase de conceito;
- 2ª fase: Desenvolvimento do produto. Essa fase termina com a "liberação para produção";
- 3ª fase: Produção, operação, serviço e descomissionamento.

O testador trabalha principalmente nas duas primeiras fases. As alterações no produto na terceira fase levaram a um retorno à primeira ou à segunda fase, dependendo de sua extensão. Portanto, o testador também participa das modificações. Com base nos requisitos relacionados à segurança, as técnicas de teste são selecionadas e os casos de teste são projetados para a verificação e validação desses requisitos. Em seguida, o testador executará essas tarefas nas respectivas subfases do desenvolvimento do produto.

O planejamento do teste normalmente ocorre na fase de conceito. No entanto, podem ser necessários ajustes nos documentos resultantes (p. ex., no plano de teste ou nas especificações de teste) em qualquer fase. A execução do teste ocorre principalmente nas transições entre as subfases individuais do desenvolvimento do produto. Por exemplo, a transição da implementação para a integração de software e, em seguida, para a integração de hardware-software. Além disso, os testadores contribuem significativamente para a transição para a terceira fase com suas atividades de teste.

2.2.3 Estrutura e teste de partes específicas da norma

2.2.3.1 Projeto e estrutura da norma [informativo]

A ISO 26262 é composta por 12 partes:

- Vocabulário (parte 1)
- Gerenciamento da segurança funcional (parte 2)
- As fases do ciclo de vida da segurança:
  - Fase de conceito (parte 3)
  - Desenvolvimento de produtos em nível de sistema, hardware e software (partes 4-6)
  - Produção, operação, serviço e descomissionamento (parte 7)
- Processos de suporte (parte 8)
- Análise orientada para a ASIL e para a segurança (parte 9)
- Diretrizes sobre a ISO 26262 (parte 10)
- Diretrizes sobre a aplicação da ISO 26262 a semicondutores (parte 11)
- Adaptação da ISO 26262 para motocicletas (parte 12

Além das partes 1, 10 e 11, cada parte contém:

- Uma introdução geral
- O escopo da aplicação

- Referências normativas
- Requisitos para conformidade com a norma

Em seguida, são apresentados os tópicos específicos da parte correspondente. A estrutura de sua descrição é a mesma em cada parte. As atividades que devem ser realizadas são descritas por meio de uma estrutura recorrente em todas as partes:

- Objetivo
- Informações gerais
- Informações introdutórias
- Pré-requisitos
- Outras informações de apoio
- Requisitos e recomendações
- Resultados do trabalho

2.2.3.2 Partes relevantes da ISO 26262 para o testador

Para o testador de software, a verificação do software e, pelo menos parcialmente, a validação do sistema são fundamentais. Além da parte 1 (vocabulário), várias outras partes também são de interesse especial: as partes 4 e 6 fornecem informações detalhadas e requisitos referentes às medidas de verificação de software recomendadas. Isso se aplica à seleção, ao projeto, à implementação e à execução das medidas de verificação correspondentes.

Ao fazer isso, essas partes se concentram nos aspectos específicos de teste e verificação do nível do sistema (ou seja, parte 4, incluindo validação de segurança) e do nível do software (ou seja, parte 6). Se os aspectos específicos do hardware também forem relevantes para este trabalho, o testador os encontrará na parte 5. Os aspectos relativos ao hardware e ao software são considerados dentro do escopo da interface hardware software (partes 4-6).

A Parte 8 tem uma função especial, pois descreve as características específicas do processo de verificação em todos os níveis de teste . Além disso, ela contém requisitos para processos de suporte que são essenciais para garantir processos padronizados, como o gerenciamento de configuração e a qualificação de ferramentas.

Na parte 12, o testador encontrará tabelas de métodos adaptados com base na recém-introduzida avaliação do Nível de Integração de Segurança de Motocicletas (MSIL) para projetos de motocicletas.

2.2.4 A influência da criticidade na extensão do teste

2.2.4.1 Os níveis de criticidade da ASIL

O ASIL é uma medida para a redução de risco necessária por meio de medidas de segurança funcional. Essas medidas podem ser, por exemplo, uma função de segurança independente para monitorar um sistema de E/E ou a implementação de métodos especificamente definidos. Para níveis de risco mais altos, podem ser necessárias medidas mais elaboradas.

No início do projeto, uma equipe de especialistas realiza uma análise de perigos e avaliação de riscos (HARA) para o produto. Para cada perigo identificado por essa análise, a equipe determina um ASIL usando as metodologias definidas na norma. Na etapa seguinte, a equipe define as metas e os requisitos de segurança. Esses têm o mesmo ASIL que o perigo subjacente.

A ISO 26262 define quatro níveis: de ASIL A, para requisitos de segurança baixos, até ASIL D, para requisitos de segurança altos.

Se os resultados da análise de perigos e da avaliação de riscos levarem a requisitos abaixo do ASIL A, então, em termos da norma, eles não são relevantes para a segurança. Para esses requisitos, não são necessárias ações funcionais relacionadas à segurança, e é suficiente cobrir esses requisitos usando sistemas de gerenciamento de

qualidade (QMS) já existentes. Por motivos práticos, esses requisitos costumam ser marcados como "QM" para distingui-los dos relacionados à ASIL.

2.2.4.2 Influência da ASIL nas técnicas de teste, nos tipos de teste e no teste resultante escopo

O ASIL determinado influencia diretamente a extensão dos testes a serem realizados pelo testador. Dependendo do nível do ASIL, a norma ISO 26262 recomenda a execução de diferentes medidas ou pacotes de medidas. Ao fazer isso, a norma recomenda medidas mais extensas e detalhadas para ASIL mais alto. Para níveis mais baixos de ASIL, a execução das medidas especificadas geralmente é opcional.

A ISO 26262 especifica três níveis de recomendação: sem recomendação, recomendado e altamente recomendado. O entendimento geral desses níveis de recomendação é que, quando um método é listado como "altamente recomendado", ele deve ser aplicado, enquanto um método listado como "recomendado" deve ser aplicado ou deve ser fornecida uma justificativa para não o usar. No caso de "nenhuma recomendação", a norma não fornece nenhuma recomendação a favor ou contra o uso da medida em questão. Ela pode ser usada como uma medida de apoio sem nenhuma preocupação. Entretanto, sua execução não substitui as medidas recomendadas ou altamente recomendadas pela ISO 26262.

Para o testador, isso significa que a norma recomenda técnicas de teste específicas e tipos de teste para sistemas relevantes para a segurança, dependendo do ASIL. O testador só pode decidir livremente até o ponto em que a norma permite o caso especial. Por exemplo, o uso de particionamento de equivalência e análise de valor de limite são meramente recomendados para ASIL A. Por outro lado, para um ASIL B ou superior, essas técnicas são altamente recomendadas (consulte a seção 2.2.5).

O ASIL não é uma característica de todo o produto. Ele está vinculado a uma meta de segurança específica e aos requisitos de segurança derivados dela. Para o mesmo produto, os requisitos de segurança com ASILs diferentes podem, portanto, resultar em esforços de teste significativamente diferentes. Isso deve ser levado em consideração pelo testador ao planejar o escopo do teste.

2.2.5 Aplicação do conteúdo do CTFL® no contexto da ISO 26262

A ISO 26262 oferece ao testador recomendações específicas para as atividades de teste na forma de tabelas de métodos. Elas podem ser encontradas nas partes 4, 5, 6, 8 e 12. Além das recomendações específicas de segurança funcional para processos de teste e atividades de teste, elas também incluem as técnicas de teste a serem usadas.

Nesse contexto, a norma usa o termo "método" relacionado a todas as técnicas de teste ou atividades de teste aplicáveis. Nesse ponto, a terminologia de segurança funcional difere um pouco dos termos do ISTQB®. Para o testador, os seguintes métodos da ISO 26262 são de interesse especial:

- Técnicas de teste (p. ex., particionamento de equivalência e análise de valor limite)
- Técnicas para execução de testes (p. ex., simulação ou protótipo de um componente ou sistema)
- Tipos de teste (p. ex., testes não funcionais, como teste de performance e teste de resistência)
- Ambientes de teste (p. ex., hardware-in-the-loop e veículos)
- Técnicas de teste estático (p. ex., revisões e análise estática)

As tabelas de métodos definem os métodos recomendados pela norma para cada nível ASIL.

Eles são sempre projetados na mesma estrutura:

		ASIL A	ASIL B	ASIL C	ASIL D
1	Método x	o	+	++	++
2	Método y	o	o	+	+
3a	Método z1	+	++	++	++
3b	Método z2	++	+	o	o

Tabela1 : Exemplo de uma tabela de métodos

Para cada método, dependendo do nível ASIL, é documentado se seu uso é recomendado (+) ou mesmo altamente recomendado (++). Para os métodos marcados como opcionais (o), não há nenhuma recomendação fornecida pela norma a favor ou contra seu uso.

A ISO 26262 também lista métodos alternativos equivalentes nas tabelas de métodos usando sufixos de letras (no exemplo acima, linhas 3a e 3b). Aqui, o testador precisa escolher uma combinação adequada para poder verificar os requisitos relevantes de forma compatível com a ASIL. O testador deve justificar a utilidade da combinação selecionada.

No caso de métodos sem alternativas (p. ex., linhas 1 e 2), essa opção de escolha não é permitida. Aqui, o testador deve aplicar todos os métodos que são altamente recomendados para o respectivo nível ASIL.

Por exemplo, ao verificar um requisito de acordo com o ASIL C, os seguintes métodos são derivados da tabela 1:

- Método x: altamente recomendado, portanto deve ser aplicado ao desenvolver de acordo com a ISO 26262
- Método y: recomendado, portanto deve ser aplicado ou deve ser apresentada uma justificativa para não aplicar o método
- Métodos z1 e z2: aqui, pelo menos o método z1 deve ser escolhido, pois tem o nível mais alto de recomendação para ASIL C

A ISO 26262 também permite que o testador use outros métodos além dos listados nas tabelas de métodos. Nesse caso, o método escolhido deve ser justificado quanto à sua utilidade e adequação.

De modo geral, a ISO 26262 permite personalizar as atividades de segurança para atender às necessidades específicas de um determinado projeto ("adaptação"). Para atividades relacionadas a testes, isso significa que a estratégia e a abordagem de teste podem ser adaptadas ao projeto específico.

2.3 AUTOSAR

AUTOSAR significa "AUTomotive Open System Architecture". AUTOSAR é tanto a arquitetura quanto a parceria de desenvolvimento por trás dela. A parceria AUTOSAR foi estabelecida em 2003 e inclui principalmente OEMs e fornecedores da indústria automotiva. O objetivo da parceria é desenvolver e estabelecer conjuntamente uma norma aberta do setor para a arquitetura de software automotivo. Atualmente, a AUTOSAR é uma norma estabelecida globalmente para sistemas E/E automotivos. Portanto, o testador certamente se depara com produtos de trabalho AUTOSAR. Para o testador, é importante conhecer os objetivos do projeto AUTOSAR, a estrutura geral da arquitetura AUTOSAR e sua influência no trabalho do testador.

2.3.1 Objetivos do projeto AUTOSAR

Os objetivos do projeto AUTOSAR são os seguintes:

- Apoiar a portabilidade do software
- Apoiar a escalabilidade em diferentes arquiteturas e variantes de hardware
- Suporte a diferentes domínios funcionais
- Apoiar o intercâmbio de dados com sistemas não AUTOSAR
- Definir uma arquitetura aberta para software automotivo
- Apoiar o desenvolvimento de sistemas confiáveis caracterizados por disponibilidade, confiabilidade, segurança, integridade, capacidade de manutenção e
- Apoiar a colaboração entre os parceiros
- Apoiar os padrões automotivos internacionais aplicáveis e as tecnologias de ponta

2.3.2 Estrutura geral do AUTOSAR [Informativo]

Há duas variantes de arquitetura no AUTOSAR: Clássica e Adaptativa. Para simplificar, este syllabus se concentra no AUTOSAR Classic. Um sistema AUTOSAR normalmente consiste em várias unidades de controle eletrônico (ECUs). Em cada ECU, a arquitetura de software do AUTOSAR Classic consiste em três camadas:

- A camada superior do aplicativo é independente do hardware. Um componente dessa camada é chamado de componente de software AUTOSAR (SW-C).
- A camada inferior orientada por hardware é o software básico padronizado (BSW).
- A camada de abstração intermediária é o ambiente de tempo de execução AUTOSAR (RTE). Como um tipo de middleware, ele implementa o fluxo de dados entre SW-Cs e entre SW-Cs e BSW.

Na metodologia AUTOSAR para o desenvolvimento de sistemas automotivos, OEMs e fornecedores trocam informações sobre o sistema usando arquivos de descrição baseados em modelos AUTOSAR (os chamados "arquivos arxml"):

- A "Descrição da configuração da ECU" inclui dados para a integração dos SW-Cs em uma única ECU.
- A "descrição da configuração do sistema" inclui dados para a integração de todas as ECUs em um veículo.
- O "ECU extract" inclui os dados da "descrição da configuração do sistema" para uma única ECU.

2.3.3 Influência da AUTOSAR no trabalho do testador

A AUTOSAR influencia o trabalho do testador, especialmente nos seguintes níveis de teste:[12]

- Teste de componentes de software e teste de integração de software em um ambiente virtual (p. ex., software-in-the-loop): Usando uma simulação do BSW e do RTE, o testador pode testar um SW-C antecipadamente.
- Teste de componentes de software e teste de integração de software na ECU real: aqui, o testador tem acesso à comunicação no RTE. Assim, o testador pode observar e controlar o comportamento de um SW-C em tempo de execução.
- O teste de aceite AUTOSAR é um teste do sistema de software que garante a conformidade da funcionalidade AUTOSAR nos níveis de comunicação e aplicação. A execução do teste de aceite AUTOSAR é opcional.
- Teste de integração do sistema : integração de várias ECUs, por exemplo, para testar a funcionalidade cuja implementação está distribuída em várias ECUs. Ao simular a funcionalidade ainda não implementada, o testador pode avaliar o comportamento do sistema antecipadamente.

2.4 Comparação de ASPICE, ISO 26262 e CTFL®

2.4.1 Objetivos do ASPICE e da ISO 26262

Existem vários padrões que propõem requisitos para o desenvolvimento de produtos. Normalmente, elas destacam diferentes aspectos do desenvolvimento. Nesta subseção, a ISO 26262 e a ASPICE são comparadas em relação aos seus objetivos.

[12] Para obter uma visão geral dos níveis de teste, consulte a seção 2.4.2

A ISO 26262 tem o objetivo de evitar riscos de falhas em hardware e software, fornecendo requisitos e processos adequados. Para o desenvolvimento de sistemas de E/E, ela define os requisitos para os processos e métodos de teste[13] a serem usados pelo testador. Esses requisitos dependem do nível ASIL do item.

A ASPICE tem a finalidade de determinar a capacidade do processo de desenvolvimento de produtos dentro da estrutura de avaliações. Para isso, a ASPICE define critérios avaliáveis para esses processos. Em contraste com a ISO 26262, esses critérios são independentes do nível ASIL do produto.

2.4.2 Comparação dos níveis de teste entre ASPICE, ISO 26262 e CTFL®

Tanto a ISO 26262 quanto a ASPICE descrevem os níveis de teste . No entanto, eles não são totalmente consistentes com os níveis de teste do CTFL®. Portanto, para uma colaboração eficiente e eficaz dentro da equipe de desenvolvimento, os testadores devem ter um entendimento comum de todos os níveis de teste.

O termo "sistema" usado no ASPICE e os termos "sistema" e "item" usados na ISO 26262 referem-se a um produto que consiste em componentes de hardware e software. O CTFL-Syllabus [ISTQB_FL_SYL], entretanto, refere-se a software apenas quando usa o termo "sistema". Portanto, os níveis de teste do ISTQB® podem ser mapeados para os níveis de teste na ISO 26262 e ASPICE da seguinte forma. Observe que, na coluna ISO 26262, os números entre parênteses referem-se a uma parte específica da norma ISO 26262 e a um capítulo específico dela. Na coluna ASPICE, o parêntese mostra a respectiva ID do processo.

ISTQB CTFL®	ISO 26262:2018	ASPICE 4.0
Teste de aceite	Validação de segurança (4-8)[14]	Validação (VAL.1)[15]
Sistema de teste de sistemas[16]	Integração e sistemas e itens (4-7)[17]	Verificação do sistema (SYS.5)
Teste de integração do sistema		Integração do sistema e verificação da integração (SYS.4)
Teste de sistema	Teste do software incorporado (6-11)[18]	Verificação de software (SWE.6)
Teste de integração de componentes	Integração e verificação de software (6-10)	Verificação de componentes de software e verificação de integração (SWE.5)
Teste de componentes	Verificação da unidade de software (6-9)	Verificação da unidade de software (SWE.4)

Tabela2 : Atribuição dos níveis de teste

[13] Para métodos na ISO 26262, consulte a seção 2.2.5

[14] A validação de segurança abrange apenas partes de um teste de aceite de acordo com o ISTQB.

[15] A validação (VAL.1) cobre apenas partes de um teste de aceite de acordo com o ISTQB.

[16] O teste de vários sistemas distribuídos heterogêneos

[17] A integração e o teste do item incluem três fases: a integração e o teste de hardware e software de um elemento, a integração e o teste de todos os elementos pertencentes ao item e a integração e o teste do item em conexão com outros itens do veículo.

[18] Teste dos requisitos de software (sistema) de segurança funcional executados no hardware de destino

No ISTQB CTFL® Syllabus [ISTQB_FL_SYL], a maioria das técnicas de teste pode ser usada em diferentes níveis de teste. Da mesma forma, a ASPICE geralmente não atribui nenhuma técnica de teste aos níveis de teste. Portanto, ambos deixam a escolha para os testadores. Na ISO 26262, há tabelas de métodos individuais para cada nível de teste (consulte as seções2.2.4 e2.2.5). Elas fornecem ao testador recomendações de técnicas de teste com base no nível ASIL.

3 Testes em um Ambiente Virtual - 160 min.

Palavras-chave

modelo de ambiente, hardware-in-the-loop, model-in-the-loop, software-in-the-loop

Palavras-chave específicas do domínio

sistema de circuito fechado, sistema de circuito aberto

Objetivos de aprendizado para o Capítulo 3

3.1 Ambiente de teste em geral

AuT-3.1.1 (K1) Relembrar a motivação por trás de um ambiente de teste no desenvolvimento automotivo

AuT-3.1.2 (K1) Recordar as partes gerais de um ambiente de teste específico para automóveis

AuT-3.1.3 (K2) Lembre-se das diferenças entre sistemas de circuito fechado e sistemas de circuito aberto

AuT-3.1.4 (K1) Relembrar as funções essenciais, os bancos de dados e os protocolos de uma ECU

3.2 Testes em ambientes de teste XiL

AuT-3.2.1.1 (K1) Relembrar a estrutura de um ambiente de teste MiL

AuT-3.2.1.2 (K2) Explicar a área de aplicação e as condições de limite de um ambiente de teste MiL

AuT-3.2.2.1 (K1) Relembrar a estrutura de um ambiente de teste SiL

AuT-3.2.2.2 (K1) Relembrar as áreas de aplicação e as condições de limite de um ambiente de teste SiL

AuT-3.2.3.1 (K1) Relembrar a estrutura de um ambiente de teste de HiL

AuT-3.2.3.2 (K2) Explicar as áreas de aplicação e as condições de limite de um ambiente de teste de HiL

AuT-3.2.4.1 (K2) Resumir as vantagens e desvantagens de testar usando critérios para os ambientes de teste XiL

AuT-3.2.4.2 (K3) Aplicar critérios para a atribuição de uma determinada extensão do teste a um ou mais ambientes de teste

AuT-3.2.4.3 (K1) Descrever os ambientes de teste XiL no modelo V

3.1 Ambiente de teste em geral

3.1.1 Motivação para um ambiente de teste no desenvolvimento automotivo

O testador enfrenta desafios especiais. Por um lado, espera-se que o testador comece a testar o mais cedo possível para encontrar defeitos no início do processo de desenvolvimento. Por outro lado, o testador precisa de um ambiente realista para testar o sistema e encontrar os defeitos que apareceriam no produto concluído. O testador pode resolver esse conflito usando ambientes de teste adequados que correspondam às diferentes fases de desenvolvimento. Ao fazer isso, o testador pode implementar e executar tarefas de teste individuais antes que a ECU totalmente produzida ou desenvolvida esteja disponível. Ao usar vários ambientes de teste, o testador pode simular situações que são difíceis de reproduzir em veículos reais, como curtos-circuitos e sobrecargas de comunicação de rede, curtos-circuitos e circuitos abertos em chicotes de fiação ou sobrecargas em comunicações de rede.

3.1.2 Partes gerais de um ambiente de teste

Para que o testador possa realizar o teste, ele precisa de um ambiente de teste no qual as partes ausentes sejam simuladas. Esse ambiente ajuda o testador a controlar as entradas do objeto de teste e a observar os resultados do teste, também chamado de "ponto de controle" (PoC) e "ponto de observação" (PoO). De acordo com a norma ISO/IEC/IEEE 29119-1, um ambiente de teste consiste nas seguintes partes:

- Hardware do ambiente de teste (p. ex., um computador de controle, um computador com capacidade para tempo real quando necessário, bancada de teste e kit de desenvolvimento)
- Software do ambiente de teste (p. ex., sistema operacional, software de simulação e modelos de ambiente)
- Instalações de comunicação (p. ex., acesso à rede e um registrador de dados)
- Ferramentas (p. ex., osciloscópio e ferramentas de medição)
- Laboratório (p. ex., proteção contra radiação eletromagnética e ruído)

Uma parte importante do ambiente de teste é o modelo de ambiente . Os modelos são elementos-chave do ambiente de teste virtual. Eles representam aspectos do mundo real, como o motor de combustão, as transmissões, os sensores e as ECUs do veículo ou até mesmo o motorista e as condições da estrada. O ambiente de teste também oferece diferentes pontos de acesso. Eles permitem que o testador observe e meça o objeto de teste e o influencie ou controle quando necessário.

3.1.3 Diferenças entre sistemas de loop fechado e de loop aberto

O ambiente de teste é usado para controlar as interfaces de entrada do dispositivo em teste e monitorar sua saída por meio das interfaces de saída. Em seguida, o comportamento nas interfaces de saída é analisado. O sucesso ocorre quando os resultados reais correspondem aos resultados esperados.

Em geral, há dois tipos de sistemas de controle, de loop fechado e de loop aberto. A diferença se baseia na forma como a ECU reage ao seu ambiente, o que exige diferentes requisitos de simulação para o ambiente de teste virtual.

3.1.3.1 Sistema de circuito fechado

A estimulação em um sistema de loop fechado, também conhecido como *in-the-loop*, leva em consideração a saída do objeto de teste. Isso é feito por meio de um modelo de ambiente , que coleta as saídas e as encaminha direta ou indiretamente para a entrada do objeto de teste. Portanto, um loop de controle é criado no ambiente de teste.

Os sistemas de circuito fechado são usados com mais frequência para testar controladores. Com isso, funções complexas podem ser testadas, como controles de motor e engrenagem e sistemas de assistência ao motorista, como o sistema de freios antibloqueio (ABS®) ou o controle de dinâmica do veículo (VDC®).

3.1.3.2 Sistema de circuito aberto

Em um sistema de loop aberto, as saídas do sistema não têm relação com as entradas. O sistema não tem um loop de feedback. Nesse caso, as entradas do objeto de teste são definidas diretamente pelo testador no procedimento de teste.

O caso de aplicação dos sistemas de loop aberto e de loop fechado depende muito do comportamento do objeto de teste. Os sistemas de circuito fechado têm saídas que influenciam as entradas, criando feedback, enquanto os sistemas de circuito aberto não têm esse mecanismo de feedback. Se o objeto de teste tiver um comportamento reativo ou se espelhar uma máquina de estado, é preferível um sistema de loop aberto. Nos componentes eletrônicos do interior e do chassi, há muitos exemplos de sistemas de loop aberto (p. ex., luzes e interruptores).

3.1.4 Bancos de dados e protocolos de comunicação de uma ECU

Uma ECU no ambiente automotivo funciona como um sistema incorporado, que consiste em hardware e software. A ECU recebe diferentes entradas analógicas e digitais, que coletam constantemente dados ambientais na forma de tensão, corrente e temperatura. Além disso, os sistemas de barramento de comunicação oferecem dados adicionais de sensores ou de outras unidades de controle. Os sistemas de barramento de comunicação fornecem informações adicionais à unidade de controle, provenientes de sensores ou de outras ECUs, que coletam e processam as informações por conta própria ou as geram. O objeto de teste gerencia os dados na memória para processar a ação, as informações ou os dados de saída. As saídas geradas também são executadas por meio de pinos de saída analógicos e digitais, sistemas de barramento ou interfaces de diagnóstico.

Os bancos de dados são depósitos de dados e definem os sinais de entrada e saída da unidade de controle. Esses dados também incluem descrições, unidades de controle e fórmulas de conversão dos sinais.

Os protocolos de comunicação descrevem a troca de dados por meio das interfaces físicas correspondentes. Esses protocolos definem qual tensão ou sequência de bits representa qual valor do sinal.

A seleção de bancos de dados e protocolos de comunicação depende da função da ECU. Por exemplo, para acessar as funções de diagnóstico na unidade de controle, o testador precisa das informações sobre o banco de dados usado (p. ex., especificação da interface de programação de aplicativos (Association for Standardization of Automation and Measuring Systems (ASAM) MCD-3 D) e diagnóstico de veículos orientado a serviços (ASAM SOVD)) e o protocolo de comunicação (p. ex., serviços de diagnóstico unificados de acordo com a ISO 14229 e especificação AUTOSAR (SOME/IP)). Outros bancos de dados específicos do setor automotivo são definidos nos padrões ASAM e AUTOSAR.

3.2 Testes em ambientes de teste XiL

No setor automotivo, são usados os seguintes tipos de ambientes de teste do XiL[19]

- Model-in-the-loop (MiL)
- Software-in-the-loop (SiL)
- Processor-in-the-loop[20] (PiL)
- Hardware-in-the-loop (HiL)
- Vehicle-in-the-loop[21] (ViL)

[19] A letra X em XiL é um espaço reservado para os ambientes de teste específicos na lista fornecida.

[20] Esse ambiente de teste não é considerado neste syllabus e é meramente informativo.

[21] Esse ambiente de teste não é considerado neste syllabus e é meramente informativo.

O testador deve se familiarizar com os ambientes de teste (ou seja, MiL, SiL e HiL) e entendê-los. Os parágrafos a seguir analisam mais detalhadamente a estrutura e as áreas de aplicação dos diferentes ambientes de teste. Nesse sentido, XiL é um termo genérico para os diferentes ambientes de teste.

3.2.1 Model-in-the-loop (MiL)

3.2.1.1 Estrutura de um ambiente de teste MiL

Em um ambiente de teste MiL, o objeto de teste está disponível como um modelo. Esse modelo é executável, mas não compilado para hardware especial. Os desenvolvedores criam modelos usando ferramentas especiais de modelagem. O testador precisa de um ambiente de teste para poder executar e testar esses modelos. Normalmente, isso é implementado no mesmo ambiente de desenvolvimento que o próprio objeto de teste. Além disso, esse ambiente de teste pode conter um modelo de ambiente . O testador pode controlar e observar o objeto de teste por meio de pontos de acesso. Os pontos de acesso podem ser colocados arbitrariamente no modelo do objeto de teste e no modelo do ambiente. O modelo do objeto de teste está conectado ao modelo do ambiente e pode ser facilmente implementado e usado como um sistema de loop fechado.

3.2.1.2 Áreas de aplicação e condições de limite de um ambiente de teste MiL

Com um ambiente de teste MiL, o testador pode testar o projeto do sistema funcional. Durante o desenvolvimento (p. ex., o modelo V geral), o testador também pode testar componentes individuais até um sistema inteiro. Para executar o teste, o testador precisa de um computador e do software de simulação correspondente, incluindo o modelo de ambiente. O modelo do ambiente se torna mais complexo à medida que o escopo das funções do objeto de teste aumenta. Os aspectos da realidade e os fatores ambientais são muito complexos. Os tempos de execução dos modelos também aumentam de forma desproporcional. Portanto, o esforço para implementar um ambiente de teste MiL deixa de valer a pena a partir de uma determinada fase do desenvolvimento. [22]

Ao usar um ambiente de teste MiL, o testador pode testar a adequação funcional dos modelos em todos os níveis de desenvolvimento em uma fase inicial do desenvolvimento (ou seja, o lado esquerdo do modelo V[23]). Mas não é comum habilitar o modelo de ambiente para simular funções de barramento ou diagnóstico ou comportamento físico, como quebras de cabos ou curtos-circuitos. Essas tarefas podem ser realizadas com mais facilidade e a um custo menor em outros ambientes de teste.

Em um ambiente de teste MiL, a execução do teste não ocorre em tempo real. Como todos os componentes estão disponíveis como um modelo, a execução do teste é feita em tempo de simulação. Quanto mais complexo for um sistema, mais tempo de execução ou potência o computador precisará para fornecer todas as informações necessárias. A duração da simulação em sistemas menores é mais curta do que a execução em tempo real. No entanto, uma grande vantagem é que o testador pode pausar a simulação a qualquer momento para análise e avaliação detalhadas.

3.2.2 Software-in-the-loop (SiL)

3.2.2.1 Estrutura de um ambiente de teste SiL

O objeto de teste é compilado para um ambiente de teste SiL específico. Isso significa que o código-fonte foi compilado para uma determinada arquitetura de computador. Esse código de máquina pode ser lido pelo ambiente de teste, pois consiste em conjuntos de dados binários. Para que o ambiente de teste possa acessar os sinais, é

[22] Isso também é válido para todos os outros ambientes XiL.

[23] Outras descrições do conteúdo do modelo V podem ser encontradas em: Sommerville, I. (2011). Software Engineering (9. Auflage). Pearson Education.

necessário um wrapper[24] . Um wrapper é um software adicional que fornece acesso a entradas que podem não estar acessíveis se o objeto de teste se comunicar com o ambiente diretamente e não por meio do wrapper. Portanto, o testador pode controlar os sinais do software e observá-los. O wrapper define os pontos de acesso ao objeto de teste, mas não executa suas tarefas funcionais.

É necessário um modelo de ambiente para a simulação. O objeto de teste é conectado ao ambiente de teste com a ajuda do wrapper. A execução do teste é realizada em um computador sem hardware especial. O testador precisa de uma ferramenta de software que possa criar um invólucro para o objeto de teste com pontos de acesso ao ambiente de teste.

3.2.2.2 Áreas de aplicação e condições de limite de um ambiente de teste SiL

Se o desenvolvedor gerar código-fonte com base em um modelo, o comportamento real do software pode ser diferente do comportamento esperado. Isso pode ser causado por diferentes tipos de dados no modelo, principalmente ponto flutuante, e no código de software compilado, principalmente ponto fixo, mas também por diferentes espaços de memória. Essas anomalias no comportamento esperado podem ser testadas pela primeira vez em um ambiente de teste SiL. O testador pode usar uma abordagem de teste como o teste back-to-back (consulte a seção4.2.2) para comparar o comportamento.

O testador executa os testes, de forma análoga ao ambiente de teste MiL, em tempo de simulação. Dependendo da técnica de cálculo e da complexidade do modelo do ambiente, esse tempo de simulação pode ser mais curto ou mais longo do que em tempo real. O testador pode pausar a execução do teste a qualquer momento para análise e avaliação detalhadas. Testes de adequação funcional, testes de interface e testes de regressão são tipos de teste muito comuns que podem ser avaliados em um ambiente de teste SiL. Por outro lado, os testes de eficiência de performance e os testes de confiabilidade são incomuns. Essas características de qualidade são afetadas principalmente pelo hardware de destino.

3.2.3 Hardware-in-the-loop (HiL)

3.2.3.1 Estrutura de um ambiente de teste HiL

Se o objeto de teste estiver disponível como um protótipo ou se já estiver completamente desenvolvido, o testador poderá usar um ambiente de teste HiL para executar os testes. As partes típicas de um ambiente de teste HiL são:

- Uma fonte de alimentação para definir diferentes tensões de alimentação
- Um computador com capacidade de tempo real para a execução do modelo de ambiente
- Várias partes reais que não estão implementadas no modelo de ambiente
- Uma unidade de processamento de sinal de tipo de sinal e amplitude de sinal
- Uma unidade de inserção de falhas (FIU) (consulte a seção 4.2.3) para a simulação de rompimentos e curtos-circuitos de cabos
- Uma breakout box como uma interface de acesso adicional no chicote de cabos
- Um ônibus remanescente para simular os participantes do ônibus inexistente

3.2.3.2 Áreas de aplicação e condições de limite de um ambiente de teste de HiL

Os pontos de acesso em um ambiente de teste HiL são diversos. O testador deve estar ciente de que o uso de pontos de acesso errados para o objeto de teste pode tornar os resultados do teste inúteis. Conhecer os diferentes

[24] Informações detalhadas sobre arquiteturas de software podem ser encontradas em: Balzert, H., Schröder, K., & Kern, U. (2022). Grundlagen der Softwaretechnik (6. Aufl.). Carl Hanser Verlag.

pontos de acesso e suas conexões no ambiente de teste HiL permite que testes eficazes sejam implementados, executados e avaliados.

O ambiente de teste HiL é mais complexo do que os ambientes de teste mencionados anteriormente (ou seja, MiL e SiL) devido às suas várias partes. O testador deve dominar essa complexidade para realizar as tarefas de teste. O ambiente de teste HiL pode ser usado para testes de componentes, testes de integração e testes de sistema. O objetivo é, entre outras coisas, encontrar defeitos funcionais e não funcionais no software e no hardware.

Com a ajuda dos ambientes de teste HiL, diferentes níveis de teste podem ser analisados. Se o objeto de teste for uma única ECU, ele será chamado de componente[25] HiL. Se o objeto de teste for uma combinação de várias ECUs, ele será chamado de HiL de sistema. O testador usa o componente HiL para testar as funções da unidade de controle. Na HiL do sistema, o foco está em testar a troca de dados entre as ECUs e o teste do sistema.

Em contraste com os ambientes de teste mencionados anteriormente (ou seja, MiL e SiL), o tempo de simulação em um ambiente de teste HiL é sempre executado em tempo real. A razão para isso é que o software está sendo executado em hardware real. Pausar ou parar não é mais possível nesse ambiente de teste. Portanto, o ambiente de teste inclui um computador com capacidade de tempo real que pode coletar e fornecer todos os sinais relevantes em um período predeterminado.

3.2.4 Comparação dos ambientes de teste XiL

3.2.4.1 Vantagens e desvantagens de testar nos ambientes de teste XiL

O testador compreende os atributos dos diferentes ambientes de teste. Ao fazer isso, o testador pode entender e avaliar as vantagens e desvantagens de testar em cada ambiente. Os critérios são mostrados na tabela 3.

Critérios	Ambiente de teste HiL	Impacto	MiL	SiL	HiL
Proximidade com a realidade	A realidade é simulada, muitas características são abstraídas e o foco está nas estruturas e na lógica.	Baixa	+	o	o
	O software real compilado pode ser executado sem hardware.	Baixo a médio	o	+	o
	Sistema integrado, capaz de executar	Alta	o	o	+
Tempo e esforço na depuração	Os defeitos são encontrados no modelo do objeto de teste (ajuste do modelo).	Baixa	+	o	o
	Os defeitos são encontrados no software programado (ajuste de software).	Médio	o	+	o
	Os defeitos são encontrados no nível do sistema (ajuste do sistema).	Alta	o	o	+
Esforço de implementação e manutenção	Criar modelo de ambiente	Baixa	+	o	o
	Criar modelo de ambiente e wrapper	Médio	o	+	o
	Criar um modelo de ambiente e conectar os componentes de hardware	Alta	o	o	+
Esforço de preparação para o teste	O ambiente pode ser configurado rapidamente.	Baixa	+	o	o
	O ambiente pode ser configurado rapidamente.	Médio	o	+	o
	O projeto, a implementação e a avaliação dos testes exigem muito esforço.	Alta	o	o	+

[25] Neste caso, o termo componente é usado para uma ECU no contexto de um E/Esystem.

Critérios	Ambiente de teste HiL	Impacto	MiL	SiL	HiL
Nível necessário de maturidade do objeto de teste	Os modelos de sistema são simulados.	Baixa	+	o	o
	As funções iniciais são testadas com o software de destino.	Médio	o	+	o
	Uma ou mais ECUs executáveis ou sistemas parciais são testados da forma mais completa possível	Alta	o	o	+
Nível necessário de detalhamento da base de teste (especificação)	São testados modelos que cobrem parcialmente uma especificação incompleta.	Médio	+	o	o
	As informações relevantes no nível do software devem estar disponíveis (p. ex., especificação detalhada do componente).	Médio a alto	o	+	o
	Os requisitos podem ser testados no nível do sistema com uma especificação completa do sistema	Alta	o	o	+
Acesso ao objeto de teste	Todos os sinais em um modelo podem ser observados e controlados.	Alta	+	o	o
	Somente os sinais disponíveis no invólucro podem ser observados e controlados.	Médio	o	+	o
	Somente os sinais disponíveis no hardware ou nos protocolos de comunicação podem ser observados e controlados.	Baixa	o	o	+

Tabela3 : Critérios e seu impacto nos ambientes de teste MiL, SiL e HiL

3.2.4.2 Aplicar critérios para a atribuição de uma determinada extensão do teste a um ou mais ambientes de teste

Na tabela a seguir, os objetivos do teste são descritos em detalhes e atribuídos a ambientes de teste adequados.

Objetivo do teste	Descrição por exemplos	MiL	SiL	HiL
Testar os requisitos do cliente	Fornecimento correto da funcionalidade necessária. Isso inclui o processamento correto das entradas, a reação correta às entradas e a saída correta de dados no ponto de saída.	O	O	+
Mecanismos de teste para detecção e tratamento de defeitos	- Detecção e tratamento de aleatórios de hardware - Detecção e tratamento de defeitos de software - Transferência para um estado seguro após a detecção de defeitos, exemplo desativação de um sistema	+	+	+
Teste de reação aos dados de configuração	Verificar a influência dos dados de configuração no comportamento do objeto de teste	O	+	+
Testar funções de diagnóstico	Fornecimento correto da funcionalidade de diagnóstico necessária, como detecção de defeitos, ativação de defeitos e requisitos de redefinição, e ativação de defeitos na memória de defeitos (p. ex., diagnóstico a bordo ou na oficina)	-	+	+

Teste a interação nas interfaces	Verificar as interfaces internas e externas do objeto de teste	O	+	+
Comprovar a usabilidade	O objeto de teste deve atender aos requisitos de usabilidade do usuário.	-	O	+
Legenda: + recomendado, o possível, - não é sensato				

Tabela 4: Comparação dos tipos de teste nos ambientes de teste MiL, SiL e HiL

Essa tabela mostra que os ambientes de teste podem ser adequados para determinados objetivos de teste. Essa abordagem diversificada fica evidente especialmente no teste dos mecanismos de detecção e tratamento de defeitos. De acordo com o turno da esquerda, a conclusão geral é que os requisitos básicos e os defeitos de projeto já são detectados antecipadamente por meio de testes. Portanto, o MiL é usado para a detecção de defeitos gerais de projeto, o SiL principalmente para defeitos técnicos de software e o HiL para defeitos técnicos de hardware/software. Além disso, é importante observar que, além da evidência de confiabilidade, eficiência de performance e usabilidade, todos os tipos de teste se concentram na adequação funcional do objeto de teste.

Na estratégia de teste, o gerente de teste atribui o escopo do teste a vários ambientes de teste diferentes. Os testadores devem se sincronizar regularmente para garantir testes eficientes e precisos em diferentes ambientes XiL. Se for o caso, recomenda-se alinhar os objetivos do teste, evitar a duplicação e verificar se cada teste é executado no ambiente mais adequado para sua fase de desenvolvimento. Essa colaboração ajuda a evitar conflitos de recursos, garante a consistência da cobertura e reduz o risco de testes em níveis inadequados. Ao promover a comunicação e aproveitar as ferramentas ou estruturas compartilhadas, as equipes podem otimizar as estratégias de teste e manter uma progressão contínua ao longo do ciclo de vida do teste. Ao combinar os critérios das tabelas 3 e 4, o gerente de teste pode escolher o ambiente de teste ideal.

3.2.4.3 Classificação dos ambientes de teste XiL no modelo V

O projeto do sistema técnico está no lado esquerdo do modelo V. O testador pode testar esse projeto com um ambiente de teste MiL. Se o modelo e o ambiente de teste forem desenvolvidos posteriormente, o testador também poderá executar testes de componentes e testes de integração com esse ambiente de teste.

O testador pode usar um ambiente de teste SiL se os componentes individuais do objeto de teste forem programados e compilados. Os testes típicos para um ambiente de teste SiL são testes de componentes e testes de integração. Eles podem ser encontrados no lado direito do modelo V.

Nos testes de sistema, determinadas funcionalidades do objeto de teste foram totalmente desenvolvidas. O testador pode executar o teste do sistema com um ambiente de teste HiL.

Com uma atribuição correta do ambiente de teste aos níveis de teste, todo o processo de teste pode ser otimizado de acordo com os três aspectos a seguir:

Minimização dos riscos do produto

- Encontrar tipos de falha específicos do nível de teste (p. ex., eficiência de performance no nível do sistema em um ambiente HiL)

Minimizar o custo do teste

- Para cada nível de teste, são necessários os tipos de teste adequados
- Transferência de testes para níveis de teste anteriores, menos onerosos e virtuais

Conformidade com os padrões

- Nas tabelas de métodos da norma ISO 26262, os ambientes de teste são recomendados dependendo do ASIL

4 Teste estático e teste dinâmico - 230 min.

Palavras-chave

testes back-to-back, injeção de falhas, baseado em requisitos

Objetivos de aprendizado para o Capítulo 4

4.1 Teste estático

AuT-4.1.1 (K2) Explicar a finalidade e os requisitos das diretrizes MISRA-C com a ajuda de exemplos

AuT-4.1.2 (K3) Aplicar a revisão de requisitos usando as características de qualidade da norma ISO/IEC/IEEE 29148 que são relevantes para os testadores

4.2 Teste dinâmico

AuT-4.2.1 (K3) Projetar casos de teste para obter cobertura de teste de condição/decisão modificada

AuT-4.2.2 (K2) Explique o uso de testes back-to-back, dando exemplos

AuT-4.2.3 (K2) Explique o teste de injeção de falhas, dando exemplos

AuT-4.2.4 (K1) Relembrar os princípios dos testes baseados em requisitos

AuT-4.2.5 (K3) Aplicar critérios dependentes do contexto para a escolha de técnicas de teste e abordagens de teste adequadas e necessárias

4.1 Testes Estáticos

Introdução

O teste estático consiste em examinar os produtos de trabalho do desenvolvimento de software sem executá-los. Isso inclui a avaliação por pessoas que realizam revisões e análise estática com suporte de ferramentas.

4.1.1 Diretrizes da MISRA-C

É o estado da arte para que o código-fonte esteja em conformidade com as diretrizes de codificação . A norma ISO 26262 também recomenda a adesão às diretrizes de codificação para software relevante para a segurança[26] . As diretrizes de codificação ajudam a evitar anomalias no código, que podem levar a defeitos. Ao mesmo tempo, elas oferecem suporte à manutenção e à portabilidade.

O MISRA-C fornece diretrizes de codificação para a linguagem de programação C e são comumente usadas em projetos de software automotivo. Ela define dois tipos diferentes de diretrizes: regras e diretivas.

- As regras podem ser verificadas por ferramentas de análise estática. Exemplo de regra: "O código-fonte não deve incluir comentários aninhados".
- As diretivas não são totalmente verificadas pelas ferramentas de análise estática. Elas se referem a detalhes do processo de desenvolvimento ou a documentos fora do software. Exemplo de diretriz: "O desenvolvedor deve documentar suficientemente o comportamento implementado".

Cada diretriz é categorizada em um dos três níveis de obrigação a seguir:

- As diretrizes "consultivas" devem ser seguidas pelo desenvolvedor se o esforço for razoável.
- As diretrizes "obrigatórias" devem ser seguidas pelo desenvolvedor; exceções são possíveis, mas devem ser oficialmente aprovadas (p. ex., pelo gerenciamento de qualidade)
- As diretrizes "obrigatórias" devem ser seguidas pelo desenvolvedor; não há exceções.

As organizações podem aumentar individualmente o nível de obrigação de uma diretriz, mas nunca podem reduzi-lo.

4.1.2 Características de qualidade para análises de requisitos

As especificações são a base para o desenvolvimento e os testes. Portanto, os defeitos nas especificações levam a atividades de acompanhamento que consomem muito tempo e custos. Isso se aplica especialmente se os defeitos forem detectados somente nas fases finais do desenvolvimento, como testes de aceite ou em operação. As revisões são uma medida eficaz para encontrar defeitos nas especificações com antecedência e, consequentemente, corrigi-los com antecedência e a baixo custo.

Durante a análise do teste, o testador deve verificar as especificações do objeto de teste, especialmente em relação à sua adequação como base de teste. As características de qualidade ajudam o testador durante a revisão das especificações a se concentrar e encontrar o maior número possível de defeitos. A ISO 29148:2018 inclui características de qualidade para requisitos únicos e para conjuntos de requisitos.

Características de qualidade da ISO 29148:2018 relevantes para os testadores incluem:

- Verificável: Cada requisito pode ser comprovado como tendo sido realizado corretamente.
- Sem ambiguidade: cada requisito contém condições de teste claras.
- Consistente: Cada requisito é consistente com todos os outros requisitos.

[26] Consulte também [ISO 26262:2018] Parte 6-5.4.3

- Completo: Cada requisito não deixa margem para interpretação e não se baseia no conhecimento implícito ou no conhecimento da experiência.
- Singular[27] : Nenhum requisito pode ser dividido em requisitos parciais significativos.

Na preparação para uma revisão de requisitos, o testador pode derivar uma lista de verificação de revisão com itens de lista de verificação dessas características de qualidade. Esses itens da lista de verificação devem ser mais específicos do que simplesmente nomear os critérios de qualidade, que são muito abstratos. Por exemplo, um item de lista de verificação para consistência poderia ser "Os termos são usados de forma consistente em todos os requisitos?". Na revisão da especificação, o testador deve responder aos itens da lista de verificação com o melhor de seu conhecimento.

De acordo com a ISO 29148:2018, os requisitos também devem ser viáveis, apropriados, corretos, conformes, compreensíveis e necessários. No entanto, geralmente é difícil para o testador avaliar essas características de qualidade.

4.2 Testes Dinâmicos

4.2.1 Teste de condição/decisão modificado

As técnicas de teste descritas aqui fazem parte das técnicas de teste caixa-branca. Para obter mais detalhes, consulte também o syllabus para Analista de Testes Técnicos [ISTQB_ALTTA_SYL]). O testador deriva os casos de teste diretamente da estrutura do objeto de teste (p. ex., código-fonte).

No teste de decisão, os casos de teste são projetados para executar todos os resultados possíveis da decisão, ou seja, os resultados "verdadeiros" e "falsos" de cada decisão. Uma decisão consiste em uma ou mais condições, cada uma das quais pode ser avaliada como "verdadeira" ou "falsa". O resultado da decisão é determinado pela combinação lógica desses valores de condição.

Se uma decisão consistir em uma condição, o teste de decisão e o teste de condição atingem a mesma cobertura. No entanto, para decisões com várias condições, estão disponíveis técnicas de teste mais detalhadas, que são descritas a seguir:

- Teste de condição (técnica de teste A na tabela 5): O testador projeta casos de teste com o objetivo de cobrir os resultados verdadeiro/falso de cada condição individual. Com uma escolha ruim dos dados de teste (consulte a tabela 5), ainda é possível obter 100% de cobertura da condição , mas não a cobertura total de todos os resultados da decisão. Para ambos os casos de teste TC1 e TC2, as condições individuais B1 e B2 são exercidas como verdadeiras e falsas, e os resultados da decisão são avaliados como "falsos".
- Teste de condições múltiplas (técnica de teste B na tabela 5): O testador projeta casos de teste com o objetivo de abranger todas as combinações de valores relacionados às condições individuais. Se cada combinação de valores for testada, cada resultado da decisão também será testado.
- Teste de condição/decisão modificado (MC/DC) (técnica de teste C na tabela 5): É como o teste de condições múltiplas (B). Entretanto, a técnica de teste considera apenas combinações nas quais as condições individuais (B1, B2) influenciam independentemente o resultado da decisão. No caso da TC4, alterar B1 ou B2 de "falso" para "verdadeiro" individualmente não altera o resultado da decisão. (ou seja, ele permanece "falso"). A cobertura de 100% do MC/DC pode ser obtida com o uso dos casos de teste TC 1, TC 2 e TC 3; não é necessário considerar o TC 4.

A Tabela 5 mostra um exemplo dos casos de teste necessários para 100% de cobertura, dependendo da técnica de teste escolhida:

[27] singular também é conhecido como *atômico*

	Condições individuais		Resultado da decisão para a expressão: E = B1 E B2	Técnica de teste		
Caso de teste	B1	B2		A	B	C
TC 1	B1=TRUE	B2=FALSE	E =FALSE	X	X	X
TC 2	B1=FALSE	B2=TRUE	E=FALSE	X	X	X
TC 3	B1=TRUE	B2=TRUE	E=TRUE		X	X
TC 4	B1=FALSE	B2=FALSE	E=FALSE		X	

Tabela 5: Comparação das técnicas de teste: teste de condição (A), teste de condição múltipla (B) e teste de condição/decisão modificado (teste MC/DC) (C)

O exemplo mostra os limites dessas técnicas de teste: No caso do teste de condição (A), apesar de uma cobertura de condição de 100%, o testador corre o risco de cobrir apenas *um* resultado de decisão. Uma escolha melhor de casos de teste corrigiria isso usando os casos de teste TC 3 e TC 4.

Com o uso do teste de condições múltiplas (B), o testador pode cobrir todas as entradas e saídas possíveis. Entretanto, o número de testes a serem executados é o mais alto para essas técnicas de teste.

Ao usar o teste de condição/decisão modificado (C), o testador pode obter cobertura completa de todas as condições únicas e de todas as decisões com um número menor de testes em comparação com o teste de condições múltiplas.

4.2.2 Teste back-to-back

Teste back-to-back é uma abordagem de teste na qual um objeto de teste é testado usando um pseudo-oráculo para gerar resultados esperados. Para fazer isso, o testador executa o mesmo caso de teste em todas as variantes e compara os resultados do teste. Se os resultados do teste forem idênticos, o teste foi aprovado. Se os resultados do teste forem diferentes, a causa da diferença detectada será analisada.

Os objetos de teste devem se basear nos mesmos requisitos em termos de conteúdo. Somente assim eles poderão demonstrar um comportamento comparável. Embora esses requisitos sejam normalmente usados como base para derivar entradas de teste, eles não servem como um oráculo de teste para definir os resultados esperados. Em vez disso, no teste back-to-back, o comportamento de um objeto de teste é comparado com o de outro, usando um como pseudo-oráculo para o outro. Dessa forma, espera-se que o teste revele diferenças não intencionais, mesmo que muito pequenas, entre os objetos de teste ou seus ambientes

No caso mais simples, os objetos de teste dos testes back-to-back são versões diferentes do mesmo software. Nesse caso, uma versão anterior do objeto de teste serve como pseudo-oráculo para o teste back-to-back, como um teste de regressão. Uma alternativa é a comparação de um modelo executável com o código gerado manual ou automaticamente. Nesse caso, é uma forma de teste baseado em modelo, no qual o modelo executável também serve como pseudo-oráculo. Portanto, essa abordagem de teste é muito adequada para o design de testes automatizados. Aqui, o testador deriva não apenas o resultado esperado do modelo, mas também casos de teste automatizados.

4.2.3 Teste de Injeção de Falhas

O teste de injeção de falhas é uma abordagem de teste projetada para avaliar a resposta de um componente ou sistema a condições adversas (p. ex., um defeito - também chamado de *falha* nesse contexto). As técnicas de programação, como o tratamento de erros, têm o objetivo de fazer com que o sistema reaja a defeitos internos e externos de forma robusta e segura. Para realizar o teste de injeção de falhas, o testador pode inserir defeitos no sistema de forma seletiva nos seguintes pontos:

- Defeitos em componentes externos: por exemplo, detecção de valores não plausíveis de sensores

- Defeitos nas interfaces: por exemplo, evitar danos causados por curtos-circuitos ou mensagens perdidas
- Defeitos no aplicativo: detecção e tratamento de defeitos internos

As falhas podem ser injetadas de diferentes maneiras, dependendo do sistema e do ambiente de teste:

- Na injeção de falha clássica , o testador insere um defeito manipulando o objeto de teste físico.
- Defeitos externos ou defeitos relacionados à interface são normalmente simulados em tempo de execução em um ambiente de teste HiL usando uma FIU
- Defeitos baseados em software, como defeitos internos ou de interface, são frequentemente simulados em ambientes de teste SiL ou diretamente no ambiente de desenvolvimento, usando ferramentas como depuradores ou o Protocolo Universal de Medição e Calibração (XCP).

4.2.4 Testes baseados em requisitos

Nos testes baseados em requisitos, o testador normalmente analisa os requisitos textuais para derivar condições de teste de requisitos atômicos individuais, projeta casos de teste que exercitam os requisitos e os executa. Essa decomposição envolve a identificação de aspectos distintos e testáveis em um requisito. Cada requisito atômico deve descrever um único comportamento ou condição que possa ser verificado de forma independente.

Ao dividir um requisito complexo ou de alto nível em tais elementos granulares, o testador garante uma cobertura abrangente e evita a perda de expectativas ocultas ou implícitas. A cobertura dos requisitos é medida como a proporção de requisitos atômicos cobertos por pelo menos um caso de teste em relação ao número total de requisitos atômicos.

Os casos de teste para testes baseados em requisitos são, em geral, casos de teste positivos, que confirmam que os requisitos declarados foram atendidos. Os casos de teste negativos geralmente são derivados usando técnicas de teste complementares, como particionamento de equivalência, adivinhação de erros ou testes exploratórios, embora os requisitos também possam definir explicitamente essas condições negativas.

4.2.5 Seleção dependente do contexto

Várias técnicas de teste e abordagens de teste são relevantes no contexto dos sistemas automotivos. Algumas delas são apresentadas no syllabus CTFL [ISTQB_FL_SYL], enquanto outras são discutidas na seção (4.2). Essas técnicas incluem:

- Testes baseados em requisitos
- Particionamento de equivalência
- Análise do valor de contorno
- Teste de declaração
- Teste de ramificação
- Teste de condição/decisão modificado
- Adivinhação de erros
- Injeção de falhas
- Testes back-to-back

A seleção de técnicas de teste e abordagens de teste adequadas depende de vários fatores

Estado da arte

A técnica de teste ou a abordagem de teste representa o estado da arte atual para essa finalidade? Nesse caso, normas como a ISO/IEC/IEEE 29119 e a ISO 26262 fornecem orientações valiosas. Particularmente, a ISO/IEC/IEEE 29119-4 oferece uma visão geral estruturada das técnicas de teste padronizadas, incluindo as medidas de cobertura correspondentes, categorizadas em técnicas de teste caixa-preta, técnicas de teste caixa-branca e técnicas de teste baseadas na experiência. A norma ISO 26262 sugere ainda técnicas de teste e

abordagens de teste aplicáveis, dependendo do nível ASIL, conforme discutido na seção 2.2. Os desvios das recomendações devem ser cuidadosamente considerados e justificados.

Base de teste

A base de teste fornece condições de teste adequadas para a técnica de teste? Por exemplo, o testador só pode formar partições de equivalência se a base de teste incluir parâmetros ou variáveis. Os valores devem poder ser agrupados em partições de equivalência. Condições semelhantes se aplicam aos valores de limite. Eles só podem ser testados se o intervalo de valores for definido de forma linear.

Testes baseados em riscos

Teste baseado em risco significa a identificação dos riscos do produto e a consideração do nível de risco para a seleção das técnicas e abordagens de teste. Por exemplo, o teste de um valor de limite só faz sentido se houver o risco de ocorrerem violações de limite e se o impacto de tais violações constituir um risco.

Nível de teste

A técnica de teste/abordagem de teste pode ser usada no nível de teste? O teste caixa-branca é particularmente adequado se o código-fonte ou a estrutura interna servir como base de teste. No caso ideal, o grau estrutural de cobertura é mensurável. Para o teste caixa-preta, o objeto de teste precisa estar disponível e ser observável. Por exemplo, o teste de uma partição de equivalência de um sensor no teste do sistema pode ser mais eficiente do que no teste de componentes. Se uma técnica ou abordagem de teste não for utilizável em um nível de teste, o testador deverá escolher um nível de teste diferente, de acordo com a estratégia de teste.

Exemplos de seleção de técnicas de teste e abordagens de teste

A tabela a seguir fornece uma lista de técnicas e abordagens de teste, complementada por um exemplo de avaliação do usuário de vários dos fatores acima e a escolha da técnica/abordagem de teste com base nisso.

	Técnica de teste / Abordagem de teste	**Recomendado para uso com o ASIL A?**	**A base de teste é adequada?**	**Risco, se o defeito não for detectado?**	**O teste de sistema em nível de teste é razoável?**	**Seleção**
1	Testes baseados em requisitos	++	SIM	++	SIM	X
2	Particionamento de equivalência	+	SIM	++	SIM	X
3	Análise do valor de contorno	+	NÃO	-	SIM	
4	Teste de declaração	++	SIM	++	NÃO	
5	Teste de decisão	+	SIM	++	NÃO	
6	MC/DC	+	SIM	+	NÃO	
7	Adivinhação de erros	+	NÃO	++	SIM	
8	Teste de injeção de falhas	+	SIM	+	NÃO	
9	Testes back-to-back	+	NÃO	++	SIM	

Tabela 6: Exemplo de escolha de técnica de teste/abordagem de teste

5 Lista de abreviações

Abreviação	Definição / Significado
ACQ	Aquisição (ASPICE)
ASIL	Nível de integridade da segurança automotiva
ASAM	Association for Standardization of Automation and Measuring Systems (Associação para Padronização de Sistemas de Automação e Medição)
ASPICE	SPICE automotivo
AUTOSAR	Arquitetura de sistema aberto automotivo
AUTOSIG	Grupo de interesse específico automotivo
BP	Prática básica (ASPICE)
BSW	Software básico (AUTOSAR)
CTFL	Certification Testing Foundation Level
E/E	Elétrico / Eletrônico
ECU	Unidade de controle eletrônico
E/E	Elétrica/Eletrônica
EES	Simulação de erro elétrico
EOP	Fim da produção
FIU	Unidade de inserção de falhas
GP	Prática Genérica (ASPICE)
HiL	Hardware-in-the-loop
IEC	Comissão Eletrotécnica Internacional
ISO	Organização Internacional de Padronização
HOMEM	Gerenciamento (ASPICE)
MC/DC	Cobertura de condição/decisão modificada
MiL	Modelo no circuito
MISRA	Associação de Confiabilidade de Software do Setor Automotivo
OEM	Fabricante do equipamento original
PA	Atributo de processo (ASPICE)
PIM	Aprimoramento de processos (ASPICE)
QM	Gerenciamento da qualidade
REU	Reutilização (ASPICE)
RTE	Ambiente de tempo de execução (AUTOSAR)
SiL	Software-in-the-loop
SOP	Início da produção
SPICE	Melhoria do processo de software e determinação de capacidade
SPL	Fornecimento (ASPICE)
SUP	Suporte (ASPICE)

Abreviação	Definição / Significado
SW-C	Componente de software (AUTOSAR)
SWE	Engenharia de software (ASPICE)
SYS	Engenharia de sistemas (ASPICE)
VAL	Validação (ASPICE)
VDA	Associação Alemã da Indústria Automotiva
XCP	Protocolo Universal de Medição e Calibração
XiL	X-in-the-loop: Um termo geral para todos os ambientes de teste in-the-loop, como MiL, SiL e HiL

6 Termos específicos do domínio

Prazo	Definição
Arquitetura de sistema aberto automotivo	Uma parceria de desenvolvimento que estabelece uma norma aberta do setor para a arquitetura de software na indústria automotiva. Observação: O termo também é usado para a arquitetura de software padronizada e a abordagem de desenvolvimento de sistema/software associada.
nível de integridade de segurança automotiva	Um nível de integridade de segurança para segurança funcional automotiva, definido na ISO 26262.
SPICE automotivo	Um modelo de referência de processo e um modelo de avaliação de processo associado no setor automotivo.
software básico (AUTOSAR)	No AUTOSAR, uma camada de software que consiste em componentes padronizados e orientados para o hardware.
caixa de interrupção	Uma unidade de medição para analisar, interromper ou manipular sinais físicos em fios.
sistema de ônibus	Uma rede de unidades de controle eletrônico que trocam informações por meio da mesma conexão.
dimensão de capacidade	Uma dimensão de um modelo de avaliação de processo que define os atributos do processo a serem avaliados.
indicador de capacidade	Um indicador de capacidade de processo usado na avaliação de capacidade de processo.
nível de capacidade	Um nível atribuído a um processo com base na avaliação dos atributos correspondentes do processo em uma avaliação da capacidade do processo.
sistema de circuito fechado	Um sistema no qual uma ação de controle ou uma entrada depende da saída ou de alterações na saída.
(MISRA-C)	Uma diretriz de programação em MISRA-C que não pode ser totalmente verificada por análise estática.
Descrição da configuração da ECU	Os dados necessários para integrar componentes de software a uma unidade de controle eletrônico.
Extrato de ECU	Dados de configuração do sistema para uma unidade de controle eletrônico.
unidade de controle eletrônico	No setor automotivo, um sistema incorporado que controla (sub)sistemas elétricos em um veículo.
MISRA-C	Uma diretriz de codificação para o uso da linguagem de programação C para sistemas críticos fornecida pela MISRA.
sistema de circuito aberto	Um sistema no qual uma ação de controle ou uma entrada é independente da saída ou das alterações na saída.
fabricante do equipamento original	No setor automotivo, um fabricante de automóveis.

Prazo	Definição
atributo do processo	Um conjunto de características mensuráveis de um processo para uma avaliação da capacidade do processo.
dimensão do processo	Uma dimensão de um modelo de avaliação de processos que define os processos a serem avaliados.
processo de desenvolvimento de produtos	Um processo que inclui todas as atividades desde a primeira ideia de produto até a produção.
em tempo real	O processamento de dados para que os resultados estejam disponíveis em um período predeterminado.
computador com capacidade para tempo real	Um computador capaz de processar sinais em tempo real.
liberação	Documentação de liberação de um item de liberação.
item de liberação	Um elemento identificável com funções implementadas, propriedades e uso pretendido.
processo de liberação	Um processo que leva a uma liberação.
recomendação de liberação	Uma recomendação para liberar ou não um item de liberação com base nos resultados do teste.
regra (MISRA-C)	Uma diretriz de programação em MISRA-C que é totalmente verificável por análise estática.
ambiente de tempo de execução (AUTOSAR)	No AUTOSAR, a camada de abstração que controla e implementa a troca de dados entre os componentes de software AUTOSAR, bem como entre o software aplicativo e o software básico, tanto dentro de uma unidade de controle eletrônico quanto entre unidades de controle eletrônico.
ciclo de vida da segurança	O ciclo de vida de um sistema relevante para a segurança, desde o início até o descarte.
tempo de simulação	O período de uma simulação de computador.
componente de software (AUTOSAR)	No AUTOSAR, um componente na camada de software de aplicativo independente de hardware.
verificação de componentes de software e verificação de integração (ASPICE)	No Automotive SPICE, uma verificação do software integrado com base no projeto arquitetônico do software.
verificação da unidade de software (ASPICE)	No Automotive SPICE, uma verificação das unidades de software quanto à consistência com seu projeto detalhado
verificação de software (ASPICE)	No Automotive SPICE, uma verificação do software integrado para verificar a consistência com os requisitos do software
Descrição da configuração do sistema	Os dados usados na integração de todas as unidades de controle eletrônico de um veículo.

Prazo	Definição
integração de sistemas e verificação de integração (ASPICE)	No Automotive SPICE, uma verificação dos elementos do sistema integrado para verificar a consistência com a arquitetura do sistema.
ciclo de vida do sistema	As fases de desenvolvimento de um sistema até sua desativação.
verificação do sistema (ASPICE)	No Automotive SPICE, uma verificação do sistema para verificar a consistência com os requisitos do sistema

7 Referências

7.1 Normas

Automotive SPICE 4.0	(2023), Automotive SPICE Process Assessment / Reference Model
ISO 14229	(2020), Road vehicles – Unified diagnostic services (UDS)
ISO 26262	(2018), Road vehicles – Functional safety
ISO 26262-1	(2018), Road vehicles – Functional safety Part 1: Vocabulary
ISO 26262-2	(2018), Road vehicles – Functional safety Part 2: Management of functional safety
ISO 26262-3	(2018), Road vehicles – Functional safety Part 3: Concept phase
ISO 26263-4	(2018), Road vehicles – Functional safety Part 4: Product development at the system level
ISO 26263-5	(2018), Road vehicles – Functional safety Part 5: Product development at the hardware level
ISO 26263-6	(2018), Road vehicles – Functional safety Part 6: Product development at the software level
ISO 26263-7	(2018), Road vehicles – Functional safety Part 7: Production, operation, service and decommissioning
ISO 26263-8	(2018), Road vehicles – Functional safety Part 8: Supporting processes
ISO 26263-9	(2018), Road vehicles – Functional safety Part 9: Automotive safety integrity level (ASIL)-oriented and safety-oriented analyses
ISO 26263-10	(2018), Road vehicles – Functional safety Part 10: Guidelines on ISO 26262
ISO/IEC 25010	(2023), Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) Product quality model
ISO/IEC 33020	(2019), Information technology – Process assessment Process measurement framework for assessment of process capability
ISO/IEC/IEEE 12207	(2017), Systems and software engineering – Software life cycle processes
ISO/IEC/IEEE 15288	(2023), Systems and software engineering – System life cycle processes
ISO/IEC/IEEE 24748-1	(2018), Systems and software engineering – Life cycle management Part 1: Guide for life cycle management
ISO/IEC/IEEE 29119-1	(2022), Software and systems engineering – Software testing Part 1: General concepts
ISO/IEC/IEEE 29119-2	(2021), Software and systems engineering – Software testing Part 2: Test processes
ISO/IEC/IEEE 29119-3	(2021), Software and systems engineering – Software testing Part 3: Test documentation
ISO/IEC/IEEE 29119-4	(2021), Software and systems engineering – Software testing Part 4: Test techniques

ISO/IEC/IEEE 29148 (2018), Systems and software engineering – Life cycle processes – Requirements engineering

MISRA C (2025), Guidelines for the use of the C language in critical systems

7.2 Documentos ISTQB®

[ISTQB_ALTTA_SYL] ISTQB® Advanced Level Technical Test Analyst Syllabus, Version 2012

[ISTQB_FL_SYL] ISTQB® Foundation Level Syllabus v4.0, 2023

7.3 Glossários

Referências para a terminologia utilizada neste syllabus:

IREB® Glossary https://cpre.ireb.org/en/downloads-and-resources/glossary

ISTQB® Glossary https://glossary.istqb.org/

8 Marcas registradas

CTFL® é uma marca registrada do German Testing Board (GTB) e. V. somente na UE

GTB® é uma marca registrada do German Testing Board (GTB) e. V. somente na UE

ISTQB® é uma marca registrada do International Software Testing Qualifications Board

Automotive SPICE® é uma marca registrada da Associação Alemã da Indústria Automotiva (VDA)

9 Apêndice A - Objetivos de aprendizagem/nível cognitivo de conhecimento

Os objetivos específicos de aprendizagem que se aplicam a este syllabus são mostrados no início de cada capítulo. Cada tópico do syllabus será examinado de acordo com o objetivo de aprendizado para ele

Os objetivos de aprendizagem começam com um verbo de ação correspondente ao seu nível cognitivo de conhecimento, conforme listado abaixo.

Nível 1: Lembrar (K1)

O candidato se lembrará, reconhecerá e recordará um termo ou conceito.

Verbos de ação: Recordar, reconhecer.

Exemplos
Relembrar os conceitos da pirâmide de teste.
Reconhecer os objetivos típicos dos testes.

Nível 2: Compreender (K2)

O candidato pode selecionar as razões ou explicações para declarações relacionadas ao tópico e pode resumir, comparar, classificar e dar exemplos para o conceito de teste.

Verbos de ação: Classificar, comparar, diferenciar, distinguir, explicar, dar exemplos, interpretar, resumir

Exemplos	Notas
Classificar as ferramentas de teste de acordo com sua finalidade e as atividades de teste que elas suportam.	
Compare os diferentes níveis de teste.	Pode ser usado para procurar semelhanças, diferenças ou ambos.
Diferenciar teste de depuração.	Procura diferenças entre os conceitos.
Distinguir entre riscos do projeto e do produto.	Permite que dois (ou mais) conceitos sejam classificados separadamente.
Explicar o impacto do contexto no processo de teste.	
Dê exemplos de por que os testes são necessários.	
Inferir a causa raiz dos defeitos a partir de um determinado perfil de falhas.	
Resumir as atividades do processo de revisão do produto de trabalho.	

Nível 3: Aplicar (K3)

O candidato pode executar um procedimento quando confrontado com uma tarefa familiar ou selecionar o procedimento correto e aplicá-lo a um determinado contexto.

Verbos de ação: Aplicar, implementar, preparar, usar

Exemplos	Notas
Aplicar a análise de valor limite para derivar casos de teste a partir de determinados requisitos.	Deve se referir a um procedimento / técnica / processo etc.

Exemplos	Notas
Implementar métodos de coleta de métricas para dar suporte aos requisitos técnicos e gerenciais.	
Preparar testes de instalabilidade para aplicativos móveis.	
Use a rastreabilidade para monitorar o progresso do teste quanto à integridade e à consistência com os objetivos, a estratégia e o plano de teste.	Pode ser usado em um LO que deseja que o candidato seja capaz de usar uma técnica ou um procedimento. Como "aplicar".

Referência

(Para os níveis cognitivos dos objetivos de aprendizagem)

Anderson, L. W. e Krathwohl, D. R. (eds) (2001) A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives (Uma revisão da taxonomia de Bloom dos objetivos educacionais), Allyn & Bacon

10 Apêndice B: Matriz de rastreabilidade entre resultados de negócio x objetivos de aprendizagem

Este capítulo lista a rastreabilidade entre os resultados de negócio e os objetivos de aprendizagem do Certified Tester Automotive Software Tester.

Resultados de negócio: Testador de software automotivo			BO1	BO2	BO3	BO4	BO5
AuT-BO1	Colaborar efetivamente em uma equipe de teste. ("colaborar")		37				
AuT-BO2	Adaptar as técnicas de teste conhecidas do ISTQB® Certified Tester Foundation level (CTFL®) aos requisitos específicos do projeto. ("adaptar")			7			
AuT-BO3	Considere os requisitos básicos dos padrões relevantes (ou seja, Automotive SPICE® e ISO 26262) para a seleção de técnicas de teste adequadas. ("selecionar")				27		
AuT-BO4	Apoiar a equipe de teste no planejamento baseado em riscos das atividades de teste e aplicar elementos conhecidos de estruturação e priorização. ("support & apply")					9	
AuT-BO5	Aplicar os ambientes de teste virtuais (ou seja, HiL, SiL e MiL) em ambientes de teste. ("aplicar")						13
LO exclusivo	Objetivo de aprendizado	Nível K					
1	Introdução						
1.1	Requisitos de objetivos de projeto divergentes e aumento da complexidade do produto						
AuT-1.1	Explicar e dar exemplos dos desafios do desenvolvimento de produtos automotivos que surgem dos objetivos divergentes do projeto e da crescente complexidade do produto	K2	X				

Resultados de negócio: Testador de software automotivo			BO1	BO2	BO3	BO4	BO5
1.2	Aspectos do projeto influenciados pelos padrões						
AuT-1.2	Recordar os aspectos do projeto que são influenciados pelos padrões (p. ex., tempo, custo, qualidade, riscos do projeto e riscos do produto)	K1	X				
1.3	Os seis estágios genéricos do ciclo de vida do sistema						
AuT-1.3	Lembre-se dos seis estágios genéricos do ciclo de vida do sistema de acordo com a ISO/IEC/IEEE 24748-1	K1	X				
1.4	A contribuição e a participação do testador no processo de liberação						
AuT-1.4	Recordar a função (ou seja, contribuição e colaboração) do testador no processo de liberação	K1	X				
2	Padrões para o teste de sistemas elétricos/eletrônicos (E/E)						
2.1	SPICE automotivo (ASPICE)						
AuT-2.1.1.1	Lembre-se das duas dimensões do Automotive SPICE (ASPICE)	K1			X		
AuT-2.1.1.2	Lembre-se das categorias de processo e dos grupos de processo da ASPICE [informativo]	K1			X		
AuT-2.1.1.3	Explicar os níveis de capacidade 0 a 3 do ASPICE	K2			X		
AuT-2.1.2.1	Relembrar a finalidade dos processos relevantes específicos do teste do ASPICE	K1	X		X		
AuT-2.1.2.2	Explicar o significado dos quatro níveis de classificação e dos indicadores de capacidade do ASPICE sob a perspectiva dos testes	K2			X		
AuT-2.1.2.3	Explicar os requisitos do ASPICE para a estratégia de teste, incluindo os critérios para verificação de regressão	K2	X		X	X	
AuT-2.1.2.4	Lembre-se dos requisitos do ASPICE para o software de teste	K1	X		X	X	
AuT-2.1.2.5	Usar medidas de verificação para verificação de unidades de software	K3	X		X	X	
AuT-2.1.2.6	Explicar os diferentes requisitos de rastreabilidade do ASPICE do ponto de vista dos testes	K2	X		X		

Resultados de negócio: Testador de software automotivo			BO1	BO2	BO3	BO4	BO5
2.2	ISO 26262						
AuT-2.2.1.1	Explicar o objetivo da segurança funcional para sistemas de E/E	K2			X		
AuT-2.2.1.2	Lembre-se da contribuição dos testadores para a cultura de segurança	K1	X		X		
AuT-2.2.2	Discutir a função do testador na estrutura do ciclo de vida da segurança de acordo com a ISO 26262	K2	X		X		
AuT-2.2.3.1	Lembre-se do design e da estrutura da ISO 26262 [informativo]	K1			X		
AuT-2.2.3.2	Relembrar as partes da ISO 26262 que são relevantes para o testador	K1	X		X		
AuT-2.2.4.1	Lembre-se dos níveis de criticidade da ASIL	K1	X		X		
AuT-2.2.4.2	Explicar a influência da ASIL nas técnicas de teste aplicáveis e nos tipos de teste para testes estáticos e dinâmicos e o escopo de teste resultante	K2	X	X	X	X	
AuT-2.2.5	Use as tabelas de métodos da ISO 26262	K3	X	X	X	X	
2.3	AUTOSAR						
AuT-2.3.1	Lembre-se dos objetivos do projeto AUTOSAR	K1			X		
AuT-2.3.2	Lembre-se do projeto geral do AUTOSAR [informativo]	K1			X		
AuT-2.3.3	Lembre-se da influência da AUTOSAR no trabalho do testador	K1	X	X	X		
2.4	Comparação entre ASPICE, ISO 26262 e CTFL®						
AuT-2.4.1	Lembre-se dos diferentes objetivos da ASPICE e da ISO 26262	K1			X		
AuT-2.4.2	Explicar as diferenças entre ASPICE, ISO 26262 e CTFL® com relação aos níveis de teste	K2	X	X	X		
3	Testes em um ambiente virtual						
3.1	Ambiente de teste em geral						
AuT-3.1.1	Lembre-se da motivação por trás de um ambiente de teste no desenvolvimento automotivo	K1	X				X

Resultados de negócio: Testador de software automotivo			BO1	BO2	BO3	BO4	BO5
AuT-3.1.2	Relembrar as partes gerais de um ambiente de teste específico para automóveis	K1	X				X
AuT-3.1.3	Lembre-se das diferenças entre sistemas de loop fechado e sistemas de loop aberto	K2	X				X
AuT-3.1.4	Recordar as funções essenciais, os bancos de dados e os protocolos de uma unidade de controle automotivo	K1	X				X
3.2	Testes em ambientes de teste XiL						
AuT-3.2.1.1	Lembre-se da estrutura de um ambiente de teste MiL	K1	X				X
AuT-3.2.1.2	Explicar a área de aplicação e as condições de limite de um ambiente de teste MiL	K2	X				X
AuT-3.2.2.1	Lembre-se da estrutura de um ambiente de teste SiL	K1	X				X
AuT-3.2.2.2	Relembrar as áreas de aplicação e as condições de limite de um ambiente de teste SiL	K1	X				X
AuT-3.2.3.1	Lembre-se da estrutura de um ambiente de teste HiL	K1	X				X
AuT-3.2.3.2	Explicar as áreas de aplicação e as condições de limite de um ambiente de teste de HiL	K2	X				X
AuT-3.2.4.1	Resumir as vantagens e desvantagens do teste usando critérios para os ambientes de teste XiL	K2	X			X	X
AuT-3.2.4.2	Aplicar critérios para a atribuição de uma determinada extensão do teste a um ou mais ambientes de teste	K3	X			X	X
AuT-3.2.4.3	Descrever os ambientes de teste XiL no modelo V	K1	X			X	X
4	Testes estáticos e testes dinâmicos						
4.1	Testes estáticos						
AuT-4.1.1	Explicar o propósito e os requisitos da diretriz MISRA-C com a ajuda de exemplos	K2	X	X			

Resultados de negócio: Testador de software automotivo			BO1	BO2	BO3	BO4	BO5
AuT-4.1.2	Aplicar a revisão de requisitos usando as características de qualidade da norma ISO 29148:2018 que são relevantes para os testadores	K3	X	X			
4.2	Testes dinâmicos						
AuT-4.2.1	Projetar casos de teste para obter cobertura de teste de condição/decisão modificada	K3	X		X		
AuT-4.2.2	Explicar o uso de testes back-to-back, dando exemplos	K2	X		X		
AuT-4.2.3	Explicar o teste de injeção de falhas, dando exemplos	K2	X		X		
AuT-4.2.4	Relembrar os princípios dos testes baseados em requisitos	K1	X		X		
AuT-4.2.5	Aplicar critérios dependentes do contexto para a escolha de técnicas de teste e abordagens de teste adequadas e necessárias	K2	X	X	X	X	

11 Apêndice C: Notas de versão

O syllabus do ISTQB® Automotive Software Tester v2.1 (2025) é uma pequena atualização da v2.0.1 (2017). Os objetivos de aprendizagem em si não foram alterados, mas o conteúdo de alguns objetivos de aprendizagem foi atualizado para refletir os padrões atualizados mencionados no syllabus. Além disso, foram consideradas as alterações entre o ISTQB CTFL 3.1 e o CTFL 4.0 e as atualizações no Glossário ISTQB. Além disso, a consistência e a compreensibilidade foram aprimoradas quando necessário.

Os seguintes padrões atualizados foram levados em consideração:

- ISO 24748-1:2024
- ISO 26262:2018
- SPICE Automotivo 4.0
- Versão AUTOSAR Classic R23-11
- MISRA-C:2025
- ISO 29148:2018

O esquema de nomenclatura dos objetivos de aprendizagem foi alterado para "AuT" mais o número da seção.

12 Apêndice D - Bancos de dados automotivos e protocolos de comunicação

Interfaces	Banco de dados	Protocolos de comunicação
Memória	ASAM MCD-2 MC (também ASAP2 ou A2L)	ASAM MCD-1 XCP (Protocolo Universal de Medição e Calibração) ASAM MCD-1 CCP (Protocolo de Calibração CAN)
Ônibus	Norma ASAM MCD-2 NET (também *FIBEX - Field Bus Exchange Format*)	FlexRay (ISO 17458) CAN (Rede de área do controlador de acordo com a ISO 11898-2)
	DBC (banco de dados de comunicação para CAN)	CAN (Rede de área do controlador de acordo com a ISO 11898-2)
Diagnóstico	ASAM MCD-3 D (Especificação da interface de programação de aplicativos) ASAM SOVD (Service-Oriented Vehicle Diagnostics) CDD (Descrição do diagnóstico do CANdelaStudio)	KWP2000 (ISO 14230) ISO-OBD (ISO 15031) UDS (ISO 14229) SOME/IP (Especificação AUTOSAR)

Tabela 6: Bancos de dados e protocolos de comunicação comuns do setor automotivo

A AUTOSAR tem um formato XML padronizado que integra os bancos de dados de um veículo completo. Ele é chamado de formato ARXML (AUTOSAR Integrated Master Table of Application Interfaces, esquema XML R21-11).