テスト技術者資格制度

Foundation Level Specialist シラバス
自動車ソフトウェアテスト担当者

V2.1.1.J01

International Software Testing Qualifications Board

ISTQB®
Certified Tester
Advanced Level
Agile Test Leadership at Scale

著作権について

変更の概要

バージョン	日付：	注
2.1.1	2025/12/21	エラッタ：GTB ワーキンググループによるドイツ語ローカライゼーションにおいて発見された問題を修正
2.1	2025/04/30	マイナーアップデート：参照標準、ISTQB 用語集、および ISTQB CTFL v4 の更新に伴い、既存コンテンツを変更。理解しやすくするために表現を改善し、現行のシラバステンプレートへ移行。詳細は付録 C（リリースノート）を参照。
2.0.2 （英語版）	2018/07/04	英語版（V2.0 DE の英語翻訳）の GA 承認
2.0 DE	2017/03/31	V.1.1 に基づく学習の目的と内容。 2017 年 3 月 31 日にフランクフルトで開催された GTB 作業グループ会合にて、対応するドイツ語版をリリース。
1.1 DE	2015/6/14	内容のレビューと、ドイツ ISTQB®テスト技術者資格制度 Foundation Level シラバス 2011 V1.0.1 と ISTQB®用語集 V2.2 に対する比較。 2015 年 3 月 15 日にミュンヘンで開催された GTB 作業グループ会合にてリリース。
1.0 DE	2011/01/19	作成者：Dr. Hendrik Dettmering、gasq GmbH の要請により執筆。 著作権はドイツテスト委員会 e.V.に完全に譲渡済み。

JSTQB®

バージョン	日付：	注
2.1.1.J01	2026/07/03	Foundation Level Specialist CTFL® Automotive Software Tester Version2.1.1 の日本語翻訳版
2018.J03	2022/08/14	些細な用語の修正 1.3、2.1.2.1
2018.J02	2021/01/05	誤記等の修正
2018.J01	2020/1/25	Foundation Level Specialist CTFL® Automotive Software Tester Version2.0.2 の日本語翻訳版

目次

謝辞

本書は、2018 年に ISTQB®総会によって正式にリリースされた。改訂版 2.1 は、所定のプロセスに則り、CT-AuT プロダクトオーナーHorst Pohlmann によって正式にリリースされた。

本書は、International Software Testing Qualifications Board（ISTQB®）のため、ドイツテスト委員会の以下のメンバーによって作成された：Ralf Bongard (lead), Klaudia Dussa-Zieger, Matthias Friedrich, Thorsten Geiselhart, Horst Pohlmann, Ralf Reißing, Alexander Schulz.

技術レビューは Michael Humm が担当した。また、レビュー担当チームおよび加盟委員会からの提案と意見に感謝する。

本シラバスのレビュー、コメント、投票には以下の者が参加した。

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.
技術編集は Gary Mogyorodi が担当した。

日本語訳については、以下の日本語翻訳ワーキンググループメンバーにより行われた。

日本語翻訳ワーキンググループメンバー：

大段 智広 （DMM.com）
大西 建児（ガイオ・テクノロジー）h
川上 朋也（デンソー）
鈴木 正人（デンソー）
須原 秀敏（ベリサーブ）
蛸島 昭之（サイバネット MBSE）
永田 哲（ASTER）
林 宏昌（デンソー）
福田 里奈（メルカリ）
町田 欣史（NTT データ）
森 貴彦（デンソー）
山上 直宏（デンソー）

本書は、「付録 - 参考文献」章にある日本語翻訳文献の慣習を参考に翻訳している。そのため、同じ英語の訳が一致しないケースがある（例えば product の訳が「製品」と「プロダクト」）。

0 イントロダクション[1]

0.1 本書の目的

本シラバスは、International Software Testing Qualifications Board（以降では ISTQB®として参照）による自動車ソフトウェアテスト担当者資格の基礎となる。ISTQB®は本シラバスを以下の目的で提供する。

1. 加盟ボードには、現地言語への翻訳や、研修プロバイダーの認定のために提供する。加盟ボードはシラバスを各国の言語事情に合わせて調整し、参考文献も現地の出版物に合わせて修正することができる。
2. 認証機関には、本シラバスの学習目標に基づき、現地言語で試験問題を作成するために提供する。
3. 研修プロバイダーには、研修教材の作成や、適切な指導方法の決定のために提供する。
4. 資格取得希望者には、認定試験の準備（研修コースの一環として、または独学で）に活用するために提供する。
5. 国際的なソフトウェアおよびシステムエンジニアリングコミュニティには、ソフトウェアおよびシステムテスト分野の発展、ならびに書籍や論文の基礎情報として提供する。

0.2 認定テスト技術者 自動車ソフトウェアテスト担当者

自動車ソフトウェアテスト担当者資格は、自動車関連のソフトウェアのテストに関係するすべての人を対象としている。この対象者には、テスト担当者、テストアナリスト、テストエンジニア、テストコンサルタント、テストマネージャー、受け入れテスト担当者、ソフトウェア開発担当者などが含まれる。また、本資格は、自動車関連のソフトウェアのテストに関して基本的な知識の習得を目指すプロジェクトマネージャー、安全マネージャー、品質マネージャー、ソフトウェア開発マネージャー、ビジネスアナリスト、IT マネージャー、マネジメントコンサルタントも対象にする。

0.3 テスト技術者のキャリアパス

ISTQB®認定テスト技術者資格制度は、テスト技術者のキャリアのあらゆる段階を支援する。
ISTQB®認定テスト技術者 自動車ソフトウェアテスト担当者資格を取得した者は、コアアドバンスドレベル（テストアナリスト、テクニカルテストアナリスト、テストマネージャー）や、さらにその先のエキスパートレベル（テストマネジメントやテストプロセス改善）へのステップアップも視野に入れることができる。スペシャリストの専門分野では、アジャイルテスト、AI テスト、モバイルアプリテストなど、特定のテスト手法や活動に特化した分野や、ギャンブルやゲームなど特定業界向けのテストノウハウを深く学ぶことができる。最新の ISTQB 認定テスト技術者資格制度の情報は、www.istqb.org を参照のこと。

[1] 本テキストの主要部分は、ISTQB® CTFL Core シラバスから引用されている [20]

0.4 ビジネス上の成果

本節では、自動車ソフトウェアテスト担当者として認定された者が習得していることが期待される、ビジネス上の成果を一覧で示す。

自動車ソフトウェアテスト担当者として認定された者は、以下の技能を習得している

AuT-BO-01	テストチーム内で効果的に*共同作業する*（「*共同作業する*」）。
AuT-BO-02	ISTQB®テスト技術者資格制度 Foundation Level（CTFL®）のテスト技法を、具体的なプロジェクト要件に*適合させる*（「*適合させる*」）。
AuT-BO-03	関連する標準（Automotive SPICE®、ISO 26262 など）の基本的な要件を考慮して、適切なテスト技法を*選択する*（「*選択する*」）。
AuT-BO-04	リスクの存在するテスト活動の計画作成においてテストチームを支援し、構造化と優先順位付けの既知の要素を*適用する*（「*支援および適用する*」）。
AuT-BO-05	仮想環境（HiL、SiL、MiL など）へテスト手法を*適用する*（「*適用する*」）。

0.5 学習目的と認知レベル

学習目的はビジネス上の成果を支援し、自動車ソフトウェアテスト担当者技術者資格試験の作成に利用される。各章の冒頭に具体的な学習目的のレベルが示されており、以下のように分類される。

- K1：記憶
- K2：理解
- K3：適用

学習目的のさらなる詳細および例については付録 A を参照のこと。章の見出しの直下にキーワードとして列挙されているすべての用語については、学習目的で明示的に記載されていなくても、ISTQB®用語集に基づく正しい名称と定義を記憶しておく必要がある（K1）。

0.6 自動車ソフトウェアテスト担当者認定資格試験

自動車ソフトウェアテスト担当者資格試験は、本シラバスに基づいて実施される。試験問題では、本シラバスの複数の章から出題される場合がある。導入部、付録、および「参考情報」と記載されたセクション以外は、本シラバスのすべての部分が試験範囲となる。試験では、すべての K レベルにおけるキーワードや学習目標の理解度が評価される。

基準や書籍は参考文献として挙げられているが、それらに関して出題されるのは、シラバス本文で要約されている内容のみである。詳細については、「試験構造および規則」ならびに「認定テスター自動車ソフトウェアテスター シラバス」文書を参照のこと。

認定自動車ソフトウェアテスト技術者試験を受験するためには、ISTQB®認定テスト技術者 - Foundation レベル（CTFL®）資格を取得済みであり、自動車開発プロジェクトにおけるテストへの関心が必要である。

本シラバスに基づき、Foundation Level スペシャリスト自動車ソフトウェアテスト担当者という、ドメイン固有の資格認定のための試験が追加された。試験問題では、本シラバスの複数の章からの内容が問われることがある。試験の各問題は、主要な用語に割り当てられた問題を除いて、一般的に 1 つの学習の目的に割り当てられる。試験の形式は多肢選択式である。試験は、認定トレーニングコースの直後に、または（例えば試験センターや公的試験場で）独立して実施してもよい。受験に際して、コース受講は必須ではない。

受験希望者は以下のことについても満たしていることが求められる。

- ソフトウェア開発またはソフトウェアテストの最低限の背景知識を有する（例えば、ソフトウェアテスト担当者、もしくはソフトウェア開発担当などとして 6 ヶ月程度の経験がある）。または、
- ISTQB®標準（ISTQB®メンバー委員会）により認定されたコースを受講している。そして/または、
- 自動車業界での E/E 開発プロジェクトでテストに関して初歩的な経験がある。

受験資格に関する注意：STQB®テスト技術者資格制度 Foundation Level（CTFL®）を、自動車ソフトウェアテスト技術者資格制度の受験前に取得しておく必要がある。

0.7 認定審査

ISTQB®のメンバー委員会にて、教育コースの教材が本シラバスに従っている教育機関を認定する。教育機関は、委員会または認定を行う機関から認定ガイドラインを入手しなければならない。教育コースがシラバスに従っていると認定されると、教育コースの一部として ISTQB の試験を実施することができる。

本シラバスの認定ガイドラインは、プロセスマネジメントおよびコンプライアンス WG が公開している一般的な認定ガイドラインに準拠する。

0.8 標準（規格）の扱い

IEEE や ISO などの国際標準化団体は、品質特性やソフトウェアテストに関連する標準を発行している。本シラバスでは、これらの標準を引用しているが、その目的は（ISO/IEC 25010 の品質特性のように）フレームワークを提供すること、あるいは読者に付加情報を提供することにある。 なお、本シラバスは標準ドキュメントを「参照」として引用しており、標準ドキュメントそのものが試験範囲となるわけではない。標準の詳細については「第 7 章：参考文献」を参照されたい。

0.9 詳細レベル

本シラバスの詳細レベルは、国際的に統一されたコースおよび試験を可能にするものである。この目的を達成するため、本シラバスは以下で構成される。

- 自動車ソフトウェアテスト技術者シラバスの意図を記述した一般教授目標
- 学習者が想起できなければならない用語のリスト
- 各知識領域において達成すべき認知的学習成果を記述した学習事項
- 認められた文献や標準規格などの参照先を含む、主要概念の解説

本シラバスの内容は、自動車ソフトウェアテストの知識領域全体を網羅したものではない。これは自動車ソフトウェアテスト技術者トレーニングコースにおいてカバーされるべき詳細レベルを反映した

ものである。また、自動車分野におけるソフトウェアプロジェクトに適用されるテスト概念および技法に焦点を当てている。

本シラバスでは、ISTQB®用語集に従ったソフトウェアテストおよび品質保証の用語（名称と定義）を使用している。 関連分野の用語については、それぞれの用語集を参照すること。機能安全エンジニアリングについては ISO 26262 を、要求工学については IREB®用語集を参照されたい。

0.10 本シラバスの構成

本シラバスの試験対象となる章は 4 つである。各章の最上位見出しに、その章に割り当てられた時間を明記している。ただし、章より下位のレベルでの時間配分は示していない。認定トレーニングコースにおいては、本シラバスは 4 つの章全体で最低 12 時間の指導時間を必要とし、その配分は以下の通りである。

- 第 1 章：自動車ソフトウェアテストへのイントロダクション（30 分）
  - テスト担当者は、相反する目標や増大する複雑性、さらには規格（スタンダード）が時間・コスト・品質・リスクに及ぼす影響など、車載製品開発における課題について学ぶ。
  - また、製品ライフサイクルの 6 つのフェーズと、製品リリースにおける自らの役割についても学習する。
- 第 2 章：E/E システムのテストに関する標準（300 分）
  - テスト担当者は ASPICE について、その能力レベル、テスト関連プロセス、ドキュメント要件、テスト戦略の役割に加え、ASPICE が求めるトレーサビリティやテスト戦略への期待事項を学ぶ。
  - ISO 26262 に関して、テスト担当者は安全文化、自動車安全整合性レベル、テスト技法に焦点を当て、安全ライフサイクル内における自身の役割を理解する。
  - また、テスト担当者は AUTOSAR がテストに与える影響を理解し、ASPICE、ISO 26262、および CTFL®の目的とそれぞれのテストレベルを比較する。
- 第 3 章：仮想環境でのテスト（160 分）
  - テスト担当者、クローズドループおよびオープンループシステムを含む、自動車仮想テスト環境の目的、構造、および主要な機能について学ぶ。また、さまざまな XiL テスト環境（すなわち MiL、SiL、および HiL）、それらの用途、利点、および限界についての理解を深める。
  - さらに、各テスト環境に対して適切なテスト範囲（スコープ）を割り当てる方法を学習する。
- 第 4 章：静的テストおよび動的テスト（230 分）
  - テスト技術者は、MISRA-C などのコードに関する主要なガイドラインや、ISO/IEC/IEEE 29148 などの要件に関するガイドラインの説明方法と適用方法を学ぶ。

テスト技術者**は、改良条件決定網羅（MC/DC）のテストケースを作成し、故障注入テストやバックツーバックテストを理解した上で、プロジェクトの状況に基づいて適切なテスト技法やテストアプローチを選択する。**

1 自動車ソフトウェアテストへのイントロダクション[30 分]

用語

なし

学習の目的

1.1 多様化するプロジェクト目標やプロダクトの複雑さの増大により発生する要件

AuT-1.1 （K2）多様化するプロジェクト目標やプロダクトの複雑さの増大により発生する自動車プロダクト開発の課題を説明し、例を挙げる。

1.2 標準の影響を受けるプロジェクトの側面

AuT-1.2 （K1）標準により影響されるプロジェクトの側面（例えば時間、コスト、品質、プロジェクト/プロダクトリスクなど）を想起する。

1.3 システムライフサイクルでの一般的な 6 つのフェーズ

AuT-1.3 （K1）ISO/IEC/IEEE 24748-1 で定義されているシステムライフサイクルの 6 つの一般的なフェーズを想起する。

1.4 リリースプロセスでのテスト担当者の役割と参画

AuT-1.4 （K1）リリースプロセスにおけるテスト担当者としての役割（例えば貢献や協力）を想起する。

イントロダクション

ソフトウェアテストの 7 原則の 1 つは、「テストは状況次第」である。本章では、「自動車ソフトウェアテスト担当者」[2]が活動する E/E システム開発のコンテキストを概説する。本環境では、コスト効率性や安全要求、市場への投入の速度などの相互に影響を与え合うゴール、複雑さの増大、イノベーティブなソリューションに対する強い要望により、特有の課題が発生している。一方で、自動車の標準とライフサイクルにより、テスト担当者が活動するフレームワークが形成される。最終的にテスト担当者は、ソフトウェアとシステムのリリースに向けて行動している。

1.1 多様化するプロジェクト目標やプロダクトの複雑さの増大により発生する要件

自動車メーカーやサプライヤーが発表する新しいモデルの数は年々増えており[3]、コストダウン圧力も増大している。このプロセスに影響を及ぼすものとして以下を挙げることができる。

- モデルの数と複雑度の増加
  個々のエンドユーザーのニーズによりよく応えるために、OEM（自動車メーカー）はますます多くの自動車モデルを提供する。ただし、このことは、モデルごとの販売台数が少なくなることを意味する。開発コストおよび生産コストの上昇に対処するために、メーカーは複数のモデルを共通のプラットフォームのバリエーションとして開発する。しかし共通プラットフォームの開発は単一モデルの開発よりもはるかに複雑である。これは、可能性のある多くのバリエーションを管理する必要があるためである。
- 機能の増加
  エンドユーザーは既存の機能を損なうことなく、ますます多くのイノベーションが要求されるため、機能が増加する。

[2] 以降では、「テスト担当者」という用語のみを使用する。この用語は「自動車 E/E ソフトウェアテスト担当者」を表す。
[3] 経営コンサルタント企業 Progenium による調査からの例：「1990 年に新たに提供されたモデルは 101 のみであったが（中略）、2014 年にその数が 453 に増加している」**エラー! 参照元が見つかりません。**

- 構成の数の増加
  エンドユーザーはそれぞれ自分の好みに合わせることのできる車が欲しい。この要求を満たすには、1 つの自動車モデルで複数の構成を用意するだけでなく、機能の範囲も増加させる必要がある。
- 品質要件の増加
  機能や複雑度が増加したにもかかわらず、エンドユーザーは自動車の機能に対して従来と同等またはそれ以上の品質を期待する。

プロジェクトの目標である、時間、コスト、およびスコープは競合する（「プロジェクト管理トライアングル」として知られており、品質はスコープの一側面として考慮される）。自動車メーカー（OEM を含む）とサプライヤーはシステムの開発の効率をさらに向上させて、複雑度や品質要件が増加し、予算が抑えられる中で、開発期間を短くする必要がある。

1.2 標準の影響を受けるプロジェクトの側面

標準は、時間、コスト、品質、プロジェクトのリスクやプロダクトのリスクなど、プロジェクトの主要な側面に影響を及ぼす。

- 標準は次の特性によってプロセスの効率性を向上させる（例：安定した品質を維持しながら開発にかかる時間またはコストを削減する）。
  - 統一性のある名前付け
  - 高い透明性
  - 協調作業の容易化（内部および外部）
  - 再利用性の向上
  - 経験（ベストプラクティス）の共有
- 確立されたテクノロジー ガイドラインを使用することで、標準はリスクの検出や欠陥の早期検出と解決に役立つ。
- 標準は監査にとって基本である。このため、監査者はプロダクトまたはプロセスの品質を評価できる。同時に、監査者は標準が要件を満たしていることをチェックできる。
- 標準は、契約上または規制上の指令およびガイドラインの一部である。

本シラバスでは、自動車ソフトウェアテストに関連する以下の標準を参照する。

- 標準化されたプロセスや手法を提供する ISO 26262 と Automotive SPICE（ASPICE）。
- ソフトウェアアーキテクチャーやプロダクトを標準化する AUTOSAR。

1.3 システムライフサイクルでの一般的な 6 つのフェーズ

自動車とすべての車載コンポーネント[4]のシステムライフサイクルは、プロダクトのアイデアで始まり、自動車の廃棄で終了する。本ライフサイクルは、開発、ビジネス、物流、生産技術プロセスを含む。開始基準と終了基準を定義してマイルストーンとして使用することで、プロセスの成熟度を高めるのとシステムライフサイクル[5]を以下の 6 つのフェーズに分類して典型的なテスト活動[6]と構成することに役立つ：

- 概念：テスト戦略とテスト目的を定義するためのテスト計画作業
- 開発：品質を確実にするためのテスト分析、テスト設計、テスト実装、テスト実行、テストモニタリング、テストコントロール、テスト完了

[4] 電子制御ユニット（ハードウェアおよびソフトウェア）とコンポーネント。
[5] ISO 26262 の安全ライフサイクルも同様のフェーズで構成される。
[6] テスト活動についてはこちらも参照されたい：基本的なテストプロセス**エラー! 参照元が見つかりません。**

- 生産：準備状態を検証するための完成品性能テスト
- 利用：特有のテスト活動は行われない
- 支援：継続的に信頼性を提供するためのメンテナンステスト
- 廃止：運用停止やデータ移行を確認するための移行テスト

自動車業界の一般的なプロダクト開発プロセスは、大まかに、構想、開発、および生産の 3 つのステージで構成され、テストやリリースの活動がその中で実施される。

1.4　リリースプロセスでのテスト担当者の役割と参画

自動車製造の環境では、目標が達成されたことが検証された場合にプロジェクトとしてリリースを宣言することでマイルストーンに到達する。これ以降、リリースアイテムはその意図した用途に必要とされる成熟度レベルを満たすと考えられる。

リリースアイテムは、パラメーターを含むソフトウェア構成、必要に応じてハードウェアとメカニクスも対象にするとサポート文書で構成される。

テスト担当者は、リリースプロセスの重要な以下の情報を最終的なテストレポートにより報告する：

- テスト済みアイテム
- 評価済みの品質特性（例えば応答時間やリソース消費）
- 既知の欠陥
- プロダクトメトリクス
- テスト終了基準が達成されたときのリリース規則に基づくリリース推奨の情報。リリース規則はベストプラクティスガイドライン（例えば、テストコースや公道でのテスト、またはインストール推奨）で提供されているものに従う。

さらに、テスト担当者は、リリースに関連する以下の成果物の作成に参加する。

- 変更の優先度付けと、判定作業。
- 機能の優先度付け（実装の順序決め）。

2　E/E システムのテストに関する標準[300 分]

用語

機能安全、手法テーブル

ドメイン特有の用語

自動車用安全度水準（ASIL）、ソフトウェア検証、システム検証

学習の目的

2.1 Automotive SPICE（ASPICE）

AuT-2.1.1.1　（K1）ASPICE の 2 つの座標軸を想起する。

AuT-2.1.1.3　（K2）ASPICE の能力レベル 0～3 を説明する。

AuT-2.1.2.1　（K1）ASPICE のテスト特有プロセスの目的を想起する。

AuT-2.1.2.2　（K2）テストの観点から、ASPICE の 4 つの評定尺度と能力指標の意味を説明する。

AuT-2.1.2.3　（K2）リグレッション検証の基準を含むテスト戦略に関する ASPICE の要件を説明する。

AuT-2.1.2.4　（K1）テストウェアに関する ASPICE の要件を想起する。

AuT-2.1.2.5　（K3）ソフトウェアユニット検証用の検証方策を使用する。

AuT-2.1.2.6　（K2）テストの観点から ASPICE のトレーサビリティ要件を説明する。

2.2 ISO 26262

AuT-2.2.1.1　（K2）E/E システムの機能安全の目的を説明する。

AuT-2.2.1.2　（K1）安全文化におけるテスト担当者の役割を想起する。

AuT-2.2.2　（K2）ISO 26262 で規定されている安全ライフサイクルのフレームワークでのテスト担当者の役割を説明する。

AuT-2.2.3.2　（K1）テスト担当者に関連する、ISO 26262 のパートを想起する。

AuT-2.2.4.1　（K1）ASIL の安全度レベルを想起する。

AuT-2.2.4.2　（K2）静的テストや動的テストのためのテスト技法や、テストタイプに関する ASIL の影響および結果としてのテスト範囲を説明する。

AuT-2.2.5　（K3）ISO 26262 の手法テーブルを解釈できるようになる。

2.3 AUTOSAR

AuT-2.3.1　（K1）AUTOSAR のプロジェクトの目的を想起する。

AuT-2.3.3　（K1）テスト担当者の作業に対する AUTOSAR の影響を想起する。

2.4 ASPICE、ISO 26262、CTFL®の比較

AuT-2.4.1　（K1）ASPICE と ISO 26262 の目的の差異を想起する。

AuT-2.4.2　（K2）テストレベルに関する ASPICE、ISO 26262、CTFL®の間の相違を説明する。

2.1 Automotive SPICE（ASPICE）

イントロダクション

プロセス改善は、システムの品質は開発プロセスの質により決定される、というアプローチに従う。プロセスモデルは、組織のある単位やプロジェクトのプロセス能力をモデルと比較して測定することにより、改善のための選択肢を提供する。さらに、評価結果を使用することで、組織のある単位やプロジェクトのプロセスを改善するためのフレームワークとしてモデルを適用できる。

2001 年、SPICE[7] User Group and the Automotive Special Interest Group（AUTOSIG）は ASPICE の策定に着手した。2005 年に発行されて以降、本標準は自動車業界で確固とした標準として認められている。本段落でのすべての記述は、ASPICE のバージョン 4.0 に基づいている。

2.1.1 標準の設計と構成

2.1.1.1 ASPICE の 2 つの座標軸

ASPICE は、アセスメントモデルを 2 つのアセスメントの座標軸で定義する。

プロセス座標において、ASPICE はプロセス参照モデル（PRM）を定義する。この PRM は、組織のプロセスの比較対象として参照され、組織の評価と改善を可能にする。各プロセスに対して、ASPICE は目的と成果、さらには要求されるアクション（すなわち、基本プラクティス）と情報項目（すなわち作業成果と作業成果物）を定義する。

ASPICE の**能力座標**は、能力レベルごとにいくつかのプロセス属性（すなわち、測定可能な特徴）を定義する。プロセスそれぞれについて、プロセス固有の属性と共通する属性がある。ISO/IEC 33020 は、プロセス能力のアセスメントの基盤として機能する。

これらのモデルを使用することで、プロセス（すなわちプロセス座標）をプロセスの能力（すなわち能力座標）という側面において評価できる。

2.1.1.2 プロセス座標でのプロセスカテゴリー［情報提供］

ASPICE はプロセスをグループにまとめており、さらにグループはカテゴリーに分類される：

主要ライフサイクルプロセスは、主要なプロセスとして機能するすべてのプロセスを含む。

- 取得プロセス群（ACQ）
- 供給プロセス群（SPL）
- システムエンジニアリングプロセス群（SYS）
- ソフトウェアエンジニアリングプロセス群（SWE）
- 機械学習エンジニアリングプロセス群（MLE）
- ハードウェアエンジニアリングプロセス群（HWE）
- 妥当性確認プロセス群（VAL）

支援ライフサイクルプロセスは他のプロセスを支援するすべてのプロセスを含む。

- 支援プロセス群（SUP）

組織ライフサイクルプロセスは企業の目的を支援するすべてのプロセスを含む。

- 管理プロセス群（MAN）
- プロセス改善プロセス群（PIM）
- 再利用プロセス群（REU）

[7] 「Software Process Improvement and Capability dEtermination」の頭字語。

テスト担当者にとっては、システムエンジニアリング（SYS）とソフトウェアエンジニアリング（SWE）の 2 つのプロセス群が主な着目点であり、また、MLE や VAL、HWE も着目点になりうる。

2.1.1.3　能力座標の能力レベル（CL）

アセッサーは 6 つのレベルのアセスメントシステム（すなわちレベル表示）を使用してプロセス能力をアセスメントする。ASPICE は以下に示すように、能力レベル 0～3[8]を定義する。

- レベル 0（不完全なプロセス）：プロセスは実装されていないか、またはそのプロセスの目的を達成していない。このレベルにおいては、プロセス成果のシステマティックな達成に関する証拠がほとんどないかまたは全くない。例：テスト担当者は要件のわずかな部分のチェックのみを行う。
- レベル 1（実施されたプロセス）：実装されたプロセスが、そのプロセス目的を達成している（ただし、一貫性なく実施された可能性あり）。例：テストプロセスに対して完全な計画が明らかになっていない。ただし、テスト担当者は要件の達成度合いを示すことができる。
- レベル 2（管理されたプロセス）：プロジェクトは計画され、実施中のプロセスは管理される。特定の状況下では、目的を満たすように、実施中に活動方針を調整する。作業成果物の要件が定義される。プロジェクトメンバーが作業成果物をレビューし、承認する。例：テストマネージャーがテスト目的の定義、テスト活動の計画、およびテストプロセスの管理を行う。ここには逸脱が発生したときの対応も含む。
- レベル 3（確立されたプロセス）：プロジェクトは標準化されたプロセスを使用し、指摘事項に従って常に改善を行う。例：組織的なテスト戦略がある。テスト完了後に、テストマネージャーはテストのさらなる改善を行う。

2.1.2　ASPICE のテストに関する要件

2.1.2.1　テスト固有のプロセス

ASPICE は、ソフトウェア開発およびシステム開発のプロセスに従ってテストプロセスを定義する[9]。

- ソフトウェアユニット検証（SWE.4）は、静的テストと動的テストを必要とする。本検証では、ソフトウェアのコンポーネントをその詳細設計（SWE.3）に基づいて検証する。
- ソフトウェアコンポーネント検証および統合検証（SWE.5）では、統合されたソフトウェアを、ソフトウェアアーキテクチャー（SWE.2）に基づいてテストする。
- ソフトウェア検証（SWE.6）は、統合されたソフトウェアを、ソフトウェア要件（SWE.1）に基づいてテストする。
- システム統合および統合検証（SYS.4）では、統合されたシステムを、システムアーキテクチャー（SYS.3）に基づいてテストする。
- システム検証（SYS.5）では、統合されたシステムを、システム要件（SYS.2）に基づいてテストする。

2.1.2.2　評定レベルと能力指標

アセッサーは、能力指標を介してプロセス能力をアセスメントできる。ASPICE は、能力指標を 9 つのプロセス属性（PA）で定義する。能力レベル 1～3 については、以下のように定義されている（カッコ内に SWE.6 の例を示す）。

- PA 1.1：プロセス実施プロセス属性（例えば、テスト担当者はそのテストプロセスに従ってプロセスを実施する）。

[8] 能力レベル 4 と 5 は、現在、自動車業界で注目されていない。

[9] 本文に示すテストプロセスに加えて、プロセスグループ VAL も存在する。VAL.1 の目的は、「エンドユーザーとの直接的なやりとりをする最終製品が、その運用ターゲット環境において意図された使用上の期待を満たしているという証拠を提供すること」です。

- PA 2.1：実施管理プロセス属性（例えば、テスト担当者はテスト活動の計画、管理、制御を行う）。
- PA 2.2：作業成果物管理プロセス属性（例えば、テスト担当者は、テストウェアの品質チェックを行う）。
- PA 3.1：プロセス定義プロセス属性（例えば、テストプロセスの責任者は組織的なテスト戦略の定義を行う）。
- PA 3.2：プロセス展開プロセス属性（例えば、テスト担当者は、PA 3.1 で定義されたテスト戦略を適用する）。

プロセス実施については、ASPICE は基本プラクティス（BP）と情報項目の 2 種類の能力指標を定義している。さらに、共通プラクティス（GP）と情報項目が定義されている。プロセス属性は以下 4 つの評定尺度の指標の実装レベルに基づきアセスメントされる[7][48]。

- **N**（None）：達成していない（0 ≦ 15%）
- **P**（Partly）：部分的に達成している（＞ 15 ～ ≦ 50%）
- **L**（Largely）：おおむね達成している（＞ 50 ～ ≦ 85%）
- **F**（Fully）：十分に達成している（＞ 85 ～ ≦ 100%）

プロセスが特定の能力レベルに到達するには、能力指標が「おおむね達成している（L）」または「十分に達成している（F）」でなければならない。

2.1.2.3　テスト戦略とリグレッション検証の基準

ASPICE は能力レベル 2 以降は各テストプロセス（2.1.2.1 参照）に対してテスト戦略（すなわち検証手段）を必要とする。テストマネージャーが、テスト計画作業時にこのテスト戦略を策定する。テスト戦略の開発作業では、テストガイドライン、プロジェクト目的、契約上および規制上の要件がベースになる。

テスト担当者は、早期のテストを原則として認識している。これは、自動車環境でのソフトウェアのテストにも適用される。ただし、車両環境では、より高いテストレベルのテスト環境は著しく高価であるため、別の観点を考慮する必要がある。例えば、より高いレベルのテストでは、特別に開発されたハードウェア（プロトタイプまたはテスト専用ハードウェア）が必要である。テスト戦略はレベルごとにテスト環境を定義するが、テスト担当者はどのテスト環境でどのテストを実施するかを定義する。

リグレッション検証の基準は、テスト戦略の重要な部分である。この基準はリグレッション検証[10]の詳細を特定する。ここでの課題として、経済的な側面を考慮してテストケースを選択する必要がある（「テストの付加価値」）。リグレッション検証の基準は、リグレッションテストを選択するためのテスト目的とテストアプローチを定義する。例えば、選択はリスクベースで行うことができる。影響度分析を使用することで、テスト担当者はリグレッションテストで重視する必要のある領域を識別できる。ただし、テストマネージャーがテスト担当者に対して、自動化されたすべてのテストケースをリリースごとに繰り返すよう依頼することもある。

2.1.2.4　ASPICE でのテストウェア

テストウェアに対し、ASPICE4.0 は要求を課していない。代わりに、ASPICE はテストの中で生成される特有の情報の文書化に関する要求を設定している。

- WP 08-60：検証手段（例えば、シミュレーションや動的テスト、静的テスト）、特にテスト計画に記述されているもの
- WP 15-52：検証結果（例えば、文書化された検証ログやテストレポート、欠陥レポート）

[10] ISTQB におけるリグレッション回避テスト戦略と類似している

各情報項目について、ASPICE は情報項目特性（IIC）と内容の例を定義している。これらはプロセス実施の客観的指標として機能する。

ISO/IEC/IEEE 29119-3 は情報項目のより詳細な構造化のために使用できる。

2.1.2.5　ソフトウェアユニット検証向けの検証手段

テスト担当者は、検証戦略に従って、ソフトウェア詳細設計および機能/非機能要件との適合性を検証する。この戦略は、テスト担当者が証跡を提供する方法を定義する。このため、テスト担当者はソフトウェアユニットを検証するために、静的テスト技法と動的テスト技法をさまざまに組み合わせることができる。

SWE.4.BP.1 で、ASPICE はソフトウェアユニット検証用の検証手段（2.1.2.4 参照）の定義を必要とする。このため、テスト担当者はソフトウェアユニットが要件を達成している割合を評価できる。以下の例はソフトウェアユニットの検証用の検証手段となりうる。

- テストデータを含むソフトウェアユニットテストケース
- ツールで支援された静的解析。それらはコーディング標準（例えば MISRA-C など、4.1.1 を参照）との適合性を評価する
- ソフトウェアユニットまたはソフトウェアユニットの特定箇所（パーツ）向けのレビュー（ツールで支援された静的解析でアセスメントできないもの）

開発担当者がソフトウェアユニットを変更した場合、テスト担当者もこの変更を評価しなくてはならない。このため、ソフトウェアユニットの検証手段は、リグレッション検証の基準も含む。これには、変更されたコードの検証、確認テスト、変更されていない部分の検証の繰り返し（静的/動的リグレッションテスト）が含まれる。

2.1.2.6　ASPICE でのトレーサビリティ

CTFL® Core シラバス[ISTQB_FL_SYL]と同じく、ASPICE も双方向トレーサビリティ[11]を必要とする。これにより、テスト担当者は以下の実施が可能になる。

- 影響度分析、例えば変更影響分析
- カバレッジ評価
- ステータス追跡

さらに、テスト担当者は、関係する要素間の一貫性を文書としても意味的にも確立できる。

ASPICE は、垂直トレーサビリティと水平トレーサビリティを区別する。

垂直トレーサビリティ — ステークホルダー要件をすべてのレベルを通してソフトウェアユニットに関係付けることを必要とする。こうすることで、開発のすべてのレベルの関係付けにより、関連する作業成果物間の一貫性を確立できる。

水平トレーサビリティ — 開発の作業成果と対応するテスト仕様およびテスト結果との間のトレーサビリティと一貫性を必要とする。

さらに、基本プラクティス SUP.10 の BP4 は、変更依頼と変更依頼により影響を受ける作業成果物との間の双方向トレーサビリティを要求する。変更依頼は欠陥により開始され、変更依頼と対応する欠陥レポートとの間の双方向トレーサビリティが必要である。関係付けが大量に発生することがあるため、一貫性がもたらされるツールチェーンを使用することが必要である。これにより、テスト担当者は、効率的に依存関係を持たせること、および管理を行うことができる。

[11] 以降では、用語「トレーサビリティ」は双方向トレーサビリティを常に意味する。

2.2 ISO 26262

2.2.1 機能安全と安全文化

2.2.1.1 E/E システムにおける機能安全の目的

組込みシステムでは、機能的にも技術的にも複雑度が定常的に増加している。同時に、ソフトウェアベースの強力な E/E システムにより、車両の自動運転機能など複雑な新機能を実現している。

複雑度の高まりにより、開発時に誤った行動が発生するリスクが増加している。結果として、認識されないフォールト状態がシステムに侵入する可能性がある。生命および身体に対する潜在的なリスクが内在するシステムでは、安全に対する責任を持つエンジニアやマネージャーが潜在的なリスクを分析する必要がある。発生しうるリスクが存在する場合、適切な対策を講じて、予見される影響度を受け入れ可能なリスクレベルに軽減する必要がある。

こういった分析の実行手法は、機能安全の標準に概説されている。基盤となる標準は、IEC 61508 である。国際標準化機構（ISO）は、この標準から ISO 26262 を策定しており、2018 年からは第二版が提供されている

機能安全は、E/E システムの機能不全の振る舞いにより引き起こされるハザードが原因となる不合理なリスクの不在を確実にすることである。この意味において、機能安全という用語は、情報安全、製品安全、作業安全などの安全に関する用語と区別される。

就労環境の安全とサイバーセキュリティは、ISO 26262 の範囲外である。サイバーセキュリティの欠落は機能安全を危険にさらす可能性があり、サイバーセキュリティは製品安全に寄与する。

2.2.1.2 安全文化におけるテスト担当者の役割

ISO 26262 に準拠した製品開発において、自身の組織のプロセスを監視するだけでは不十分である。全関係者がプロセスのみには依存しないアプローチをとる必要がある。全関係者が開発プロセスと最終製品の安全に対して、自身が与える影響を理解する必要がある。関係者には、外部パートナーやサプライヤーも含まれる。

関係者は自分の行動が他のプロセスから独立して生じるわけではないことを理解する必要がある。開発の各ステップは、機能安全に関連する要求への適合と実装に非常に関与している。この責務は、製品の販売開始により終わるわけではない。安全ライフサイクルが終了するまで継続する。

テスト担当者は、ソフトウェア開発ライフサイクルのフェーズに責任を持って参加し、製品開発の全体的な状況を継続して把握しながら業務に従事することによって、安全文化に貢献する。

2.2.2 安全ライフサイクルにおけるテスト担当者の役割

安全ライフサイクルは、製品開発における安全指向の活動を定義する。安全ライフサイクルは、初期の製品アイデア起案および発生しうるリスクの調査を開始することから始まる。安全要求の仕様を決定した後に、具体的な製品への実装を行う。安全ライフサイクルは、製品が寿命を迎え廃棄されることで終了する。

ISO 26262 では、安全ライフサイクルを以下のフェーズで構成する。

- 第 1 フェーズ：コンセプトフェーズ
- 第 2 フェーズ：製品開発。このフェーズは「生産フェーズへのリリース」で終了する
- 第 3 フェーズ：生産および運用、保守、廃棄

サプライヤーのテスト担当者は、主に最初の 2 つのフェーズに従事する。第 3 フェーズでの製品に対する変更は、その影響度に応じて、第 1 フェーズまたは第 2 フェーズの再実施が発生する。このため、

テスト担当者は変更作業にも関与する。安全関連の要求に基づき、テスト担当者は、製品開発内での検証とこれらの要求の妥当性確認を行うためにテスト技法を選択し、テストケースを設計する。その後、テスト担当者は製品開発の関連するサブフェーズでこれらのタスクを実行する。

テスト計画作業は、通常、コンセプトフェーズで実施する。ただし、成果物の文書（テスト計画やテスト仕様など）に対する調整は、いずれのフェーズでも発生する可能性がある。テスト実行は、ほとんどの場合、製品開発の個々のサブフェーズの移行時に発生する。例えば、実装とソフトウェア統合との間、さらにはハードウェア-ソフトウェア統合への移行時に発生する。また、テスト担当者は、第3フェーズへの移行時にも、テスト活動の点において大きな役割を担う。

2.2.3 標準の構成と特定のパートにおけるテスト

2.2.3.1 標準の設計と構成[情報提供]

ISO 26262 は、12 のパートで構成される。

- 用語集（パート 1）
- 機能安全の管理（パート 2）
- 安全ライフサイクルのフェーズ：
  - コンセプトフェーズ（パート 3）
  - システムレベル、ハードウェアレベル、およびソフトウェアレベルにおける製品開発（パート 4～6）
  - 生産および運用（パート 7）
- 支援プロセス（パート 8）
- ASIL 指向および安全指向の分析（パート 9）
- ISO 26262 のガイドライン（パート 10）
- ISO 26262 の半導体向けの適用ガイドライン（パート 11）
- ISO 26262 の二輪車向けの適用（パート 12）

パート 1 と 10、11 を除いて、各パートは共通の構成で書かれている。その内容の一部を以下に記載する。

- 概要説明
- 適用スコープ
- 参照される標準
- 標準に適合するための要求

これらの内容の後に、パートに応じて以下の個別のトピックが続く。説明の構成は各パートで同じである。実行する活動は、全パートを通じて同様の構成を使用して記載されている。

- 目的
- 全般的な情報
- 概要説明
- 前提要求
- 詳細なサポート情報
- 要求と推奨
- 作業成果物

2.2.3.2 テスト担当者に関連する ISO 26262 のパート

ソフトウェアテスト担当者にとって、ソフトウェア検証と（少なくとも部分的に）システム妥当性確認は、最も重要なものである。パート 1（用語）を除いて、他のいくつかのパートでも、着目すべき点がある。パート 4 と 6 では、ソフトウェア検証の推奨される方策について詳細な情報と要求を説明している。これらは、対応する検証方策の選択、設計、実装、および実行に適用される。

こうすることで、これらのパートは、システム（パート 4、安全妥当性確認を含む）とソフトウェアレベル（すなわちパート 6）のテストと検証の固有の側面に焦点を当てている。もしシステムやソフトウェアレベルの作業にハードウェアの観点が関係する場合は、パート 5 を参照するとよい。ハードウェアとソフトウェアに関連する観点は、ハードウェア-ソフトウェアインターフェースのスコープ内とみなされる（パート 4～6）。

パート 8 は特別な役割であり、全テストレベルにおける検証のプロセス固有の特性を説明している。文書化とツール認定など、標準化されたプロセスを確実にするために重要な支援プロセスの要求を説明している。

パート 12 においては、テスト担当者は新しく二輪車向けの二輪車用安全度水準（MSIL）のアセスメントに基づく手法テーブルを参照することができる。

2.2.4 テスト範囲へのクリティカリティ（クリティカル度合い）の影響

2.2.4.1 ASIL のクリティカリティレベル

ASIL は、機能安全の各種方策によって要求されるリスクを軽減するための評価尺度である。このような方策は、例えば、E/E システムの監視するための、または特別に定義された手法を実装するための独立した安全機能である。リスクのレベルが高くなるほど、より綿密な方策が必要になる。

プロジェクトの開始時に、専門家チームはプロジェクトのハザード分析とリスクアセスメントを行う。この分析で識別された各リスクに対して、専門家チームは標準で定義されている方法論の 1 つを使用して、ASIL を判定する。次のステップでは、安全目標と安全要求の草案を作成する。これらはベースとするリスクと同じ ASIL を使用する。

ISO 26262 は一番低い ASIL A から一番高い ASIL D まで、4 つのレベルを定義する。

ハザード分析とリスクアセスメントの結果で ASIL A より低い要求が導かれた場合、この標準の観点では、非安全関連とされる。これらの要求に対しては、機能安全関連の行動は不要であり、既存の品質マネジメントシステム（QMS）の要求に適合することで十分である。ASIL 関連の要求と区別するという実務的な理由で、それらの要求は「QM」と分類される。

2.2.4.2 テスト技法、テストタイプ、およびテスト範囲に対する ASIL の影響

判定された ASIL は、テスト担当者が実装するテスト範囲に直接影響する。ISO 26262 標準は、ASIL の特定のレベルに応じて異なる方策や方策のパッケージの実行を推奨する。この場合のルールは、より高い ASIL ではより広範でより詳細な方策が推奨されている。より低いレベルの ASIL では、指定されている方策の実行は、多くの場合、オプションである。

ISO 26262 は、推奨策のレベルとして、「推奨策なし」、「推奨される」、「強く推奨される」の 3 つを指定する。一般的な理解として、「強く推奨される」の場合は必ず適用されなければならず（MUST）、「推奨される」の場合は適用すべき（SHOULD）、または、適用しない理由の明示が求められる。「推奨策なし」の場合、標準は対応する方策に関して、使用/不使用のいずれの推奨策も提供しないが、方策自体の適用を阻むものではない。ただし、その実行は、ISO 26262 によって推奨されるか、強く推奨される方策を置き換えるものではない。

テスト担当者にとって標準は、ASIL に応じて機能安全に関連するシステムの特定のテスト設計技法とテストタイプを推奨することを意味する。テスト担当者が独自に判断できるのは、ASIL によって特定された標準のフレームワークの範囲に限られる。例えば、同値分割法と境界値分析の使用は、ASIL A では推奨されている。一方で、ASIL B 以上では、これらの技法の使用は強く推奨されている（2.2.5 節参照）

ASIL は製品全体の特性ではない。特定の安全目的と導かれる安全要求に結び付けられている。このため、同一の製品の中でも ASIL が異なる安全要求があるため、安全要求ごとにテスト工数が大きく異なる可能性がある。テスト担当者はテスト範囲を計画する際に、このことを考慮する必要がある。

2.2.5 ISO 26262 の文脈における CTFL®からのコンテンツの適用

ISO 26262 は、テスト担当者に対して、固有の推奨策を手法テーブルの形式で提供している。これらのテーブルは、パート 4、5、6、8 および 12 で提供されている。テストプロセスやテスト活動に関する機能安全固有の推奨策とは別に、テスト担当者が使用するテスト技法も含まれている。

この文脈では、標準は「手法」という用語を、適用可能なテスト技法やテスト活動すべてに関連して使用している。この点において、機能安全の用語は、ISTQB®の用語とは若干異なっている。テスト担当者にとって、ISO 26262 の以下の手法は、特に重要である。

- テスト設計技法（例：同値分割法、境界値分析）
- テスト実行の技法（例：システムまたはコンポーネントのシミュレーションまたはプロトタイプ）
- テストタイプ（例：性能テストなどの非機能性テスト、耐久テスト）
- テスト環境（例：ハードウェアインザループ、車両）
- 静的テスト技法（例：レビュー、静的解析）

手法テーブルは、各 ASIL レベルに対して標準で推奨される手法を定義する。

手法テーブルは、常に同じ構造で定義する。

表 1：手法テーブルの例

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

ASIL レベルに応じて、各手法に対して、その使用が推奨される（+）、または強く推奨される（++）が示されている。オプション（o）と記される手法については、標準では、その使用について推奨、非推奨のいずれも指定しない。

ISO 26262 では、同等の代替手法も文字記号を用いて手法テーブルに記載する（前記例の行 3a と 3b）。このような場合、テスト担当者は、関連する要求を ASIL に適合する方法でチェックできる適切な組み合わせを選択する必要がある。テスト担当者は組み合わせの選択理由を説明できなくてはならない。

代替のない手法（例の行 1 と 2）は例外なく適用する必要がある。テスト担当者は、該当する ASIL レベルで強く推奨される全手法を適用する必要がある。

例えば、ASIL C の要求の実証のために導かれる手法は、 table 1 を参照すると以下のようになる。

- 手法 x：強く推奨される：ISO 26262 に従って開発する場合、適用しなければならない（MUST）。
- 手法 y：推奨される：適用すべきである（SHOULD）、または、適用しない理由を提供する。
- 手法 z1 と z2：少なくとも z1 は ASIL C で最高レベルであるため、選択すべきである（SHOULD）。

ISO 26262 では、手法テーブルに記載されている手法以外の手法もテスト担当者は使用できる。その場合、テスト担当者は選択した手法の有用性と合目的性を説明する必要がある。

一般的に、ISO 26262 は特定のプロジェクトにおけるニーズに合わせて安全活動をカスタマイズすることを許容している（テーラーリング）。テスト関係の活動においても、テスト戦略とテストアプローチをテーラーリングすることを許容されていると言える。

2.3 AUTOSAR

AUTOSAR は「AUTomotive Open System ARchitecture」の頭字語で、アーキテクチャーと開発パートナーシップで構成される。AUTOSAR パートナーシップは 2003 年に創設され、主に自動車業界のメーカーとサプライヤーで構成される。パートナーシップの目的は、「車両環境におけるソフトウェアアーキテクチャー向けの自由に利用可能な標準を作成し確立する」ことである。現在、AUTOSAR は自動車の E/E システム向けの世界的に確立された標準となっている。したがって、テスト担当者は必然的に AUTOSAR の作業成果物に取り組むことになる。また、テスト担当者は、AUTOSAR のプロジェクト目的、AUTOSAR アーキテクチャーの基本的な設計、および自身の作業との接点を把握しておくことが重要である。

2.3.1 AUTOSAR のプロジェクト目的

AUTOSAR の以下のプロジェクト目的は、「標準内でのコラボレーション、実装での競争」の原則に基づいている。

- ソフトウェアの移行性（移植性）をサポート
- 異なる車両や派生したプラットフォームへの拡張性をサポート
- 異なる機能ドメインをサポート
- AUTOSAR ではないシステムとのデータ交換をサポート
- オープンアーキテクチャーの定義
- 信頼性の高いシステムの開発をサポート - 可用性、信頼性、安全性、完全性、メンテナンス性、使用性によって特徴付けられる
- さまざまなパートナー間のコラボレーションをサポート
- 車両向けの自動車開発標準および先端技術を適用可能にするサポート

2.3.2 AUTOSAR の全般的な構成[情報提供]

AUTOSAR に Classic と Adaptive という 2 つのアーキテクチャーの種類がある。簡潔さの維持のために、このシラバスでは AUTOSAR Classic にフォーカスする。AUTOSAR システムはいくつかの電子制御ユニット（ECU）から構成されることが多い。それぞれの ECU において、AUTOSAR Classic のソフトウェアアーキテクチャーは次の 3 つのレイヤーを含む。

- ハードウェアから独立した上層のアプリケーションレイヤー。このレイヤーのコンポーネントは AUTOSAR ソフトウェアコンポーネント（SW-C）と呼ばれる。
- 標準化された基盤ソフトウェア（BSW）を含むハードウェア指向の下層のレイヤー。
- AUTOSAR Runtime Environment（RTE）を含む中間抽象化レイヤー。ミドルウェアの一種として、このレイヤーは、電子制御ユニット内外のデータ交換を制御し、SW-C 間および SW-C と BSW 間のデータ交換を実装する。

自動車システムを開発するための AUTOSAR メソドロジーにおいて、自動車メーカー（OEM）とサプライヤーは arxml ファイルと呼ばれる AUTOSAR テンプレートを介して、記述ファイルに関する情報を交換する[9][16]。

- 「ECU Configuration Description」には、電子制御ユニットの SW-C を統合するためのデータを記述する。
- 「System Configuration Description」には、1 つの車両内のすべての制御ユニットを統合するためのデータを記述する。
- 「ECU extract 」には、単一の電子制御ユニットのための「System Configuration Description」のデータを記述する。

2.3.3 AUTOSAR のテスト担当者の作業への影響

AUTOSAR は特に次のテストレベル[12]でテスト担当者の作業に影響を及ぼす。

- 仮想環境での SW-C およびソフトウェア統合テスト（例：ソフトウェアインザループ（SiL））。シミュレーションされた BSW および RTE を利用することで、テスト担当者はアプリケーションの SW-C を早い段階でテストできる[17][18]。
- 実制御ユニットでのソフトウェアコンポーネントテストとソフトウェア統合テストでは、テスト担当者は RTE 上の通信にアクセスする。これにより、SW-C の実行時の振る舞いを観測および制御できる[19]。
- AUTOSAR 受け入れテストは、ソフトウェアシステムのテストであり、AUTOSAR の機能性が通信およびアプリケーションのレベルで標準と適合していることを確認する。AUTOSAR 受け入れテストはオプションである[26][36]。
- システム統合テストでは、複数の ECU の統合、例えば複数の ECU に分散して実装されている機能性のテスト。テスト担当者は、未実装の機能をシミュレーションすることで、システムの振る舞いを早い段階でテストできる。

[12] テストレベル。2.4.2 も参照されたい。

2.4 ASPICE と ISO 26262、CTFL®の比較

2.4.1 ASPICE と ISO 26262 の目的

複数の標準が、プロダクト開発の要件を提案している。一般的に、これらはそれぞれ開発の異なる側面に重点を置いている。ここでは、ISO 26262 と ASPICE について、それらの目的の違いを説明する。

ISO 26262 は、ソフトウェアとハードウェアの故障のリスクを回避することを目的として、適切な要件とプロセスを提示している。E/E システムの開発については、テスト担当者が使用するテストプロセスと手法に対する要件を定義する。これらは、アイテムの ASIL レベルに依存する。

ASPICE はプロダクト開発プロセスの能力を判定するためのアセスメントのフレームワークを提供する。このため、ASPICE は、これらのプロセスを評価するための基準を定義する。ISO 26262 とは異なり、これらは重要度やプロダクトの ASIL レベルに依存しない。

2.4.2 ASPICE と ISO 26262、CTFL®のテストレベルの比較

ISO 26262 と ASPICE の両方がテストレベルを定義している。しかし、CTFL®のテストレベルと完全に一貫性が保たれているわけではない。したがって、複数のテスト担当者が効率的かつ効果的に開発チームの中で協調するためには、すべてのテストレベルについて共通の理解を持つ必要がある。

ASPICE で使用されている用語「システム」と、ISO 26262 で使用されている用語「システム」と「アイテム」は、ハードウェアとソフトウェアのコンポーネントで構成されるプロダクトを意味する。しかし、CTFL®シラバス [ISTQB_FL_SYL] では、「ソフトウェア」を意味する。したがって、ISTQB®[20]のテストレベルは、ISO 26262 と ASPICE のテストレベルと以下のように対応付けられる。なお、ISO 26262 の列においてカッコ内の数字は ISO 26262 標準のパートを示し、ASPICE の列においてカッコ内はそれぞれのプロセス ID を示す。

表 2：テストレベルの対応付け

ISTQB®	ISO 26262	ASPICE 4.0
受け入れテスト	安全妥当性確認（4～8）[13]	妥当性確認（VAL.1）[14]
システムオブシステムズテスト[15]	システムとアイテムの統合とテスト（4～7）[16]	システム検証（SYS.5）
システム統合テスト		システム統合と統合検証（SYS.4）
システムテスト	ソフトウェア安全要件の検証（6～11） ソフトウェアの統合とテスト（6～10）	ソフトウェア検証（SWE.6）
コンポーネント統合テスト[17]		ソフトウェアコンポーネント統合と統合検証（SWE.5）

[13] 安全妥当性確認は、ISTQB の受け入れテストの一部にしか対応しない。
[14] 妥当性確認（VAL.1）は ISTQB における受け入れテストのみカバーしている
[15] 複数の異種分散システムのテスト**エラー! 参照元が見つかりません。エラー! 参照元が見つかりません。**
[16] 「アイテムの統合とテスト」は、要素のハードウェアとソフトウェアの統合とテスト、アイテムに属するすべての要素の統合とテスト、および車両内の他のアイテムとの接続に関するアイテムの統合とテストの 3 つのフェーズを含む。
[17] ASPICE (SWE.5) と ISO 26262 (6-10)は追加で CTFL におけるコンポーネントテスト（「コンポーネント」という意味で）をカバーする

コンポーネントテスト[18]	ソフトウェアユニットテスト（6～9）	ソフトウェアユニット検証（SWE.4）

ISTQB® CTFL® Core シラバス[ISTQB_FL_SYL]によると、テスト技法はほとんどの場合、テストレベルに関わらず適用できる。ASPICE も、基本的には、いずれのテスト技法もテストレベルに割り当てていない。したがって、両方とも技法の選択はテスト担当者に任せられている。ISO 26262 では、各テストレベルに対して個別の手法テーブルが存在する（2.2.4 節および 2.2.5 節参照）。これらのテーブルは、ASIL レベルに応じて使用すべきテスト技法に関する推奨策をテスト担当者に提供している。

[18] ASPICE (SWE.4) と ISO 26262 (6-9)は追加で CTFL におけるコンポーネントテスト（「ユニット」という意味で）をカバーする

3　仮想環境でのテスト[160 分]

用語

環境モデル、ハードウェアインザループ、モデルインザループ、ソフトウェアインザループ

ドメイン特有の用語

オープンループシステム、クローズドループシステム

学習の目的

AuT-3.1.1　（K1）自動車開発におけるテスト環境に対するモチベーションを想起する。

AuT-3.1.2　（K1）自動車固有のテスト環境の一般的な構成要素を想起する。

AuT-3.1.3　（K2）自動車固有のテスト環境の一般的な構成要素を想起する。

AuT-3.1.4　（K1）ECU にとって重要な機能、データベース、およびプロトコルを想起する。

AuT-3.2.1.1　（K1）MiL テスト環境の構成を想起する。

AuT-3.2.1.2　（K2）MiL テスト環境の適用領域と境界条件を説明する。

AuT-3.2.2.1　（K1）SiL テスト環境の構成を想起する。

AuT-3.2.2.2　（K1）SiL テスト環境の適用領域と境界条件を説明する。

AuT-3.2.3.1　（K1）HiL テスト環境の構成を想起する。

AuT-3.2.3.2　（K2）HiL テスト環境の適用領域と境界条件を説明する。

AuT-3.2.4.1　（K2）XiL テスト環境を使用してテストすることの長所と短所について判断基準を用いて説明する。

AuT-3.2.4.2　（K3）1 つ以上のテスト環境に対し基準を適用して、特定のテスト範囲を割り当てる。

AuT-3.2.4.3　（K1）V 字モデルにおける XiL テスト環境の概要を把握する。

3.1　一般的なテスト環境

3.1.1　自動車開発におけるテスト環境に対するモチベーション

テスト担当者は固有の課題に対処する必要がある。可能な限り早期にテストを開始して、開発プロセスの早い段階で欠陥を見つける必要がある。一方で、実環境を使用してシステムをテストし、完成したプロダクトで出現する可能性がある欠陥を見つける必要がある。テスト担当者は、異なる開発フェーズに適したテスト環境を使用して、この矛盾を解決できる。このようにすることでテスト担当者は、開発完了した、または製造された ECU が利用可能になる前に、自身のテストタスクを実装および実行できる。テスト担当者はさまざまなテスト環境を使用することで、ワイヤーハーネスの短絡や開放、ネットワーク通信の過負荷など、実際の車両で再現が困難な状況をシミュレーションできる。

3.1.2　テスト環境の一般的な構成要素

テスト担当者はテストを実行するために、不足している構成要素をシミュレーションできるテスト環境を必要とする。このような環境を使用することで、テスト対象の入力を制御しテスト結果を観測できる（これらはそれぞれ「制御ポイント（PoC：point of control）」および「観測ポイント（PoO：

point of observation）」とも呼ばれる）。ISO/IEC/IEEE 29119-1 によれば、テスト環境は以下の構成要素から成り立っている。

- テスト環境のハードウェア（例えば、制御コンピューター、必要に応じてリアルタイム対応コンピューター、テストベンチ、開発キットなど）
- テスト環境のソフトウェア（例えば、オペレーティングシステム、シミュレーションソフトウェア、環境モデル）
- 通信設備（例えば、ネットワークへのアクセス、データロガー）
- ツール（例えば、オシロスコープ、計測器）
- 実験室（例えば、電磁放射とノイズからの隔離）

テスト環境の重要な構成要素は、環境モデルである。環境モデルは仮想テスト環境で重要な要素である。モデルは、燃焼機関、トランスミッション、車両センサー、ECU、さらには運転手や道路状況など、実世界のある側面を表す。テスト環境は、さまざまなアクセスポイントを提供する。テスト担当者はこれらのアクセスポイントを使用して、テスト対象の測定と観測、必要な場合は制御を行う。

3.1.3 クローズドループとオープンループの違い

テスト環境は、テスト対象のデバイスの入力インターフェースを制御し、出力インターフェースを介してその出力をモニターするために使用される。その後、出力インターフェースの振る舞いを分析する。観測された振る舞いが期待結果と一致した場合、成功とみなす。

一般的に、制御システムには、クローズドループとオープンループの 2 つのタイプがある。 この 2 つのタイプの違いは、電子制御ユニットがその環境に対して反応する方法である。ECU のテスト環境への反応の仕方に起因する。このことが、仮想テスト環境に対する異なるシミュレーション要件を生む。

3.1.3.1 クローズドループシステム

クローズドループシステム（インザループを含む）の刺激はテスト対象の出力を考慮する。環境モデルは出力を収集し、テスト対象の入力に直接的または間接的にフィードバックする。これによって、テスト環境内に制御ループが生成される。

コントローラーのテストでは、クローズドループシステムがよく使われる。テスト担当者はクローズドループシステムを使用することで、モーター制御やギア制御などの複雑な機能、さらにはアンチロックブレーキシステム（ABS®）や車両運動制御（VDC®）などの運転支援システムをテストできる。

3.1.3.2 オープンループシステム

オープンループシステムでは、システムの出力は入力に関係しない。システムはオープンであり、出力から入力へのフィードバックがない。この場合、テスト対象の入力はテスト担当者がテスト手順で直接定義する。

オープンループシステムおよびクローズドループシステムのどちらを適用するかは、テスト対象の振る舞いに強く依存する。オープンループシステムがフィードバックメカニズムを持っていない一方で、クローズドループシステムは出力が入力に影響したり、フィードバックを作ったりする。テスト対象がリアクティブな振る舞いをしたり、状態機械に則っとった振る舞いをしたりする場合は、オープンループシステムが適切である。インテリアやボディーエレクトロニクス（例えば、ライトやスイッチ）において、オープンループシステムは多くの事例がある。

3.1.4 ECU にとって重要なインターフェース、データベース、およびプロトコル

自動車環境の ECU は組込みシステムであり、ハードウェアとソフトウェアで構成される。ECU はさまざまなアナログおよびデジタルの入力を受け取り、電圧、電流、および温度の形式で環境データを絶えず収集する。さらに、通信バスシステムがセンサーや他の ECU からのさらなる情報を ECU に提

供する。通信バスシステムはその情報を自身で収集および処理するか、または生成する。テスト対象はメモリー内のデータを管理し、処理した結果を出力する。また、生成された出力はアナログまたはデジタルの出力ピン、バスシステム、または診断インターフェースを介して転送される。

データベースは、制御ユニットの入力信号および出力信号を定義する。これらのデータには、信号の説明、単位、および変換式も含まれる。

通信プロトコルは、対応する物理インターフェースを介するデータ交換を定義する。これらのプロトコルは、どの電圧またはビットシーケンスが信号のどの値を表すかを定義する。

データベースと通信プロトコルの選択は、ECU の機能に基づいて決定する。例えば、制御ユニットの診断機能にアクセスするテスト担当者は、使用されているデータベース（例：API 仕様（ Association for Standardization of Automation and Measuring Systems （ASAM） MCD-3 D）やサービス指向の車両診断（ASAM SOVD）、通信プロトコル（例えば、ISO 14229 の unified diagnostic services や AUTOSAR 仕様（SOME/IP））） に関する情報を必要とする。自動車固有のその他のデータベース例が、ASAM や AUTOSAR 標準に定義されている。

3.2 XiL テスト環境でのテスト

自動車業界では、以下のタイプの XiL テスト環境を使用する。

- モデルインザループ（MiL）。
- ソフトウェアインザループ（SiL）。
- プロセッサインザループ[19]（PiL）。
- ハードウェアインザループ（HiL）。
- ビークルインザループ[20]（ViL）。

テスト担当者はテスト環境（すなわち MiL、SiL、および HiL）に精通し、それらを理解する必要がある。以下の段落では、さまざまなテスト環境の構造と適用領域について詳細に説明する。ここでの XiL とは、さまざまなテスト環境の総称である。

3.2.1 モデルインザループ（MiL）

3.2.1.1 MiL テスト環境の構成

MiL テスト環境では、テスト対象はモデルとして利用できる。このモデルは実行可能であるが、特定のハードウェア用にはコンパイルされない。このようなモデルは、開発担当者が特別なモデリングツールを使用してモデル化する。テスト担当者はこれらのモデルを実行およびテストするために、テスト環境を必要とする。多くの場合、これはテスト対象自身と同じ開発環境で実装される。このテスト環境には、環境モデルを追加して含めることができる。テスト担当者はアクセスポイントを介して、テスト対象を制御および観測できる。アクセスポイントはテスト対象のモデルの中だけでなく、環境モデルの中にも任意に配置できる。テスト対象のモデルは環境モデルに接続でき、クローズドループシステムとして容易に実装および使用できる。

3.2.1.2 MiL テスト環境の適用領域と境界条件

テスト担当者は MiL テスト環境を使用することにより、機能的なシステム設計をテストできる。開発（例えば一般的な V 字モデル）においては、テスト担当者は単一のコンポーネントからシステム全体までテストすることもできる。テスト担当者はテストを実行するために、コンピューターや対応するシミュレーションソフトウェア（環境モデルを含む）を必要とする。環境モデルは、テスト対象の機能のスコープが拡大するにつれて複雑さが増してくる。現実と環境要因の対応付けは非常に時間がかかる。モデルの実行時間も非線形に増加する。このため、MiL テスト環境を実装することは、開発の特定のフェーズからは価値がなくなる。[21]

テスト担当者は MiL テスト環境を使用することにより、開発の初期フェーズ（すなわち V 字モデルの左側）からモデルの機能的な適合性をテストできる。ただし、環境モデルを使用してバス機能や診断機能、または物理的な振る舞い（ケーブルの断線や短絡）をシミュレーションすることは一般的ではない。これらのタスクは、別のテスト環境を使用することにより、より容易に、より少ないコストで実行できる。

MiL テスト環境では、テスト環境を実時間で実行できない。全コンポーネントがモデルとして利用できれば、テストはシミュレーションとして実行できる。システムが複雑になるにつれて、必要な全情報を提供するためにさらに多くの実行時間またはコンピューター処理能力が必要になる。小規模システムのシミュレーションの実行時間は、実際の実行時間よりも短くなりうる。大きな長所として、テスト担当者はシミュレーションを任意のタイミングで一時停止し、詳細な分析と評価を実施できる。

[19] このテスト環境は本シラバスで説明されていない。情報提供のみである。
[20] このテスト環境は本シラバスで説明されていない。情報提供のみである。
[21] これは、他のすべての XiL 環境にも当てはまる。

3.2.2 ソフトウェアインザループ（SiL）

3.2.2.1 SiL テスト環境の構成

テスト対象は特定の SiL テスト環境向けにコンパイルする。これは、ソースコードを特定のコンピューターアーキテクチャー向けにコンパイルすることを意味する。このマシンコードはバイナリデータであり、特定のテスト環境で使用できる。テスト環境では信号にアクセスできるようにするためのラッパーが必要である。ラッパーは、テスト対象がラッパーを経由せず環境と直接コミュニケーションする場合にアクセスできない入力へのアクセスを提供する追加ソフトウェアである。したがって、テスト担当者は、ソフトウェアの入出力を制御および観測できる。ラッパーはテスト対象に対するアクセスポイントを定義するが、機能的な処理を実行することはない。

シミュレーションでは、環境モデルが必要である。テスト対象はラッパーを介することで、テスト環境に接続できる。テストは、特別なハードウェアを使用せずコンピューター上で実行する。テスト担当者は、テスト対象をラップし、テスト環境へのアクセスポイントを提供するためのソフトウェアツールを必要とする。

3.2.2.2 SiL テスト環境の適用領域と境界条件

開発担当者がモデルに基づいてソースコードを生成すると、ソフトウェアの実際の振る舞いは、期待される振る舞いとは異なることがある。この原因の 1 つは、浮動小数点が使われるモデルと、固定小数点が使われるコンパイル後のソフトウェアコードではデータ型が異なることである。別の原因として、メモリー領域が異なることが挙げられる。これらの期待される振る舞いからの逸脱は、SiL テスト環境以降でのみテストできる。テスト担当者はバックツーバックテスト（4.2.2節も参照）のようなテストアプローチを使用して、振る舞いを比較できる。

テスト担当者は、MiL テスト環境の場合と同様に、SiL テスト環境においては仮想時間でテストを実行する。計算技法と環境モデルの複雑度に応じて、このシミュレーションにかかる時間は、実時間よりも短くなったり長くなったりする。テスト担当者は任意のタイミングでテスト実行を一時停止し、詳細な分析と評価を実行できる。機能適合性テスト、インターフェーステスト、リグレッションテストは、SiL テスト環境で評価できるごく一般的なテストタイプである。一方、性能効率性テストおよび信頼性テストは一般的ではない。これらの品質特性は、対象となるハードウェアによって大きく影響を受ける。

3.2.3 ハードウェアインザループ（HiL）

3.2.3.1 HiL テスト環境の構成

テスト対象がプロトタイプとして利用できる場合、または開発が既に完了している場合、テスト担当者は HiL テスト環境を使用してテストを実行できる。HiL テスト環境は、一般的に以下の要素で構成される。

- さまざまな電圧を供給できる電源
- 環境モデルを実行するためのリアルタイム対応コンピューター
- 環境モデルに実装されていない複数の実構成要素
- 信号のタイプと信号の振幅の処理ユニット
- ケーブル断線や短絡をシミュレーションするためのフォールトインジェクションユニット（FIU）（4.2.3 節参照）
- ケーブルハーネスの追加アクセスインターフェースとしてのブレークアウトボックス
- 存在しないバス接続要素をシミュレーションするためのその他のバス

3.2.3.2 HiL テスト環境の適用領域と境界条件

HiL テスト環境にはアクセスポイントが広範に存在する。テスト担当者は、テスト対象に対して誤ったアクセスポイントを使用するとテスト結果が無意味なものになることに注意する必要がある。HiL テスト環境におけるさまざまなアクセスポイントとそれらの接続方法を理解することで、効果的なテストの実装、実行、および評価が可能になる。

HiL テスト環境は構成要素が多いため、これまでに説明している MiL や SiL のテスト環境よりも複雑である。テスト担当者はテストタスクを実施するにあたり、この複雑性を克服しなければならない。HiL テスト環境は、コンポーネントテスト、統合テスト、およびシステムテストで使用できる。さまざまな目的の中の 1 つは、ソフトウェアおよびハードウェアの機能性/非機能性欠陥を見つけることである。

HiL テスト環境を使用することで、さまざまなテストレベルにおいてテスト対象を分析できる。テスト対象が単一の ECU の場合、コンポーネント[22]HiL と呼ぶ。テスト対象が複数の ECU の組み合わせである場合、システム HiL と呼ぶ。テスト担当者はコンポーネント HiL を使用して、制御ユニットの機能をテストする。システム HiL では、ECU 間のデータ交換のテストとシステム全体のシステムテストに重点を置く。

HiL テスト環境のシミュレーションは、前述のテスト環境（MiL および SiL）と異なり、常に実時間で実行する。これは、ソフトウェアが実際のハードウェアで実行しているためである。本テスト環境では、テストの一時停止や強制終了を行うことはできない。したがって、本テスト環境では、事前定義された時間内にすべての関連する信号を収集して処理することができるリアルタイム対応コンピューターを使用する。

3.2.4 XiL テスト環境の比較

3.2.4.1 XiL テスト環境でテストすることの長所と短所

テスト担当者は、さまざまなテスト環境の属性の違いを理解する必要がある。こうすることで、各環境でテストすることの長所と短所を理解し評価できる。判断基準を表 3 に示す。

表 3：MiL、SiL、および HiL テスト環境の判断基準とそれらの影響

判断基準	HiL テスト環境	影響	MiL テスト環境	SiL テスト環境	HiL テスト環境
実環境との類似性	現実のシミュレーション、多くの特性の抽象化、構造とロジックに重点	低	+	o	o
	コンパイルされた実際のソフトウェアを実行可能（ハードウェア不使用）	低～中	o	+	o
	統合されたシステムを実行可能	高	o	o	+
デバッグにかかる時間と工数	テストアイテムのモデルで見つかった欠陥（モデル修正）	低	+	o	o
	プログラム済みソフトウェアで見つかった欠陥（ソフトウェア修正）	中	o	+	o
	システムレベルで見つかった欠陥（システム修正）	高	o	o	+

[22] 本書で「コンポーネント」という用語は、E/E システムの文脈での電子制御ユニット（ECU）を表す。

実装とメンテナンスにかかる工数	環境モデルの作成	低	+	o	o
	環境モデルとラッパーの作成	中	o	+	o
	環境モデルの作成とハードウェアコンポーネントの結線	高	o	o	+
テストの準備にかかる工数	環境は迅速にセットアップ可能	低	+	o	o
	環境は迅速にセットアップ可能	中	o	+	o
	テスト環境の設計、実装、評価に多大な工数が必要	高	o	o	+
テスト対象の完成度	システムモデルがシミュレーションされている	低	+	o	o
	対象ソフトウェアを使用して初期機能がテストされている	中	o	+	o
	1 つ以上の実行可能な電子制御ユニットまたは部分的なシステムが極力全体的にテストされている	高	o	o	+
テストベース（仕様）に必要な完全性	完全な仕様ではないにせよ、それに基づきモデルをテストされている	中	+	o	o
	ソフトウェアレベルに相当する情報が利用できる必要がある（詳細なコンポーネント仕様）	中～高	o	+	o
	要件をシステムレベルでテストできる（完全なシステム仕様）	高	o	o	+
テスト対象へのアクセス	モデル内の全信号の観測と制御が可能	高	+	o	o
	観測と制御が可能なのはラッパーで利用可能な信号のみ	中	o	+	o
	観測と制御が可能なのはハードウェアまたは通信プロトコルで利用可能な信号のみ	低	o	o	+

3.2.4.2　1 つ以上のテスト環境へのテストの割り当て基準

次の表は、テスト目的を詳細に説明し、適切なテスト環境へ割り当てている。

表 4：MiL、SiL、および HiL テスト環境のテストタイプの比較

テスト目的	例を用いた説明	MiL	SiL	HiL
顧客要件のテスト	要求された機能が正確に提供されていることの確認。入力の正しい処理、入力への正しい応答、正しいデータ出力を含む。	o	o	+

故障検出と対処のメカニズムのテスト	• ランダムハードウェア故障の検出と対処 • ソフトウェア故障の検出と対処 •故障が検出された後の安全状態への遷移（例：システムの停止）	+	+	+
コンフィギュレーションデータへの応答のテスト	テスト対象の振る舞いに対するコンフィギュレーションデータの影響度の確認	o	+	+
診断機能のテスト	必要な診断機能が正しく提供されていることの確認。故障検出/確定/クリア、メモリーへの故障情報保存（例：自己診断または整備場での診断）など	-	+	+
インターフェースでの相互作用のテスト	テスト対象の内部/外部インターフェースの確認	o	+	+
使用性の確認	テスト対象は要求通りかつユーザーの使用性要求通りに使用できる。	-	o	+
キー：+ 推奨、o 可能、 -不可能				

本表が示していることは、それぞれのテスト環境はあるテスト目的に向いていることである。このアプローチの多様性は、特に故障検出と処理のメカニズムをテストする際に明らかになる。「シフトレフト」の原則に従うと、ほとんどの場合、基本的な要件と設計の欠陥は、テストを介して早期の段階で既に検出されている。したがって、一般的に、MiL は全般的な設計欠陥の検出、SiL は技術的なソフトウェア欠陥の検出、HiL は技術的なハードウェア/ソフトウェア欠陥の検出を目的として使用する。さらに、安定性と信頼性、性能効率性、および使用性の証跡とは別に、全テストタイプは、テスト対象の機能的合目的性に重点を置くことに注意する必要がある。

テスト戦略では、テストマネージャーはテストのスコープをさまざまな異なるテスト環境に割り当てる。テスト担当者は効率的で適切なテストが XiL 環境を通じて行われていることを確認し続けるべきである。もし適用可能な場合、テスト目的に合致しているか、重複を避けられているか、それぞれのテストが最適な環境で実行されているかを開発フェーズにおいて確認するのがよい。この協力がリソースの競合を回避し、カバレッジの一貫性を担保し、不適切なテストレベルでテストされるというリスクを軽減することにつながる。コミュニケーションの促進、ツールやフレームワークの共有を通じて、チームはテスト戦略の最適化、シームレスな進捗の維持をテストのライフサイクルを通して実現できる。テストマネージャーは表 3 と 4 の基準を組み合わせて、最適なテスト環境を選択できる。

3.2.4.3 V 字モデルでの XiL テスト環境の分類

技術的なシステム設計は、V 字モデルの左側に位置する。テスト担当者はこの設計を MiL テスト環境によりテストできる。テスト対象と MiL テスト環境をさらに開発すると、テスト担当者は、このテスト環境でコンポーネントテストと統合テストも実行できる。

テストアイテムの単一コンポーネントのプログラムとコンパイルが終了している場合、テスト担当者は SiL テスト環境を使用できる。SiL テスト環境のテストは、通常、コンポーネントテストと統合テストである。これらは V 字モデルの右側に位置する。

システムテストを行う場合、テスト対象の特定の機能が全体的に開発済みである必要がある。テスト担当者は HiL テスト環境を使用してシステムテストを実行できる。

適切なテスト環境をテストレベルに割り当てることで、以下の 3 つの観点に従ってテストプロセス全体を最適化できる。

プロダクトリスクの最小化

- テストレベル固有の故障タイプの検出（例：HiL 環境内でのシステムレベルでの性能効率性テスト）

テストコストの最小化

- 全テストレベルを、適切なテストタイプに割り当てることが必要である
- より早く、より安く、仮想的なテストレベルへのテストの移行

標準への準拠

- ISO 26262 標準の手法テーブルで、テスト環境は ASIL に基づいて推奨される。

4 静的テストおよび動的テスト [230 分]

用語

バックツーバックテスト、フォールトインジェクション、 要件ベースドテスト

学習の目的

静的テスト

AuT-4.1.1 （K2）MISRA-C ガイドラインの目的と要件について例を用いて説明する。

AuT-4.1.2 （K3）ISO/IEC/IEEE 29148 が定義するテスト担当者に関係する品質特性を用いて要件をレビューする。

動的テスト

AuT-4.2.1 （K3）MC/DC テストカバレッジを達成するためのテストケースを作成する。

AuT-4.2.2 （K2）バックツーバックテストについて例を用いて説明する。

AuT-4.2.3 （K2）フォールトインジェクションテストについて例を用いて説明する。

AuT-4.2.4 （K1）要件ベースドテストの原則を想起する。

AuT-4.2.5 （K3）適切かつ必須のテスト技法とテストアプローチを選択する際にコンテキストに依存した基準を適用する。

4.1 静的テスト

イントロダクション

静的テストでは、ソフトウェア開発の作業成果物を実行することなく、それらをテストする。このテストには、レビュー担当者による評価と、ツールによる静的解析が含まれる。

4.1.1 MISRA-C ガイドライン

開発者がコーディングガイドラインに則っとってプログラミングすることは、現在最も優れた技術的プラクティスの一環である。ISO 26262 標準でも、安全関連ソフトウェア[23]向けにコーディングガイドラインの準拠を推奨している。コーディングガイドラインは、欠陥を引き起こす可能性のある不正がソフトウェアに混入するのを防止するのに役立つ。また、開発担当者がソフトウェアの保守性と移植性の改善にも役立つ。

MISRA-C ガイドラインは、プログラミング言語 C 向けのコーディングガイドラインを提供し、自動車のソフトウェアプロジェクトで広く使用されている。このガイドラインでは、ルールと指針という 2 つのタイプのガイドラインを定義している。

- ルール：静的解析ツールを使用して検証できるものである。例えば、ソースコードは入れ子になったコメントを含まない、など。
- 指針：静的解析ツールでは完全には検証できないものである。指針は、ソフトウェアの外部にある開発プロセスや文書の詳細を対象にしている。例えば、開発者は実装した振る舞いを十分に文書化する必要がある、などである。

各ガイドラインは、以下の 3 つの義務レベルの 1 つに分類される。

[23] [ISO 26262：2018]パート 6-5.4.3 も参照されたい。

- 「推奨」ガイドライン：従うことに伴う労力が合理的である場合、従う必要がある。
- 「必須」ガイドライン：従う必要がある。従わない場合、正式な逸脱手続き（例えば品質マネジメント）が必要となる。
- 「義務」ガイドライン：従う必要がある。例外は認められない。

組織は個別にルールまたは指針の要件を強化できるが、緩和することはできない。

4.1.2 要件レビューのための品質特性

仕様は、開発とテストの土台である。このため、仕様に欠陥が存在すると、対処するための活動に多大なコストや時間がかかる。このことは、受け入れテストなどの開発フェーズ終盤や運用時にのみ欠陥が検出される場合に顕著である。レビューは、仕様の欠陥を早期に検出するのに効果的な手段の 1 つであり、結果として欠陥を早期かつ低コストで修正するのに役立つ。

テスト担当者はテスト分析時に、テスト対象の基となる仕様をチェックする必要がある。こうすることで、特に仕様がテストベースとして合目的性を満たしているかどうかをチェックできる。テスト担当者は仕様のレビュー時に品質特性を利用することで焦点を絞り、可能な限り多くの欠陥を検出できる。ISO 29148：2018 は、個々の要件および要件の集合に対する品質特性を定義している。

テスト担当者に関連する、ISO 29148：2018 の要件特性

- 検証可能である：各要件が正しく実現されていることを検証できる。
- 曖昧性がない：各要件のテスト条件は明確である。
- 一貫性がある：各要件はそれ自体に一貫性があり、他の要件とも一貫している。
- 完全である：各要件は解釈の余地を残さず、暗黙の知識や経験に基づく知識に依存しない。
- 単独である[24]：いずれの要件も、意味のある部分的な要件に分割できない。

テスト担当者は要件レビューの準備として、要件の品質特性からレビューチェックリストを導出することができる。これらのチェックリストアイテムは、抽象的すぎる品質基準を単に列挙するよりも具体的である必要がある。例えば、一貫性に関するチェックリストアイテムとしては「全要件を通じて用語が一貫して使用されているか？」が挙げられる。要件レビューにおいて、テスト担当者は自身の知る限りでチェックリストアイテムに回答しなければならない。

ISO 29148:2018 によると、先に述べた品質特性だけでなく、要件は実現可能、適切、正確、適合、理解可能、かつ必要であるという品質特性を満たすべきである。しかし、テスト担当者がこれらの品質特性を評価することは、ほとんどの場合困難である。

4.2 動的テスト

4.2.1 MC/DC テスト

ここで説明するテスト技法は、ホワイトボックステスト技法の一部である（詳細については、テクニカルテストアナリスト シラバス[ISTQB_ALTTA_SYL]も参照されたい）。テスト担当者は、テストケースをテスト対象の構造（例えば、ソースコード）から直接導き出す。

デシジョンテストではテストケースはすべての可能な判定結果、すなわち各判定の「真」と「偽」の結果を実行するように設計するすべての判定の取りうる出力（すなわち、各デシジョンの真と偽 ）。判定は 1 つ以上の条件を含み、各条件は真か偽と評価される。判定結果は、これらの条件の論理的な組み合わせによって決定される。

[24] 「単独である」は「不可分である」としても

判定が 1 つの条件だけで構成される場合、デシジョンテストと条件テストは同等のカバレッジを達成する。ただし、複数の条件を持つ判定については、より詳細なテスト技法が利用可能であり、以下のように記述される：同じカバレッジを得られる詳細なテスト技法が適用できる

- 条件テスト（表 5 のテスト技法 A）：テスト担当者は、個々の条件の真/偽の結果を網羅する目的でテストケースを設計する。テストデータを不適切に選択した場合（表 5 参照）、100%の条件カバレッジは達成できるが、すべての判定結果は網羅できない。以下の表における TC1 と TC2 ではいずれもテストケース TC1 と TC2 の両方において以下の表では、個々の条件 B1 および B2 は、真および偽の両方に対して実行できるが、両方のテストケースの判定結果は「偽」と判定される。
- 複合条件テスト（表 5 のテスト技法 B）：テスト担当者は、個々の条件のすべての組み合わせを網羅する目的でテストケースを設計する。すべての組み合わせをテストすると、すべての判定結果もテストしたことになる。
- 改良条件判定テスト（MC/DC）（表 5 のテスト技法 C）：これは複合条件テスト（B）に類似している。ただし、このテスト技法は、各条件（B1、B2）が独立して判定結果に影響する組み合わせのみを考慮する。テストケース TC4 では、B1 または B2 のいずれかを個別に「偽」から「真」に変更しても、判定結果は変化しない（「偽」のままである）。100%の MC/DC カバレッジはテストケース TC1、TC2、および TC3 により達成でき、TC4 を考慮する必要はない。

表 5 は、選択したテスト技法ごとに 100%カバレッジを達成するために必要なテストケースを例示している。

表 5：条件テスト（A）、複合条件テスト（B）、および改良条件判定テスト（MC/DC テスト）（C）の各テスト技法の比較

	個々の条件		次の式に対する判定結果 E = B1 AND B2	テスト技法		
テストケース	B1	B2		A	B	C
TC 1	B1=真	B2=偽	E=偽	X	X	X
TC 2	B1=偽	B2=真	E=偽	X	X	X
TC 3	B1=真	B2=真	E=真		X	X
TC 4	B1=偽	B2=偽	E=偽		X	

この例は、テスト技法の限界を示している。条件テスト（A）の場合、100%の条件カバレッジは達成できるが、テスト担当者は判定結果が *1* つだけであるというリスクを負う。より適切なテストケースの選択（TC3 と TC4 を選択）として、TC 3 と TC 4 を使用することで、この状況を改善できる。

テスト担当者は複合条件テスト（B）を使用することにより、可能性のあるすべての入力と出力をカバーできるが、実行する必要のあるテストの数は、このテスト技法が最も多い。

改良条件判定テスト（C）を使用することにより、すべての単一条件とすべての判定結果を、複合条件テストに比べてより少ない数のテストで完全なカバレッジを達成できる。

4.2.2 バックツーバックテスト

バックツーバックテストは、疑似オラクルを使用してテスト対象を検証し、期待される結果を生成する過去オラクルを期待結果の生成に使用してテスト対象をテストするテストアプローチである。このため、テスト担当者はすべてのバリアントに対して同じテストケースを実行し、テスト結果を比較す

る。テスト結果が同じであれば、テストは合格となる。テスト結果が異なる場合、検出された差異の原因を分析する。

テスト対象は、内容の観点から同じ要件に基づく必要がある。この場合に限り、テスト対象は比較可能な振る舞いを示す。これらの要件は、通常はテストの入力の生成のための基盤として使用するが、期待される結果を定義するテストオラクルとしては使われない。テストオラクルベースその代わりに、バックツーバックテストでは、テスト対象の振る舞いが過去オラクルの振る舞いと比較される。このようにすることで、バックツーバックテストでは、テスト対象間一方のテスト対象をもう一方の疑似オラクルとして使用し、両者の動作を比較する。これにより、テスト対象アイテム間またはテスト環境間の意図しないごくわずかな差異でも明らかにすることが期待される。

最も単純なケースでは、バックツーバックテストのテスト対象は、同じソフトウェアの異なるバージョンである。この場合、古いバージョンのテスト対象はバックツーバックテストの過去オラクルとして機能し、リグレッションテストと同様の役割を果たす。別のケースは、実行可能なモデルと手動または自動で生成されたコードとの比較である。これはモデルベースドテストの 1 つの形式であり、実行可能なモデルが過去オラクルとしても機能する。したがって、このテストアプローチは自動化されたテスト設計に非常に適している。ここでは、テスト担当者はモデルから期待結果だけでなく、自動化されたテストケースも導き出す。

4.2.3 フォールトインジェクションテスト

コンポーネントまたはシステムがさまざまな条件（例えば、欠陥、この文脈においてはフォールト）に対する反応を評価するために設計するテストアプローチであるフォールトインジェクションテストは、コンポーネントやシステムが好ましくない状態（例：欠陥、この文脈では、故障とも呼ばれる）に対してどのように反応するかを評価するために設計されたテストアプローチである。エラー処理などのプログラミング技法は、堅牢かつ安全な方法でシステムが内部および外部の欠陥に対処できるようにすることを目的とする。テスト担当者はフォールトインジェクションテストを実施するために、システムの以下のポイントに欠陥を選択的に挿入する。

- 外部コンポーネントの欠陥：例えば、センサーからのあり得ない値の検出
- インターフェースの欠陥：例えば、短絡や信号途絶による被害の回避危険の回避
- アプリケーションの欠陥：内部欠陥の検出とハンドリング

フォールトはシステムとテスト環境に依存してさまざまな方法で注入される。

欠陥は、システムやテスト環境に応じて、さまざまな方法で挿入される場合がある。

- 従来のフォールトインジェクションでは、テスト担当者は物理的なテスト対象を操作することで欠陥を挿入する。
- 外部コンポーネントの欠陥（またはインターフェース関連の欠陥）は、通常は HiL テスト環境においてフォールトインジェクションユニット（FIU）を使用して実行時にシミュレーションできる。
- ソフトウェアベースの欠陥（インターフェース欠陥や内部欠陥など）は、デバッガーなどのツールや汎用的測定およびキャリブレーションプロトコル（XCP）を用いて、SiL テスト環境や開発環境内で直接シミュレーションされることが多い。
- 外部コンポーネントの欠陥（およびインターフェースの欠陥）は、実行時にHiLテスト環境（FIUを用いる）でシミュレーションされることが多い。
- インターフェースや内部欠陥などのソフトウェアベースの欠陥は、多くの場合、SiL テスト環境または直接開発環境においてデバッガーや Universal Measurement and Calibration Protocol（XCP）を使用してシミュレーションされる

4.2.4 要件ベースドテスト

要件ベースのテストでは、テスト担当者は通常、テキスト形式の要件を分析し、個々の単独である要件からテスト条件を導出する。その後、要件を実行するテストケースを設計し、それらを実行する。この分析では、要件内の明確に識別可能でテスト可能な部分を特定する。個々の単独である要件は、独立して検証可能な単独の動作または条件を記述すべきである。

複雑またはハイレベルの要件をこのような細分化された要素に分解することで、テスト担当者は完全なカバレッジを確保し、隠れた期待や暗黙の期待を見落とすことを回避できる。要件カバレッジは、少なくとも 1 つのテストケースでカバーされた単独である要件の数が、単独である要件の総数に対する比率として測定される。

要件ベーステストのテストケースは一般に正のテストケースであり、明示された要件が満たされていることを確認する。負のテストケースは、同値分割法、エラー推測、探索的テストなどの補完的なテスト技法を用いて導出されることが多いが、要件が明示的にそのような否定の条件を定義する場合もある。

4.2.5 状況に依存した選択

自動車システムにおいて関連するテスト技法およびテストアプローチがいくつか存在する。

これらのうちの一部は、CTFL シラバス[ISTQB_FL_SYL]や本シラバスの 4.2 節で言及されているものが含まれる。

- 要件ベースドテスト
- 同値分割法
- 境界値分析
- ステートメントテスト
- デシジョンテスト
- MC/DC テスト
- エラー推測
- フォールトインジェクション
- バックツーバックテスト

ただし、使用する技法を決定する際には、特に以下の要因を考慮する。

最先端

テスト技法およびテストアプローチは目的にとって現在最高水準のものであるか？これについては、ISO/IEC/IEEE 29119 や ISO 26262 などの標準が役立つ。特に ISO/IEC/IEEE 29119-4 は、標準化されたテスト手法（対応するカバレッジ指標を含む）を体系的に整理し、ブラックボックステスト技法、ホワイトボックステスト技法、経験ベースのテスト技法に分類されている。ISO 26262 標準は、さらに 2.2 節で言及されている ASIL レベルに応じて適用できるテスト技法およびテストアプローチも提案している。標準の推奨から逸脱は慎重に検討し、正当化されなければならない。

テストベース

テストベースはテスト技法に対して適切なテスト条件を提供するか？ 例えば、テストベースにパラメーターや変数が含まれている場合のみ、テスト担当者は同値パーティションを構築できる。テスト担当者はそれらの値を適切な同値にグループ化できる必要がある。同様な条件が境界値にも適用される。値が連続した範囲で定義されている場合のみ、テストは実行できる。

リスクベースドテスト

リスクベースドテストでは、プロダクトリスクを特定し、テスト技法およびテストアプローチの選択に対するリスクレベルを考慮する。例えば、境界値のテストは、境界の誤りが発生するリスクが存在する場合のみ意味がある。

テストレベル

テスト技法およびテストアプローチは該当するテストレベルで適切に使用できるか？ ホワイトボックステストは、ソースコードまたは内部構造がテストベースとして機能する場合に特に適切である。理想的なケースでは、構造カバレッジを測定できる。ブラックボックステストでは、テスト対象が利用可能で観察できる必要がある。例えば、センサーの同値パーティションのテストは、コンポーネントテストで実行するよりもシステムテストで実行した方が効率的となる可能性がある。あるテストレベルでテスト技法およびテストアプローチが使用できない場合、テスト担当者はテスト戦略に従って、別のテストレベルを選択する必要がある。

テスト技法およびテストアプローチの選択例

以下の表は、テスト技法およびテストアプローチの一覧を示し、これまでに言及した要因のいくつかを評価し、それに基づきテスト技法 / テストアプローチを選択した例を示している。

表 6：テスト技法 / テストアプローチの選択例

	テスト技法 / テストアプローチ	ASIL A での使用は推奨されるか？	テストベースは適切か？	欠陥が検出されない場合のリスクは？	テストレベル「システムテスト」は適切か？	選択
1	要件ベースドテスト	++	はい	大	はい	X
2	同値分割法	+	はい	大	はい	X
3	境界値分析	+	いいえ	-	はい	
4	ステートメントテスト	++	はい	大	いいえ	
5	デシジョンテスト	+	はい	大	いいえ	
6	改良条件判定テスト（MC/DC）	+	はい	小	いいえ	
7	エラー推測	+	いいえ	大	はい	
8	フォールトインジェクションテスト	+	はい	小	いいえ	
9	バックツーバックテスト	+	いいえ	大	はい	
キー：++ 強く推奨、+ 推奨、- 対象外						

5 略語一覧

略語	定義 / 意味
ACQ	Acquisition（調達） (ASPICE)
ASIL	Automotive safety integrity Level（自動車用安全度水準）
ASAM	Association for Standardization of Automation and Measuring Systems
ASPICE	Automotive SPICE
AUTOSAR	Automotive Open System Architecture
AUTOSIG	Automotive Specific Interest Group
BP	Base Practice（基本プラクティス）(ASPICE)
BSW	Basic Software（基盤ソフトウェア）(AUTOSAR)
CTFL	Certified Tester Foundation Level（テスト技術者資格制度 Foundation Level）
E/E	Electric / Electronic（電気/電子）
ECU	Electronic Control Unit（電子制御ユニット）
EES	Electrical Error Simulation（電気エラーシミュレーション）
EOP	End of Production（量産終了）
FIU	Fault Insertion Unit（フォールト生成ユニット）
GP	Generic Practice（共通プラクティス） (ASPICE)
HiL	Hardware-in-the-loop（ハードウェアインザループ）
IEC	International Electrotechnical Commission（国際電気標準会議）
ISO	International Organization for Standardization（国際標準化機構）
MAN	Management（マネジメント）(ASPICE)
MC/DC	Modified Condition/Decision Coverage（改良条件判定カバレッジ）
MiL	Model-in-the-loop（モデルインザループ）
MISRA	Motor Industry Software Reliability Association
OEM	Original Equipment Manufacturer
PA	Process Attribute（プロセス属性）(ASPICE)
PIM	Process Improvement（プロダクト進化プロセス）(ASPICE)

略語	定義 / 意味
QM	Quality Management（品質マネジメント）
REU	Reuse（再利用）(ASPICE)
RTE	Run Time Environment（実行環境）(AUTOSAR)
SiL	Software-in-the-loop（ソフトウェアインザループ）
SOP	Start of Production（量産開始）
SPICE	Software Process Improvement and Capability Determination
SPL	Supply（供給）(ASPICE)
SUP	Support（サポート）(ASPICE)
SW-C	Software Component（ソフトウェアコンポーネント）(AUTOSAR)
SWE	Software Engineering（ソフトウェアエンジニアリング）(ASPICE)
SYS	Systems Engineering（システムエンジニアリング）(ASPICE)
VAL	Validation（妥当性確認） (ASPICE)
VDA	German Association of the Automotive Industry（ドイツ自動車工業会）
XCP	Universal Measurement and Calibration Protocol
XiL	MiL、SiL、HiL などさまざまな仮想テスト環境の総称

6 ドメイン固有の用語

用語	定義
Automotive Open System Architecture	自動車産業におけるソフトウェアアーキテクチャーのオープンな業界標準を確立する開発パートナーシップ。 注：この用語は、標準化されたソフトウェアアーキテクチャーおよび関連するシステム／ソフトウェア開発手法を指す場合にも使用される。
自動車用安全水準	ISO 26262 で定義される自動車の機能安全における安全水準。
Automotive SPICE	自動車産業におけるプロセス参照モデルおよび関連するプロセス評価モデル。
基盤ソフトウェア (AUTOSAR)	AUTOSAR では、標準化されたハードウェア指向のコンポーネントで構成されるソフトウェアレイヤー。
ブレークアウトボックス	ワイヤー内の物理信号の分析、割り込み、操作を行う測定ユニット。
バスシステム	同一の信号線を介して情報を交換する電子制御ユニットのネットワーク。
能力座標	プロセスアセスメントモデルにおける評価対象となるプロセス属性を定義する座標。
能力指標	プロセス能力アセスメントで使用されるプロセス能力の指標。
能力レベル	プロセス能力アセスメントにおける対応するプロセス属性の評定に基づいてプロセスに割り当てられるレベル。
クローズドループシステム	制御アクションまたは入力が、出力または出力の変化に依存するシステム。
指針(MISRA-C)	静的解析では完全に検証できない MISRA-C のプログラミングガイドライン。
ECU configuration description	ソフトウェアコンポーネントを電子制御ユニットと統合するために必要なデータ。
ECU extract	電子制御ユニットのシステム構成データ。
電子制御ユニット	自動車分野において、車両内の電気（サブ）システムを制御する組込みシステム。
MISRA-C	MISRA が提供するクリティカルシステム向け C 言語プログラミングのためのコーディングガイドライン。
オープンループシステム	制御アクションまたは入力が、出力または出力の変化から独立しているシステム。
original equipment manufacturer	自動車業界における自動車メーカー。
プロセス属性	プロセス能力アセスメントのためのプロセスの測定可能な特性。
プロセス座標	プロセスアセスメントモデルにおける評価対象となるプロセスを定義する座標。
プロダクト開発プロセス	初期のプロダクト構想から生産に至るまでの全活動を含むプロセス

用語	定義
リアルタイム	所定の時間内に結果が得られるようにデータを処理すること。
リアルタイム対応コンピューター	リアルタイムで信号を処理できるコンピューター。
リリース	リリースアイテムの提供に関する文書記録。
リリースアイテム	実装された機能、特性および意図された用途を有する識別可能な要素。
リリースプロセス	リリースに向けて進行するプロセス。
リリース推奨	テスト結果に基づいて、リリース対象のリリースアイテムを公開すべきか否かの推奨事項。
ルール(MISRA-C)	静的解析によって完全に検証可能な MISRA-C のプログラミングガイドライン。
ランタイム環境 (AUTOSAR)	AUTOSAR において、電子制御ユニット内および電子制御ユニット間で、AUTOSAR ソフトウェアコンポーネント間ならびにアプリケーションソフトウェアと基盤ソフトウェア間のデータ交換を制御・実装する抽象化レイヤー。
安全ライフサイクル	安全性に関連するシステムの設計から廃棄までのライフサイクル。
シミュレーション時間	コンピューターシミュレーションの時間枠。
ソフトウェアコンポーネント(AUTOSAR)	AUTOSAR において、ハードウェア非依存アプリケーションソフトウェアレイヤーにおけるコンポーネント。
ソフトウェアコンポーネント検証および統合検証 (ASPICE)	Automotive SPICE において、ソフトウェアアーキテクチャー設計に基づく統合されたソフトウェアの検証。
ソフトウェアユニット検証 (ASPICE)	Automotive SPICE において、ソフトウェアユニットの詳細設計との整合性に関する検証
ソフトウェア検証(ASPICE)	Automotive SPICE において、統合されたソフトウェアがソフトウェア要件と整合していることの検証
system configuration description	車両内のすべての電子制御ユニットを統合する際に使用されるデータ。
システム統合および統合検証(ASPICE)	Automotive SPICE において、統合されたシステム要素がシステムアーキテクチャーと整合していることの検証。
システムライフサイクル	システムが廃棄されるまでの開発フェーズ。
システム検証(ASPICE)	Automotive SPICE において、システム要件との整合性に関するシステムの検証

7 引用文献

7.1 標準

規格番号	規格名称
Automotive SPICE 4.0 (2023)	Automotive SPICE プロセスアセスメント / リファレンスモデル
ISO 14229 (2020)	道路運送車両 ― 統合診断サービス (UDS)
ISO 26262 (2018)	道路運送車両 ― 機能安全
ISO 26262-1 (2018)	第 1 部：用語集
ISO 26262-2 (2018)	第 2 部：機能安全の管理
ISO 26262-3 (2018)	第 3 部：コンセプトフェーズ
ISO 26262-4 (2018)	第 4 部：システムレベルの製品開発
ISO 26262-5 (2018)	第 5 部：ハードウェアレベルの製品開発
ISO 26262-6 (2018)	第 6 部：ソフトウェアレベルの製品開発
ISO 26262-7 (2018)	第 7 部：生産、運用、保守および廃棄
ISO 26262-8 (2018)	第 8 部：支援プロセス
ISO 26262-9 (2018)	第 9 部：ASIL（自動車安全度水準）および安全指向の分析
ISO 26262-10 (2018)	第 10 部：ISO 26262 に関するガイドライン
ISO/IEC 25010 (2023)	システムおよびソフトウェア工学 ― システムおよびソフトウェア品質要件と評価 (SQuaRE) ― 製品品質モデル
ISO/IEC 33020 (2019)	情報技術 ― プロセスアセスメント ― プロセス能力アセスメントのためのプロセス測定フレームワーク
ISO/IEC/IEEE 12207 (2017)	システムおよびソフトウェア工学 ― ソフトウェアライフサイクルプロセス
ISO/IEC/IEEE 15288 (2023)	システムおよびソフトウェア工学 ― システムライフサイクルプロセス
ISO/IEC/IEEE 24748-1 (2018)	システムおよびソフトウェア工学 ― ライフサイクル管理 ― 第 1 部：ライフサイクル管理ガイド
SO/IEC/IEEE 29119-1:2022	ソフトウェアおよびシステム工学 ― ソフトウェアテスト 第 1 部：一般概念
ISO/IEC/IEEE 29119-2:2021	ソフトウェアおよびシステム工学 ― ソフトウェアテスト 第 2 部：テストプロセス
ISO/IEC/IEEE 29119-3:2021	ソフトウェアおよびシステム工学 ― ソフトウェアテスト 第 3 部：テストドキュメント
ISO/IEC/IEEE 29119-4:2021	ソフトウェアおよびシステム工学 ― ソフトウェアテスト 第 4 部：テスト技法
ISO/IEC/IEEE 29148:2018	システムおよびソフトウェア工学 ― ライフサイクルプロセス ― 要件工学
MISRA C:2025	クリティカルなシステムにおける C 言語利用のためのガイドライン

7.2 ISTQB® 関連文書

ISTQB_ALTTA_SYL	ISTQB® テスト技術者資格試験 Advanced Level テクニカルテストアナリスト シラバス 2012 年版
ISTQB_FL_SYL	ISTQB® テスト技術者資格試験 Foundation Level シラバス v4.0 (2023 年)

7.3 用語集の引用

本シラバスで使用されている用語については、以下の用語集を参照してください：

IREB® 用語集 https://cpre.ireb.org/en/downloads-and-resources/glossary
ISTQB® 用語集 https://glossary.istqb.org

8 商標

CTFL® は、EU（欧州連合）内における German Testing Board (GTB) e. V. の登録商標

GTB® は、EU（欧州連合）内における German Testing Board (GTB) e. V. の登録商標

ISTQB® は、International Software Testing Qualifications Board の登録商標

Automotive SPICE® は、ドイツ自動車工業会（VDA）の登録商標

9 付録 A：学習の到達目標と知識の認知レベル

本シラバスに適用される個別の学習目標は、各章の冒頭に示されている。シラバス内の各トピックは、それぞれに設定された学習目標に従って試験が行われる。 各学習目標は、以下に挙げる知識の認知レベルに対応した「動作動詞（アクション・バーブ、action verb）」から始まる。

レベル 1：記憶（K1）

受験者は、用語や概念を記憶し、認識し、思い出すことができる。

動作動詞（アクション・バーブ）： 想起する（思い出す）、認識する

例
テストピラミッドの概念を想起する。
テストの典型的な目的を認識する。

レベル 2：理解（K2）

受験者は、トピックに関連する記述に対してその理由や根拠を選択でき、テストの概念について要約、比較、分類、および具体例を挙げることができる。

動作動詞（アクション・バーブ）： 分類する、比較する、相違を述べる、区別する、説明する、例を挙げる、解釈する、要約する

例	備考
テストツールを、その目的とサポートするテスト活動に従って分類する。	
異なるテストレベルを比較する。	類似点、相違点、またはその両方を探すために用いられる。
テストとデバッグを区別（相違を特定）する。	概念間の相違点を探す。
プロジェクトリスクとプロダクトリスクを区別する。	2 つ（またはそれ以上）の概念を個別に分類できるようにする。
テストプロセスに対するコンテキスト（状況・文脈）の影響を説明する。	
なぜテストが必要なのか、その具体例を挙げる。	
提示された故障のプロファイルから、欠陥の根本原因を推論する。	
作業成果物レビュープロセスの活動を要約する。	

レベル 3：適用（K3）

受験者は、使い慣れたタスクに直面した際に手順を実行できる。または、与えられた状況に対して適切な手順を選択し、それを適用することができる。

動作動詞（アクション・バーブ）： 適用する、実装する（実施する）、準備する、使用する

例	備考
与えられた要件からテストケースを導出するために、境界値分析を適用する。	手順、技法、プロセスなどを指すべきである。
技術的要件および管理上の要件をサポートするために、メトリクス収集方法を実装（実施）する。	
モバイルアプリの設置性テスト（インストーラビリティテスト）を準備する。	
テスト目標、テスト戦略、およびテスト計画に対する網羅性と一貫性を監視するために、トレーサビリティを使用する。	技法や手順を使用できることを受験者に求める学習目標（LO）で使用される。「適用する」と同様。

参考文献（学習目標の認知レベルに関するもの）

Anderson, L. W. and Krathwohl, D. R. (eds) (2001) A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives, Allyn & Bacon

10 付録 B：学習目標とビジネス成果のトレーサビリティ・マトリクス

本章では、ビジネス成果と「自動車ソフトウェアテスト技術者」の学習目標との対応関係を一覧に示します。

ビジネス成果(BO): 自動車ソフトウェアテスト担当者		BO1	BO2	BO3	BO4	BO5
AuT-BO1	テストチーム内で効果的に共同作業する（「共同作業する」）	37				
AuT-BO2	ISTQB®テスト技術者資格制度 Foundation Level（CTFL®）のテスト技法を、具体的なプロジェクト要件に適合させる（「適合させる」）		7			
AuT-BO3	関連する標準（Automotive SPICE®、ISO 26262 など）の基本的な要件を考慮して、適切なテスト技法を選択する（「選択する」）			27		
AuT-BO4	リスクの存在するテスト活動の計画作成においてテストチームを支援し、構造化と優先順位付けの既知の要素を適用する（「支援および適用する」）				9	
AuT-BO5	テスト環境において、仮想テスト環境（HiL、SiL、および MiL）を適用する。（「適用する」）					13

ビジネス成果(BO): 自動車ソフトウェアテスト担当者			BO1	BO2	BO3	BO4	BO5
ユニーク LO	学習の目的	K レベル					
1	イントロダクション						

1.1	多様化するプロジェクト目標やプロダクトの複雑さの増大により発生する要件						
AuT-1.1	多様化するプロジェクト目標やプロダクトの複雑さの増大により発生する自動車プロダクト開発の課題を説明し、例を挙げる。	K2	X				
1.2	標準の影響を受けるプロジェクトの側面						
AuT-1.2	標準により影響されるプロジェクトの側面（例えば時間、コスト、品質、プロジェクト/プロダクトリスクなど）を想起する	K1	X				
1.3	システムライフサイクルでの一般的な6つのフェーズ						
AuT-1.3	ISO/IEC/IEEE 24748-1 で定義されているシステムライフサイクルの6つの一般的なフェーズを想起する	K1	X				
1.4	リリースプロセスでのテスト担当者の役割と参画						
AuT-1.4	リリースプロセスにおけるテスト担当者としての役割（例えば貢献やと協力）を想起する	K1	X				
ビジネス成果(BO): 自動車ソフトウェアテスト担当者			BO1	BO2	BO3	BO4	BO5
ユニーク LO	学習の目的	K レベル					
2	E/E システムのテストに関する標準						
2.1	Automotive SPICE（ASPICE）						
AuT-2.1.1.1	ASPICE の 2 つの座標軸を想起する	K1			X		
AuT-2.1.1.2	ASPICE のプロセス区分およびプロセス群を想起する［参考情報］	K1			X		
AuT-2.1.1.3	ASPICE の能力レベル 0～3 を説明する	K2			X		
AuT-2.1.2.1	ASPICE のテスト特有プロセスの目的を想起する	K1	X		X		
AuT-2.1.2.2	テストの観点から、ASPICE の 4 つの評定尺度と能力指標の意味を説明する。	K2			X		
AuT-2.1.2.3	リグレッション検証の基準を含むテスト戦略に関する ASPICE の要件を説明する	K2	X		X	X	
AuT-2.1.2.4	テストウェアに関する ASPICE の要件を想起する	K1	X		X	X	
AuT-2.1.2.5	ユニット検証用の検証戦略（テスト戦略との対比）と基準を設計する	K3	X		X	X	
AuT-2.1.2.6	テストの観点から ASPICE のトレーサビリティ要件を説明する	K2	X		X		
ビジネス成果(BO): 自動車ソフトウェアテスト担当者			BO1	BO2	BO3	BO4	BO5
2.2	ISO 26262						
AuT-2.2.1.1	E/E システムの機能安全の目的を説明する	K2			X		

AuT-2.2.1.2	安全文化におけるテスト担当者の役割を想起する	K1	X		X		
AuT-2.2.2	ISO 26262 で規定されている安全ライフサイクルのフレームワークでのテスト担当者の役割を説明する	K2	X		X		
AuT-2.2.3.1	ASPICE のテスト特有プロセスの目的を想起する	K1			X		
AuT-2.2.3.2	テスト担当者に関連する、ISO 26262 のパートを想起する	K1	X		X		
AuT-2.2.4.1	ASIL の安全度レベルを想起する	K1	X		X		
AuT-2.2.4.2	静的テストや動的テストのためのテスト技法や、テストタイプに関する ASIL の影響および結果としてのテスト範囲を説明する	K2	X	X	X	X	
AuT-2.2.5	ISO 26262 の手法テーブルを解釈できるようになる	K3	X	X	X	X	
2.3	AUTOSAR						
AuT-2.3.1	AUTOSAR のプロジェクトの目的を想起する	K1			X		
AuT-2.3.2	AUTOSAR の一般的な設計を想起する［参考情報］	K1			X		
AuT-2.3.3	テスト担当者の作業に対する AUTOSAR の影響を想起する	K1	X	X	X		
2.4	ASPICE、ISO 26262、CTFL®の比較						
AuT-2.4.1	ASPICE と ISO 26262 の目的の差異を想起する	K1			X		
AuT-2.4.2	テストレベルに関する ASPICE、ISO 26262、CTFL®の間の相違を説明する	K2	X	X	X		
ビジネス成果(BO): 自動車ソフトウェアテスト担当者			BO1	BO2	BO3	BO4	BO5
3	仮想環境でのテスト						
3.1	一般的なテスト環境						
AuT-3.1.1	自動車開発におけるテスト環境に対する目的/モチベーションを想起する	K1	X				X
AuT-3.1.2	自動車固有のテスト環境の一般的な構成要素を想起する	K1	X				X
AuT-3.1.3	自動車固有のテスト環境の一般的な構成要素を想起する	K2	X				X
AuT-3.1.4	ECU にとって重要な機能、データベース、およびプロトコルを想起する	K1	X				X
3.2	XiL テスト環境でのテスト						
AuT-3.2.1.1	MiL テスト環境の構成を想起する	K1	X				X
AuT-3.2.1.2	MiL テスト環境の適用領域と境界条件を説明する	K2	X				X
AuT-3.2.2.1	SiL テスト環境の構成を想起する	K1	X				X
AuT-3.2.2.2	SiL テスト環境の適用領域と境界条件を説明する	K1	X				X
AuT-3.2.3.1	HiL テスト環境の構成を想起する	K1	X				X
AuT-3.2.3.2	HiL テスト環境の適用領域と境界条件を説明する	K2	X				X

AuT-3.2.4.1	XiL テスト環境（MiL、SiL、および HiL）を使用してテストすることの長所と短所について判断基準を用いて説明する	K2	X			X	X
AuT-3.2.4.2	1 つ以上のテスト環境に対し基準を適用して、特定のテスト範囲を割り当てる	K3	X			X	X
AuT-3.2.4.3	3 つの XiL テスト環境（MiL、SiL、および HiL）を V 字モデルに対応付ける	K1	X			X	X
ビジネス成果(BO): 自動車ソフトウェアテスト担当者			BO1	BO2	BO3	BO4	BO5
4	自動車ドメイン固有の静的および動的テスト技法						
4.1	静的テスト						
AuT-4.1.1	MISRA-C：2012 ガイドラインの目的と要件について例を用いて説明する	K2	X	X			
AuT-4.1.2	ISO/IEC 29148 が定義するテスト担当者に関係する品質特性を用いて要件をレビューする	K3	X	X			
4.2	動的テスト						
AuT-4.2.1	MC/DC テストカバレッジを達成するためのテストケースを作成する	K3	X		X		
AuT-4.2.2	バックツーバックテストについて例を用いて説明する	K2	X		X		
AuT-4.2.3	フォールトインジェクションテストの原則について例を用いて説明する	K2	X		X		
AuT-4.2.4	要件ベースドテストの原則を想起する	K1	X		X		
AuT-4.2.5	適切かつ必須のテスト技法とテストアプローチを選択する際にコンテキストに依存した基準を適用する。	K3	X	X	X	X	

11 付録 C：リリースノート

ISTQB 自動車ソフトウェアテスト技術者シラバス v2.1 (2025) は、v2.0.1 (2017) のマイナーアップデート版である。学習の目的（LO）自体に変更はないが、シラバス内で引用されている標準規格の更新を反映させるため、一部の LO の内容を更新した。加えて、ISTQB CTFL 3.1 から CTFL 4.0 への変更や、ISTQB 用語集の更新についても考慮している。また、必要に応じて整合性と分かりやすさの向上を図った。

本改訂では、以下の更新された標準規格を考慮している：

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

なお、LO の命名規則は「AuT」に節番号を付与する形式に変更された。

12 付録 D：自動車用データベースおよび通信プロトコル

インターフェース	データベース	通信プロトコル
メモリー (Memory)	ASAM MCD-2 MC (ASAP2 または A2L)	ASAM MCD-1 XCP (汎用計測・キャリブレーション（適合）プロトコル) ASAM MCD-1 CCP (CAN キャリブレーション（適合）プロトコル)
バス (Bus)	ASAM MCD-2 NET 規格 (FIBEX - Field Bus Exchange Format)	FlexRay (ISO 17458) CAN (ISO 11898-2 に準拠したコントローラー・エリア・ネットワーク)
	DBC (CAN 用通信データベース)	CAN (ISO 11898-2 に準拠したコントローラー・エリア・ネットワーク)
診断 (Diagnostics)	ASAM MCD-3 D (API 仕様) ASAM SOVD (サービス指向車両診断) CDD (CANdelaStudio 診断記述ファイル)	KWP2000 (ISO 14230) ISO-OBD (ISO 15031) UDS (ISO 14229) SOME/IP (AUTOSAR 仕様)

表 6：自動車業界における一般的なデータベースおよび通信プロトコル

AUTOSAR には、車両全体のデータベースを統合する標準化された XML 形式が存在する。これは **ARXML 形式**（AUTOSAR アプリケーション・インターフェース統合マスターテーブル、XML スキーマ R21-11）と呼ばれる。