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

Version 2018.J03

International Software Testing Qualifications Board

変更の概要

バージョン	日付：	注
1.0	2011/1/19	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。
1.1	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。
2.0	2017/3/31	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。
2.0.1（英語版）	2017/6/30	主要な用語に対する小規模な変更のみ。 2017 年 3 月に最初に行われた内部アルファレビュー後に挙がった各国のレビューアからの意見（主に用語法）を反映（「謝辞」を参照）。 対応する英語版の参考文献を追加（「参考文献」を参照）。
2.0.1（英語版）	2017/8/13	ISTQB®用語作業グループの意見に基づいて用語を微調整。2017 年 7 月に行われた 2 回目の内部アルファレビュー後に挙がった各国のレビューアからの意見（主に用語法）を反映（「謝辞」を参照）。
2.0.1（英語版）	2017/8/20	ISTQB®用語作業グループの新たな意見に基づいて用語を微調整。 レビューアからの最新の意見を反映。
2.0.1（英語版）	2017/9/15	ミュンヘンで開催された GTB 作業グループ会合でのレビューアからの意見を反映。
2.0.1（英語版）	2017/9/16	3.2.2 節を改訂。
2.0.1（英語版）	2017/9/22	GA ベータドラフト版用最終版。
2.0.2（英語版）	2018/5/28	ベータレビューを反映した、GA リリース用の最終版。
2.0.2（英語版）	2018/7/4	GA 承認後に、ISTQB®公開用にすかし模様を除去し、商標制限を追加。

JSTQB®

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

目次

謝辞

ドイツテスト委員会（GTB）は、ドイツ語版 2017、V2.0 の著作者およびレビューチーム（アルファベット順）に感謝したい。

Graham Bath, André Baumann, Arne Becher, Ralf Bongard (Lead Syllabus and Co-Chair WG), Kai Borgeest, Tim Burdach, Mirko Conrad, Klaudia Dussa-Zieger, Matthias Friedrich, Dirk Gebrath, Thorsten Geiselhart, Matthias Hamburg, Uwe Hehn, Olaf Janßen, Jacques Kamga, Horst Pohlmann (Lead Exam and Chair WG), Ralf Reißing, Karsten Richter, Ina Schieferdecker, Alexander Schulz, Stefan Stefan, Stephanie Ulrich, Jork Warnecke and Stephan Weißleder.

ドイツテスト委員会（GTB）と「自動車ソフトウェアテスト担当者資格制度」作業グループは、英語バージョン 2018、V.2.0.x の拡大レビューチームに感謝したい。Graham Bath, Thomas Borchsenius, Ádám Bíró, Zsolt Csatári, Attila Farkas, Attila Fekete , Ferenc Hamori, Ádám Jezsoviczki, Gábor Kapros, Miguel Mancilla, Roland Milos, Kenji Onishii, Miroslaw Panek, Mirosław Panek, Barthomiej Predki, Stefan Stefan, Stuart Reid, Ralf Reissing, Hidetoshi Suhara, Tamás Széplakin, Eshraka Zakaria and Csaba Zelei.

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：
大西 建児（ガイオ・テクノロジー）
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）
林 宏昌（デンソー）
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」）。

このドキュメントが作成されるまでの経緯

本シラバス 1.0 は、Global Association for Software Quality AISBL（gasq）からの要請に基づいて、Dr. Hendrik Dettmering により 2010 年から 2011 年にかけて作成された。

文書のレビューに向けて OEM の著名な専門家が指名され、本シラバスの目標と品質が適切であることをチェックおよび評価した。このため、本文書は、自動車ソフトウェアテスト担当者認定のためのシラバスであり、同時にトレーニング教材や認定試験向けの試験問題のベースとなる。

2014 年 1 月 1 日以降、ドイツテスト委員会（GTB）の「自動車ソフトウェアテスト担当者資格制度」作業グループが本シラバスの開発を引き継ぎ、本トピックの迅速な開発を可能にした。また、業界に依存しない Core シラバスの作成と、確立された ISTQB® Foundation Level に、自動車固有の側面に関するスペシャリストの資格の付加という業界からの 2 つの要請にも応えるようにした。

バージョン 1.1 は、2015 年 6 月 15 日にリリースされた。本バージョンはバージョン 1.0 を踏襲し、ISTQB® Foundation Level と重複する部分は削除された。

イントロダクション[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）の Foundation Level のソフトウェアテストトレーニングプログラムに対するスペシャリストを定義する。教育機関は本シラバスに基づいて、コース教材を作成し、認定に向けて適切な指導方法を定義する。資格取得希望者は、本シラバスを活用して試験に備える。

本シラバスの履歴と背景の詳細については、本シラバスの「このドキュメントができるまでの経緯」を参照されたい。

ISTQB® CTFL®-スペシャリスト：自動車ソフトウェアテスト担当者

テスト技術者資格制度 Foundation Level（CTFL®）トレーニングプログラムの現在のスペシャリストモジュールは、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にしている。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、リリーステスト担当者、ソフトウェア開発担当者などが含まれる。Foundation Level 資格の対象者には、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獲得と理解を必要とするプロジェクトマネージャー、品質マネージャー、ソフトウェア開発マネージャー、システムアナリスト（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。

[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]

ビジネス上の価値

本段落では、CTFL®自動車ソフトウェアテスト担当者として認定された人が習得していることが期待される、ビジネス上の価値（ISTQB®の「ビジネス成果」）を概説する。

CTFL®自動車ソフトウェアテスト担当者（CTFL®-AuT）は以下の技能を習得している。

AUTFL-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*」）。
AUTFL-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*」）。
AUTFL-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*」）。
AUTFL-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*」）。
AUTFL-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*」）。

学習する知識の目的と認知レベル

本シラバスの各段落は、1 つの認知レベルに割り当てられている。

- K1：記憶
- K2：理解
- K3：適用
- K4：分析

学習の目的は、対応する段落/章/モジュールを学習し終えた段階で資格取得希望者が習得していることが期待される事柄を定義する。

学習の目的のうち[情報提供]と記されている内容は、適切な時間枠内で教育機関により教えられるが、試験とは関連しない。

例：AUTFL-2.2.3.1　　ISO 26262 の設計と構造を想起する[情報提供]。

用語

資格取得希望者は、見出し「用語」（K1）の直下にある段落で列挙されているすべての用語を、学習の目的で明示的に列挙されていない場合でも、想起できる必要がある。ISTQB®用語集および承認済みバージョンの各国語用語集の定義（現在のシラバスのその他の用語も含む）が適用される。

試験

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。

受験要件

「自動車ソフトウェアテスト技術者資格制度」の受験希望者は、ISTQB®テスト技術者資格制度 Foundation Level（CTFL®）の認定を有しており、自動車開発プロジェクトのテストに関心を持っている必要がある。

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、システムテストやユーザー受け入れテストなどのテスト担当者、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。

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。

教育機関に対する更なる情報を付録に掲載する。

詳細レベル

本シラバスの詳細レベルは一貫した教育と試験を可能にする。このゴールを達成するために本シラバスは以下のようになっている。

- 全般的な学習の目的。（拡張された）基本レベルの意図を説明する。
- 必須の学習内容。説明と、必要に応じて他の文献への参照を含む。
- 各知識領域の学習の目的。トレーニングが目標とする認知結果と、資格取得希望者が到達すべきマインドセットを説明する。
- 資格取得希望者が理解し想起できる必要のある用語の一覧。
- 学習する必要のある重要な概念の説明。内容が確立されている技術文献や標準などの情報源を含む。

本シラバスは、「車載電子機器開発プロジェクトにおけるソフトウェア指向システムのテスト」分野の知識を完全に説明するものではない。学習の目的に関連して必要となるスコープや詳細のレベルに対応しているにすぎない。

本シラバスの構成

本シラバスは、4 つの主要な章で構成される。各章の一番上の見出しは、各章で説明される最も重要な学習の目的のカテゴリー/最高の認知レベルを示す。また、認定コースで当該の章に割り当てられる最小の学習時間を指定する。

例：

イントロダクション（K2）[30 分]

この例は、章見出し「イントロダクション（K2）」で K1[2]と K2 が期待され（K3 は期待されない）、この章の教材のトレーニングには 30 分を計画することを説明している。

各章は、複数の節で構成される。各節に対しても、学習の目的と時間枠が定義される。節に対して時間が定義されていない場合、章の時間に含まれている。

ジェンダーニュートラルな表現

読みやすくするために、男性ユーザーや女性ユーザーなどジェンダーを意識させる表現は使用しない。平等の原則に従い、全般的に、すべてのジェンダーに有効な役割名を使用する。

[2] より高いレベルに分類される学習の目的は、より低いレベルに分類される学習の目的を包含する。

1 イントロダクション（K2）[30 分]

用語

テスト固有の用語はなし

学習の目的

AUTFL-1.1.1 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）

AUTFL-1.2.1 時間、コスト、品質、プロジェクト/プロダクトリスクなどの標準により影響されるプロジェクトの側面を想起する。（K1）

AUTFL-1.3.1 ISO/IEC 24748-1 [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）

AUTFL-1.4.1 リリースプロセスにおけるテスト担当者としての貢献と協力を想起する。（K1）

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」[20]である。本段落では、「自動車ソフトウェアテスト担当者」[3]が活動する E/E 開発の環境を概説する。本環境では、目標の多様化、複雑さの増大、イノベーションに対する大きなプレッシャーにより、特別な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貢献する。

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（K2）[15 分]

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[4]、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。
- 構成の数の増加

[3] 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。
[4] 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」[44]

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、および品質は競合する（「プロジェクト管理トライアングル」）。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（K1）[5 分]

標準は、時間、コスト、品質、プロジェクト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：安定した品質の下で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[20]、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、監査者はプロダクトやプロセスが要件を満たしていることをチェックできる[1]。
- 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、他のシラバスと同様、以下の標準を参照する。

- プロセスや手法を標準化する ISO 26262 [8]や Automotive SPICE（ASPICE）[2]などの標準。
- プロダクトを標準化する AUTOSAR[3]などの標準。

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（K1）[5 分]

自動車とすべての車載コンポーネント[5]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。本ライフサイクルの間においては、開発プロセス、ビジネスプロセス、物流プロセス、生産技術に関連するプロセスが関与する。開始基準と終了基準を事前に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に役立つ。これらは、システムライ

[5] 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。

フサイクル[6]を以下の 6 つのフェーズに分類して同期する[1]。（典型的なテスト活動[7]を括弧内に示す）：

- 概念（テスト計画作業）
- 開発（テストの分析、設計、実装、実行、評価、および報告）
- 生産（完成品性能テスト）
- 利用（テスト活動はなし）
- 支援（メンテナンステスト）
- 廃止（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、および生産で構成される。

1.4　リリースプロセスでのテスト担当者の役割と参画（K1）[5 分]

自動車製造の環境では、プロジェクトはリリースを宣言することでマイルストーンに到達し、証跡を確認することで、目標の到達を判断する。これ以降、リリースアイテムはその用途や目的に必要とされる成熟度レベルを満たす。

リリースプロセスは、リリースアイテムのリリースに向けて進行することが予見される。リリースアイテムは、テストアイテム（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）とその他の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[2]：

- テスト済みアイテムと性能特性（それらのバージョンを含む）
- 既知の欠陥
- プロダクトメトリクス
- リリース規則に基づくリリース推奨の情報（テスト終了基準が達成された場合）。例えば、ベストプラクティスガイドライン（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。

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結果の作成に参加する[4]。

- 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）。

[6] 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。
[7] 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[2]

2　E/E システムのテストに関する標準（K3）[300 分]

用語

Automotive SPICE（ASPICE）

Automotive SPICE（ASPICE）、ソフトウェア適格性確認テスト（ASPICE）、システム適格性確認テスト（ASPICE）

ISO 26262[8]

自動車用安全度水準（ASIL）、機能安全、手法テーブル（ISO 26262）

AUTOSAR

テスト固有の用語はなし

比較

テスト固有の用語はなし

学習の目的

Automotive SPICE（ASPICE）

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

AUTFL-2.1.1.2 ASPICE の 3 つのプロセスカテゴリーと 8 つのプロセス群を想起する[情報提供]。（K1）

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

AUTFL-2.1.2.1 ASPICE の 5 つのテスト関連プロセスの目的を想起する。（K1）

AUTFL-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）

AUTFL-2.1.2.3 リグレッションテスト戦略を含むテスト戦略に関する ASPICE の要件を説明する。（K2）

AUTFL-2.1.2.4 テスト文書に関する ASPICE の要件を想起する。（K1）

AUTFL-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）

AUTFL-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）

ISO 26262

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

AUTFL-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）

AUTFL-2.2.2.1 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）

[8] 機能安全に関する用語の日本語訳は、基本的に日本規格協会発行の ISO 26262 和英対訳版に従う。

AUTFL-2.2.3.1 ISO 26262 の設計と構造を想起する[情報提供][9]。

AUTFL-2.2.3.2 テスト担当者に関連する、ISO 26262 のパートのタイトルを想起する。（K1）

AUTFL-2.2.4.1 ASIL の安全度レベルを想起する。（K1）

AUTFL-2.2.4.2 静的テストや動的テストに適用できるテスト設計技法や、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）

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

AUTOSAR

AUTFL-2.3.1 AUTOSAR の目的を想起する。（K1）

AUTFL-2.3.2 AUTOSAR の全般的な設計を想起する[情報提供][10]。（K1）

AUTFL-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）

比較

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

AUTFL-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）。

2.1 Automotive SPICE（ASPICE）（K3）[140 分]

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。この場合のプロセスモデルは、組織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[5]。

2001 年、SPICE[11] User Group and the AUTOSIG（Automotive Special Interest Group）は Automotive SPICE（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。

2015 年 7 月、ドイツ自動車工業会（VDA）は ASPICE バージョン 3.0[7]をリリースした。2017 年から、ASPICE 3.0 の改訂バージョン V.3.1[48]が、実績のあるバージョン 2.5[2]を置き換えつつある。このため、本段落でのすべての記述は、ASPICE のバージョン 3.1 に基づいている[48]。

[9] 試験で必須ではない
[10] 試験で必須ではない。
[11] 「Software Process Improvement and Capability dEtermination」の頭字語。

2.1.1 標準の設計と構成（K2）[25 分]

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を定義する。これらは、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（基本プラクティス）と作業成果（作業成果物）を定義する。組織が ASPICE よりも詳細な参照プロセスの情報を必要とする場合は、ISO/IEC 12207[22]または ISO/IEC 15288[14]を参照されたい。

ASPICE の**能力座標**は、多くのプロセス属性を定義する。これらは、プロセスの能力についての測定可能な特徴を提供する。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[40]。

これらのモデルを使用することで、プロセス（プロセス座標）をプロセスの能力（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

ASPICE はプロセスを 8 つのプロセスグループにまとめており、プロセス群は 3 つのプロセスカテゴリーにまとめられる[7][48]：

主要プロセスは、企業の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。

- プロダクトそして/またはサービスの取得（ACQ）
- プロダクトそして/またはサービスの供給（SPL）
- システムエンジニアリング（SYS）
- ソフトウェアエンジニアリング（SWE）

支援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（SUP）

組織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。

- プロジェクトまたはプロセスの管理（MAN）
- プロセス改善（PIM）
- システムおよびコンポーネントの再利用（REU）

テスト担当者にとっては、システム開発（SYS）とソフトウェア開発（SWE）の 2 つのプロセス群が主な着目点である。これらは、Automotive SPICE V 字モデルのプロセスを構築する（[7]付録 D「主要な概念」）。

2.1.1.3 能力座標の能力レベル

アセッサーは 6 つのレベルのアセスメントシステム（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[12]を定義する[7][48]。

- 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。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チェックし、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、およびプロセスの管理を行う。逸脱が発生した場合は、状況に応じて対応する。
- 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：組織全体向けに共通のテスト戦略がある。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支援する（「基本的なテストプロセス」を参照）。

2.1.2 ASPICE のテストに関する要件（K3）[115 分]

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[6]。

- 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。
- ソフトウェア統合テスト（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（SWE.2）に基づいてテストする。
- ソフトウェア適格性確認テスト（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。
- システム統合テスト（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（SYS.3）に基づいてテストする。
- システム適格性確認テスト（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 アセスメントレベルと能力指標

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（括弧内に SWE.6 の例を示す）。

- PA 1.1：プロセス実施（テスト担当者は基本的なテストプロセスに従ってプロセスを実施する）。

[12] 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。

- PA 2.1：実施管理（テスト担当者は、主にテスト活動の計画、管理、制御を行う）。
- PA 2.2：作業成果物管理（テスト担当者は、主にテスト文書の品質チェックを行う）。
- PA 3.1：プロセス定義（テストプロセスの責任者は主に全般的なプロジェクト戦略の定義を行う）。
- PA 3.2：プロセス展開（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（PA 1.1）については、ASPICE は基本プラクティス（BP）と作業成果物（WP）の 2 種類の指標を定義している。さらに、共通プラクティス（GP）と共通リソース（GR）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**（Not achieved）：達成していない（0 ≦ 15%）
- **P**（Patially achieved）：部分的に達成している（＞ 15 ～ ≦ 50%）
- **L**（Largely achieved）：おおむね達成している（＞ 50 ～ ≦ 85%）
- **F**（Fully achieved）：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、達成されるべき能力レベルの指標が「おおむね達成している（L）」でなければならない。より低い能力レベルの指標は「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テスト戦略

基本的プラクティスとして、ASPICE は各テスト固有のプロセス（2.1.2.1 参照）に対してテスト戦略[13]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。

リグレッションテスト戦略は、テスト戦略の重要な部分である。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」）。リグレッションテスト戦略は、リグレッションテストを選択するための目的と技法を定義する。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 でのテスト文書

テスト活動における文書化のために、ASPICE は CTFL®に記載されている多くの作業成果物（WP）を必要とする[7]。

- WP 08-50：テスト仕様書（テスト設計、テストケース、テスト手順仕様を含む）
- WP 08-52：ISO/IEC/IEEE 29119-3[35]に従ったテスト計画書と、それに含まれる戦略（WP 19-00）

[13] CTFL [20]では、プロジェクト固有のテスト戦略は、テストアプローチとも呼ばれる。

- WP 13-50：テスト結果、テストログ、インシデント/逸脱レポート、テストサマリーレポート

各作業成果物について、ASPICE は特性と内容の例を定義している。アセッサーは抜き取りチェックを行うことで、それらを評価できる。作業成果物はアセッサーにとって、プロセス実施の客観的指標として機能する。

テスト計画書について、ASPICE は ISO/IEC/IEEE 29119-3[14]を直接参照する。本標準は、他の必須作業成果物に対して使用でき、特定の目的に適用できるテンプレートも提供している。このテンプレートを使う際には、特定の状況の下で、プロセスの意図される目的に役立つことを明確にしておかなくてはならない。

2.1.2.5　ユニット検証（SWE.4）向けの検証戦略と合格/不合格基準

ソフトウェアユニット検証（SWE.4）について、ASPICE は検証戦略[15]を必要とする。SWE.5/SWE.6/SYS.4/SYS.5 のテスト固有のプロセスについて、ASPICE はテスト戦略（2.1.2.3 を参照）を必要とする。テスト戦略は、動的テストでのみ使用される。これは検証戦略への追加事項であり、コードレビューと静的解析も考慮する（両方の技法とも、CTFL®では「静的テスト」として参照される）。

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。

開発担当者が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、ユニット検証戦略は、リグレッション戦略も含む。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。

SWE.4.BP.2 で、ASPICE はユニット検証用の合格/不合格基準の開発を必要とする。これらの合格/不合格基準は、満たすべき条件を定義する。このため、テスト担当者はユニットが非機能要件を達成している割合と詳細設計に一致している割合を評価できる。以下の合格/不合格基準はユニットの検証用の合格/不合格基準となりうる。

- ユニットテストケース（テストデータを含む）
- テストカバレッジの目的（例：デシジョンカバレッジ）
- ツールで支援された静的解析（コーディング標準（MISRA-C など、4.1.1 を参照）との適合性を評価する）
- ユニットまたはユニットの特定箇所（パーツ）向けのコードレビュー（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）

Automotive SPICE（ASPICE）では、検証戦略の文書化は、ユニットレベルのテスト計画（[15]段落 6.2.7）の一部として行われる。内容は ISO/IEC/IEEE 29119-3 に従って編成され、静的テストの観点により補強される。

[14] これは、ISTQB シラバスで引き続き使用されている IEEE 829：1998 および IEEE 829：2008 を置き換える。
[15] 「検証戦略」と「テスト戦略」の用語について、ASPICE では「戦略」という用語が、ISTQB のプロジェクト固有の「アプローチ」の代わりに使用されている。

2.1.2.6 Automotive SPICE（ASPICE）でのトレーサビリティ

CTFL® Core シラバス[20]と同じく、ASPICE も双方向トレーサビリティ[16] 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
- 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[7]。

垂直トレーサビリティ — ステークホルダー要件をソフトウェアコンポーネン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成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 BP8 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。変更依頼は問題により開始され、変更依頼と対応する問題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。

2.2 ISO 26262（K3） [125 分]

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、（検知されない）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担当者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 を策定している。

ISO 26262 では、機能安全を、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として定義している。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[8][41]。就労環境の安全とサイ

[16] 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。

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。システムの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全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[8]。

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（K2）[15 分]

安全ライフサイクルは、安全指向の製品開発のフェーズを定義する。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（1.3 節を参照）。

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：製品コンセプト
- 第 2 フェーズ：製品開発
- 第 3 フェーズ：製品生産および保守（「生産フェーズへのリリース」後）

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（2.2.4 節参照）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を実行する。

テスト計画作業の活動は、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[8]。

2.2.3 標準の構成とテスト固有のパート（K1）[10 分]

2.2.3.1 標準の設計と構成[情報提供]

ISO 26262 は、10 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）

パート 1 と 10 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[8]。

- 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、システム妥当性確認を含む）とソフトウェアレベル（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見なされる（パート 4、5、および 6）。

ISO 26262 のパート 8 は特別な位置付けであり、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、テスト担当者にとって重要な支援プロセスの要求を説明している[8]。

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（K2）[20 分]

2.2.4.1 ASIL のクリティカリティレベル

ASIL（Automotive Safety Integrity Level：自動車用安全度水準）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。これらの要求は、既存の品質マネジメント（QM）に適合することで対処される[8]。

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。「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 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[8]。

2.2.5 ISO 26262 の文脈における CTFL®からのコンテンツの適用（K3）[60 分]

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、および 8 で提供されている。プロセスや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、適用可能な技法や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：システムまたはその一部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、ソークテスト[17]）
- テスト環境（例：HiL、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。

前述の例で、以下の手法は、ASIL C の要求の実証のために導かれる。

- 手法 x：強く推奨される：ISO 26262 に従って開発する場合、通常適用される。
- 手法 y：推奨される：証跡として役立つ場合に適用される。

[17] パフォーマンステストの 1 種 [37]

- 手法 z1 と z2：少なくとも z1 は ASIL C でより高いレベルであるため、選択される必要がある。

ISO 26262 では、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。ただし、その場合、テスト担当者は代わりに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[8]。

2.3 AUTOSAR（K1）[15 分]

イントロダクション

AUTOSAR は「AUTomotive Open System ARchitecture」の頭字語で、開発パートナーシップで構成される。本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。このため、本標準はソフトウェアの重要性と複雑性の高まりに対処することを目的とする[9]。現在、AUTOSAR は 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 のプロダクトに取り組むことになる。また、テスト担当者は、AUTOSAR の目的、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 の目的（K1）[5 分]

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[9][15]。

1. ソフトウェアの移行性（移植性）をサポート
2. 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
3. 異なる機能ドメインをサポート
4. オープンアーキテクチャの定義、メンテナンス性、適応性、および拡張性
5. 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（機能的およびサイバーセキュリティ：「セーフティ&セキュリティ」） - 完全性、メンテナンス性によって特徴付けられる
6. 元来のリソースの持続可能な使用をサポート
7. 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
8. 車載電子制御ユニット（ECU）の基盤ソフトウェアの機能の標準化
9. 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（K1）[情報提供] [5 分]

AUTOSAR のアーキテクチャは次の 3 つの異なるレイヤーで構成される。

- ハードウェアから独立し、AUTOSAR ソフトウェアコンポーネント（SW-C）を含むレイヤー。
- 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レイヤー。
- AUTOSAR Runtime Environment（RTE）を含む抽象化レイヤー。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 と基盤ソフトウェア間のデータ交換を実装する。

他に特筆すべきは、整合性を維持して制御ユニットソフトウェアを開発するための AUTOSAR メソドロジーである。この点において、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（EcuC）」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 of System Configuration Description（EcuEx）」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（K1）[5 分]

AUTOSAR は特に次のテストレベル[18]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））。仮想 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誘発および測定できる[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、異なる電子制御ユニットの（もしくは車両内における）機能的な統合と接続を確認する。テスト担当者は、おそらく分散配置されていることにより不足している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[17]。

[18] テストレベル。2.4.2 も参照されたい。

2.4　比較（K2）[20 分]

2.4.1　ASPICE と ISO 26262 の目的（K1）[5 分]

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 2626[8]は、開発時に作りこんだソフトウェアのシステマチック故障、および運用時のハードウェア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[7]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　テストレベルの比較（K2）[15 分]

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[20]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®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。

表 2：テストレベルの対応付け

ISTQB®	ISO 26262	ASPICE 3.1
受け入れテスト	安全妥当性確認（4～9）[19]	同等のものなし
システムオブシステムズテスト[20]	アイテムの統合とテスト（4～8）[21]	システム適格性確認テスト（SYS.5）
システム統合テスト		システム統合テスト（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）	ソフトウェア適格性確認テスト（SWE.6）
コンポーネント統合テスト		ソフトウェア統合テスト（SWE.5）

[19] 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。
[20] 複数の異種分散システムのテスト[34][39]
[21] 「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。

コンポーネントテスト	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[20][49]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。一方、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（2.2.5 節および 2.2.4.2 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技法に関する推奨策をテスト担当者に提供している。

3　仮想環境でのテスト（K3）[160 分]

用語

モデルインザループ（MiL：Model in the Loop）、ソフトウェアインザループ（SiL：Software in the Loop）、ハードウェアインザループ（HiL：Hardware in the Loop）、オープンループシステム、クローズドループシステム、環境モデル（自動車）

学習の目的

AUTFL-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）

AUTFL-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）

AUTFL-3.1.3　クローズドループシステムとオープンループシステムの違いを想起する。（K2）

AUTFL-3.1.4　車載制御ユニット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）

AUTFL-3.2.1.1　MiL テスト環境の構成を想起する。（K1）

AUTFL-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）

AUTFL-3.2.2.1　SiL テスト環境の構成を想起する。（K1）

AUTFL-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）

AUTFL-3.2.3.1　HiL テスト環境の構成を想起する。（K1）

AUTFL-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）

AUTFL-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）

AUTFL-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）

AUTFL-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）

3.1　一般的なテスト環境（K2）[30 分]

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（K1）[5 分]

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、または製造された電子制御ユニット（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。テスト担当者は異なる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し、テストケースを実行できる[24]。

3.1.2 テスト環境の一般的な構成要素（K1）[5 分]

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、テストアイテムの入力を刺激し出力を観測できる（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 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（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（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（ネットワークへのアクセス、データロガー）
- ツール（オシロスコープ、計測器）
- 実験室（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構成要素である。モデルは、燃焼機関、トランスミッション、車両センサー、電子制御ユニット、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。テスト環境には、さまざまなアクセスポイントが存在する。テスト担当者はこれらのアクセスポイントを使用して、テストアイテムの測定と観測を行う[25]。

3.1.3 クローズドループとオープンループの違い（K2）[15 分]

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刺激し、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。テストが正常終了した場合、観測された振る舞いは、期待結果と一致する。

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。電子制御ユニット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 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、テストアイテム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、テストアイテムの動作原理に強く依存する。テストアイテムがリアクティブな振る舞いをしたり、状態機械にの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。

3.1.3.2 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）の刺激はテストアイテムの出力を考慮する。環境モデルは出力を収集し、テストアイテム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®）や横滑り防止装置（ESP®）などの運転支援システムをテストできる[27][30]。

3.1.4 電子制御ユニット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（K1）[5 分]

自動車環境の制御ユニット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。電子制御ユニット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さらなる情報を制御ユニットに提供する。これらの情報は、センサーや他の電子制御ユニットから提供される。これらの機器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、電子制御ユニット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：ASAM MCD2 D や「Open Diagnostic Data Exchange」）と通信プロトコル（ISO 14229 の「Unified Diagnostic Services」）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 標準に定義されている[30][31]。

3.2　XiL テスト環境でのテスト（K3）[130 分]

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[22]（PiL）。
- ハードウェアインザループ（HiL）。
- ビークルインザループ[23]（ViL）。

ここで、テスト担当者はテスト環境（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）（K2）[20 分]

3.2.1.1　MiL テスト環境の構成

MiL テスト環境では、テストアイテム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、これはテストアイテム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、テストアイテムを刺激および観測できる。アクセスポイントはテストアイテム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。テストアイテム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。（一般的な V 字モデルに類する）開発においては、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、テストアイテム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。[24]

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（V 字モデルの左側）からモデルの機能をテストできる。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ことを認識する必要がある。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

[22] 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。
[23] 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。
[24] これは、他のすべての XiL 環境にも当てはまる。

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。

3.2.2 ソフトウェアインザループ（SiL）（K1）[10 分]

3.2.2.1 SiL テスト環境の構成

テストアイテムは特定の 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向けのソフトウェアツールを使用して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のみ使用できる。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、そのマシンコードに対して特別なアクセスインターフェー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刺激および観測できる。ラッパーはテストアイテム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。テストアイテム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、テストアイテムをラップし、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）のような技法を使用して、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実行を一時停止し、詳細な分析と評価を実行できる。機能テスト、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、性能テストおよび信頼性テストは一般的ではない。これらのソフトウェア特性は、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）（K2）[20 分]

3.2.3.1 HiL テスト環境の構成

テストアイテム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
- 信号のタイプの処理と信号の振幅の処理
- 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 節も参照）
- 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シミュレーション

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、テストアイテム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アイテムを分析できる。テストアイテムが単一の電子制御ユニット（ECU）の場合、コンポーネント[25]HiL と呼ぶ。テストアイテムが複数の電子制御ユニット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、電子制御ユニット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（K3）[80 分]

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

[25] 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。

判断基準	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	低	低～中	高
	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	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）	統合されたシステムを実行可能
デバッグにかかる時間と工数	低	中	高
	テストアイテムのモデルで見つかった欠陥（モデル修正）	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）	システムレベルで見つかった欠陥（システム修正）
実装とメンテナンスにかかる工数	低	中	高
	環境モデルの作成	環境モデルとラッパーの作成	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
テストの準備にかかる工数	低	中	高
	環境は迅速にセットアップ可能	環境は迅速にセットアップ可能	テスト環境の設計、実装、評価に多大な工数が必要
テストアイテムの完成度	低	中	高
	システムモデルがシミュレーションされている	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	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
テストベース（仕様）に必要な完全性	中	中～高	高
	完全な仕様がない場合でも、モデルをテストでき、また仕様を部分的であっても確認できる	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）	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）
テストアイテムへのアクセス	高	中	低
	モデル内の全信号の観測と制御が可能	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	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

3.2.4.2　1 つ以上のテスト環境へのテストケースの割り当て

次の表は、テスト目的をさらに詳細に説明し、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

テストタイプ	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（パラメーターセットやバリアントコーディングなど）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	テストアイテムの内部/外部インターフェースの確認	o	+	+
使用性の確認	観測対象のテストアイテムは要求通りかつユーザーの期待通りに使用できる。	-	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。「フロントローディング」[26]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、効率性と性能、および使用性の証跡とは別に、全テストタイプは、テストアイテム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、テスト担当者は（テストマネージャーの役割で）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　一般的な V 字モデルでの XiL テスト環境（MiL、SiL、HiL）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。テストアイテムと 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、テストアイテム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[24]。

[26] 欠陥は、早く検出されればされるほどよい

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テスト）

テストコストの最小化

- 全テストタイプを、適切なテストレベル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　自動車ドメイン固有の静的および動的テスト技法[230 分]

用語

コーディング標準、バックツーバックテスト

学習の目的

静的テスト技法

AUTFL-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）

AUTFL-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）

動的テスト技法

AUTFL-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）

AUTFL-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）

AUTFL-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）

AUTFL-4.2.4　要件ベースドテストの原則を想起する。（K1）

AUTFL-4.2.5　適切かつ必須のテスト設計技法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）

4.1　静的テスト技法（K3）[75 分]

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、担当者による評価（レビュー）と、ツールによる静的解析が含まれる。

4.1.1　MISRA-C：2012 ガイドライン（K2）[15 分]

開発者がコーディングガイドラインにの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[27]向けにコーディングガイドラインの使用を推奨している。コーディング標準は、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：2012 ガイドライン[12]は、プログラミング言語 C 向けのガイドラインを含み、2 つのタイプのガイドラインを定義している。

- ルール：全般的に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。

[27] [ISO 26262：2011]パート 9 表 6 も参照されたい。

- 指針：静的解析ツールでは完全には検証できないもの。この理由は、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ためである。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。

- 「推奨」ガイドライン：極力従う必要がある。従わない場合も違反を文書化すべきである。
- 「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（K3）[60 分]

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、テストアイテムの基となる仕様をチェックする必要がある[20]。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/IEC/IEEE 29148：2011[38]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/IEC/IEEE 29148：2011 の要件特性

個々の要件および要件の集合の特性：

- 検証可能である：各要件は静的テストまたは動的テストにより検証できる。
- 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：各要件はそれ自体および他の要件に対して一貫している。
- 完全である：各要件はすべてのケース（エラーシナリオ、異常終了シナリオ、例外シナリオを含む）を考慮している。同時に、すべての表と図はラベル付けされており、使用する略語と用語は定義されている。
- 追跡可能である：各要件は（例えば ID によって）明確に識別されている。これにより、影響度分析が可能となり、テストケースによるカバレッジが可視化される。
- （要件の集合に対して）定義範囲がはっきりしている：開発のスコープが明確であり、したがってテスト対象も明確である。
- 単独である：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。

テスト担当者はレビュー用のツールの 1 つとして、要件の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。これらのレビューチェックリストには、前述の特性に沿った質問を含む。テスト担当者は、最大限の知識と信念に基づきそれらの質問に答える必要がある。以下に、各要件に対してテスト担当者が答える必要があると考えられる質問の一部を示す。

- 検証可能である：要件は対応するテストレベルにおいて静的テストまたは動的テストで検証できるか。

- 曖昧性がない：要件が複数の意味に解釈されることはないか。また、暗黙的な知識または経験知識に基づいて構築されていないか。
- 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か。
- 単独である：要件はさらに部分的な要件に分割されないか。例えば、要件内にある if-then-else のような論理的つながりを分解することで、複数の部分要件として表現できるか。

Hobbs 著の『Embedded Software Development for Safety-Critical Systems』[31]によると、要件は、実現可能であり、実現手段から独立している必要がある。テスト担当者にとってこれらの特性を評価することはほとんどの場合困難であるが、これらの特性はテスト設計に少なからず影響を及ぼす。

4.2 動的テスト技法（K3）[155 分]

4.2.1 条件テスト、複合条件テスト、MC/DC テスト（K3）[60 分]

これらの技法は、ホワイトボックステスト設計技法の一部である（詳細については、シラバス CTAL-TTA も参照されたい）。テスト担当者は、テストケースをテストアイテムの構造（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（[20]を参照）ではコード内の判定のカバレッジに着目してテストケースを設計するのに対し、条件テストでは判定内の個々の条件に着目する。このため、これらの技法では、判定を行う*方法*に重点を置く。各判定は、1 つ以上の「不可分条件」で構成される。テスト担当者がテストケースを実行する場合、これらの各条件は「真」または「偽」の値となる。判定の全体的な値は、これらの個々の値の論理的な組み合わせから導かれる[37]。

判定が一つの条件だけで構成される場合、これらの技法はデシジョンテストと同一視できる。そうでない場合、これらの技法は以下の点で異なる[37]：

- （単純な）条件テスト（表 5 の技法 A）：テスト担当者は、個々の条件の真/偽の結果をカバー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（単純な）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カバー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「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 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 は例を使用して、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	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、技法の限界を示している。（単純な）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。テストケースを適切に選択することにより、この状況を改善できる（TC 3 と TC 4 を選択した場合）。

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（K2）[15 分]

バックツーバックテスト（別名：比較テスト[32]）は、テスト（設計）技法というよりはむしろテストのアプローチである。このテストでは、テストアイテムの 2 つ以上のバリアントを比較する。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、結果を比較する。結果が同じであれば、テストは合格となる。結果が異なる場合、検出された差異の原因を分析する。

テストアイテムは、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、テストアイテムは比較可能な振る舞いを示す。要件はテスト設計のテストベースとして機能しない。一方、バックツーバックテストでは、テスト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。したがって、このテストは要件ベースドテストの代わりとはならない。

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アイテムは、同じソフトウェアの異なるバージョンである。この場合、例えば、古いバージョンのテストアイテムはバックツーバックテストのテストオラクルとして機能する（リグレッションテストと同じ）[33]。別のケースは、（手動または自動で）生成されたコードの比較である[32]。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テストオラクルとしても機能する[34]。したがって、この技法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（K2）[15 分]

フォールトインジェクションテストは、特別なテスト（設計）技法というよりはむしろロバストネステストのためのアプローチである。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。テスト担当者はこれらの機能をテストするために、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をシステムが安全に検出する必要がある場合。
- インターフェースの欠陥：例えば、短絡や信号途絶によりシステムが機能不全にならないようにする必要がある場合。
- ソフトウェアの欠陥：システムが内部欠陥を検出して処理する必要がある場合。

従来のフォールトインジェクションでは、テスト担当者は実際のコンポーネントを操作して欠陥を注入する。

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、実行時にテスト担当者がシミュレーションできる。フォールトインジェクションテストは、通常、HiL テスト環境で実行する。ここでは、フォールトインジェクションユニット（FIU）[18]を物理欠陥のドライバーとして使用する。これらの欠陥の中でも、特定の回路の短絡や開放を重要視する。ソフトウェアのインターフェース欠陥のシミュレーションは、多くの場合、SiL テスト環境で既に行われている。

ソフトウェアの欠陥は、例えばデバッガーや XCP を使用して開発環境でのみ注入できることが多い。したがって、ほとんどの場合、実行には非常に多くの時間がかかる。

4.2.4 要件ベースドテスト（K1）[5 分]

要件ベースドテストは、テスト（設計）技法というよりはむしろテスト向けのアプローチ（プラクティス）である[21]。この手法の目的は、テストケースを使用して要件をカバーすることである。したがって、テスト担当者はテストアイテムが要件を満たしていることを判定する。

この手法では、テスト担当者は要件の分析、テスト条件の導出、テストケースの設計、およびテストケースの実行を行う。その後、テスト結果を分析して、テストを洗練する。この手順を行うことで、さらに別のテストを作成することもできる。また、テスト担当者は別のテストプラクティス（経験ベースのテスト）を適用する。こうすることで、例えば探索的テストの形態でリグレッションテストを実行して、欠陥のリスクを軽減できる。

要件が不完全であるか一貫性がない場合、その要件に基づいて設計されたテストは同じ問題を抱えることになる。一方、要件が詳細すぎる場合、テスト担当者はすべての要件をテストできない可能性がある。このような場合、テストケースに優先順位を割り当てる必要がある[3]。

4.2.5 状況に依存したテスト技法の選択（K3）[60 分]

ISO 26262 標準（パート 6）は、テスト担当者が ASIL レベルに応じてテスト設計技法（2.2 節参照）を適用するよう提案している。これらの技法には、特に CTFL®や本シラバスの 4.2 節で言及されている技法が含まれる。

- 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
- 改良条件判定テスト
- エラー推測

- フォールトインジェクションテスト
- 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

技法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。ISO 26262 標準は、ASIL レベルに応じて適用できる技法も提案している。標準の推奨から逸脱する技法については、ISO 26262 に関する 2.2 節の説明を参照されたい。

テストベース

テストベースは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クラスを構築できる。テスト担当者はそれらの値を適切な同値クラス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、技法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

技法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、テストアイテムが利用可能で観察できる必要がある。例えば、センサーの同値クラスのテストは、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設計技法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。

テスト技法の選択例

以下の表は、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設計技法を選択した例を示している。

表 6：テスト技法の選択例

	テスト設計技法	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						

付録

車両データベースおよび通信プロトコル

表 7：自動車業界で一般的な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	ASAM MCD-2 MC （および ASAP2 または A2L）	ASAM MCD-1 XCP （Universal Measurement and Calibration Protocol） ASAM standard CCP （CAN Calibration Protocol）
バス	ASAM MCD2 NET 標準 （および *FIBEX（Field Bus Exchange Format）*）	FlexRay （ISO 17458） CAN （ISO 11898-2 準拠の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 準拠のコントローラーエリアネットワーク）
診断	ASAM MCD2 D （および *ODX*） CDD （CANdelaStudio 診断記述）	*KWP2000（ISO 14230）* *ISO-OBD（ISO 15031）* UDS *（ISO 14229）*

AUTOSAR は、完成車のデータベースを統合できるように XML 形式で標準化した。これは ARXML 形式（AUTOSAR Integrated Master Table of Application Interfaces、XML スキーム R3.0）と呼ばれる。

ASAM は、「Association for Standardization of Automation and Measuring Systems」の頭字語である。

表索引

参考文献

[1] International Organization for Standardization (ISO), International Electrotechnical Commission (IEC), *ISO/IEC TS 24748-1:2016 Systems and software engineering - Life cycle management - Part 1: Guide for life cycle management,* 2016.

[2] Verband der Automobilindustrie e.V. (VDA) / QMC Working Group 13 / Automotive SIG, *Automotive SPICE Process Assessment Model,* Berlin: Verband der Automobilindustrie e. V. (VDA), 2008.
JSTQB 訳注) 日本語版は
"http://www.automotivespice.com/download/,"
において発行されている。

[3] AUTOSAR, „http://www.autosar.org/specifications/,“ [Online]. [Zugriff am 04 04 2016].

[4] ZVEI, *Best Practice Guideline - Software Release,* Frankfurt am Main,: ZVEI, 2016.

[5] International Software Testing Qualifications Board (ISTQB) / German Testing Board e.V. (GTB), *ISTQB/GTB Certified Tester Advanced Level (CTAL) Syllabus - Technical Test Analyst (TTA) - Deutsche Ausgabe,* German Testing Board e.V. (GTB), 2012.
JSTQB 訳注） 日本語版は
"http://jstqb.jp/dl/JSTQB-Syllabus.Advanced_TTA_Version2012.J02.pdf,"
において発行されている。

[6] Verband der Automobilindustrie e.V. (VDA) / QMC Working Group 13, „Status and outlook VDA QMC working group 13 - Automotive SPICE 3.0, Blue-Gold Volume,“ in *Sixth VDA Automotive SYS Conference*, Berlin, 2016.
JSTQB 訳注） 日本では「VDA Automotive SPICE ガイドライン」(QMC ジャパン株式会社, 2017 年)として発行されている。

[7] Verband der Automobilindustrie e.V. (VDA) / QMC Working Group 13 / Automotive SIG, *Automotive SPICE Process Assessment / Reference Model,* http://www.automotivespice.com/download/, 2015 Version 3.0.
JSTQB 訳注）日本語版は
"http://www.automotivespice.com/download/,"
において発行されている。

[8] International Organization for Standardization (ISO), *ISO 26262:2011 Road Vehicles - Functional Safety,* Genf, 2011.

[9] AUTOSAR, „Glossary AUTOSAR Release 4.2.2,“ [Online]. Available: http://www.autosar.org/fileadmin/files/releases/4-2/main/auxiliary/AUTOSAR_TR_Glossary.pdf. [Zugriff am 03 03 2016].

[10] H. Wallentowitz, Handbuch Kraftfahrzeugelektronik : Grundlagen, Komponenten, Systeme, Anwendungen ; mit zahlreichen Tabellen, Wiesbaden: Vieweg, 2016.

[11] K. Borgeest, *Elektronik in der Fahrzeugtechnik,* Springer Vieweg, 2014.

[12] MISRA Electrical Group MIRA Ltd., *MISRA-C:2012-Programmierrichtlinien – Version 3,* UK, Warwickshire, 2013.
JSTQB 訳注）日本では"MISRA-C:2004-Programmierrichtlinien --Version 2"が「自動車用 C 言語利用のガイドライン（第 2 版）」(自動車技術会, 2006 年)として発行されている。

[13] 754-2008 - IEEE Standard for Floating-Point Arithmetic, *754-2008 - IEEE Standard for Floating-Point Arithmetic,* 2008.

[14] International Organization for Standardization (ISO), International Electrotechnical Commission (IEC), Institute of Electrical and Electronics Engineers (IEEE), *ISO/IEC/IEEE 15288:2015 Systems and software engineering - System life cycle processes,* 2015-15-05.

[15] Verband der Automobilindustrie e.V. (VDA), *Entwicklung softwarebestimmter Systeme - Forderungen an Prozesse und Produkte,* Bd. 13, Verband der Automobilindustrie e.V. (VDA), 2004.

[16] Measuring, Association for Standardization of Automation and, „http://asam.net/,“ 2016. [Online]. [Zugriff am 2016].

[17] K. Hoermann, M. Mueller, L. Dittmann und J. Zimmer, Automotive SPICE in Practice in der Praxis – Interpretationshilfe für Anwender und Assessoren, Heidelberg: dpunkt verlag GmbH, 2. Auflage, 2016.

[18] National Instruments Germany GmbH, „Einsatz von Fault Insertion Units (FIUs) für die Überprüfung elektronischer Steuergeräte,“ Nr. 25. Juni, 2015.

[19] Patzer und Zaiser, „Einsatzgebiete für XCP,“ in *XCP-Das Standardprotokoll für die Steuergeräte Entwicklung*, Stuttgart, Vector Informatik GmbH, 2014.

[20] International Software Testing Qualifications Board (ISTQB) / German Testing Board e.V. (GTB), *ISTQB/GTB Certified Tester Foundation Level (CTFL) Syllabus - Version 2011 1.0.1 - Deutsche Ausgabe,* German Testing Board e.V. (GTB), 2011.
JSTQB 訳注） 日本語版は
"http://jstqb.jp/dl/JSTQB-SyllabusFoundation_Version2011.J02.pdf,"
において発行されている。

[21] International Organization for Standardization (ISO), International Electrotechnical Commission (IEC), Institute of Electrical and Electronics Engineers (IEEE), *ISO/IEC/IEEE 29119-1:2013 Software and systems engineering - Software testing - Part 1: Concepts and definitions,* 2013-09-01.

[22] International Organization for Standardization (ISO), International Electrotechnical Commission (IEC), *ISO/IEC 12207:2008 Systems and software engineering - Software life cycle processes,* International Organization for Standardization (ISO), 2008-02-01.

[23] A. Spillner, T. Roßner, M. Winter und T. Linz, Praxiswissen Softwaretest Testmanagement: Aus- und Weiterbildung zum Certified Tester - Advanced Level nach ISTQB-Standard, Heidelberg: dpunkt.verlag, 2008.

[24] Verband der Automobilindustrie e.V. (VDA), *Sicherung der Qualität in der Prozesslandschaft,* Bd. Band 4, Verband der Automobilindustrie e.V. (VDA), 2011.

[25] dpa, „www.motor-talk.de," 24 02 2015. [Online]. Available: http://www.motor-talk.de/news/die-zahl-der-modelle-waechst-der-absatz-nicht-t5219608.html. [Zugriff am 12 12 2016].

[26] AUTOSAR, „Requirements on Acceptance Test AUTOSAR TC Release 1.1.0," [Online]. Available: http://www.autosar.org/fileadmin/files/standards/tests/tc-1-1/general_auxiliary/AUTOSAR_ATR_Requirements.pdf. [Zugriff am 2016 12 12].

[27] R. Schönfeld, Regelungen und Steuerungen in der Elektrotechnik, Verlag Technik GmbH, 1993.

[28] AUTOSAR, „Project Objectives AUTOSAR Release 4.2.1," [Online]. Available: http://www.autosar.org/fileadmin/files/releases/4-2/main/auxiliary/AUTOSAR_RS_ProjectObjectives.pdf. [Zugriff am 03 03 2016].

[29] AUTOSAR, „Main Requirements AUTOSAR Release 4.2.1," [Online]. Available: http://www.autosar.org/fileadmin/files/releases/4-2/main/auxiliary/AUTOSAR_RS_Main.pdf. [Zugriff am 03 03 2016].

[30] G. Baumann, „Was verstehen wir unter Test？ Abstraktionsebenen, Begriffe und Definitionen," FKFS 1. AutoTest; Fachkonferenz zum Thema Test und Diagnose in der Automobilentwicklung., Stuttgart, 2006.

[31] C. Hobbs, Embedded Software Development for Safety-Critical Systems, Taylor & Francis Group, 2016.

[32] AUTOSAR, „AUTOSAR - The worldwide Automotive Standard for E/E systems,“ *ATZ extra,* p. 5, 2013.

[33] A. Spillner und T. Linz, Basiswissen Softwaretest [Elektronische Ressource] : Aus- und Weiterbildung zum Certified Tester - Foundation Level nach ISTQB-Standard, Heidelberg: dpunkt.verlag, 2012.

[34] International Software Testing Qualifications Board (ISTQB) / German Testing Board e.V. (GTB), *ISTQB/GTB Standardglossar der Testbegriffe Version 3.1,* Erlangen: German Testing Board e.V. (GTB), 13. April 2016.

[35] International Organization for Standardization (ISO), International Electrotechnical Commission (IEC), Institute of Electrical and Electronics Engineers (IEEE), *ISO/IEC/IEEE 29119-3:2013 Software and systems engineering - Software testing - Part 3: Test documentation,* 2013-09-01.

[36] AUTOSAR, „Acceptance Test Main Requirements AUTOSAR TC Release 1.1.0,“ [Online]. Available: http://www.autosar.org/fileadmin/files/releases/tc-1-1/general_auxiliary/AUTOSAR_ATR_Main.pdf. [Zugriff am 2016 03 03].

[37] International Organization for Standardization (ISO), International Electrotechnical Commission (IEC), Institute of Electrical and Electronics Engineers (IEEE), *ISO/IEC/IEEE 29119-4:2015 Software and systems engineering - Software testing - Part 4: Test techniques,* Bd. 4, 2015.

[38] International Organization for Standardization (ISO), International Electrotechnical Commission (IEC), Institute of Electrical and Electronics Engineers (IEEE), *ISO/IEC/IEEE 29148:2011 - Systems and software engineering - Life cycle processes - Requirements engineering,* 2011-12-01.

[39] M. Winter, M. Ekssir-Monfared, H. M. Sneed, R. Seidl und L. Borner, Der Integrationstest: Von Entwurf und Architektur zur Komponenten- und Systemintegration, München: Carl Hanser Verlag GmbH & Co. KG, 2012.

[40] International Organization for Standardization (ISO), International Electrotechnical Commission (IEC), *ISO/IEC 33020-03:2015 Informationstechnik – Prozessbewertung – Rahmenwerk für Prozessmessungen zur Beurteilung der Prozessfähigkeit,* 01-03-2015.

[41] M. Conrad und G. Sandmann, „A Verification and Validation Worflow for IEC 61508 Applications,“ *SAE International,* 2009.

[42] H.-W. Wiesbrock, M. Conrad, I. Fey und H. Pohlheim, „Ein neues automatisiertes Auswertverfahren für Regressions- und Back-to-Back-Tests eingebetteter Regelsysteme,“ *Softwaretechnik-Trends,* Bd. 22, 2002.

[43] U. Freund, V. Jaikamal und J. Löchner, „Multilevel System Integration of Automotive ECUs based on AUTOSAR,“ [Online]. Available: http://papers.sae.org/2009-01-0918/. [Zugriff am 27 09 2016].

[44] T. Ringler, C. Dziobek und F. Wohlgemuth, „Tagungsband Modellbasierte Entwicklung eingebetteter Systeme - Chancen und Herausforderungen bei der virtuellen Absicherung verteilter Body&Comfort-Funktionen auf Basis von AUTOSAR - S.83 - 93,“ [Online]. Available: https://www.in.tu-clausthal.de/fileadmin/homes/GI/Documents/MBEES15Proceedings.pdf. [Zugriff am 27 09 2016].

[45] Bakshi, U.A.; Baksi; V.U.: Control Systems, Edition 2010. （英語版）

[46] International Organization for Standardization (ISO), International Electrotechnical Commission (IEC), ISO/IEC 2382:2015-05 Information technology - Vocabulary, 2015-05.

[47] International Organization for Standardization (ISO), International Electrotechnical Commission (IEC), ISO/IEC 24765:20170-09 Information technology - Vocabulary, 2017-09.

[48] German Association of the Automotive Industry (VDA) / QMC Working Group 13 / Automotive SIG, Automotive SPICE Process Assessment / Reference Model, http://www.automotivespice.com/download/, 2017, Version 3.1.

JSTQB 訳注） 日本語版は
"http://www.automotivespice.com/download/,"
において発行されている。

[49] International Software Testing Qualifications Board (ISTQB®) – ISTQB® Certified Tester Foundation Level (CTFL®) Syllabus - Version 2018.

JSTQB 訳注） 日本語版は
"http://jstqb.jp/dl/JSTQB-SyllabusFoundation_Overview__Version2018.J01.pdf,"
において発行されている。

定義

ISTQB®用語集(https://glossary.istqb.org/en/search)の用語（太字）以外に、シラバス固有の用語（太字ではない）が使用されている。これらの用語はここに定義されている意味で使用される。

用語	定義/意味	用語集の重要な用語	参考文献
Automotive Open System Architecture（AUTOSAR）	2003 年に創設された開発パートナーシップ。自動車業界のソフトウェアアーキテクチャ用のオープンな業界標準の策定と確立を目的とする。		
Automotive SPICE	**自動車業界のプロセス用のプロセス参照モデルおよび関連するプロセスアセスメントモデル。ISO/IEC 33002：2015 の要件に準拠する。**	**X**	[2]
E/E システム	電気または電子要素からなる、機能性を持ったシステム。		[7]
ECU Configuration Description	電子制御ユニットの SW-C を統合するためのデータ。		[3]
ECU extract	System Configuration Description から抽出した電子制御ユニット用データ。		[3]
Original equipment manufacturer（OEM）	自動車業界では、この用語は完成車メーカーを意味する。「Tier1… n」も参照のこと。		[2]
Runtime Enviroment（RTE） （AUTOSAR）	制御ユニット内外の AUTOSAR ソフトウェアコンポーネント間、およびアプリケーションと基盤ソフトウェア（BSW）間のデータ交換を制御および実装する抽象化レイヤー。		[3]
ソフトウェアコンポーネント（SW-C） （AUTOSAR）	ハードウェアに依存しないソフトウェアレイヤーで、個々のアプリケーションや機能を含む。		[3]
System Configuration Description	車両内のすべての電子制御ユニットの統合に使用されるデータ。		[3]
Tier1...n	サプライチェーンのさまざまなレベルのサプライヤーは、Tier1…n と呼ばれる。OEM と直接取引するサプライヤーは Tier1、Tier1 のサプライヤーは Tier2 と呼ばれる。		[2]

用語	定義/意味	用語集の重要な用語	参考文献
XiL テスト環境	**さまざまな仮想テスト環境の動的テストに対する総称。** **「ハードウェアインザループ」、「ソフトウェアインザループ」、「モデルインザループ」も参照のこと。**	**X**	
安全ライフサイクル	安全性に関連するシステムのプロダクトライフサイクル。プロダクトのアイデアで始まり、ライフサイクルの終了時にプロダクトを廃棄することで終わる。		[8]
安全文化	機能的に安全なプロダクトを共通して開発するという会社全体の姿勢。		[8]
インストール推奨	SW リリース時に添付されるもので、サプライヤーはこれを使用して、リリースアイテムが公道でのリリースに関して何ら制限のないことと、公道で使用/テストできることを OEM に宣言する。		
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に依存しないシステム。** **クローズドループシステムも参照のこと。**	**X**	[44]
改良条件判定テスト（MC/DC テスト）	『ISTQB®用語集』を参照のこと。		
環境モデル（車両）	**リアルタイムシミュレーションにおいて、コンポーネントまたは他のコンポーネントを含むシステム、車両プロセス、環境条件をシミュレーションするための実環境を抽象化したもの。**	**X**	[7]
基盤ソフトウェア	（AUTOSAR）：ハードウェア指向の標準化されたソフトウェアコンポーネント。		[9]
機能安全	電気/電子（E/E）システムの誤動作によって危険な状況が引き起こされるという不合理なリスクが存在しないこと。	X	[8]
機能リスト	リリース向けに実装する機能は、リリース計画時に指定し、機能リストに記述する。		[4]
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。** **オープンループシステムも参照のこと。**	**X**	[44]

用語	定義/意味	用語集の重要な用語	参考文献
欠陥リスト	修正済みまたは未修正の欠陥のリスト。通常、テストレポートに含まれる。		[4]
フォールト生成ユニット	テスト環境を構成する要素の 1 つで、コンポーネントまたはシステムのインターフェースのフォールトをシミュレーションできる。		
検証基準	検証基準は、テストアイテムを適切に検証するために満たす必要のある質的および量的基準を定義する。		[7]
検証戦略	アイテムを検証するためのハイレベル計画。これには、検証基準、関連する手法、技法、ツールを用いた検証活動、および検証対象の作業成果物やプロセスが含まれる。		[7]
検証用の基準	ソフトウェア検証用の一連のテストケースおよび合格/不合格基準。		[7]
固定小数点	桁数が固定されている数値。小数点の位置が固定されている。		
コーディング標準	**データまたはプログラムコンポーネントの設計または設計記述の特性を定義する標準。**	**X**	[46]
コードレビュー	計画されている目的に対するコードの適合性チェック、および提供されている仕様および標準の逸脱分析。		[14]
コンポーネント HiL	単一の電子制御ユニット（ECU）のイメージ用のテスト環境。		[7]
サイバーセキュリティ（車両）	電子犯罪から安全に保たれている状態、およびこれを達成するために実施される手段。		TBD
参照プロセス	『ISTQB®用語集』を参照のこと。		
指針（MISRA）	MISRA-C：2012 のプログラミングガイドラインであり、静的解析ツールによって完全には検証されない。		[12]
システム HiL	車両全体を範囲とした電子制御ユニットグループのイメージ用のテスト環境。		[7]
システムオブシステムズテスト	システムオブシステムズが、定められた要件を満たすことを検証するためのテスト。		
システムライフサイクル	PEP を含み廃棄までのシステムの開発と実装のさまざまなフェーズ。		[1][14][15]

用語	定義/意味	用語集の重要な用語	参考文献
システム適格性確認テスト（ASPICE）	**ソフトウェアコンポーネント、ハードウェアコンポーネント、およびメカニクスから成る完成済みの統合されたシステムで実施するテスト。システム要件に準拠しており、完成済みシステムをデリバリーする準備が整っていることの証拠を提供する。**	**X**	[7]
システム統合テスト（ASPICE）	システムのアーキテクチャ設計に対するテスト。統合されたシステムアイテムが、システムアイテム間のインターフェースを含むシステムのアーキテクチャ設計に準拠していることの証拠を提供する。		[7]
自動車用安全度水準（Automotive Safety Integrity Level）	**ISO 26262 のアイテムまたは要素の必要要件と、不合理な残余リスクを回避するための安全方策を、4 つのレベルで指定する。D が最も厳しく、A が最も緩い。**	**X**	[8]
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向けに有効な時間枠。		[7]
手法テーブル（自動車）	**自動車用安全度水準（ASIL）およびテスト対象の状況に応じて必要とされるさまざまなテストアプローチ、テスト技法、およびテストタイプを記載するテーブル。**	**X**	[8]
条件カバレッジ	『ISTQB®用語集』を参照のこと。		
条件テスト	『ISTQB®用語集』を参照のこと。		
生産	開発済みのプロダクトを量産すること。 車両環境の PEP では、製造/量産とも呼ばれる。		[1][15]
ソークテスト	ソークテストは実運用に基づくテストに類似しているが、大きなサンプルサイズに対して、通常のユーザーをテスト担当者として、事前に定義したテストシナリオに限定されず実施するテストである。また、ソークテストは実際の利用環境に基づいて実施される。テスト担当者の安全性を確保する必要がある場合（安全対策の追加やアクチュエーターの無効化など）、これらのテストには制限が加えられることがある。		[37]
ソフトウェアインザループ（SiL）	**シミュレーションされた環境で、または実験的なハードウェアと統合して、実際のソフトウェアを使用して実施する動的テスト。**	**X**	[4]

用語	定義/意味	用語集の重要な用語	参考文献
ソフトウェア適格性確認テスト（ASPICE）	**完成済みの統合されたソフトウェアで実施するテスト。ソフトウェア要件に準拠していることの証拠を提供する。**	**X**	[7]
テストアイテム	『ISTQB®用語集』を参照のこと。 車両の世界におけるテストアイテムは、基本的なパラメーター化などのソフトウェア構成を含む。また、ほとんどの場合、ハードウェアとメカニクスも含む[6]。		[4]
テスト戦略	『ISTQB®用語集』を参照のこと。		
テスト文書	システムまたはコンポーネントのテストの計画、または結果を記述する文書[ISO/IEC/IEEE 24765]。		[47]
電気エラーシミュレーション	「フォールト生成ユニット」を参照のこと。		
トレーサビリティ	『ISTQB®用語集』を参照のこと。		
能力レベル	1 つ以上のプロセス属性。これらを十分に達成すると、プロセス能力が大幅に向上する。		[7]
能力座標	能力レベルを構成する多くのプロセス属性で定義される。プロセス属性は、プロセスの能力を測定可能にする特性を提供する。		[7]
能力指標	プロセス能力アセスメントの実行と説明に使用できる指標。		[7]
ハードウェアインザループ（HiL）	**統合されたソフトウェアと実際のハードウェアを使用して、シミュレーションされた環境で実施する動的テスト。**	**X**	[4]
バスシステム	同一の信号線を介して情報を交換する複数の電子制御ユニットのネットワーク。		[10]
バックツーバックテスト	**テストアイテムの、または同じテストアイテムのシミュレーションモデルの 2 つ以上のバリアントを比較するテスト。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、結果を比較する。** **比較テストも参照のこと。**	**X**	[32]

用語	定義/意味	用語集の重要な用語	参考文献
フォールトインジェクション	『ISTQB®用語集』を参照のこと。		
複合条件テスト	『ISTQB®用語集』を参照のこと。		
不在バスシミュレーション	存在しない電子コントロールユニットのバス通信インターフェースの仮想化。		
浮動小数点	実際の値を近似した値。		[13]
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。		[7]
プロセスモデル	『ISTQB®用語集』を参照のこと。		
プロセス改善	『ISTQB®用語集』を参照のこと。		
プロセス座標	すべての関連プロセスは、まずプロセスカテゴリーで、次にプロセス群で定義されて組み合わされる。		[11]
プロセス属性	プロセスの能力をアセスメントするためのプロセスの測定可能な特性。		[11]
プロダクトライフサイクル	「システムライフサイクル」を参照のこと。		
プロダクト開発プロセス	プロダクトのアイデアを初めて考えたときから生産に至るまでのすべての活動を含むプロセス。		[14]
モデルインザループ（MiL）	**システムのシミュレーションモデルを使用して、シミュレーション環境で実施する動的テスト。**	**X**	[4]
リアルタイム	コンピューターシステムの動作の 1 つの種類。データを処理するプログラムは、処理結果が事前定義された時間内に利用可能になるように、常に準備を整えて待機している。アプリケーションに応じて、データは、時間的にランダムに送信されることも、事前定義された時間に送信されることもある。		[45]
リアルタイム対応コンピューター	定義された時間枠内に信号処理を必ず完了することができるコンピューティング ユニット。		[8]

用語	定義/意味	用語集の重要な用語	参考文献
リグレッションテスト戦略	リグレッションテスト戦略は、テストアイテムの変更が発生した際に、どんな基準を使用してリグレッションテストケースを選択するかを定義する。		
リリース	リリースアイテムに実装された機能、プロパティ、および意図される使用に関するステートメント。		[14]
リリースアイテム	規定された機能、プロパティ、および目的を持つ一義的に識別可能な要素		[6]
リリースプロセス	リリースに向けて進行するプロセス。		[4]
リリース推奨	テスト結果に基づいてリリースアイテムをリリースする（またはリリースしない）ための、テスト担当者またはテストマネージャーによる推奨。		[4]
リリース目的	リリースアイテムの使用目的。		[4]
ルール（MISRA）	MISRA-C：2012 のプログラミングガイドラインで、静的解析ツールによって検証できるもの。		[12]

略記

本シラバスでは、以下の略記を使用する。

略記	定義/意味	参考資料
ACQ	Acquisition（調達）（ASPICE）	[7]
ASIL	Automotive Safety Integrity Level（自動車用安全度水準）	[8]
ASAM	Association for Standardisation of Automation and Measuring Systems	[16]
ASPICE	Automotive SPICE	
AUTOSAR	Automotive Open System Architecture	[9]
AUTOSIG	Automotive Specific Interest Group	[17]
BP	Base Practice（基本プラクティス）	[7]
BSW	Basic Software（基盤ソフトウェア）	[9]
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	Electronical 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）	[17]
EOP	End-of-Production（量産終了）	
FIU	Fault Insertion Unit（フォールト生成ユニット）	[18]
GP	Generic Practice（共通プラクティス）	[7]
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）	
ISTQB®	International Software Testing Qualifications Board	
MAN	Management（マネジメント）（ASPICE）	[7]
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）	[7]
PEP	Product Evolution Process（プロダクト進化プロセス）	[14]
PIM	Process Improvement（プロセス改善）（ASPICE）	[7]
QM	Quality Management（品質マネジメント）	
REU	Reuse（再利用）（ASPICE）	[7]
RTE	Runtime Environment（実行環境）	[9]
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	[7]
SPL	Supply（供給）（ASPICE）	[7]
SUP	Support（サポート）（ASPICE）	[7]
SW	Software（ソフトウェア）	
SW-C	Software Component（ソフトウェアコンポーネント）	[9]
SWE	Software Engineering（ソフトウェアエンジニアリング）（ASPICE）	[7]
SYS	System Engineering（システムエンジニアリング）（ASPICE）	[7]
VDA	Verband der Automobilindustrie（ドイツ自動車工業会）	
WP	Work Product（作業成果物）	[7]
XCP	Universal Measurement and Calibration Protocol	[19]
XiL	MiL、SiL、HiL などさまざまな仮想テスト環境の総称	

索引

ふ

ま

も

よ

り

れ