車載システム開発時に使用するソフトウェアツールをISO 26262の要求事項に準拠させるための作業項目の抽出と考察

実施報告書

2013 年 2 月

独立行政法人情報処理推進機構
Information-technology Promotion Agency, Japan

はじめに

IPA/SEC では、ソフトウェア品質説明力を強化すべく様々な観点からの検討を実施してきました。その一環として、ソフトウェア品質を説明するための手法等について具体的な実施方法、そのための作業量、実施にあたっての課題等を整理し、実際にソフトウェア品質を説明する際の参考とできるようにするために、公募により、観点ごとに分けられた実験を別々に実施しました。本書は、それらの結果を、実験ごとにまとめた報告書のうちの１つです。

本報告書の実験は、「2011 年度 システムエンジニアリング実践拠点事業」として、日本ノーベル株式会社に委託し実施しました。

「車載システム開発時に使用するソフトウェアツールを ISO 26262
の要求事項に準拠させるための作業項目の抽出と考察」
【実施報告書】

独立行政法人情報処理推進機構

目次

1. 実験の背景・目的

本実験では品質説明力強化に関わる要素として、既存の認証制度への適合性を取り扱う。既存の認証制度下でのソフトウェアシステムの開発に用いられるソフトウェアツールに関して、一般的なツールに対する品質確認及び品質レベル（製品に不具合が生じた時の社会的な影響度に応じて設定されるレベル）毎に要求されるツールの安全性の調査に関わる作業項目を洗い出し、品質レベル毎に、既存の安全規格の求める作業項目とのギャップを明らかにすることで、品質説明力の強化に関連する諸作業の実現可能性等の評価等、参考となるデータを収集することを目的とする。

具体的には開発時に使用されるコンパイラツールを対象に、機能安全規格などの既存認証制度で要求されている、そのコンパイラツール自体の不具合がシステム（製品）にまで影響しないことを確認する[1]為の品質作業を抽出し、品質レベル毎にまとめることで、作業項目のギャップを調査する。

[1] そのソフトウェアツールの品質が要求される信頼性レベルに達しているかを確認する。

2. 実験について

2.1. 概要

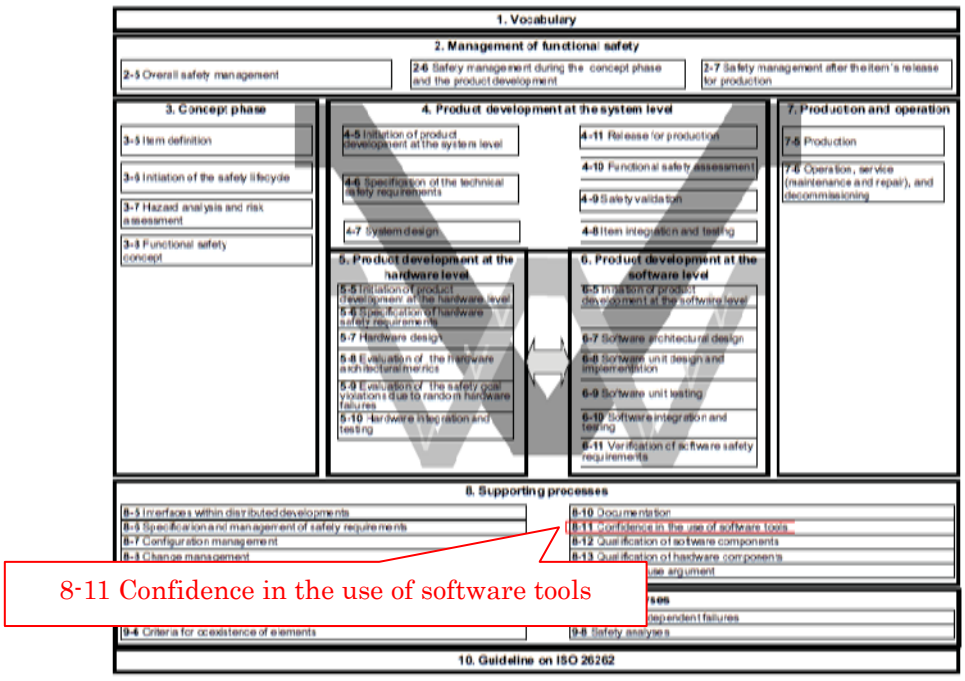
本実験では「図 1.ISO 26262 Overview」の赤線で囲まれている ISO 26262-8:2011 の、「Confidence in the use of software tools」で規定されている「開発に使用するソフトウェアの品質評価」のフェーズに対して、以下のシミュレーションを行う。

※ 本実験は、一般的なシステム開発でのライフサイクルのフェーズ（ISO 26262 では図 1 の 「6 . Product development at the software level」にあたる）ではなく、開発に用いられるソフトウェアツールの品質確認のフェーズを作業対象とした実験となる。

- 車載システムと開発ツールに求められる安全性レベルの設定
- 安全性レベルから導き出される、ソフトウェアツールの認定方法の選定
- ソフトウェアツールに対して ISO 26262 の安全要求事項を満たす為に必要な具体的な作業を抽出

このシミュレーションから、「開発に使用するソフトウェアの品質評価」として ISO 26262-8:2011 にて要求されている作業を洗い出し、また併せて一般的にソフトウェア品質評価に要求される作業を洗い出し、これらに考察を加えることにより、ソフトウェア品質説明力の強化にとって有効となる作業項目を明らかにする。

図 1.ISO 26262 Overview



2.2. 前提

開発時に使用するソフトウェア（ソフトウェアツール）には様々な種類のものがあるが、下記に挙げる点を考慮し、今回の実験の対象ソフトウェアとしては「組込みソフトウェア開発向けコンパイルツール」とする。

- 非常に多くの組込みソフトウェア開発現場で採用されており、使用頻度が高い
- 使用ソフトウェアの品質が製品（開発されるソフトウェア）の品質に直接影響する
- 入出力の組合せが莫大であるため、品質評価が難しい

その中でも、機能安全の観点から特にソフトウェアの品質に対する要求事項の多い、「車載システムの開発で、選定・使用されるコンパイルツール」に対して、ISO 26262 で要求されている開発時に使用されるソフトウェアの品質確認(ソフトウェアツール認定)のための作業として、「C 言語コンパイラの品質評価」のシミュレーションを行う。
また、ISO 26262-8:2011 では品質評価の際に使用する方法として後述の「図 3.TCL3 に分類された場合のASIL[2]別の認定方法」に挙げられる 4 つの方法が規定されており、品質レベルに応じて推奨される方法を選択するが、本シミュレーションではこの規定されている全ての方法に対してシミュレーションを行う。

2.3. 本実験にて設定する品質レベル

本実験では、製品が不具合を起こした時の影響度のレベルに応じた段階を設け、その段階ごとに結果をまとめる。この段階のことを以下では「品質レベル」と呼称する。

本実験では、ISO 26262-8:2011 「11 Confidence in the use of software tools」を、作業抽出とギャップ分析の対象とする。上記の規格では、車載システムの開発に使用するソフトウェアツールに対して、その品質が安全性の観点から妥当なレベルにあるか、ソフトウェアツールの利用者が検証し、認定することが求められている。
利用者は車載システムの機能に対して適切なASILを設定し、ソフトウェアツールに対しても適切なTCL[3]を設定することで、各ソフトウェアツールに対して推奨される認定方法が決定され、それにしたがってツール認定レポートを作成しなければならない。
そのため、本実験ではこの推奨される認定方法の根拠となる ASIL、TCL を品質レベルの指標とする。

---

[2] ASIL：Automotive Safety Integrity Level
リスクを許容水準に抑えることを達成するための安全性の要求
A(要求レベル低)~D(要求レベル高)までのレベルがある。

[3] TCL ：Tool Confidence Level
ソフトウェアツールの誤動作により、安全性の要求に反してしまうリスクのレベル
1~3 までのレベルがあり、高いほどリスクが回避し難く影響が大きいとみなせる。

2.4. 実験内容

「2.2 前提」、「2.3 本実験にて設定する品質レベル」で決定した実験の前提条件と品質レベルを使用して以下の手順で実験を実施した。

2.4.1. 規格で要求されているツールに対する安全性要求事項の抽出

ISO 26262-8:2011 の規格にて、開発に使用するソフトウェアツールの品質確認の為に規定された「ソフトウェアツール認定」の全体的な要求事項を以下の 3 つの手順で抽出した。

「2.4.1.1 車載システムと開発ツールに求められる安全性レベルの設定」
「2.4.1.2 安全性レベルから導き出される、ソフトウェアツールの認定方法の選定」
「2.4.1.3 認定方法毎の安全性要求事項の抽出」

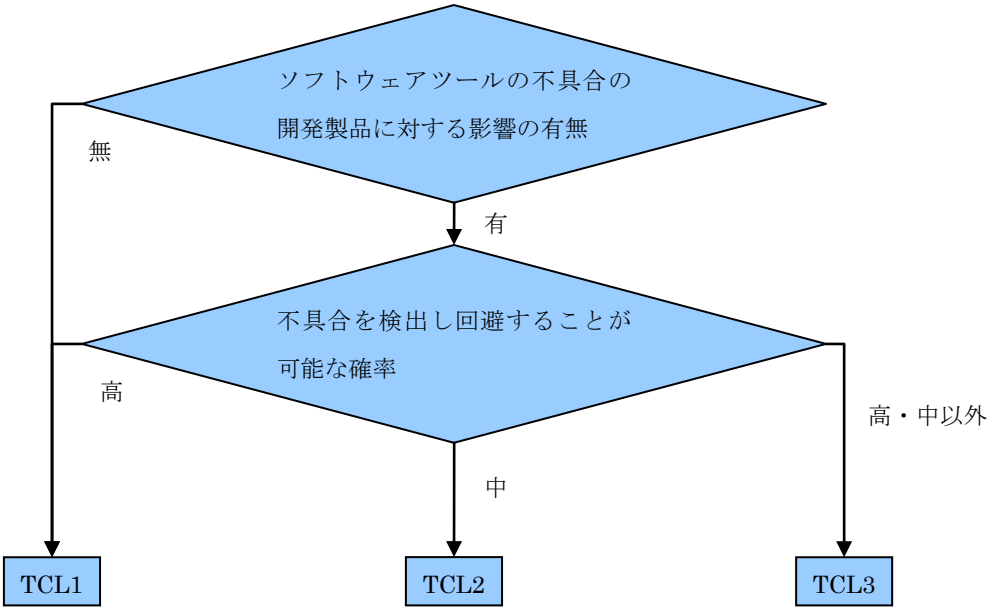
2.4.1.1. 車載システムと開発ツールに求められる安全性レベルの設定

詳細な要求事項はソフトウェアツールの認定方法により異なる。
ソフトウェアツール認定方法は ISO 26262 で使用される安全性レベル ASIL と TCL の組合せにより奨励される認定方法が決定されるため、まず本実験において用いる ASIL と TCL のレベルを設定した。

本実験の対象ソフトウェアツールである C 言語コンパイラには、以下の特徴がある。
・コンパイラの不具合が開発製品の不具合に直結する
・インプット、アウトプットともに膨大なケースが考えられる
　※インプット　＝　ソースコード、コンパイルオプションなど
　　アウトプット　＝　オブジェクトコード
・アウトプットの検証がシステマティックには行えない
　※実際にテストしてみなければ、品質を確認できない

以上の特徴から、コンパイラツールの TCL は要求される品質確認レベルの最も高い TCL 3 を用いることにした。(参考: 図 2. TCL 確定フロー)

図 2. TCL 確定フロー


ソフトウェアツールの不具合の
開発製品に対する影響の有無
無
有
不具合を検出し回避することが
可能な確率
高
中
高・中以外
TCL1
TCL2
TCL3

2.4.1.2. 安全性レベルから導き出される、ソフトウェアツールの認定方法の選定

ソフトウェアツールの TCL が 3 の場合には「図 3.TCL3 に分類された場合の ASIL 別の認定方法」に従って、強く推奨される認定方法を選択することが望ましいとされている。
しかし本実験ではソフトウェアツールの利用者が ASIL　A～D のいずれの要求レベルを選択した場合にも対応するため、4 つの認定方法それぞれが選択されたケースを想定して、作業を進める。

図 3.TCL3 に分類された場合の ASIL 別の認定方法

認定方法	ASIL			
	A	B	C	D
利用実績	++	++	+	+
ツール開発プロセスの評価	++	++	+	+
ソフトウェアツールの検証	+	+	++	++
安全規格に従った開発	+	+	++	++

＋＋　：強く推奨
＋　　：推奨

2.4.1.3. 認定方法毎の安全性要求事項の抽出

前述したように、「図 3.TCL3 に分類された場合の ASIL 別の認定方法」にある 4 つの認定方法全てを対象として、共通となる部分を含めて、下記の 5 つの点からの要求事項を抽出した。

- ISO 26262 認定方法で共通となる安全性要求事項の抽出
- ISO 26262 認定方法「ツール開発プロセスの評価」での安全性要求事項の抽出
- ISO 26262 認定方法「利用実績」での安全性要求事項の抽出
- ISO 26262 認定方法「ソフトウェアツールの検証」での安全性要求事項の抽出
- ISO 26262 認定方法「安全規格に従った開発」での安全性要求事項の抽出

※各プロセスにおける要求事項の詳細は本報告書末尾の添付資料にある別紙[要求事項一覧]を参照されたい。

2.4.1.3.1. Automotive SPICE で要求されている安全性要求事項

認定方法「ツール開発プロセスの評価」を採用する際に参照する規格として Automotive SPICE を選択した。Automotive SPICE は車載システムの品質確保を目的として、欧州の完成車メーカーが共同で策定したプロセスモデルである。また、認定方法「ツール開発プロセスの評価」で「ソフトウェアツールの開発に適用される開発プロセス」の適切な規格の例として挙げられているうちの 1 つでもある。

Automotive SPICE には計 31 のプロセスが存在するが、今回は完成車メーカーがサプライヤーに要求することが多い 15 プロセスを使用した(表 1 参照)。

表 1.今回使用するプロセスの一覧

プロセス ID	プロセス名
ENG.1	要件抽出
ENG.2	システム要件分析
ENG.3	システムアーキテクチャ設計
ENG.4	ソフトウェア要件分析
ENG.5	ソフトウェア設計
ENG.6	ソフトウェア構築
ENG.7	ソフトウェア統合テスト
ENG.8	ソフトウェアテスト
ENG.9	システム統合テスト
ENG.10	システムテスト
SUP.1	品質保証
SUP.8	構成管理
SUP.9	問題解決管理
SUP.10	変更依頼管理
MAN.3	プロジェクト管理

※各プロセスにおける要求事項の詳細は本報告書末尾の添付資料にある別紙[要求事項一覧]を参照されたい。

2.4.1.3.2. 安全規格で要求されている安全性要求事項

認定方法「安全規格に従った開発」を採用する際に参照する安全規格としては、車載システムの機能安全規格である ISO 26262-6:2011 及び ISO 26262-8:2011 を選択した。

ソフトウェアツールに完全に適合可能な安全規格はまだ存在していない。そのため「安全規格に従った開発」の規格文書内で参考規格として挙げられている ISO 26262 の製品開発フェーズから、ソフトウェアツールの設計、実装、テストを行うプロセスを抜粋して使用した。

※各プロセスにおける要求事項の詳細は本報告書末尾の添付資料にある別紙[要求事項一覧]を参照されたい。

2.4.2. 一般的な開発で行われる品質確認要求の抽出

ISO 26262 では規定されていないが、必要と考えられる品質確認の要求事項は他にも存在する。

一般的なシステム開発において、開発に使用するソフトウェアツールを導入する際に求められる品質確認要求事項をソフトウェア選定時・導入時・導入後の各フェーズでまとめた。

これを基にコンパイラ導入に際した一般的な品質確認のための作業項目を抽出した。

この抽出により、ISO 26262 での要求事項とのギャップが明確になった。

※要求事項の詳細は本報告書末尾の添付資料にある別紙[要求事項一覧]を参照されたい。

2.4.3. 抽出された安全要求事項を満たす為に必要な具体的な作業の考察

ここまでの作業で抽出された要求事項を基に、それを満たすために必要な具体的なソフトウェアツールの品質確認の作業内容を考察した。考察は以下の 2 つの視点で行った。

- ISO 26262 における安全性要求事項

  2.4.1 の結果を基に、ISO 26262-8:2011 で開発に使用するソフトウェア(ソフトウェアツール)の品質確認の為に規定された「ソフトウェアツール認定」を行う際に必要となる作業内容を具体的に考察した。

  規格では「評価する項目」は記載されているが、それを「だれが」「なにを」「どのように」行うかはほとんど示されていない。

  そのあいまいになっている箇所を明確にすることを目的とした。

- 一般的な品質確認要求事項

  2.4.2 の結果を基に、一般的なシステム開発において行われている、使用するソフトウェアツールに対する品質確認要求事項の具体的な作業を考察した。

  この考察を行うことで ISO 26262 とは違った傾向の、品質確認に必要な視点、注意点、そして、作業内容を確認することができた。

結果としてまとめられた作業内容の概略は[3.開発ツールとしてのソフトウェアの品質確認の為に求められる品質作業]に記載した。

2.4.4. 抽出した作業項目と規格要求とのギャップの分析

これまでに得られた要求事項と具体的な作業内容を一覧表にまとめた。

ソフトウェアツールの認定方法ごとに要求事項を分類し、その要求事項を満たすための作業項目、作業内容、要求規格名と共にまとめることで、品質レベル毎の要求事項のギャップが明らかになった。抽出された要求事項のうち、一般的に行うべき要求事項や 4 つのソフトウェアツールの認定方法の全てで必要とされた要求事項は[共通]として分類した。

要求事項と作業項目をリストアップすることで作業量の目安といったことも把握しやすくなった。

まとめた結果は[4.実験結果]を参照されたい。

2.4.5. 本実験の工数

本実験において、各作業にかけた工数を「表 2 各実験内容に対する作業工数」に示す。

表 2 各実験内容に対する作業工数

実験内容	工数
規格で要求されているツールに対する安全性要求事項の抽出	53.5 時間
車載システムと開発ツールに求められる安全性レベルの設定	7.5 時間
安全性レベルから導き出される、ソフトウェアツールの認定方法の選定	7.5 時間
認定方法毎の安全性要求事項の抽出	101.5 時間
一般的な開発で行われる品質確認要求の抽出	40.0 時間
抽出された安全要求事項を満たす為に必要な具体的な作業の考察	78.5 時間
抽出した作業項目と規格要求とのギャップの分析	34.0 時間
実験レポート作成	80.0 時間
合計	402.5 時間

作業期間：2012 年 4 月 4 日開始　 2012 年 5 月 31 日完了

3. 開発ツールとしてのソフトウェアの品質確認の為に求められる品質作業

本実験で抽出された、「開発ツールとしてのソフトウェアの品質」を確認するための要求事項から考察した具体的な作業内容について、以下の 2 つの場合に分けてまとめた。

- ISO 26262 のソフトウェアツール認定
  ISO 26262-8:2011 で規定された「ソフトウェアツール認定」を行う際に必要となる作業内容
- 一般的な品質確認要求事項
  一般的なシステム開発において行われる、ソフトウェアツール導入の際に必要とされる作業内容

3.1. ISO 26262 のソフトウェアツール認定

ISO 26262-8:2011 で開発に使用するソフトウェア(ソフトウェアツール)の品質確認の為に規定された「ソフトウェアツール認定」で要求される作業内容について記述する。

3.1.1. 全体像

3.1.1.1. 要件定義

開発システムに対する認識を共有・統一する為に、開発システムの要件定義を行う。

3.1.1.2. ASIL の決定

その開発システムに対する安全要求のレベル(ASIL)の算定を行う為に、開発システムの誤作動が発生した際に発生する様々な現象の洗い出しを行う。それらの現象に対して下記の観点から分析を行い、最も厳しい安全要求レベル(Max ASIL)の算定を行う。

- 不具合の発生頻度
  不具合が、他の機能との組み合わせた時のみ発生するなど、特定の環境下で発生するのか、あるいは任意の環境下で発生するのか等の考察を行い、どの程度の発生頻度があるのかを分析する。
- 引き起こされる危険度
  発火、制御不能等、実際に発生する現象を挙げてその現象の影響を抽出し、物損・死亡事故等の被害状況の分析を行う。
- 回避(検知・制御)の容易性
  不具合が発生した際の車の速度によって回避の容易性が異なるように、不具合が生じた時の回避可能性について分析する。

3.1.1.3. TCL の決定

ソフトウェアツールに対して想定される使用方法や要求される特性についての情報を抽出する。そして、これらの情報からそのソフトウェアツールの不具合が開発システムの挙動への影響の有無と、ソフトウェアツールの不具合を検出する為の仕組みの信頼度を検討し、これらの情報からソフトウェアの信頼度(TCL)を確定させる。

3.1.1.4. ソフトウェアツール認定方法の選択

「2.4.1.2 安全性レベルから導き出される、ソフトウェアツールの認定方法の選定」でも述べたように、ここまでに確定した 最も厳しい安全要求レベル(Max ASIL) とソフトウェアの信頼度(TCL) から ISO 26262-8:2011 で定められた下記の認定方法から適切な方法を選択する。
(認定方法の詳細に関しては「3.1.2 ソフトウェアツールの認定方法」で記載する)

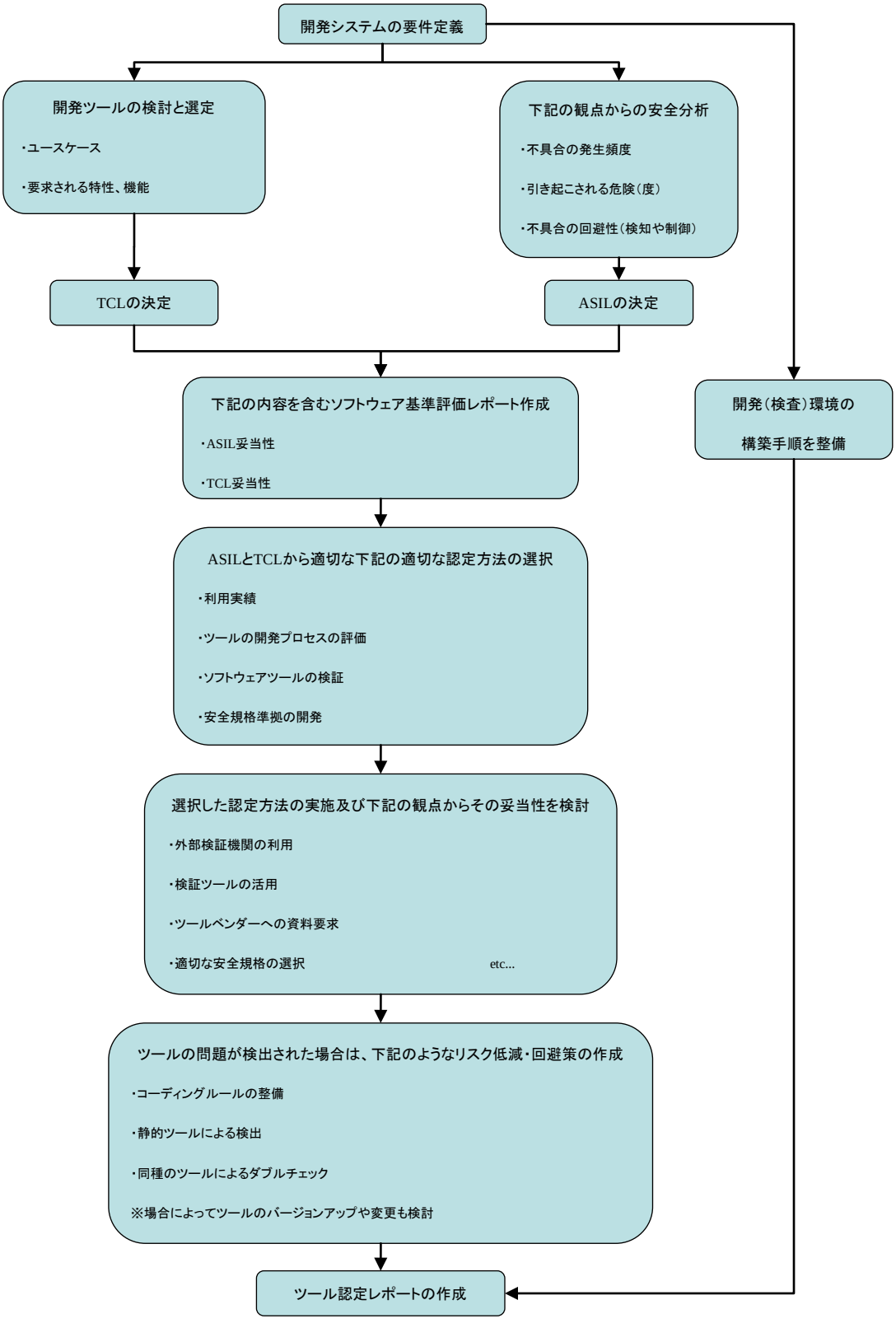
- 利用実績
- ツール開発プロセスの評価
- ソフトウェアツールの検証
- 安全規格に従った開発

3.1.1.5. エビデンスの作成

そして、上記を検討・考慮したエビデンスとして、この品質評価の為に必要な基準や実施するための環境等をまとめたソフトウェアツール基準評価レポート(software tool criteria evaluation report)を作成する。

その後、選択した認定方法での認定の実施とそこで検出した不具合の回避・検出策を作成する。これらの実際の認定作業に関わる活動のエビデンスとして、ソフトウェアツール認定レポート(software tool qualification report) を作成する。

図 4.ソフトウェアツール認定のフロー



3.1.2. ソフトウェアツールの認定方法

本節では、ISO 26262-8:2011 にて規定されたソフトウェアツール認定の各方法について説明する。ソフトウェアツール認定を行う為の方法として下記の 4 つの方法が規定されている。

- 利用実績
- ツール開発プロセスの評価
- ソフトウェアツールの検証
- 安全規格に従った開発

これらの方法の概要と要求事項は以下の通りとなる。

3.1.2.1. 利用実績

これは、そのソフトウェアツールの過去の利用実績をもって、認定を行う方法である。過去に開発で利用した実績があるソフトウェアツールにおいては、利用実績によるソフトウェアツール認定が可能であるが、これは主にソフトウェアのバージョン変更の場合のみに有効である。そして、この方法では下記のエビデンスが必要となる。

- ソフトウェアツールが過去の開発と同様のユースケース、運用環境、機能制限、目的で使用されていること
- 根拠となる十分な量の適切なデータ
- ソフトウェアツールの仕様が変更されていないこと
- 以前の開発で検出された不具合がシステマティックな方法で蓄積されていること

また、下記を含むソフトウェアツールの過去の利用実績の結果も合わせて必要である。

- 固有の識別子とソフトウェアツールのバージョン番号
- ソフトウェアツールの設定（オプション類の設定）
- 利用期間の詳細と関連データ
- 不具合のドキュメント
- 以前のバージョンと、それぞれの関連バージョンで修正された不具合のリスト
- 不具合の予防・回避策、誤出力の検出方法

3.1.2.2. ツール開発プロセスの評価

これは、ソフトウェアツールの開発プロセスが（Automotive SPICE、CMMI、ISO 15504 等の）適切な基準に従っていることの確認をもってソフトウェアツールの品質確認作業とする認定方法である。
適用される開発プロセスが国内又は国際的な規格に基づいて評価されており、対象のソフトウェアツールがその開発プロセスに従って適切に開発されていると評価されている[4]場合は、その評価結果を使用することができる。

本実験では Automotive SPICE を対象とした。
Automotive SPICE ではプロセスごとに基本プラクティスと呼ばれる作業項目と作業成果物が定められており、作業成果物の存在をもってプロセス実施のエビデンスとすることが可能である。

※Automotive SPICE の要求事項の詳細は本報告書末尾の添付資料にある別紙[要求事項一覧]を参照されたい。

3.1.2.3. ソフトウェアツールの検証

この認定方法は、ソフトウェアツールが下記の基準を満たすことが必要である。

- 規定の要求に準拠していることの立証
  （コンパイラの場合は言語規格が規定の要求に相当する。）
- 不具合の回避方法・検出方法の分析の実施
- 特異な環境下での検査（予測可能な誤用、不完全な入力データ、禁止された設定 など）の実施

3.1.2.4. 安全規格に従った開発

これは、安全規格に準拠して開発されているコンパイラツールを利用することで、安全性の認定を行う方法であり、要求される品質レベルが高い場合（ASIL C 又は D）には"強く推奨"とされている。しかし、現状では、ソフトウェアツール開発に完全に適応可能な安全規格は存在しない。その為、ISO 26262 などいくつかの安全規格から関連する項目を抽出し、それに準拠して開発が行われていると評価されれば、その評価結果を使用することができる。

---

[4] 開発プロセスが適切かどうかの評価は、国際規格に準拠している場合は比較的容易であるが、独自基準の開発プロセスなど世間に認知されていない基準を採用した場合は、適切な開発プロセスであることの証明に多大なコストが発生する。

3.1.3. ISO 26262-6:2011 及び ISO 26262-8:2011 の要求事項

利用者が認定の為に行う作業は、要求事項が満たされた適切なエビデンスを示したツールベンダーからの提供確認となる。

3.1.3.1. 設計・実装フェーズ

設計・実装フェーズでは下記の実施とそのエビデンスの作成が要求される。

- 不具合が混入するリスクを減らす為の設計・実装作業における基準の作成
- ハードウェア・ソフトウェアインターフェースの明確化
- 設計・実装時に不具合が混入するリスクを減らす様々な仕組みの作成やチェックの実施

3.1.3.2. 検証フェーズ

検証フェーズでは下記の実施とそのエビデンスの作成が要求される。

- 検証の実施を余裕を持って確実に行う為の検証計画の作成
- 検査のチェックポイント・仕様適合の確認

※要求事項の詳細は本報告書末尾の添付資料にある別紙[要求事項一覧]を参照されたい。

3.2. 一般的な品質確認要求事項

ISO 26262 に対応しつつ、同規格がカバーしていない可能性のある一般的な品質確認要求事項も満たすことが、ソフトウェアツールの安全性を強化することにつながる。
そこで、以下に記したような、開発に使用するソフトウェアツールの選定時、導入前、導入後、それぞれで利用者が行う品質確認要求事項を検討した結果、ISO 26262-8:2011 のソフトウェア認定とは異なる視点を持つ内容が抽出された。

- 選定時
  なぜソフトウェアツールを導入するのか、という大前提を吟味する。
  メリットだけでなくデメリットも認識する必要がある。
  ここで、複数の導入候補が選定される場合もある。
- 導入前
  ソフトウェアツールを使用する開発プロジェクトの現状分析からソフトウェアツールに求める機能、品質などを決定し、導入候補それぞれを評価する。
  また、ソフトウェアツールの選定者と使用者が異なる場合はソフトウェアツール導入に対するコンセンサスが重要となる。
- 導入後
  導入したソフトウェアツールが継続的に適切に使用される方策を策定し、実施する。

※ここで抽出された要求事項の詳細は本報告書末尾の添付資料にある別紙[要求事項一覧]を参照されたい。

4. 実験結果

ここまでの作業の結果として、「開発に使用するソフトウェアの品質評価」のための作業内容を抽出する為の入力として下記の要求事項を抽出し、本報告書末尾の添付資料にある別紙 [要求事項一覧]としてまとめた。

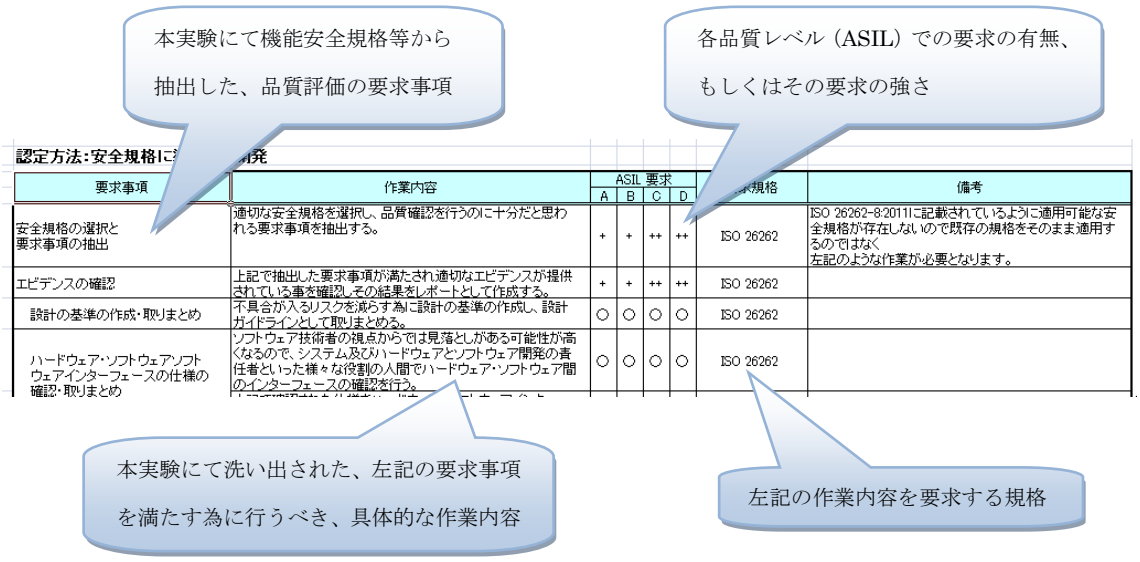
・ISO 26262 でのソフトウェアツール認定の要求事項

- Automotive SPICE の要求事項
  (ソフトウェアツールの認定方法として Automotive SPICE を使用してのツール開発プロセスの評価を選択した場合)
- ISO 26262-6:2011 及び ISO 26262-8:2011 の要求事項
  (ソフトウェアツールの認定方法として ISO 26262-6:2011 及び ISO 26262-8:2011 の一部を使用しての安全規格に従った開発を選択した場合)

・一般的に行われているソフトウェアの品質確認での要求事項
(ISO 26262 等の今回参照した既存の規格では要求されていない要求事項)

それを基に「3.開発ツールとしてのソフトウェアの品質確認」の章での要求事項に基づいた、「開発に使用するソフトウェアの品質評価」を行う際に必要となる、上記で抽出した各要求事項を満たす為の具体的な作業を下図のフォーマットで本報告書末尾の添付資料にある別紙[実験結果]に認定方法毎にレポートにまとめた。これにより、規格で要求される作業のギャップを明らかにした。

図 5.[実験結果]の各項目の説明



最後に、上記の抽出作業結果から要求事項及び作業項目の数を集計した結果を下記「表 3 要求事項・作業項目数一覧」に記載する。

表 3 要求事項・作業項目数一覧

対応			要求事項	作業項目
認定方法に依存しない 全ての認定方法で共通する作業		ISO 26262 対応	16	34
		一般的に行われているソフトウェアの品質確認対応	33	33
認定方法毎の作業	利用実績	ISO 26262 対応	3	10
		一般的に行われているソフトウェアの品質確認対応	1	1
	ツール開発プロセスの評価	ISO 26262 対応	2	2
		Automotive SPICE 対応	15	15
	ソフトウェアツールの検証	ISO 26262 対応	19	21
		一般的に行われているソフトウェアの品質確認対応	3	3
	安全規格に従った開発	ISO 26262 対応	3	3
		ISO 26262-6／ISO 26262-8 対応	10	20

※色付きのセルはソフトウェアツールの開発時に必要となる作業項目である。
そのため利用者自身がソフトウェアツールを開発する場合に、その過程で発生する作業項目となる。
またその際に準拠させる規格によって、作業内容は異なる。

5. 本実験結果からの考察

5.1. 全体的な考察

以上説明してきたように、本実験ではISO 26262でのソフトウェアツールの品質に関わる要求事項を洗い出した。この規格では品質レベルに応じて推奨される 4 つの認定方法のうちのいずれかを選択しなければならない。そこでまず各認定方法で共通の作業を抽出した上で、この 4 つの認定方法の全てに対してもそれぞれ実施するべき作業を抽出し、実験結果として一覧にまとめた。

「開発に使用するソフトウェアの品質評価」での作業項目数は「表 3 要求事項・作業項目数一覧」にまとめたが、そのうちISO 26262 で要求されるコンパイラツールの品質評価の作業項目数だけを抽出したもの[5]が「表 4 ISO 26262 での認定方法別の作業項目数」となる。表を見ればわかる通り、各認定方法で必要とされる作業項目数は大きく異なる。
また同じ認定方法でも市販のコンパイラを利用する場合と、利用者自身が開発したコンパイラの場合とでは、必要とされる作業項目数は大きく異なるため注意が必要である。

表 4 ISO 26262 での認定方法別の作業項目数

	適切な規格のプロセスで開発された市販コンパイラを利用する場合	利用者自身が適切な規格のプロセスでコンパイラを開発して利用する場合
共通する作業	34 項目	
利用実績	10 項目	
ツール開発プロセスの評価	2 項目	17 項目
ソフトウェアツールの検証	21 項目	
安全規格に従った開発	3 項目	23 項目

各認定方法に関する考察は後述（5.3、5.4、5.5、5.6 節）するが、それぞれの認定方法を比較する場合には、作業項目数だけでなく、その適用条件も考慮しなければならない。例えば利用実績は作業項目数が少ないため実施は容易に見えるが、同種の開発プロジェクトにしか適用できないといった制限があり、またソフトウェアツールの検証は作業項目数が多く実施にはコストがかかるが、利用実績のような制限はない。
ツール開発プロセスの評価や安全規格に従った開発では、開発に用いるコンパイラツールの選定時点で、それが適切な規格に準拠して開発されていなければ適用できない。

したがって、本実験にて抽出されたソフトウェアツールの品質検証作業の内容と規模、そ

[5] 「一般的に行われているソフトウェアの品質確認対応」の作業項目を除いたもの

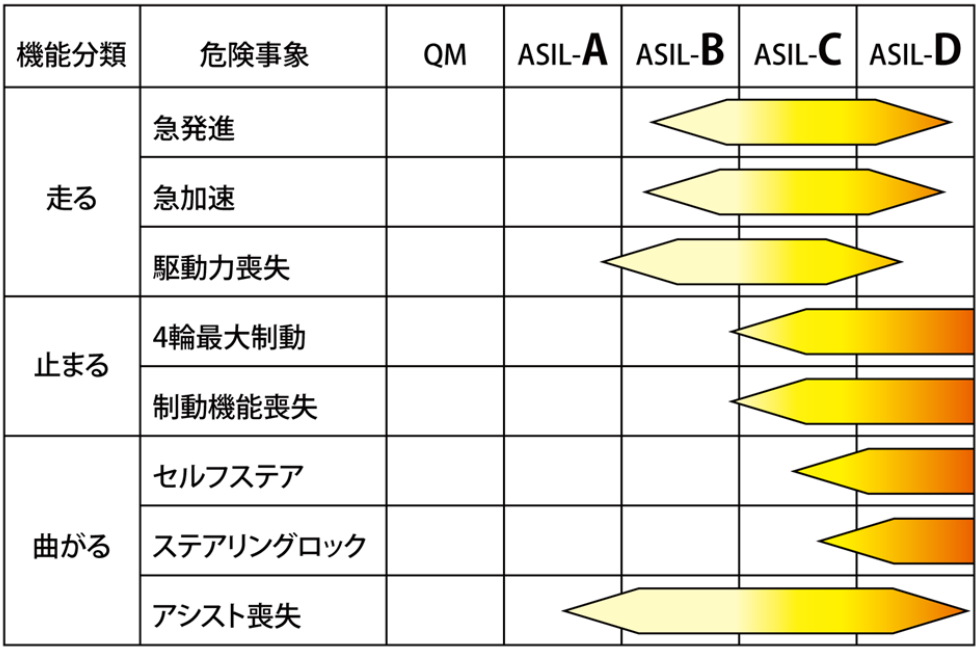
の制限事項などは、ソフトウェア品質説明力の強化に関連した検証基準等を検討する際の参考になるものと考えられる。

5.2. 品質レベルを特定する指標について

ソフトウェアツールの検証にあたっては、最初に開発システムに求められる安全性レベルと、そのツール自体のリスクや影響度の分析が必要になる。本実験中では ASIL と TCL の分析・決定の作業がこれに該当し、作業抽出の指標となる品質レベルとしている。

しかし ISO 26262 などの安全規格でもこの品質レベルを決める際の基準、目安といったものは 2012 年 5 月現在提示されていない。これまでの開発でも既にリスク分析の手法や基準を定義して、設計・開発に生かしていた開発者の場合は、分析結果の表現を各規格に合わせるだけで済む。しかしながら、これから機能安全対策に着手しようという開発者にとっては、個別に独自の基準を定義していかねばならず、対策のための負担が少なくない。そこで、公的機関や業界団体で統一した品質レベルを構築する動きが進み、基準や目安、相場観、例えば「図 6.ASIL 相場観の一例」のような指標となるものが、業界全体で早期に整備されることが望ましいと考えられる。

図 6.ASIL 相場観の一例



※QM（Quality Management）：安全性の要求なし

5.3. 認定方法：利用実績について

過去のコンパイラツールの利用実績からツールの安全性の認定を行う方法であり、要求される品質レベルが低い場合（ASIL A 又は B）には"強く推奨"とされている。適切な利用実績の記録がされている場合には要求される作業項目も少なく、一見すると利用しやすい認定方法と思われがちだが、この方法の前提条件として、①過去のバージョンから動作仕様が大きく変わっていない　②ツール利用時の動作条件・設定を変更していない　③運用環境、ユースケースも類似している（同種の開発プロジェクトにしか適用できない)、等といった制約もあり、これを適用できるシーンは限定される。

また第三者への説明の必要が生じた際には、「過去の利用では問題が発生していなかった」という根拠だけでは説得力に欠ける。その場合には過去のプロジェクトでのコンパイラツールの安全性を認定した際のエビデンスを求められるかもしれない。

5.4. 認定方法：ソフトウェアツール開発プロセスの評価について

コンパイラツールの開発プロセスが適切な規格[6]に準拠して開発されているかを認定の根拠とした方法であり、要求される品質レベルが低い場合（ASIL A 又は B）には"強く推奨"とされている。

サードパーティ製のコンパイラツールの場合は、開発プロセスのエビデンスを提供してもらう必要があるため、ツールベンダーの対応状況に依存するが、利用者側で行う作業が最も少なくなる。

その場合には第三者への品質説明のための説得力という観点からも、信頼のある外部認証機関からの認証を受けたコンパイラツールを採用することが望ましいと言えるだろう。

自社開発したコンパイラツールの場合は、そのコンパイラの開発プロセスが国際的に認知された規格に準拠して開発されている必要があるので、利用者はコンパイラを開発する段階で開発プロセスを適切な規格に準拠させておく作業が必要になる。

利用実績以外の方法でツールの安全性の認定を行う場合には、ツールベンダーからの資料提供の有無などによって、利用者にかかる作業の負荷が大きく変わる。

そのためコンパイラツールベンダーとのサポート契約の内容も重視する必要があるだろう。

[6] 例えば Automotive SPICE や CMMI、ISO 26262 などである。

5.5. 認定方法：ソフトウェアツールの検証について

コンパイラツールの品質を利用者自身が検証することで安全性の認定を行う方法であり、要求される品質レベルが高い場合（ASIL C 又は D）には"強く推奨"とされている。実際にコンパイラツールの検証を行うため、コンパイラに潜在する問題点を検出する可能性も高く、検証データが手元に残るので第三者への品質の妥当性を説明し易いというメリットもある。

しかし、実際にコンパイラツールの検証作業を行うにはかなりの期間と費用がかかり、開発体制とは別にコンパイラの専門知識を持った要員を集めなければならないかもしれない。

コンパイラツールは機能仕様が規格で定められているため、市販の検証ツールなどもあるが、その多機能さと無限ともいえる入力の組合せが、利用者による検証の実施を困難にしている。

そのため、使用するコンパイラツールの機能を想定される利用状況から分析し、検証範囲を適切に絞り込むことが肝要である。

さらに、利用者のツール利用形態が多様なため、コンパイラベンダー側からツール認定のためのエビデンスを入手するのは困難である。第三者検証機関への検査の委託、市販テストスイートの導入など、外部リソースの活用の有無によってもツール認定に要するコストは大きく影響を受ける。

これに対して、自社開発したコンパイラツールの場合は、その開発規模にもよるが、ツールの仕様やマニュアル、テストデータなど開発時に作成したものが既に揃っているはずなので、検証に要するコストはそれほど大きくはならないはずである。

5.6. 認定方法：安全規格に従った開発について

既に説明したように、コンパイラツールなどのソフトウェアツールの開発に完全に適用可能な安全規格というものは、現段階では存在しない為、複数の安全規格から適切な要求事項を選択して、独自の安全規格を作成する必要があるため、コンパイラツールの開発前にやるべき作業が多くなる。また作成した独自の安全規格が妥当であるということを第三者に説明する必要もある。

現段階ではツールベンダーが安全規格に準拠したコンパイラを提供することもできないので、開発に内製ツール（コンパイラ以外の）を利用する場合の、そのツールの認定に用いるのが、現実的である。

6. 最後に（所感等）

今回の実験:「車載システム開発時に使用するソフトウェアツールに対して ISO 26262 の安全要求事項を満たす為に必要な具体的な作業項目の考察」を実施して実感したことは、開発者に課せられるツール認定作業を支援するためのガイドラインやツールなどが十分に整備されているとは言えないことである。

本実験での規格要求事項の抽出作業では参考となる文献が規格書以外にほとんどなかったため、規格の要求を正しく理解するために労力を費やした。要求されている作業項目の洗い出し作業においても、規格書内には具体的な説明が乏しく、また参考となる書籍やツール認定事例なども現段階では存在しない。そのため現状では機能安全のための判断基準や作業の取捨選択は、開発者個々の判断に任されているが、「これが正解」というものが無い以上、その取組みのスピードは遅くならざるを得なかった。

また、品質レベルによってはツール認定作業のための負荷が大きくなり過ぎるため、一開発部門で実施するには期間・コスト的に困難なケースがある。系列グループ内で基準や手法を統一し、組織力で対応することで、作業のハードルを下げるという企業ごとの動きもあるが、業界全体で共有できる環境の整備という段階にまでには至っていない。

ソフトウェア品質説明力強化に向けた取り組みが広く進むことは、「何から手を付けていいか分からない」という開発者に対して最初の一歩を踏み出すきっかけとなり、それが業界全体の足並みを揃える上できわめて重要な役割を果たすものと考えられる。

Appendix A 参考文献

本実験にて参照した文献は以下の通りとなる。

- ISO 26262-1:2011
- ISO 26262-6:2011
- ISO 26262-8:2011
- Automotive SPICE Process Assessment Model (PAM) - RELEASE v2.5 - 10 May 2010
- Automotive SPICE Process Reference Model (PRM) - RELEASE v4.5 - 10 May 2010
- Automotive SPICE 実践ガイドブック [改訂版] （株式会社日経 BP）
- 情報システムの信頼性向上に関するガイドライン 第 2 版（経済産業省）
- SQuBOK ガイド 第 1 版 β 版（一般財団法人日本科学技術連盟）

Appendix B 実験で作成されたデータ等の資料

本実験にて入手、又は作成した成果物及び関連資料は以下の通りとなる。

— 実験結果

開発で使用する C 言語コンパイラの品質確認を行う際に、ISO 26262-8:2011 をメインとした既存規格の要求事項と一般的なソフトウェア品質評価での要求事項及びそれらの要求事項を満たす為の具体的な作業内容をまとめた資料である。

— 要求事項一覧

上記の実験結果の入力にあたるデータとなる、既存規格及び一般的なソフトウェア品質評価での要求事項をまとめた資料である。

これらは添付資料として本報告書の末尾に添付した。

添付資料

別紙
要求事項一覧

実験で作成されたデータ等の資料
要求事項一覧

ISO 26262-6:2011及びISO 26262-8:2011
での要求事項

ISO 26262-6:2011

章番号	要求事項	備考
5.5.2 6.5.3 9.5.1	ソフトウェア検証計画	
5.5.3	設計ガイドライン	
	コーディングガイドライン	
6.4.7	システム及びハードウェアとソフトウェア開発の責任者による仕様確認	
6.5.2	ハードウェアーソフトウェアインターフェース仕様書	
6.5.4 7.5.6 8.5.3 9.5.3	ソフトウェア検証報告書	
7.4.2	設計時は下記を考慮するものとする ・検証可能性 ・設計と実装可能性 ・テストの容易性 ・保守性	
7.4.3	高い複雑性による不具合の発生を避ける為に下記の特性を示す ・モジュール性 ・カプセル化 ・単純さ	
7.4.18	ISO 26262-8:2011 の9 節に従った設計の検証	
7.5.1	ソフトウェアアーキテクチャデザイン仕様書	
8.4.4	設計と実装時に下記の実現 d) 単純さ e) 読みやすさと理解しやすさ f) ロバスト性 g) 適切なソフトウェアの変更 h) テストの容易性	
8.4.5	設計と実装時に対する下記の検証 a)ハードウェアインターフェース仕様への準拠 c) ソースコードの設計仕様への準拠 d) ソースコードのコーディングガイドラインへの準拠	
8.5.1	ソフトウェアユニットデザイン仕様書	
9.4.2	ISO 26262-8:2011 9 節に従ったソフトウェアユニットテストの計画、規定、実施	
9.4.3	ソフトウェアユニットが下記を達成している事の証明 a) ソフトウェアユニットデザイン仕様への適合 b) ハードウェアインターフェースの仕様への適合 c) 規定された機能 d) 意図していない機能が存在していなこと e) ロバスト性 f) 機能をサポートするのに十分なリソース	
9.4.5	テストケースの網羅性の評価	
	意図されていない機能が存在しないこと	
	カバレッジの測定 (カバレッジが十分でない場合は、テストの追加かその論拠の説明)	
9.4.6	テスト環境がターゲット環境に可能な限り近いこと (異なる場合はソースとオブジェクトコード・テスト環境とターゲット環境の相違点の分	
9.5.2	ソフトウェア検証仕様	

ISO 26262-8:2011

章番号	要求事項	備考
9	なし(表題)	
9.1	なし(目的)	
9.2	なし(概要)	
9.3	なし(表題)	
9.3.1	各フェーズでの前提条件	
9.3.2	各フェーズでの追加支援情報	
9.4	なし(表題)	
9.4.1	なし(表題)	
9.4.1.1	下記の割り当て a) 検証されるべき作業成果物の内容 b) 検証方法 c) 検証の合否基準 d) 該当する場合は、検証環境 e) 該当する場合は、検証の際に使用したツール f) 異常を検知した際に、行うべき対応 g) リグレッションストラテジー	
9.4.1.2	下記の検討 a) 適用する検証方法の妥当性 b) 確認する作業成果物の複雑さ c) 対象の検証に関連する以前の経験 d) 使用技術の成熟度やその技術使用に関連するリスク	
9.4.2	なし(表題)	
9.4.2.1	下記の明確化 a) チェックリストのレビューや分析 b) シミュレーションシナリオ c) テストケース、テストデータ及び検査対象	
9.4.2.2	各テストケースにて下記の明確化 a) 固有の識別 b) 検証すべき関連作業成果物のバージョンへの言及 c) 前提条件と設定 d) 該当する場合は、環境条件 e) 入力データ、時系列及びそれらの値 f) 出力データを含む期待される挙動、出力値の許容範囲、時間挙動及び許容挙動	
9.4.2.3	下記の規定 a) テスト環境 b) 理論上の一時的な依存 c) リソース	
9.4.3	なし(表題)	
9.4.3.1	なし(前項で記載されている事の実施の要求)	
9.4.3.2	検証結果として下記を含む a) 検証作業成果物の固有の識別 b) 検証計画と検証仕様の言及 c) 該当するならば、検証環境と使用検証ツールの設定及び検証中に使用するキャリブレーションデータ d) 期待される結果に対する検証結果の準拠レベル e) 検証の合否の明確な状態；検証に失敗した場合に、状態は失敗の論理的根拠と検証された作業成果物の変更の為の提案を含むものとする。 f) 実施されなかったあらゆる検証段階の理由	
9.5	なし(表題)	
9.5.1	9.4.1.1 及び 9.4.1.2 の要求を満たす検証計画	
9.5.2	9.5.2. 9.4.2.1 から 9.4.2.3 までの要求を満たす検証仕様	
9.5.3	9.5.3. 9.4.3.1 及び 9.4.3.2 の要求を満たす検証レポート	
11	なし(表題)	
11.1	なし(目的)	
11.2	なし(概要)	
11.3	なし(表題)	
11.3.1	ISO 26262-4:2011 5.5.2 に沿った安全計画	一般的には、これより前の段階で作成・抽出されているものなので、コンパイラのソフトウェアツール認定時に新たに作業をしなければならないケースは稀であると考えます。
	ソフトウェアツールが使用される安全ライフサイクルの適切な要件	
11.3.2	予め定義された最大 ASIL	
	(外部からの)ソフトウェアツールのユーザーマニュアル	
	(外部からの)ソフトウェアツールの環境・制限	
11.4	なし(表題)	
11.4.1	なし(表題)	
11.4.1.1	なし(該当範囲の定義)	
11.4.2	なし(表題)	
11.4.2.1	TCLの妥当性確認は、ASIL によって要求が異なる 認定の妥当性確認は、ASIL によって要求が異なる	具体的には ASIL A -> 特に要求は無し
11.4.3	なし(表題)	

11.4.3.1	下記が評価基準や認定に準拠している事 ・使用方法 ・環境と機能制限 ・動作条件	
11.4.4	なし(表題)	
11.4.4.1	下記の決定 a) ソフトウェアツールのバージョン番号の識別 b) ソフトウェアツールの設定 c) ソフトウェアツールのユースケース d) ソフトウェアツールの実行環境 e) ソフトウェアツールの誤作動やその誤った出力の生成の際に、アイテムやエレメントに割り当てられた抵触する可能性のある全ての安全要求の最大 ASIL f) 確定した信頼レベルに基づき要求される場合にソフトウェアツールの認定方法	
11.4.4.2	下記の情報が利用可能なこと a) ソフトウェアツールの機構、機能、及び技術的な性質の記述 b) 該当する場合は、ユーザーマニュアルやユーザーガイド c) 運用環境の記述 d) 該当する場合は、特異な運用環境下にてソフトウェアツールが期待される挙動の記述 e) 該当する場合は、既知のソフトウェアツールの異常と適切な予防対策、回避策や次善策の記述 f) ソフトウェアツールの信頼性の要求レベル確定中に確認されているソフトウェアツールの誤動作とそのエラー出力誤動作の検出方法	
11.4.5	なし(表題)	
11.4.5.1	下記の情報を含むソフトウェアツールの使用方法に関する記述 a) 用途 b) 入力と期待する出力 c) 該当する場合は、環境と機能制限	
11.4.5.2	なし(TI/TD の分類の定義)	
11.4.5.3	なし(TI/TD の分類の定義)	
11.4.5.4	なし(TI/TD の分類の定義)	
11.4.5.5	なし(TCLの確定方法の定義)	
11.4.6	なし(表題)	
11.4.6.1	なし(適切な認定方法の定義)	

11.4.6.2	ソフトウェアツールの認定の際に下記を明記する事 a) 固有の識別とソフトウェアツールのバージョン番号 b) 評価分析の参考共に分類されたソフトウェアツールの最大のツール信頼レベル c) ソフトウェアツールの誤動作やその誤った出力によって抵触する可能性のある安全要求のうち予め決められた最大 ASIL (または特定の ASIL) d) ソフトウェアツール認定のための設定と環境 e) 認定を実施した人物または組織 f) 適用した認定方法 g) 適用したソフトウェアツール認定方法の結果 h) 該当する場合は、利用制限と認定中に確認された誤動作	ソフトウェアツールの信頼性レベルは、下記の二つの観点の分析によって確定します。 ・ソフトウェアツールの不具合が開発システムの不具合を引き起こす可能性の有無 ・ソフトウェアツールの不具合の検出・混入防止策の信頼性
11.4.7	なし(表題)	
11.4.7.1	なし(定義に関する記述)	
11.4.7.2	利用実績によるソフトウェアツール認定の際は下記のエビデンスを要求 a) 類似のユースケース、類似の確定運用環境及び、類似の機能制限で同じ目的でソフトウェアツールが以前に使用されている。 b) 利用実績の根拠は、十分な量の適切なデータに基づく c) ソフトウェアツールの仕様が変更されていない d) 誤動作の発生と以前の開発で発生したソフトウェアツールの対応する誤った出力はシステマティックな方法で蓄積される。	類似のソフトウェアツールや互換を謳っているソフトウェアツールの認定の為にはこの利用実績による認定は認められず、バージョン検討の場合のみ有効
11.4.7.3	利用実績によるソフトウェアツール認定の際は以前の下記の情報を検討材料とする事 a) 固有の識別とソフトウェアツールのバージョン番号 b) ソフトウェアツールの設定 c) 利用期間の詳細と利用の関連データ d) 誤動作とそれらを発生させる条件の詳細と共にソフトウェアツールの対応する誤った出力の文書 e) 測定した以前のバージョンのリスト、それぞれの関連バージョンで修正されたリスト化された誤動作 f) 該当するならば、既知の誤動作の予防手段、回避策や次善策、またはそれに対応する誤った出力の検出方法	
11.4.7.4	なし(制限事項)	
11.4.8	なし(表題)	
11.4.8.1	なし(認定の適用に関する記述)	
11.4.8.2	安全規格準拠によるソフトウェアツール認定の際は、適切な規格に準じている事	規格が国際規格の場合は妥当性確認は比較的容易だが、ローカルやプライベートな規格の場合は妥当性確認はその規格の内容に精通していなければならず困難
11.4.8.3	安全規格準拠によるソフトウェアツール認定の際は、適切に適用されている事の証明	

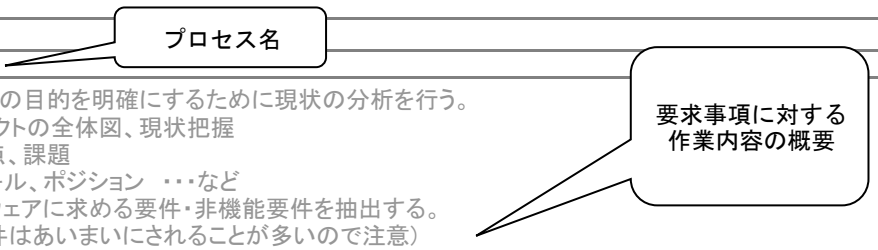
11.4.9	なし(表題)	
11.4.9.1	なし(認定の適用に関する記述)	
11.4.9.2	ソフトウェアツールの認証によるソフトウェアツール認定の際は下記の基準に準拠する事 a) ソフトウェアツールが規定の要求に準拠している事を立証するものとする b) 検証中に発生した誤動作とそれに対応するソフトウェアツールの誤った出力は、引き起こされる可能性のある結果とそれらを回避や検出する方法と共に分析されます。 c) 特異な運用環境下でのソフトウェアツールの反応が検査されるものとする	
11.4.10	TCL の妥当性確認	ASIL-A 要求無し ASIL-B 推奨 ASIL-C, D 必須
	認定の妥当性確認	
11.5	なし(表題)	
11.5.1	なし(Software tool criteria evaluation report の内容)	
11.5.2	なし(Software tool qualification report の内容)	

実験で作成されたデータ等の資料
要求事項一覧

一般的なソフトウェア品質評価での要求事項

ソフトウェア導入前の品質確認要求事項

品質確認要求事項	
	概要
2.1	現状分析
	ソフトウェア導入の目的を明確にするために現状の分析を行う。 ・対象プロジェクトの全体図、現状把握 ・現在の問題点、課題 ・使用者のスキル、ポジション ・・・など その上でソフトウェアに求める要件・非機能要件を抽出する。 (特に非機能要件はあいまいにされることが多いので注意) ソフトウェアが使用される環境についても確認をする。 (特殊な環境の場合には要注意)



ソフトウェア選定時の品質確認要求事項

品質確認要求事項	
	概要
1.1	ソフトウェアを導入するメリットの認識
	開発効率を向上させる。 新しい技術に対応した開発を行う。 生産性を高め、開発コストを下げる。 バグを減らし、成果物の品質を向上させる。
1.2	ソフトウェアを導入するデメリットの認識
	新ソフトウェアへの対応体制の整備を行う。 新ソフトウェアを活用するための教育コストがかかる場合があり、保守・維持コストのような間接的なコストの評価を行う。 新ソフトウェアの効果がすぐにでるわけではないので、スケジュールの評価を行う。
1.3	ソフトウェアの影響の認識
	ソフトウェアを導入する際は開発プロセス全体からの視点も重要となり、ソフトウェアを使用するプロセスではそのソフトウェアが有効であっても、開発プロセス全体の生産性を向上させるとは限らず、個人のソフトウェアへの評価が高くても、使用者全員の評価が高いとは限らないので、開発プロセス全体からの視点から評価を行う。

ソフトウェア導入前の品質確認要求事項

品質確認要求事項	
	概要
2.1	現状分析
	ソフトウェア導入の目的を明確にするために現状の分析を行う。 ・対象プロジェクトの全体図、現状把握 ・現在の問題点、課題 ・使用者のスキル、ポジション ・・・など その上でソフトウェアに求める要件・非機能要件を抽出する。 (特に非機能要件はあいまいにされることが多いので注意) ソフトウェアが使用される環境についても確認をする。 (特殊な環境の場合には要注意)
2.2	ソフトウェア導入の目的
	現状分析を行い下記のような観点からソフトウェアを導入する目的を明確にする。 ・ソフトウェアに期待すること ・ソフトウェア使用によるプロジェクト全体の理想目標と最低限クリアすべき目標 ・組織的な問題の解決等 ソフトウェアに対する信頼性・安全性の水準を決定する。
2.3	ソフトウェアの評価
	ソフトウェアの評価を行い、導入目的に合致するか確認する。 (客観的な評価を行い、できるだけ評価結果は数値化し、メリット・デメリットの両面から評価) 類似ソフトウェアが複数存在する場合は比較検討を行う。
2.3.1	ソフトウェアの導入効果の見積もり
	導入前に導入効果を見積もるのは難しいが、導入事例を参考にする。 (これまでの経験が基となるが、ソフトウェア導入の機会自体が頻繁にあるわけではない)
2.3.2	ソフトウェアの品質の評価
	以下の項目を評価内容として考慮する。 機能性 移行性 操作性 互換性 第三者評価・導入事例

2.3.2.1	機能性の評価
	ソフトウェアに求める要件、非機能要件との適合性を確認するために評価期間を設けたり、ソフトウェアの評価版の使用を検討する。(導入後に機能不足が発見されるとプロジェクト全体に影響があるため、特に注意して評価) 必要であれば、テストケースを作成し確認する。
2.3.2.2	移行性の評価
	ソフトウェアのセットアップ手順を確認する。 ソフトウェアを入れ替える場合は既存ソフトウェアから新ソフトウェアへの移行の難易度を評価する。 (既存開発環境から新開発環境への移行についても同様) 使用者が容易に新ソフトウェアに対応できるかも考慮して円滑に新ソフトウェアが導入できるかどうかを評価する。
2.3.2.3	操作性の評価
	操作方法の難易度により教育コストも変化し、操作性に問題があると、作業効率低下や誤った使用により作りこまれる不具合などが考えられ、操作性は開発効率に関係するので、一般的なソフトウェアと同様に操作できることや直観的に操作できることを確認する。(結果出力が見やすいことも重要)
2.3.2.4	互換性の評価
	既存の開発環境との相性、同時に使用される他のソフトウェアとの組み合わせに問題はないかを確認する。 (ソフトウェアを導入することによる開発プロセスの複雑化は避ける。)
2.3.2.5	第三者評価・導入事例の評価
	導入事例、利用実績なども参考にして客観的、専門的な評価結果として利用し、評価する。
2.3.3	適用評価
	下記のような観点からソフトウェアを導入することでどのような効果があるかを評価する。 ・過去プロジェクトと比較 ・現在進行中のプロジェクトと比較
2.3.3.1	過去の開発プロジェクトに適用
	過去の開発プロジェクトにソフトウェアを使用し、その結果を評価する。 (ソフトウェアの適用前後の工数を確認することで見積もりの精度があがる)
2.3.3.2	現在進行中のプロジェクトに適用
	現在進行中のプロジェクトにソフトウェアを使用し、その結果を評価する。 (ソフトウェアを使用しないプロセスと比較することでプロセスの差異が明確になり、ソフトウェアの導入方法の試行という位置づけにもなる)
2.4	導入コストの評価
	ソフトウェア導入に関わるコストを評価する。 ・購入コスト ・移行コスト ・教育コスト ・保守コスト
2.4.1	購入コストの評価
	購入費やライセンス費について評価する。
2.4.2	移行コストの評価
	移行にかかるコストを評価する。 (ソフトウェア、開発環境の移行作業だけではなく、これまでの成果物が利用できるかにも左右される)
2.4.3	教育コストの評価
	教育にコストをかけることで使用者の新ソフトウェアに対する不安要素が減りソフトウェアが活用され、生産性があがることが期待でき、一時的にコストが発生してもトータルのコスト削減となるケースもあるので、 使用者が円滑にソフトウェアを使用するために必要なコストを評価する。 (ソフトウェアの使用に前提知識が必要な場合、その知識の有無、知識習得の難易度によっても教育コストは変動)
2.4.4	保守コストの評価
	メンテナンスをだれが、どのように行うのか、といったメンテナンス方法について考慮し、ソフトウェアを長期間適切に使用するのメンテナンスに必要なコストを評価する。 (OSSやフリーソフトでも保守コストが発生する場合がある)
2.5	制限事項の確認
	ソフトウェアの制限事項を確認し、その内容が既存開発環境、既存成果物に影響がないか、影響があっても許容範囲内であるかを評価する。
2.6	導入者と使用者の同意
	導入者と使用者が異なる場合には両者のコンセンサスが重要になるので、ソフトウェアの導入目的をしっかりと共有する。 同意が得られていないと、希望するソフトウェアが導入されない、導入されたソフトウェアが活用されないといった状況が起こりえるので、使用者以外の関係者にも同意を得る。 必要であればソフトウェアの評価結果を活用する。

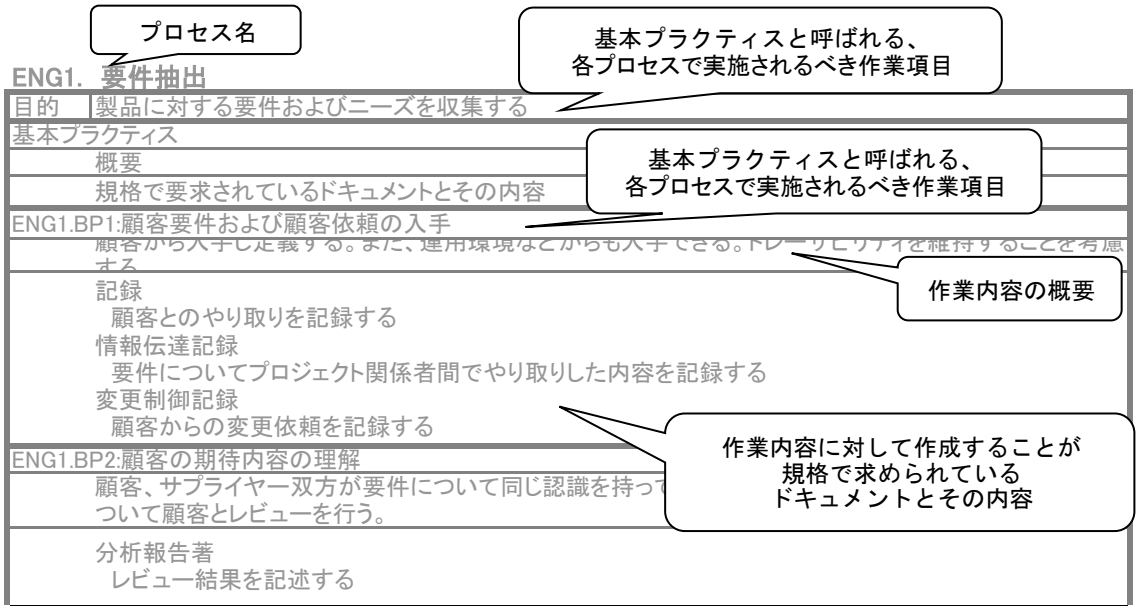
2.7	障害発生時の対策の策定
	障害発生時の対策を定めておき、影響を最小限に抑える。 障害発生を抑えるための未然防止策と、障害発生時に影響を最小限に抑える事後対策の2種類の対策を行う。
2.7.1	未然防止策の策定
	使用者ミスによる障害を抑えるためのソフトウェア使用方法の周知する。 定期的なソフトウェアのバージョン確認と必要なメンテナンスを行う。 ソフトウェアの出力結果をチェックする仕組みを作成する。 類似ソフトウェアの既知不具合の傾向を把握する。
2.7.2	事後対策の策定
	関係者への状況通知を行う。 原因の究明と対策を行う。 障害による影響の拡大を阻止する。 再発・類似障害発生を防止する。

ソフトウェア導入後の品質確認要求事項

品質確認要求事項	
	概要
3.1	ソフトウェアの導入支援
	使用者のソフトウェアに対する不安感を軽減させ、組織内へのソフトウェア定着を促進させる為に 開発中に適切な導入支援を行うことで、ソフトウェアによる作業効率アップを図る。 ソフトウェアの導入状況が思わしくない場合は改善策を検討し、実施する。
3.2	保守
	脆弱性の解消等、更新が必要となる変更が行われている可能性があるのでバージョンアップ情報を確認する。 バージョンアップする場合には既存バージョンとの互換性を確認する。 （開発環境の変化により、ソフトウェアの使われ方が変わることもある。）
3.3	ソフトウェアの再評価
	今後別ソフトウェアを導入するための見積もりの精度アップにつながるので、実際の導入結果から定期的にソフトウェアの効果を測定し、見積もりと比較する。 （定期的にソフトウェアの評価を行い使用者に示すことで、ソフトウェアの効果を実感することができ、一方でソフトウェアの評価結果をソフトウェアの開発者に提供することで、評価結果を反映したソフトウェアのバージョンアップが検討される可能性もある。）
3.4	ソフトウェア導入の評価
	今後の見積もり精度を高める為に導入後のソフトウェアの評価結果からソフトウェアを導入したことへの判断を行う。

実験で作成されたデータ等の資料
要求事項一覧

Automotive SPICEでの要求事項


プロセス名
基本プラクティスと呼ばれる、
各プロセスで実施されるべき作業項目
ENG1. 要件抽出
目的
製品に対する要件およびニーズを収集する
基本プラクティス
概要
規格で要求されているドキュメントとその内容
基本プラクティスと呼ばれる、
各プロセスで実施されるべき作業項目
ENG1.BP1:顧客要件および顧客依頼の入手
作業内容の概要
記録
顧客とのやり取りを記録する
情報伝達記録
要件についてプロジェクト関係者間でやり取りした内容を記録する
変更制御記録
顧客からの変更依頼を記録する
作業内容に対して作成することが
規格で求められている
ドキュメントとその内容
ENG1.BP2:顧客の期待内容の理解
顧客、サプライヤー双方が要件について同じ認識を持って
ついて顧客とレビューを行う。
分析報告書
レビュー結果を記述する

ENG1. 要件抽出

目的	製品に対する要件およびニーズを収集する
基本プラクティス	
	概要
	規格で要求されているドキュメントとその内容
ENG1.BP1:顧客要件および顧客依頼の入手	
	顧客から入手し定義する。また、運用環境などからも入手できる。トレーサビリティを維持することを考慮す
	記録 顧客とのやり取りを記録する。 情報伝達記録 要件についてプロジェクト関係者間でやり取りした内容を記録する。 変更制御記録 顧客からの変更依頼を記録する。
ENG1.BP2:顧客の期待内容の理解	
	顧客、サプライヤー双方が要件について同じ認識を持っていることを確認する。顧客要件および依頼について顧客とレビューを行う。
	分析報告著 レビュー結果を記述する。
ENG1.BP3:要件の合意	
	要件について関係するすべての部署から明確な合意を得る。
	分析報告書 合意された顧客要件を記述する。
ENG1.BP4:顧客要件ベースラインの確立	
	顧客要件をベースラインとして確立する。明示されていないが必要となる要件を洗い出し、ベースラインに含める。
	変更制御記録 要件変更に伴ってベースラインが変更された場合にその内容を記述する。 分析報告書 顧客要件のベースラインを記述する。
ENG1.BP5:顧客要件の変更の管理	
	ベースラインに合わせて顧客要件の変更を管理し、変更方法を確立する。変更には顧客によるものの他に技術的なものも含まれる。変更については影響、リスクを評価する。
	変更制御記録 要件の変更内容をベースラインに盛り込む。 変更の方法を記述する。 分析報告書 変更について評価された内容を記述する。 リスク管理計画書 識別された変更とその影響について記述する。
ENG1.BP6:顧客とサプライヤー間の照会情報伝達の仕組みの確立	
	顧客が要件について情報を把握できる仕組みを確立し、伝達する。顧客との合同会議など。
	記録 顧客とのやり取りを記録する。

ENG2. システム要件分析

目的	製品の要件に基づいてシステム要件を定義する。システム要件は製品に対する要件を網羅していること。
基本プラクティス	
	概要
	規格で要求されているドキュメントとその内容
ENG2.BP1:システム要件の識別	
	顧客要件からシステムに必要な機能および能力をシステム要件としてシステム要件仕様書に記述する。システム要件は識別可能なようにする。
	アプリケーションパラメーター システムのパラメーターを定義する。 システム要件仕様書 顧客要件からシステム要件を確立する。
ENG2.BP2:システム要件の分析	
	識別されたシステム要件を実現可能性、リスク、テスト可能性の観点から分析する。分析結果よりシステムの検証基準も作成される。
	分析報告書 システム要件の分析結果を記述する。 検証基準 システム要件仕様書を作成するための条件を定義する。 システム要件仕様書のレビュー基準も作成する。
ENG2.BP3:運用環境における影響の特定	
	システム要件と運用環境において他のコンポーネントとのインターフェース、影響について分析する。 要件同士の矛盾はないか、システム要件は顧客要件を実現するか、など。 要件がさまざまな条件下で運用環境に与える影響を分析する。
	インターフェース要件仕様書 他のコンポーネントとのインターフェースを記述する
ENG2.BP4:システム要件の優先順位付けおよび分類	
	識別、分析されたシステム要件に優先順位を付け、分類し、リリース予定にマッピングする。
	分析報告書 システム要件の分析結果を記述する。 検証基準 システム要件仕様書を作成するための条件を定義する。 システム要件仕様書のレビュー基準も作成する。 リリース計画書 分析、分類されたシステム要件のリリース計画を記述する。
ENG2.BP5:システム要件の評価および更新	
	システム要件の内容を評価する。変更される場合はコスト、スケジュールなどの面から評価し、システム要件仕様書を更新する。顧客はシステム要件仕様書を承認する。 システム要件の評価は以下の点を評価する。 ・システム要件は顧客要件を網羅していること ・システム要件は少なくとも1つの顧客要件とリンクしていること ・システム要件間に矛盾がないこと ・検証基準、合格基準が明確であること
	リリース計画書 分析、分類されたシステム要件のリリース計画を記述する。 変更制御記録 変更内容、担当者などの変更内容属性を記述する。 分析報告書 システム要件の分析結果を記述する。 システム要件仕様書 顧客要件からシステム要件を確立する。
ENG2.BP6:システム要件に対する顧客要件の一貫性および双方向トレーサビリティの確証	
	システム要件と顧客要件の一貫性を確実にする。双方向のトレーサビリティを確立することで一貫性を確保する。
	トレーサビリティ記録 システム要件、顧客要件とのマッピングを記述する。
ENG2.BP7:システム要件の情報伝達	
	システム要件および変更内容を関係するすべての部署に周知するための方法を確立し、伝達する。
	情報伝達記録 要件についてプロジェクト関係者間でやり取りした内容を記録する。

ENG3. システムアーキテクチャ設計

目的	システム要件に基づいてシステムアーキテクチャを設計する。システムアーキテクチャはシステム要件を網羅していること。
基本プラクティス	
	概要
	規格で要求されているドキュメントとその内容
ENG3.BP1:システムアーキテクチャ設計書の定義	
	定義されたシステム要件を満たすシステムアーキテクチャを設計する。要件には機能の他に非機能要件も含まれる。 以下の点を明確にする。 ・システム要素の概要、性能 ・システム要素間の関係 ・セキュリティ要件　など
	システムアーキテクチャ設計書 システムアーキテクチャを記述する。
ENG3.BP2:システム要件の割り当て	
	すべてのシステム要件をシステムアーキテクチャ設計の要素に割り当てる。
	システムアーキテクチャ設計書 システム要件をシステムアーキテクチャにマッピングする。
ENG3.BP3:インターフェースの定義	
	各システム要素の内部・外部インターフェースを識別、設計し、文書化する。
	システムアーキテクチャ設計書 タスク間やプロセス間の通信、システムアーキテクチャ間の依存関係を記述する。
ENG3.BP4:検証基準の作成	
	システムアーキテクチャ設計より機能、非機能要件に対するシステムの各要素の検証基準を定義する。
	検証基準 システムテストのテスト仕様書を作成するための条件を定義する。 システムアーテキチャ設計書のレビュー基準も作成する。
ENG3.BP5:システムアーキテクチャ設計の検証	
	システムアーキテクチャ設計のレビューを行う。システム要件を網羅していることを確実にする。 システムアーキテクチャ設計の評価は以下の点を評価する。 ・システムアーキテクチャはシステム要件を網羅していること ・検証基準が明確であること
	システムアーキテクチャ設計書 定義されたシステム要件を満たすシステムアーキテクチャを記述する。 検証結果 システムアーキテクチャ設計書のレビュー基準およびレビュー結果を記述する。
ENG3.BP6:システムアーキテクチャ設計に対するシステム要件の一貫性および双方向トレーサビリティの確証	
	システムアーキテクチャ設計とシステム要件の一貫性を確実にする。双方向のトレーサビリティを確立することで一貫性を確保する。
	トレーサビリティ記録 システム要件−システムアーキテクチャ設計間のマッピングを記述する。
ENG3.BP7:システムアーキテクチャ設計の情報伝達	
	システムアーキテクチャ設計を関係するすべての部署に周知するための仕組みを確立し、伝達する。
	構成品目 プロセスの成果物を構成品目として記述する。 情報伝達記録 要件についてプロジェクト関係者間でやり取りした内容を記録する。

ENG4. ソフトウェア要件分析

目的	システム要件に基づいてソフトウェア要件を定義する。ソフトウェア要件はシステム要件、システムアーキテクチャ設計を網羅していること。
基本プラクティス	
	概要
	規格で要求されているドキュメントとその内容
ENG4.BP1:ソフトウェア要件の識別	
	システム要件分析、システムアーキテクチャ設計から、ソフトウェア要件を識別し、ソフトウェア要件仕様書として文書化する。 以下の点を明確にする。 ・ソフトウェアの機能 ・ソフトウェア間の関係 ・インターフェース要件 ・セキュリティ要件 ・パフォーマンス要件（リソース消費） ・制約事項　など
	アプリケーションパラメーター ソフトウェア要件に対するアプリケーションパラメーターを記述する。 トレーサビリティ記録 システム要件、システムアーキテクチャとのマッピングを記述する。 インターフェース要件仕様書 ソフトウェア要件間のインターフェースを定義する。 ソフトウェア要件仕様書 機能要件、非機能要件を定義する。
ENG4.BP2:ソフトウェア要件の分析	
	ソフトウェア要件を以下の観点から分析する。 ・実現可能性：技術的、スケジュール的、コスト的に実現可能か ・リスク：システム全体に与える影響はどうか ・テスト可能性：要件は検証、テスト可能であるか
	分析報告書 分析結果を記述する。 ソフトウェア要件仕様書 分析結果を記述する。 検証基準 ソフトウェアテスト仕様書を作成するための条件を定義する。 ソフトウェア要件仕様書のレビュー基準も作成する。
ENG4.BP3:運用環境における影響の特定	
	ソフトウェア要件が運用環境において及ぼす影響と他のコンポーネントとのインターフェースについて分析する。
	分析報告書 分析結果を記述する。
ENG4.BP4:ソフトウェア要件の優先順位付けおよび分類	
	ソフトウェア要件を識別する。識別したソフトウェア要件に優先順位を付け、リリース予定にマッピングする。
	分析報告書 分析結果を記述する。 ソフトウェア要件仕様書 分析結果を記述する。 検証基準 ソフトウェアテスト仕様書を作成するための条件を定義する。 ソフトウェア要件仕様書のレビュー基準も作成する。 リリース計画書 リリースごとに納入される機能を記述する。 ソフトウェア要件仕様書 分析結果を記述する。
ENG4.BP5:ソフトウェア要件の評価および更新	
	ソフトウェア要件の内容を評価する。変更される場合はコスト、スケジュールなどの面から評価し、ソフトウェア要件仕様書を更新する。顧客はソフトウェア要件仕様書を承認する。 ソフトウェア要件の評価は以下の点を評価する。 ・ソフトウェア要件はシステム要件を網羅していること ・ソフトウェア要件は少なくとも1つのシステム要件とリンクしていること ・システム要件間に矛盾がないこと ・検証基準が明確であること
	リリース計画書 リリースごとに納入される機能を記述する。 ソフトウェア要件仕様書 分析結果を記述する。 変更制御記録 変更内容、担当者などの変更内容属性を記述する。 分析報告書 要件をコスト、スケジュールなどの観点から分析した結果を記述する。

ENG4.BP6:ソフトウェア要件に対するシステム要件の一貫性および双方向トレーサビリティの確証
ソフトウェア要件とシステム要件の一貫性を確実にする。双方向のトレーサビリティを確立することで一貫性を確
トレーサビリティ記録 システム要件、システムアーキテクチャとのマッピングを記述する。 インターフェース要件仕様書 システム要件、システムアーキテクチャとのマッピングを記述する。
ENG4.BP7:ソフトウェア要件に対するシステムアーキテクチャ設計の一貫性および双方向トレーサビリティの確証
ソフトウェア要件とシステムアーキテクチャ設計の一貫性を確実にする。双方向のトレーサビリティを確立すること で一貫性を確保する。
トレーサビリティ記録 システム要件、システムアーキテクチャとのマッピングを記述する。 インターフェース要件仕様書 システム要件、システムアーキテクチャとのマッピングを記述する。
ENG4.BP8:ソフトウェア要件の情報伝達
ソフトウェア要件および変更内容を関係するすべての部署に周知するための方法を確立し、伝達する。
構成品目 プロセスの成果物を構成品目として記述する。 情報伝達記録 要件についてプロジェクト関係者間でやり取りした内容を記録する。

ENG5. ソフトウェア設計

目的	ソフトウェア要件に基づいてソフトウェア設計を行う。ソフトウェア設計はソフトウェア要件を網羅しているこ
基本プラクティス	
	概要
	規格で要求されているドキュメントとその内容
ENG5.BP1:ソフトウェアーキテクチャ設計書の作成	
	定義されたソフトウェア要件を満たすソフトウェアーキテクチャを設計する。ソフトウェアコンポーネントの識別や内部/外部インターフェースなども含める。
	ソフトウェアーキテクチャ設計書 ソフトウェアコンポーネントを記述する。 定義されたソフトウェア要件を満たすソフトウェアーキテクチャを記述する。
ENG5.BP2:ソフトウェア要件の割り当て	
	すべてのソフトウェア要件をソフトウェアーキテクチャ設計のコンポーネントに割り当て、確実に網羅する。
	ソフトウェアーキテクチャ設計書 ソフトウェア要件をソフトウェアコンポーネントにマッピングする。 ソフトウェア詳細設計書 ソフトウェアコンポーネントをモジュールにマッピングする。 トレーサビリティ記録 ソフトウェア要件-ソフトウェアーキテクチャ設計、 ソフトウェアーキテクチャ設計-ソフトウェア詳細設計のマッピングを記述する。
ENG5.BP3:インターフェースの定義	
	ソフトウェアコンポーネントとの内部インターフェース、外部ソフトウェアコンポーネントとの外部インターフェースを識別、設計し、文書化する。
	ソフトウェアーキテクチャ設計書 タスク間やプロセス間の通信、ソフトウェアコンポーネント間の依存関係を記述。する ソフトウェア詳細設計書 モジュール間のインターフェースを記述する。
ENG5.BP4:動的な振る舞いの記述	
	ソフトウェアコンポーネント間の依存関係を表す、動的振る舞い、相互作用を評価し、文書化する。
	ソフトウェアーキテクチャ設計書 動的な振る舞いを記述する。 ソフトウェア詳細設計書 動的な振る舞いを明確にする。 検証結果 ソフトウェアーキテクチャ設定、ソフトウェア詳細設計のレビュー基準およびレビュー結果を記述する。
ENG5.BP5:リソース消費目標の定義	
	ソフトウェアコンポーネントのリソース消費目標を決定し、文書化する。
	ソフトウェアーキテクチャ設計書 リソースの消費目標を記述する。 ソフトウェア詳細設計書 リソースの消費目標を明確にする。 検証結果 ソフトウェアーキテクチャ設定、ソフトウェア詳細設計のレビュー基準およびレビュー結果を記述する。
ENG5.BP6:詳細設計書の作成	
	ソフトアーキテクチャ設計からソフトウェアコンポーネントを細分化していき、ソフトウェアコンポーネントの詳細設計を行う。トレーサビリティは維持すること。
	ソフトウェア詳細設計書 各モジュールについての詳細設計を記述する。 検証結果 ソフトウェアーキテクチャ設定、ソフトウェア詳細設計のレビュー基準およびレビュー結果を記述する。 検証基準 ソフトウェア統合テストのテスト仕様書を作成するための条件を定義する。 ソフトウェアテキチャ設計書、ソフトウェア詳細設計書のレビュー基準も作成する。
ENG5.BP7:検証基準の作成	
	ソフトウェアアーキテクチャ設計、動的な振る舞い、インターフェース、リソース消費量に関する、各コンポーネントの検証基準を定義する。
	ソフトウェア詳細設計書 各モジュールについての詳細設計を記述する。 検証結果 ソフトウェアーキテクチャ設定、ソフトウェア詳細設計のレビュー基準およびレビュー結果を記述する。 検証基準 ソフトウェア統合テストのテスト仕様書を作成するための条件を定義する。 ソフトウェアテキチャ設計書、ソフトウェア詳細設計書のレビュー基準も作成する。

ENG5.BP8:ソフトウェア設計の検証
ソフトウェア設計がソフトウェア要件を網羅していることを確実にする。
ソフトウェアーキテクチャ設計書 動的な振る舞い、リソース消費目標を記述する。 ソフトウェア詳細設計書 各モジュールについての詳細設計を記述する。 検証結果 ソフトウェアーキテクチャ設定、ソフトウェア詳細設計のレビュー基準およびレビュー結果を記述する。 検証基準 ソフトウェア統合テストのテスト仕様書を作成するための条件を定義する。 ソフトウェアテキチャ設計書、ソフトウェア詳細設計書のレビュー基準も作成する。
ENG5.BP9:ソフトウェアーキテクチャ設計に対するソフトウェア要件の一貫性及び双方向トレーサビリティの確証
ソフトウェアーキテクチャ設計とソフトウェア要件の一貫性を確実にする。双方向のトレーサビリティを確立することで一貫性を確保する。
ソフトウェアーキテクチャ設計書 ソフトウェア要件−ソフトウェアーキテクチャ設計間のマッピングを記述する。 ソフトウェア詳細設計書 ソフトウェアーキテクチャ設計−ソフトウェア詳細設計間のマッピングを記述する。 トレーサビリティ記録 ソフトウェア要件−ソフトウェアーキテクチャ設計、 ソフトウェアーキテクチャ設計−ソフトウェア詳細設計のマッピングを記述する。
ENG5.BP10:ソフトウェア詳細設計に対するソフトウェアテキチャ設計の一貫性及び双方向トレーサビリティの確証
ソフトウェア詳細設計とソフトウェアーキテクチャ設計の一貫性を確実にする。双方向のトレーサビリティを確立することで一貫性を確保する。
トレーサビリティ記録 ソフトウェア要件−ソフトウェアーキテクチャ設計、 ソフトウェアーキテクチャ設計−ソフトウェア詳細設計のマッピングを記述する。

ENG6. ソフトウェア構築

目的　ソフトウェア設計と整合性があり、検証されたソフトウェアユニットを生成する。
基本プラクティス
概要
規格で要求されているドキュメントとその内容
ENG6.BP1:ユニット検証戦略の定義
ソフトウェアユニットを検証するための戦略を作成する。各種検証手法など。
テスト計画書 各種検証手法などを定義する。 テスト仕様書 ユニットテストのテストケースやテスト手順などを記述する。
ENG6.BP2:ソフトウェアユニットの分析
ソフトウェアユニットを互換性、相互作用、クリティカリティ、技術的な複雑性、リスク、テスト可能性の観点から分析する。
–
ENG6.BP3:ソフトウェアユニットの優先順位付けおよび分類
識別されたソフトウェアユニットに優先順位を付け、分類し、リリースにマッピングする。
–
ENG6.BP4:ソフトウェアユニットの作成
ソフトウェアユニットを作成する。
ソフトウェアユニット ソフトウェア詳細設計に従ってソフトウェアユニットを作成する。
ENG6.BP5:ユニット検証基準の作成
ソフトウェアユニットが設計、機能、非機能要件を満たしているかの検証基準を作成し、文書化する。
検証基準 ソフトウェア詳細設計からソフトウェアユニットをテストするためのテスト仕様書を定義する。 ソースコードレビュー基準も用意する。
ENG6.BP6:ソフトウェアユニットの検証
検証戦略に従い、ユニット検証基準よりソフトウェアユニットを検証する。コードレビュー含まれる。
テスト仕様書 ユニットテストのテストケースやテスト手順を記述する。 テスト結果 ソフトウェアユニットテストの結果を記録する。 検証結果 コードレビューの基準およびレビュー結果を記述する。
ENG6.BP7:ユニット検証結果の記録
ユニット検証結果を文書化し、関係する部署に伝達する。
テスト結果 ソフトウェアユニットテストの結果を記録する。 検証結果 コードレビューの基準およびレビュー結果を記述する。
ENG6.BP8:ソフトウェアユニットに対するソフトウェア詳細設計の一貫性および双方向トレーサビリティの確証
ソフトウェアユニットとソフトウェア詳細設計の一貫性を確実にする。双方向のトレーサビリティを確立することで一貫性を確保する。
トレーサビリティ記録 ソフトウェア詳細設計–ソフトウェアユニット、 ソフトウェアユニット–ソフトウェアユニットテスト仕様、 ソフトウェア要件–ソフトウェアユニット間のマッピングを記述する。
ENG6.BP9:ソフトウェアユニットに対するソフトウェア要件の一貫性および双方向トレーサビリティの確証
ソフトウェアユニットとソフトウェア要件の一貫性を確実にする。双方向のトレーサビリティを確立することで一貫性を確保する。
トレーサビリティ記録 ソフトウェア詳細設計–ソフトウェアユニット、 ソフトウェアユニット–ソフトウェアユニットテスト仕様、 ソフトウェア要件–ソフトウェアユニット間のマッピングを記述する。
ENG6.BP10:ソフトウェアユニットテスト仕様に対するソフトウェアユニットの一貫性および双方向トレーサビリティの確証
ソフトウェアユニットテスト仕様とソフトウェアユニットの一貫性を確実にする。双方向のトレーサビリティを確立することで一貫性を確保する。
トレーサビリティ記録 ソフトウェア詳細設計–ソフトウェアユニット、 ソフトウェアユニット–ソフトウェアユニットテスト仕様、 ソフトウェア要件–ソフトウェアユニット間のマッピングを記述する。

ENG7. ソフトウェア統合テスト

目的　ソフトウェアユニットを統合し、ソフトウェアユニット間の検証を行う。
基本プラクティス
概要
規格で要求されているドキュメントとその内容
ENG7.BP1:ソフトウェア統合戦略の策定
ソフトウェア品目を統合するためのリリース戦略、統合手順を策定する。
テスト計画書 統合手順、統合手法を定義する。
ENG7.BP2:ソフトウェア統合テスト戦略の策定
統合されたソフトウェア品目をテストするための戦略を策定する。
テスト計画書 統合テスト手順を定義する。
ENG7.BP3:ソフトウェア統合テスト仕様書の作成
ソフトウェア統合テストのテスト仕様書を作成する。テストケースはソフトウェアーキテクチャ設計、詳細設計との適合性を実証する必要がある。
テスト計画書 ソフトウェアアーキテクチャ設計と詳細設計から、統合のステップごとに検証するためのテスト仕様書を作成 テスト仕様書 統合のステップごとに検証するためのテスト仕様を記述する。
ENG7.BP4:ソフトウェアユニットおよびソフトウェア品目の統合
ソフトウェア統合戦略に従って、ソフトウェアユニットをソフトウェア品目に、ソフトウェア品目を統合ソフトウェアに統合する。
ビルドリスト 統合や再統合のためにソフトウェア品目の統合リスト、統合手順を記述する。 ソフトウェア品目 統合ソフトウェア
ENG7.BP5:統合ソフトウェアの検証
ソフトウェア統合テスト仕様のテストケースに従って、各統合ソフトウェア品目を検証する。
テスト結果 ソフトウェア統合テストの結果を記述する。
ENG7.BP6:ソフトウェア統合テスト結果の記録
ソフトウェア統合テスト結果を文書化し、関係する部署に伝達する。
テスト結果 ソフトウェア統合テストの結果を記述する。
ENG7.BP7:ソフトウェア統合テスト仕様に対するソフトウェアアーキテクチャ設計、詳細設計の一貫性および双方向トレーサビリティの確証
ソフトウェア統合テスト仕様とソフトウェアアーキテクチャ設計、詳細設計の一貫性を確実にする。双方向のトレーサビリティを確立することで一貫性を確保する。
トレーサビリティ記録 ソフトウェアアーキテクチャ設計・詳細設計−ソフトウェア統合テスト仕様間のマッピングを記述する。 ビルドリスト 統合や再統合のためにソフトウェア品目の統合リスト、統合手順を記述する。
ENG7.BP8:回帰テスト戦略の策定および回帰テストの実施
変更されたソフトウェア品目が統合された場合、回帰テストを行うための戦略を策定する。回帰テストを実施し、その結果は文書化する。
テスト計画書 回帰テスト手法を定義する。 ビルドリスト 統合や再統合のためにソフトウェア品目の統合リスト、統合手順を記述する。

ENG8. ソフトウェアテスト

目的	統合ソフトウェアがソフトウェア要件を満たしている事を確認する。
基本プラクティス	
	概要
	規格で要求されているドキュメントとその内容
ENG8.BP1:ソフトウェアテスト戦略の策定	
	ソフトウェアのリリースに基づき、ソフトウェアの全体的なテスト計画を定義する。テスト手法、テスト環境など。テスト自動化計画やテストケース設計手法を含むこともある。
	テスト計画書 ソフトウェアテスト手法を記述する。
ENG8.BP2:ソフトウェアテスト仕様書の作成	
	テストケースを含むソフトウェアテスト仕様書を作成する。テストケースは要件、シナリオ、リスクなどの観点から作成する。
	テスト計画書 テスト仕様書を作成する。 テスト仕様書 ソフトウェア要件を検証するためのテスト仕様書を作成する。
ENG8.BP3:統合ソフトウェアの検証	
	テスト仕様に従って、ソフトウェアの検証を行う。テストログを出力するなど可視化する。
	テスト結果 ソフトウェアテストの結果を記録する。
ENG8.BP4:ソフトウェアテスト結果の記録	
	ソフトウェアテストの結果を文書化し、関係する部署に伝達する。
	テスト結果 ソフトウェアテストの結果を記録する。
ENG8.BP5:ソフトウェアテスト仕様に対するソフトウェア要件の一貫性および双方向トレーサビリティの確証	
	ソフトウェアテスト仕様とソフトウェア要件の一貫性を確実にする。双方向のトレーサビリティを確立することで一貫性を確保する。
	トレーサビリティ記録 ソフトウェア要件−ソフトウェアテスト仕様間のマッピングを記述する。
ENG8.BP6:回帰テスト戦略の策定および回帰テストの実施	
	ソフトウェア品目に変更がある場合、回帰テストを行うための戦略を策定する。回帰テストを実施し、その結果は文書化する。
	テスト計画書 回帰テスト手法を記述する。

ENG9. システム統合テスト

目的　ソフトウェアを統合し、ソフトウェア間の検証を行う。
基本プラクティス
概要
規格で要求されているドキュメントとその内容
ENG9.BP1:システム統合戦略の策定
システム要素を統合するためのリリース戦略、統合手順を策定する。
テスト計画書 統合手順、統合手法を定義する。
ENG9.BP2:システム統合テスト戦略の策定
統合システムをテストするための戦略を策定する。
テスト計画書 統合テスト手順を定義する。
ENG9.BP3:システム統合テスト仕様書の作成
システム統合テストのテスト仕様書を作成する。テストケースはシステムアーキテクチャ設計との適合性を実証する必要がある。
テスト計画書 システムアーキテクチャ設計から、統合のステップごとに検証するためのテスト仕様書を作成する。 テスト仕様書 統合のステップごとに検証するためのテスト仕様を記述する。
ENG9.BP4:システム要素の統合
システム統合戦略に従って、システム要素を統合システムに統合する。
システム
ENG9.BP5:統合システムの検証
システム統合テスト仕様のテストケースに従って、各統合システム要素を検証する。
テスト結果 システム統合テストの結果を記述する。
ENG9.BP6:システム統合テスト結果の記録
システム統合テスト結果を文書化し、関係する部署に伝達する。
テスト結果 システム統合テストの結果を記述する。
ENG9.BP7:システム統合テスト仕様に対するシステムアーキテクチャ設計の一貫性および双方向トレーサビリティの
システム統合テスト仕様とシステムアーキテクチャ設計の一貫性を確実にする。双方向のトレーサビリティを確立することで一貫性を確保する。
トレーサビリティ記録 システムアーキテクチャ設計−システム統合テスト仕様間のマッピングを記述する。
ENG9.BP8:回帰テスト戦略の策定および回帰テストの実施
変更されたシステム要素が統合された場合、回帰テストを行うための戦略を策定する。回帰テストを実施し、その結果は文書化する。
テスト計画書 回帰テスト手法を定義する。

ENG10. システムテスト

目的	システムがシステム要件を満たし、納入準備がなされていることを確認する。
基本プラクティス	
	概要
	規格で要求されているドキュメントとその内容
ENG10.BP1:システムテスト戦略の策定	
	システムのリリースに基づき、システムの全体的なテスト計画を定義する。テスト手法、テスト環境など。
	テスト計画書 システムテスト手法を記述する
ENG10.BP2:システムテスト仕様書の作成	
	テストケースを含むシステムテスト仕様書を作成する。テストケースは要件、シナリオ、リスクなどの観点から作成する。
	テスト計画書 テスト仕様書を作成する。 テスト仕様書 システム要件を検証するためのテスト仕様書を作成する。
ENG10.BP3:統合システムの検証	
	テスト仕様に従って、システムの検証を行う。テストログを出力するなど可視化する。
	テスト結果 システムテストの結果を記録する。
ENG10.BP4:システムテスト結果の記録	
	システムテストの結果を文書化し、関係する部署に伝達する。
	テスト結果 システムテストの結果を記録する。
ENG10.BP5:システムテスト仕様に対するシステム要件の一貫性および双方向トレーサビリティの確証	
	システムテスト仕様とシステム要件の一貫性を確実にする。双方向のトレーサビリティを確立することで一貫性を確保する。
	トレーサビリティ記録 システム要件-システムテスト仕様間のマッピングを記述する。
ENG10.BP6:回帰テスト戦略の策定および回帰テストの実施	
	システム要素に変更がある場合、回帰テストを行うための戦略を策定する。回帰テストを実施し、その結果は文書化する。
	テスト計画書 回帰テスト手法を記述する。

SUP1. 品質保証
目的　プロセスの各プロセスがあらかじめ定義された規定、計画に適合していることを保証する。
基本プラクティス
概要
規格で要求されているドキュメントとその内容
SUP1.BP1:プロジェクトの品質保証戦略の策定
プロジェクトの品質保証を実施するための戦略を策定する。
品質計画書 品質目標、検証基準などプロジェクトの品質保証に関する事項を記述する。
SUP1.BP2:品質保証が独立し実施および報告されることを確実にする組織構造の作成および維持
品質保証活動はプロジェクトから独立して行う。活動と報告ルートの独立性を確保する。
品質計画書 品質保証組織がプロジェクトから独立していることを記述する。 レビュー記録 レビュー結果を記録する。
SUP1.BP3:品質保証戦略に基づくプロジェクトの品質保証計画の策定および実装
品質保証戦略に従って、プロジェクトの品質計画について策定する。計画には要件、設計、コーディング、テストに対する品質保証活動と活動のスケジュールを含む。
品質計画書 品質保証活動の結果を記述する。 問題記録 検出された問題の内容、対応担当者、終結基準、終結予定日などを記述する。 品質記録 品質保証活動で作成される成果物。 レビュー記録 レビューの結果を記録する。 是正処置記録 レビューで検出された問題とその解決策を記述する。
SUP1.BP4:品質保証のエビデンスの維持
実施された品質保証活動の実施結果を記録、維持し、エビデンスとする。
品質計画書 品質保証活動の結果を記述する。 問題記録 検出された問題の内容、対応担当者、終結基準、終結予定日などを記述する。 品質記録 品質保証活動で作成される成果物。 レビュー記録 レビューの結果を記録する。 是正処置記録 検出された問題とその解決策を記述する。
SUP1.BP5:作業成果物の品質評価
品質保証計画に従って、品質評価を対象成果物に対して行う。
問題記録 検出された問題の内容、対応担当者、終結基準、終結予定日などを記述する。 品質記録 品質保証活動で作成される成果物。 レビュー記録 レビューの結果を記録する。 品質基準 チェックリストなど評価の基準となる指針。
SUP1.BP6:プロセス活動の品質保証
プロジェクトにおいて、行うべきプロセスが実施されていることを確認する。
問題記録 検出された問題の内容、対応担当者、終結基準、終結予定日などを記述する。 品質記録 品質保証活動で作成される成果物。 レビュー記録 レビューの結果を記録する。 品質基準 チェックリストなど評価の基準となる指針。

SUP1.BP7:品質保証活動のトラッキングおよび記録
品質保証活動が行われている事をトラッキングし、記録する。結果が記録されていることを確認する。
品質計画書 品質保証活動の結果を記述する。 問題記録 検出された問題の内容、対応担当者、終結基準、終結予定日などを記述する。 品質記録 品質保証活動で作成される成果物。 レビュー記録 レビューの結果を記録する。 是正処置記録 検出された問題とその解決策を記述する。 情報伝達記録 品質保証活動に関して、関係者でやり取りした文書を記録する。 品質基準 チェックリストなど評価の基準となる指針。
SUP1.BP8:品質保証活動および結果の報告
品質保証活動の実施状況、結果、などを関係部署に定期的に連絡する。
品質計画書 品質保証活動の結果を記述する。 レビューの結果を記録する。 是正処置記録 検出された問題とその解決策を記述する。 情報伝達記録 品質保証活動に関して、関係者でやり取りした文書を記録する。
SUP1.BP9:不適合事項の解決の確証
検出された問題を分析し、是正処置を行う。されには予防のための方策を立てる。
品質計画書 品質保証活動の結果を記述する。 レビューの結果を記録する。 是正処置記録 検出された問題とその解決策を記述する。 情報伝達記録 品質保証活動に関して、関係者でやり取りした文書を記録する。
SUP1.BP10:エスカレーションの仕組みの実施
問題解決の為に適切なマネジメント層にエスカレーションするための仕組みを作成する。
品質計画書 エスカレーションの仕組みを記載する。

SUP8. 構成管理

目的	プロジェクトの各作業成果物のバージョン管理、トレーサビリティ維持を行う。
基本プラクティス	
	概要
	規格で要求されているドキュメントとその内容
SUP8.BP1:構成管理戦略の策定	
	構成品目を管理するための活動、ライフサイクルモデル、手法を定義し、文書化する。
	構成管理計画書 定義された構成管理戦略を記述する。
SUP8.BP2:構成品目の識別	
	構成管理戦略に従って、対象の品目を識別する。対象には納入成果物、指定内部成果物、取得成果物、これらの作成・維持に必要な品目など。
	構成品目 識別された構成品目。 構成管理計画書 対象の構成品目とそのベースライン基準を記述する。
SUP8.BP3:構成管理システムの確立	
	構成管理戦略に従って、構成管理システムを構築する。システムに必要な手順、ツールなどを定義する。
	構成品目 識別された構成品目。 チェックイン、チェックアウト基準を明確にする。 バックアップ、アーカイブ、納入基準を明確にする。 取り扱いおよび保管ガイド 製品となるコード、文書の保管取り扱い方法について記述する。 構成管理計画書 対象の構成品目とそのベースライン基準、バックアップ、アーカイブ、納入基準を明確にする。 復旧計画書 復旧対象の対象、復旧方法などを記述する。 納入記録 納入品目、納入先、納入日などを記述する。 製品リリース承認記録 納入物のリリース日などを記述する。 変更履歴 変更内容、担当者、変更日などを記述する。 構成品目ライブラリ 構成品目のライブラリを構築する。
SUP8.BP4:ブランチ管理戦略の確立	
	ブランチが発生するプロジェクトの場合はブランチの管理戦略を策定する。
	構成品目 識別された構成品目。 チェックイン、チェックアウト基準を明確にする。 バックアップ、アーカイブ、納入基準を明確にする。 取り扱いおよび保管ガイド 製品となるコード、文書の保管取り扱い方法について記述する。 構成管理計画書 対象の構成品目とそのベースライン基準、バックアップ、アーカイブ、納入基準を明確にする。 復旧計画書 復旧対象の対象、復旧方法などを記述する。 納入記録 納入品目、納入先、納入日などを記述する。 製品リリース承認記録 納入物のリリース日などを記述する。 変更履歴 変更内容、担当者、変更日などを記述する。 構成品目ライブラリ 構成品目のライブラリを構築する。
SUP8.BP5:ベースラインの確立	
	構成管理戦略に従って、ベースラインを確立する。
	構成品目 構成品目のベースラインを明確にする。 変更履歴 構成品目の変更履歴を記述する。 構成品目ライブラリ 構成品目のライブラリを構築する。

SUP8.BP6:構成品目の記述の維持

構成品目の記録を最新にする。

構成品目
識別された構成品目。
チェックイン、チェックアウト基準を明確にする。
構成管理計画書
対象の構成品目とそのベースライン基準を明確にする。
復旧計画書
復旧対象の対象、復旧方法などを記述する。
変更履歴
変更内容、担当者、変更日などを記述する。
構成品目ライブラリ
構成品目のライブラリを構築する。

SUP8.BP7:修正およびリリースの制御

構成品目を変更する場合の手順、手法を明確にする。

構成品目
構成品目のベースラインを明確にする。
復旧計画書
復旧対象の対象、復旧方法などを記述する。
構成管理記録
構成品目のステータスを記録する。
変更履歴
構成品目の変更履歴を記述する。
構成品目ライブラリ
構成品目のライブラリを構築する。

SUP8.BP8:構成品目の履歴の維持

以前のベースラインバージョンを構築するために必要な構成品目ごとの履歴を保持する。

構成品目
構成品目のベースラインを明確にする。
復旧計画書
復旧対象の対象、復旧方法などを記述する。
変更履歴
構成品目の変更履歴を記述する。
構成品目ライブラリ
構成品目のライブラリを構築する。

SUP8.BP9:構成ステータスの報告

各構成品目のステータスを定期的に報告する。

構成管理記録
構成品目のステータスを記録する。

SUP8.BP10:構成品目に関する情報の検証

構成品目、そのベースラインに関する情報が完全であることを検証し、構成品目の一貫性を維持する。

記録
検証結果を記録する。

SUP8.BP11:構成品目のバックアップ、保管、アーカイブ、取り扱い、および納入の管理

構成品目に対する管理を適切に行うことで、構成品目の完全性、一貫性を維持する。

構成品目
バックアップ、アーカイブ、納入基準を明確にする。
取り扱いおよび保管ガイド
製品となるコード、文書の保管取り扱い方法について記述する。
構成管理計画書
対象の構成品目のバックアップ、アーカイブ、納入基準を明確にする。
復旧計画書
復旧対象の対象、復旧方法などを記述する。
納入記録
納入品目、納入先、納入日などを記述する。
構成管理記録
構成品目のステータスを記録する。
製品リリース承認記録
納入物のリリース日などを記述する。
構成品目ライブラリ
構成品目のライブラリを構築する。

SUP9. 問題解決管理

目的	プロジェクト中で発見された問題の解決を確実に行い、類似問題の予防を行う。
基本プラクティス	
	概要
	規格で要求されているドキュメントとその内容
SUP9.BP1:問題解決管理戦略の策定	
	問題を解決するための活動、ライフサイクルモデル、手法を定義し、文書化する。
	問題管理計画書 定義された問題解決管理戦略を記述する。
SUP9.BP2:一貫性のある問題解決管理手順の確立	
	問題解決管理戦略に従って、一貫性があり、トレース可能な方法で問題を検出し、解決、予防までできる仕組みを作成する。
	問題管理計画書 問題解決の仕組み、手順を記述する。
SUP9.BP3:問題の識別および記録	
	問題を識別し、記録する。
	問題記録 検出された問題について、内容、ステータス、終結予定日などを記述する。
SUP9.BP4:問題の原因および影響の調査および診断	
	適切な解決策を決定し、分類するために問題の原因と影響を調査する。
	問題記録 検出された問題について、内容、ステータス、終結予定日などを記述する。 分析報告書 問題の分析結果を記述する。 評価報告書 問題の評価結果を記述する。
SUP9.BP5:必要時に緊急の解決策の実行	
	問題を解決するまでに、即時の解決策が必要な場合は実行する。
	問題記録 検出された問題について、内容、ステータス、終結予定日などを記述する。 分析報告書 問題の分析結果を記述する。 評価報告書 問題の評価結果を記述する。
SUP9.BP6:必要時に警告の通知	
	問題が「高」と分類され、他のシステムまたはユーザにも影響がある場合は、警告を通知する。
	問題記録 検出された問題について、内容、ステータス、終結予定日などを記述する。
SUP9.BP7:変更依頼の開始	
	問題に対する変更依頼を開始する。手順は定義された問題解決管理手順に従う。
	問題記録 検出された問題について、内容、ステータス、終結予定日などを記述する。
SUP9.BP8:終結まで問題のトラッキング	
	問題の解決状況のステータスを確認し、解決活動が適切に行われている事を、問題解決まで定期的に実施
	問題記録 検出された問題について、内容、ステータス、終結予定日などを記述する。 問題ステータス報告書 問題のステータス状況を記述する。
SUP9.BP9:問題の傾向分析	
	類似問題の防止のため、過去に発生した問題に対して、内容の分析を行い、傾向を把握する。この結果から対策を立てる。
	問題ステータス報告書 問題の傾向を記述する。

SUP10. 変更依頼管理

目的	発生した変更依頼を適切に行う。
基本プラクティス	
	概要
	規格で要求されているドキュメントとその内容
SUP10.BP1:変更依頼管理戦略の策定	
	変更依頼を管理するための活動、ライフサイクルモデル、手法を定義し、文書化する。
	変更依頼管理計画書 定義された変更依頼管理戦略を記述する。
SUP10.BP2:一貫性のある変更依頼管理手順の確立	
	変更依頼管理戦略に従って、一貫性があり、トレース可能な方法で変更依頼を取り扱う仕組みを作成する。 変更依頼によって影響を受ける関係部署との連絡方法を定義する。
	変更依頼管理計画書 定義された変更依頼の手順を記述する。
SUP10.BP3:変更依頼の識別および記録	
	発生した変更依頼を識別し、変更依頼内容、依頼者などを記録する。
	変更依頼 変更依頼内容、依頼者を記述する。
SUP10.BP4:変更依頼管理のステータスの記録	
	変更依頼の管理、トラッキングのために変更依頼のステータスを記録する。
	変更制御記録 変更依頼のステータス、実施された変更内容を記録する。
SUP10.BP5:他の変更依頼との依存性および関係性の確立	
	変更依頼の分析、変更実施の可否判断のために他の変更依頼との依存性、関係性を識別する。
	変更依頼 他の変更依頼との依存性、関係性を記述する。
SUP10.BP6:変更の影響の評価	
	変更依頼の技術的な影響と利点を評価する。
	変更依頼 評価結果を記述する。
SUP10.BP7:変更依頼の分析および優先順位付け	
	変更依頼を工数、スケジュール、リスク、利点などの観点から分析する。分析結果から変更依頼の優先度を決定する。
	変更依頼 評価結果の分析結果と優先順位を記述する。
SUP10.BP8:実装前に変更依頼の承認	
	変更依頼の実施を承認してから実装を開始する。顧客要望の変更依頼の場合は顧客にも承認を求める。
	変更依頼 承認基準と承認者を記述する。
SUP10.BP9:実装された変更に対して実施すべき検証および妥当性確認活動の識別および計画	
	変更依頼の実施結果に対する検証基準、妥当性確認の計画を作成する。
	変更依頼 変更依頼の検証基準、妥当性確認の手順を記述する。
SUP10.BP10:変更依頼のスケジュールおよび割り当て	
	承認された変更依頼の実施スケジュールを作成し、実装、検証、妥当性確認の各担当者に作業を割り当て
	変更依頼 変更依頼のステータスを記述する。 作業成果物 変更依頼の結果作成される、作業成果物。
SUP10.BP11:実装された変更のレビュー	
	実装された変更依頼が目的、検証基準を満たしている事を確認する。変更箇所のみでなく、影響を受ける箇所も対象となる。
	変更依頼 変更依頼のステータスを記述する。 変更制御記録 変更依頼のステータス、実施された変更内容を記録する。 作業成果物 変更依頼の結果作成される、作業成果物。
SUP10.BP12:終結まで変更依頼のトラッキング	
	変更依頼の対応状況のステータスを確認し、活動が適切に行われている事を、対応終結まで定期的に実施
	変更依頼 変更依頼のステータスを記述する。 変更制御記録 変更依頼のステータス、実施された変更内容を記録する。 作業成果物 変更依頼の結果作成される、作業成果物。

MAN3. プロジェクト管理

目的	プロジェクトの目標を達成するためにプロジェクト全体の計画を行う。
基本プラクティス	
	概要
	規格で要求されているドキュメントとその内容
MAN3.BP1:作業範囲の定義	
	プロジェクトに含まれる作業を明確にする。プロジェクトの目的が現実的に実現可能であることを確認する。
	プロジェクト計画書 プロジェクトで作成すべき作業成果物、プロジェクトのリソースなどを記述する。
MAN3.BP2:プロジェクトのライフサイクルの定義	
	プロジェクトの範囲や規模などによって、適したライフサイクルを定義する。
	プロジェクト計画書 プロジェクトのライフサイクルモデルなどを記述する。
MAN3.BP3:プロジェクト属性の見積もりの決定および維持	
	プロジェクトの事業目標、品質目標、リソース、スケジュールなどの見積もりを行う。
	プロジェクト計画書 プロジェクトの見積もり結果を記述する。
MAN3.BP4:プロジェクト活動の定義	
	プロジェクトのライフサイクル、見積もり結果から必要なプロジェクト活動を計画し、活動間の依存関係を分
	プロジェクト計画書 プロジェクトのWBSを作成する。 リスク管理計画書 分析されたプロジェクトのリスクについて記述する。 スケジュール プロジェクト間の依存関係から、各活動のスケジュールを立てて記述する。 WBS 必要なプロジェクト活動とそのインプット、アウトプットとなる作業成果物を記述する。
MAN3.BP5:スキルニーズの定義	
	プロジェクトに必要なスキルを把握し、適切にプロジェクトメンバーをアサインする。
	プロジェクト計画書 必要なスキルとアサインされたプロジェクトメンバーを記述する。 WBS タスクの担当者を記述する。
MAN3.BP6:プロジェクトのスケジュールの定義および維持	
	各活動の依存関係を分析し、必要なリソースの割り当てや必要な期間を見積もりを行う。分析結果より、プロジェクトのスケジュールを立てる。スケジュールは各活動の状況により、再計画されることもあり得る。
	プロジェクト計画書 スケジュールを記述する。 リスク管理計画書 分析結果よりプロジェクトのリスクを記述する。 スケジュール 各活動の開始・終了日、担当者を記述する。 WBS 各作業間の依存関係を記述する。
MAN3.BP7:プロジェクトの窓口の識別および監視	
	プロジェクトの関係者間の連絡窓口と伝達方法を決定する。必要な情報が共有されていることを継続的に
	プロジェクト計画書 プロジェクトの情報伝達方法を記述する。 プロジェクト作業ネットワーク プロジェクト活動の関係をネットワーク図で表す。 プロジェクトステータス報告書 プロジェクトのリスクの現状を随時記録する。
MAN3.BP8:プロジェクト計画の確立	
	プロジェクトの各活動の計画書が適切に作成されていることを確認し、レビューすることで、活動間の一貫性を確保する。
	プロジェクト計画書 レビュー結果を反映する。 リスク管理計画書 レビュー結果を反映する。 スケジュール レビュー結果を反映する。

MAN3.BP9:プロジェクト計画の実装

計画した各活動を実行する。

プロジェクト計画書
　実行結果を反映する。
リスク管理計画書
　実行結果を反映する。
スケジュール
　実行結果を反映する。

MAN3.BP10:プロジェクト属性の監視

プロジェクトのリソース、工数、スケジュールなどが計画通りに実行されていること、進捗状況を監視する。

情報伝達記録
　プロジェクトに関して関係者でやり取りした内容を記録する。
進捗ステータス記録
　プロジェクトの各活動の進捗状況を記録する。
プロジェクトステータス記録
　プロジェクトの各活動の現状を記録する。計画から大きな逸脱があれば、それも記録する。

MAN3.BP11:プロジェクトの進捗のレビューおよび報告

プロジェクトの進捗状況を定期的にレビューし、関係者に報告をする。

情報伝達記録
　プロジェクトに関して関係者でやり取りした内容を記録する。
進捗ステータス記録
　プロジェクトの各活動の進捗状況を記録する。
プロジェクトステータス記録
　プロジェクトの各活動の現状を記録する。計画から大きな逸脱があれば、それも記録する。

MAN3.BP12:逸脱の是正

プロジェクトのリソース、工数、スケジュールなどで計画から大きな逸脱があれば、その問題についてレビューを行う。必要があれば、是正処置を検討する。また、類似問題の再発防止のために対策を立てる。

変更依頼
　是正処置のために行った変更依頼について記述する。
レビュー記録
　レビュー結果を記録する。
是正処置登録
　レビューにより指摘された問題点と是正処置を記録する。

別紙

実験結果

作業が行われるプロセス
共通
　全ての認定手法共通で必要となる作業
認定方法：xxx
　xxx という認定方法を選択した際に必要となる作業

機能安全規格等の要求事項

コンパイラ(TCL3)の各ツール認定作業に対してISO 26262 における各ASIL の要求の強さ
〇：必須
++：強く推奨(選択しない場合は合理的な説明が必要)
＋：推奨
－：要求無し(規格上要求されていないが、必要に応じて作業を行う)
／：ISO 26262 の要求事項ではない

共通

要求事項	作業内容	ASIL要求 A	B	C	D	要求規格	備考
安全計画の作成	全体として機能安全規格(ISO 26262)に準拠したシステム開発を行い全体としての安全性を担保する為に必要な活動・リソースの抽出、スケジュールを作成する。	〇	〇	〇	〇	ISO 26262	
…	…					…	
コンパイラを導入するメリットの認識	開発プロセスに対してコンパイラを導入するメリットを認識する。					－	
…	…					…	

要求事項	作業内容	ASIL要求 A	B	C	D	要求規格	備考
障害発生時の対策の策定	障害発生時の対策を定めておき、影響を最小限に抑える。					－	ここでの記録が次の利用実績による認定でのインプットとなります。
実績の分析	前提条件としてこの方法はバージョン検討の場合にのみ有効なので、過去の実績と認定を行うコンパイラがバージョンのみが異なっている事を確認する。	++	++	＋	＋	ISO 26262	互換を謳っているコンパイラ間での実績の比較はできません。
	…					…	

共通

要求事項	作業内容	ASIL 要求				要求規格	備考
		A	B	C	D		
安全計画の作成	全体として機能安全規格(ISO 26262) に準拠したシステム開発を行い全体としての安全性を担保する為に必要な活動・リソースの抽出、スケジュールを作成する。	○	○	○	○	ISO 26262	
ライフサイクル内の適切な要件定義	システム開発全体としての安全性を担保する為に構想から廃棄まで全体のライフサイクルを通じてコンパイラが使用されるフェーズでの必要条件の洗い出しを行う。	○	○	○	○	ISO 26262	
最も高い安全要求レベルの把握	開発対象となっているソフトウェアの誤動作が発生させる事象を抽出する。	○	○	○	○	ISO 26262	一般的には、開発システムの設計フェーズで実施さる作業なので、コンパイラのソフトウェアツール認定時に作業を実施するようなケースは稀です。
	抽出された各事象に対して発生確率・深刻度・回避のしやすさを考慮しリスク分析を行い、安全要求レベル(ASIL) を割り当てる。	○	○	○	○	ISO 26262	
	割り当てられた 各ASIL から最も高い安全基準が求められる ASIL を確定する。	○	○	○	○	ISO 26262	
コンパイラを導入するメリットの認識	開発プロセスに対してコンパイラを導入するメリットを認識する。					－	技術面(新技術対応など)、コスト面(開発効率アップなど)、品質面(バグ現象など)のメリットが考えられます。
コンパイラを導入するデメリットの認識	開発プロセスに対してコンパイラを導入するデメリットを認識する。					－	アセンブラでの開発と比べて処理速度などが劣るケースなど考えられます。 コンパイラ毎の環境依存が発生するリスクや、逆にハードウェアによってコンパイラの選択肢が限られることがあります。
開発システムの要件定義	開発システムに対する認識を共有・統一する為に、開発システムの要件定義を行う。	○	○	○	○	ISO 26262	
コンパイラの影響の認識	コンパイラのシステムまたは開発全体に対する影響を確認する。 コンパイラの性能や対応言語規格により使用できる機能に制限がかかる点を考慮する。 またサードパーティ製ライブラリとの関連も考慮すること。					－	
現状分析	コンパイラを使用するシステム開発についての現状分析を行う。 ・システムの要件定義が基となる ・システムの要件、機能、非機能要件 ・システムの動作環境（チップなどのハードウェア含む） ・開発環境、動作環境 ・開発要員の規模、スキル ・開発手法 ・開発拠点の状況（分散・集中） ・システムまたは開発の問題点、課題など					－	
コンパイラ導入の目的	システムの現状分析からコンパイラに求める要件、機能、非機能要件を明確にする。 その上でコンパイラに求める要件・非機能要件を抽出し、選定条件を決定する。 ハードウェア面からの条件も考慮する。					－	

共通

要求事項	作業内容	ASIL 要求 A	ASIL 要求 B	ASIL 要求 C	ASIL 要求 D	要求規格	備考
コンパイラの評価	コンパイラの評価を行う。 導入候補が複数ある場合は品質、性能の比較検討を行う。 よって、この時点で開発時の動作条件（コンパイルオプションなど）が決まられていることが望ましい。 比較検討はハードウェアとの組み合わせで行うとより正確な比較が可能になる。					–	
	コンパイラの評価を行う為に下記の使用状況を明確にする。 a) 用途 b) 入力と期待する出力 c)（該当する場合は)環境と機能制限	○	○	○	○	ISO 26262	
コンパイラの導入効果の見積もり	コンパイラ導入によるシステムへの効果を見積もる。 コンパイラの性能、機能に大きく依存する。 他には実行スピード、計算精度なども考慮する。 また性能が良くても品質に難があるコンパイラでは、実装、テスト時のコストが大きくなる可能性がある点も考慮する。					–	
コンパイラの品質の評価	コンパイラの品質の評価を行う。					–	ベンダーから品質に関するエビデンスを入手できるのならば、その内容を確認することも重要です。
機能性の評価	コンパイラの機能、動作条件がコンパイラに求められている要件、機能、非機能要件を満たすかを確認する。 コンパイラの評価版があれば、購入前に試用を行い、実装担当者の意見も収集する。 過去のプロジェクトコードを流用したり、ベンチマークのためのテストケースを作成して、動作確認を行う。 また、コンパイラの独自仕様、制限についてもあらかじめ確認し、システムに対するデメリットを確認する。					–	デバッグ機能や静的・動的解析ツール、リビジョン管理ツールとの連携などの機能も重要です。
移行性の評価	コンパイラのセットアップ手順をハードウェア面も含めて確認する。 既存環境、既存ソースコードとの移行性を確認する。 これらを総合的にコンパイラの動作環境の構築または利用者の移行の難易度を評価する。					–	同系統のコンパイラでも言語規格の対応バージョンなどによって結果は変化します。
操作性の評価	コンパイラの操作性について確認する。 ・コンパイルの動作方法が一般的な方法であること 例えばコマンドラインからのビルド方法がなければ、ビルド管理システムとの連携は難しくなる。 ・コンパイルオプションの指定方法、オプションの内容が一般的な内容であること オプションの機能が資料として提供されていること。 通常、デバッグとビルドではオプションが異なるので、それぞれの動作の差異を把握しておくことが大事である。 ・コンパイルエラーが発生した場合に適切なメッセージが表示されること、発生エラーが検出できること コーディング指針などで禁止されているコードは、可能ならばコンパイルエラーになるように設定しておくことが望ましい。					–	一般的なコンパイラと同様に操作できることや直観的に操作できることを確認します。

共通

要求事項	作業内容	ASIL 要求 A	ASIL 要求 B	ASIL 要求 C	ASIL 要求 D	要求規格	備考
互換性の評価	既存ソースコードが使用可能なことを確認する。 また設計ツールなどから自動生成されたソースコードが使用可能なことを確認する。 コンパイラにより作成されたオブジェクトが次工程で問題なく動作することを確認する。 新コンパイラを導入したことで前工程、次工程に問題が起きないことを確認する。					－	動作仕様の大きな変更などがあった場合は、過去のプロジェクトコードの再利用が困難になります。 利用者が混乱しないためのマニュアルの整備が必要です。 またコンパイラを他のベンダーのものに変更する場合には、特に注意が必要です。 例えば、Linuxのカーネルコードは GCCコンパイラを前提とした実装がされている箇所があり、他のコンパイラに移行した場合、コードのビルドすらできない場合があり、その修正にかなりのコストがかかります。
第三者評価・導入事例の評価	第三者によるコンパイラの評価結果、導入事例があれば取得し、その内容を評価する。					－	第三者評価は客観的な品質評価として説明責任を果たす上では大きな意味を持つため、機能安全規格に準拠したシステム開発では、できる限り実施するのが望ましいです。
規定されていない環境下での挙動の評価	規定されていない環境で動作した際の挙動を把握し、その情報を利用可能な形へと整備する。	○	○	○	○	ISO 26262	
適用評価	新コンパイラを導入することでシステムにどのような効果があるかを評価する。					－	
過去の開発プロジェクトに適用	過去開発のシステムに導入予定コンパイラを使用し、その結果を確認する。 ・コンパイル結果、コンパイル速度 ・実行結果、実行時間、計算精度・・・など 問題が発生した場合は原因の分析を行い、コーディング指針などを設定して問題を回避する。					－	
現在進行中のプロジェクトに適用	現在開発中のシステムにて試用する。 試用した結果を現行システムと比較する。 試用手順を実際の導入手順を策定する参考にする。					－	
導入コストの評価	コンパイラ導入に関わるコストを評価する。					－	
購入コストの評価	購入費やライセンス費について評価する。 コンパイラの購入費、ライセンス費を確認する。 開発拠点が分散する場合は、ライセンスの扱いや範囲に特に注意する。 特定チップで強みを発揮する場合、そのチップのコストも考慮する必要がある。					－	ソース開示の義務の有無や、製品配布などのライセンス条項の確認も忘れてはなりません。
移行コストの評価	新コンパイラに移行する場合の作業コストを評価する。 開発環境の移行コストと、必要であれば既存ソースコードの修正対応に要するコストとなる。					－	既存のソースコードとの互換性により大きく異なり、初めからコードを書き直したほうが早い場合もあり得ます。
教育コストの評価	コンパイラを円滑に使用するためのコストを評価する。 これはコンパイラの使用方法、使用者のスキルなどに依存する。 コーディング指針作成に要するコストも必要となる（その指針やコンパイラの制限の周知を含む）。					－	開発に採用する言語規格／バージョンを変える場合にはさらに教育コストが膨らむため、従来の規格／バージョンのまま開発することも検討すべきでしょう。

共通

要求事項	作業内容	ASIL 要求 A	ASIL 要求 B	ASIL 要求 C	ASIL 要求 D	要求規格	備考
保守コストの評価	コンパイラのバージョンアップ情報、不具合情報を定期的に取得するしくみが必要となる。 そのようなメンテナンス方法を策定し、必要なコストを評価する。 開発期間が長い場合には、保守期間分のコストを計算に入れる。 バージョンアップに別料金がかかる場合などは保守契約も確認する。					–	ベンダーの公開サイトなど、最新のツール情報にアクセスできる体制が望ましいです。
制限事項の確認	コンパイラの制限事項、不具合情報をマニュアルや公開サイトから確認する。 これまでの評価結果からも制限事項の有無を確認する。 そして、これらの制限事項、不具合情報が開発に及ぼす影響（その内容が既存開発環境、既存成果物に影響がないか、影響があっても許容範囲内であるか）を評価する。					–	
コンパイラの仕様の取りまとめ	ベンダーからユーザーマニュアルなどの資料を入手する。	○	○	○	○	ISO 26262	パッケージ付属のドキュメントでは不足するかもしれないため、ベンダーの協力を仰ぐことが望ましいです。
	コンパイラの仕様・特性を把握して誤った環境下で使用をしない為に、ドキュメントやベンダーからの制限事項の入手・取りまとめをする。	○	○	○	○	ISO 26262	
	コンパイラの特性を把握し、ベンダー側で想定されている使用を行う為に、下記の情報を整備する。 a)コンパイラの機構、機能、及び技術的な性質の記述 b) 該当する場合は、ユーザーマニュアルやユーザーガイド c) 運用環境の記述	○	○	○	○	ISO 26262	
コンパイラの信頼性レベル(TCL)の評価	コンパイラの不具合が開発システムの不具合を引き起こす可能性の有無(TI)を検討する。	○	○	○	○	ISO 26262	一般的な利用方法では、コンパイラの不具合は開発システムの不具合を引き起こす可能性があるので、TI 2 に分析されます。
	コンパイラの不具合の検出・回避策の精度(TD)を分析する。	○	○	○	○	ISO 26262	コンパイラの入力パターンは無限で、システマティックに不具合を検出する方法が存在しないので、 一般的には TD3 となります。
	上記の検討・分析から 信頼性レベル(TCL)を確定する。	○	○	○	○	ISO 26262	
コンパイラの影響・要求安全レベルの評価	求められる安全要求の高さが変わってくるので、開発の中でコンパイラがどのような形で利用されるかを明確に示す為にコンパイラのユースケースを作成する。	○	○	○	○	ISO 26262	
	コンパイラの不具合が製品に影響する範囲を抽出し、その中で最も高い ASIL となるケースを洗い出す。	○	○	○	○	ISO 26262	
	上述のコンパイラの信頼レベルに基づき、適切な認定方法を選択する。	○	○	○	○	ISO 26262	以下の4つの認定方法から選択します。 1a.利用実績 1b.ツール開発プロセスの評価 1c.ソフトウェアツールの検証 1d.安全規格に準拠した開発

共通

要求事項	作業内容	ASIL 要求				要求規格	備考
		A	B	C	D		
評価環境の妥当性確認	実際の開発環境を抽出する。	○	○	○	○	ISO 26262	評価の段階で開発環境・条件が定まっていないのは望ましくないため、プロジェクトの早い段階で確定しておきます。
	評価時の動作環境を抽出する。	○	○	○	○	ISO 26262	
	開発・評価環境が下記のコンパイラの仕様制限内である事を確認する。 ・使用方法 ・環境と機能制限 ・動作条件	○	○	○	○	ISO 26262	
	開発環境と評価環境を比較し、評価環境の妥当性を確認する。	○	○	○	○	ISO 26262	開発環境と著しく異なる環境でしか評価を行えないケースもあります。 その場合には、それでもその評価が妥当である、合理的な説明を行う必要があります。
評価環境の確定	評価環境を一意にする為にコンパイラのバージョン番号を確定させ、その識別を実施する。	○	○	○	○	ISO 26262	
	コンパイラの動作設定・実行環境を確定する。	○	○	○	○	ISO 26262	
	評価環境構築手順書の作成					－	
コンパイラの不具合がシステムの不具合を引き起こさない仕組みの作成	認識済みの不具合が製品へ混入されるのを回避する為に、把握している不具合情報とその回避・検出策を整備する。	○	○	○	○	ISO 26262	回避、検出は自動で行われるのが望ましいです。 例えば、静的開発ツールの検索ルールに追加し、ソースコード更新時に自動でチェックするといった方法があります。
導入者と使用者の同意	コンパイラが顧客指定の場合など、使用者選定でない場合はそのコンパイラを使用する妥当な理由などについて、合意を形成する。					－	顧客側でツール認定を行っているならば、その資料を入手するのが望ましいです。
障害発生時の対策の策定	障害発生時の対策を定めておき、影響を最小限に抑える。 不具合管理の体制を定めておき、次の開発のインプットとする。					－	
未然防止策の策定	コーディング指針を作成し、コーディングミスによる障害を回避する。 コンパイラのバージョンアップ情報、不具合情報を定期的に取得して、防止策を更新する。					－	静的解析ツールへのルール追加などにより、自動で障害を回避できる仕組みを構築することが望ましいです。
事後対策の策定	未知の不具合によると見られる障害が発生した場合の対応策を定める。 不具合の原因・発生条件・回避策を調査検討し、記録する。 システムへの影響度の測定と既存ソースコードへのチェックと修正を行い、特定コーディングで回避できる場合には未然防止策として採用する。 また可能な場合はベンダーに修正を依頼する。					－	また、速やかに開発関係者への状況通知も行い、障害による影響の拡大を阻止し、再発・類似障害発生を防止してください。
コンパイラの導入支援	コンパイラが特殊な環境で動作する場合、動作方法が特殊な場合などは、特別な導入支援を整備する。					－	開発中に適切な導入支援を行うことで、コンパイラによる作業効率のアップを図ります。

共通

要求事項	作業内容	ASIL 要求 A	ASIL 要求 B	ASIL 要求 C	ASIL 要求 D	要求規格	備考
保守	コンパイラのバージョンアップ情報、不具合情報を定期的に取得する。 必要な場合はバージョンアップ版に更新する。 その際は既存ソースコード、既存開発環境との互換性を確認する。 変更履歴などを確認し、必要であればコーディング規約などで周知する。					-	
コンパイラの再評価	新コンパイラによる効果を再評価する。 再評価結果を導入前の効果の見積もりと比較し、今後の効果の見積もりに役立てる。 使用中に発見された不具合、要望などは記録する。					-	実際の導入結果からコンパイラの効果を測定し、見積もり精度の向上に努めてください。 不具合、要望はベンダーにも報告し、修正の予定などを確認して、開発計画を調整するべきです。
コンパイラ導入の評価	新コンパイラが開発システム全体に及ぼした影響を評価する。 これらの結果をもとに、その後の開発プロジェクトでのコンパイラ導入について検討する。					-	
評価基準の妥当性確認	TCL　確定プロセスを記述する。	○	○	○	○	ISO 26262	ASIL-A 要求無し ASIL-B 推奨 ASIL-C, D 必須 ※ASILの確定プロセスの記述は規格上では求められてないですが、説明責任を果たす上では、ASILの記述も行うのが望ましいです。
	認定方法の選定・認定内容を記述する。	○	○	○	○	ISO 26262	
エビデンスの用意	適切なツール認定方法を選択した根拠と、ツールの検証結果を、各レポートにまとめる。	○	○	○	○	ISO 26262	
ソフトウェアツール基準評価レポートの作成	検査環境を一意にする為に、固有の識別とコンパイラのバージョン番号及び、認定を行う動作設定と環境を明示する。	○	○	○	○	ISO 26262	途中でコンパイラのバージョンを更新した場合などは、ツール認定を再実施する必要があるかもしれません。
	コンパイラの不具合に対する分析を行ったエビデンスとして、コンパイラの信頼性レベルとしてTCLを明示する。	○	○	○	○	ISO 26262	
	コンパイラの使用に関して分析を行ったエビデンスとして、最大(または特定) ASIL を明示する。	○	○	○	○	ISO 26262	
	安全規格の要求事項とレポートの記載事項の対応一覧を作成する。					-	この一覧は、要求事項に漏れがないか確認する為のチェックリストにも使え、第三者機関で認証を行う場合には認証期間の短縮が期待できます。
ソフトウェアツール認定レポートの作成	検査環境を一意にする為に、固有の識別とコンパイラのバージョン番号及び、認定を行う動作設定と環境を明示する。	○	○	○	○	ISO 26262	途中でコンパイラのバージョンを更新した場合などは、ツール認定を再実施する必要があるかもしれません。
	認定者を曖昧にしない為に、認定を行った個人または組織を明示する。	○	○	○	○	ISO 26262	
	認定プロセス及びその結果をエビデンスとして残す為に、適用した認定方法及びその結果を記録する。	○	○	○	○	ISO 26262	各認定方法毎の作業は、各シートを参照してください。
	コンパイラの不具合が開発システムの不具合を引き起こす事を防ぐ為のコンパイラの不具合の検出・回避策を作成する為に、コンパイラの不具合を明示する。	○	○	○	○	ISO 26262	発生条件（サンプルコード）、現象、回避策（同じ処理を行うが、正常な出力が得られるサンプルコード）など。
	安全規格の要求事項とレポートの記載事項の対応一覧を作成する。					-	この一覧は、要求事項に漏れがないか確認する為のチェックリストにも使え、第三者機関で認証を行う場合には認証期間の短縮が期待できます。

認定方法：利用実績

要求事項	作業内容	ASIL 要求 A	ASIL 要求 B	ASIL 要求 C	ASIL 要求 D	要求規格	備考
障害発生時の対策の策定	障害発生時の対策を定めておき、影響を最小限に抑える。					–	ここでの記録が次の利用実績による認定でのインプットとなります。
実績の分析	前提条件としてこの方法はバージョン検討の場合にのみ有効なので、過去の実績と認定を行うコンパイラがバージョンのみが異なっている事を確認する。	++	++	+	+	ISO 26262	互換を謳っているコンパイラ間での実績の比較はできません。
	動作条件を比較する必要があるので、以前の固有の識別とコンパイラのバージョン番号及び、認定を行う動作設定と環境の抽出・分析を行う。	++	++	+	+	ISO 26262	
	利用実績データ分析の為に、利用期間とその詳細の関連データ及び発生条件も含めた不具合情報の記録を見直す。	++	++	+	+	ISO 26262	
	修正情報も含めた不具合情報の把握の為、以前に利用したバージョンとそれぞれのバージョンで修正された不具合のリスト及びその不具合の検出・防止策を見直す。	++	++	+	+	ISO 26262	
エビデンスの作成	過去の利用実績と大きく異なる場合にはこの認定方法の信頼性に乏しい認定となるので、認定を行うケースが下記の比較対象の利用実績と類似であるこを確認する。 ・ユースケース ・運用環境 ・機能制限 ・利用目的	++	++	+	+	ISO 26262	
	少ない実績データでは利用実績の根拠として不十分となるので、十分な量のデータに基づいている事を確認する。	++	++	+	+	ISO 26262	
	コンパイラの仕様が変更されている場合は過去のデータは利用実績としての価値は無いので、比較対象時の仕様から変更されていない事を確認する。	++	++	+	+	ISO 26262	一般的に1.xx から 2.xx のようなメジャーバージョンの変更は大きな仕様変更が行われる事を示すので、 メジャーバージョンの変更があった際には利用実績による認定は行えないと考えるのが妥当です。
	コンパイラの不具合情報が利用しやすい形式で蓄積されていない場合やそれらの情報に漏れがある場合は、過去の実績情報として致命的なので、不具合情報の適切な管理と蓄積を行う仕組みを構築する。	++	++	+	+	ISO 26262	バグ管理システム(BTS) を使用して不具合情報の管理・蓄積が現実的な対応方法となります。
	作業を行ったエビデンスを残す為に、上記の作業の記録を残します。	++	++	+	+	ISO 26262	
レポート作成	以下の内容をツール認定レポートに記載する。 ・レポート自身の改版履歴 ・検討材料となる実績データとその妥当性 ・認定対象コンパイラと実績対象コンパイラの下記の情報及びその比較結果 – 仕様 – ユースケース – 利用目的 – 動作環境、設定 – バージョン ・認識をしている不具合の回避・検出策 ・不具合情報の管理・蓄積方法	++	++	+	+	ISO 26262	

認定方法：ツール開発プロセスの評価

要求事項	作業内容	ASIL 要求				要求規格	備考
		A	B	C	D		
エビデンスの確認	下記の要求事項が満たされ適切なエビデンスが提供されている事を確認する。	++	++	+	+	ISO 26262	
要件抽出	コンパイラに対する要件およびニーズが収集されていることを確認する。					AutomotiveSPICE（ENG1.BP1～ENG1.BP6）	
システム要件分析	コンパイラの要件に基づいてシステム要件が定義されていることを確認する。 システム要件はコンパイラに対する要件を網羅していること確認する。					AutomotiveSPICE（ENG2.BP1～ENG2.BP7）	
システムアーキテクチャ設計	システム要件に基づいてシステムアーキテクチャが設定されていることを確認する。 システムアーキテクチャはシステム要件を網羅していることを確認する。					AutomotiveSPICE（ENG3.BP1～ENG3.BP7）	
ソフトウェア要件分析	システム要件に基づいてソフトウェア要件が定義されていることを確認する。 ソフトウェア要件はシステム要件、システムアーキテクチャ設計を網羅していることを確認する。					AutomotiveSPICE（ENG4.BP1～ENG4.BP8）	
ソフトウェア設計	ソフトウェア要件に基づいてソフトウェア設計がされていることを確認する。 ソフトウェア設計はソフトウェア要件を網羅していることを確認する。					AutomotiveSPICE（ENG5.BP1～ENG5.BP10）	
ソフトウェア構築	ソフトウェア設計と整合性があり、検証されたソフトウェアユニットが生成されていることを確認する。					AutomotiveSPICE（ENG6.BP1～ENG6.BP10）	
ソフトウェア統合テスト	ソフトウェアユニットを統合し、ソフトウェアユニット間の検証が行われていることを確認する。					AutomotiveSPICE（ENG7.BP1～ENG7.BP8）	
ソフトウェアテスト	統合ソフトウェアがソフトウェア要件を満たしている事を確認する。					AutomotiveSPICE（ENG8.BP1～ENG8.BP6）	
システム統合テスト	ソフトウェアを統合し、ソフトウェア間の検証が行われていることを確認する。					AutomotiveSPICE（ENG9.BP1～ENG9.BP8）	
システムテスト	システムがシステム要件を満たし、納入準備がなされていることを確認する。					AutomotiveSPICE（ENG10.BP1～ENG10.BP6）	
品質保証	プロジェクトの各プロセスがあらかじめ計画されているを確認する。					AutomotiveSPICE（SUP1.BP1～SUP1.BP10）	
構成管理	プロジェクトの各作業成果物がバージョン管理され、トレーサビリティ維持されていることを確認する。					AutomotiveSPICE（SUP8.BP1～SUP8.BP11）	
問題解決管理	プロジェクト中で発見された問題の解決が確実に行われていることを確認する。					AutomotiveSPICE（SUP9.BP1～SUP9.BP9）	
変更依頼管理	プロジェクト中で発生した変更依頼が適切に行われていることを確認する。					AutomotiveSPICE（SUP10.BP1～SUP10.BP12）	
プロジェクト管理	プロジェクトの目標を達成するためのプロジェクト全体の計画が行われていることを確認する。					AutomotiveSPICE（MAN3.BP1～MAN3.BP12）	

認定方法:ツール開発プロセスの評価

要求事項	作業内容	ASIL 要求				要求規格	備考
		A	B	C	D		
レポート作成	以下の内容をツール認定レポートに記載する。 ・レポート自身の改版履歴 ・安全規格の内容 ・安全規格が国際規格に基づいていることの証明 ・開発プロセスが安全規格に準拠していることの証明	++	++	+	+	ISO 26262	使用する安全規格はいくつかの規格を組み合わせることも可能です。 安全規格の証明には認証機関の結果を用いるなど、第三者の証明があることが望まれます。

認定方法：ソフトウェアツールの検証

要求事項	作業内容	ASIL 要求 A	ASIL 要求 B	ASIL 要求 C	ASIL 要求 D	要求規格	備考
ツール検証の計画	コンパイラの検証のために、リソース(要員、機材)、期間、テスト範囲、テスト方針といったことを策定する。					－	機材については、開発環境と同じものが必要となります。
品質基準	既定の要求に準拠していることを立証する基準を決定する。 C言語コンパイラの場合は、以下の言語規格のいずれかが基準となる。 ・ISO/IEC 9899：1990 ・ISO/IEC 9899：1999	+	+	++	++	ISO 26262	コンパイルオプションによっても対応する言語規格は切り替わるケースもあります。
網羅度	C言語規格に記載されている機能を一通り検証することで、検証の網羅度を担保する。 そのため、言語規格と作成したテストケースの対応表も作成しておく。これは検証の妥当性の証明のために必要となる。 ただし組込み向けコンパイラの場合は、言語規格にある一部の機能が未実装であることも多く、また、コーディングガイドなどで開発での使用を禁じられているケースも考えられるため、これらの機能については検証対象から除外することを明示して、検証に要するコストと時間を節約する。	+	+	++	++	ISO 26262	市販のコンパイラ検証ツールや第三者検証機関のサービスを利用することで、ツール検証に関わる作業コストを大幅に短縮することが可能となるため、これらの利用も検討するべきです。 最適化に起因する不具合等のコンフォーマンステストでは検出しにくい不具合もあるので、 コンパイラの場合は、コンフォーマンスとは異なった視点からの検査を強く推奨します。
ツールの仕様確認	環境構築とテストケース作成のため、コンパイラ仕様の情報を収集する。					－	
マニュアルの確認	コンパイラのマニュアルの記載内容を確認する。 特に以下の項目は要確認 　動作環境 　インストール方法 　動作方法 　コンパイルオプション 　エラーメッセージ 　リリースノート 　機能制限 検証環境が動作環境として問題ないことを確認する。	+	+	++	++	ISO 26262	
改版履歴の確認	過去の改版履歴を確認する。 必要があれば、その内容がコンパイラに反映されていることを確認するためのテスト項目をテストスイートに追加する。(後述)	+	+	++	++	ISO 26262	修正履歴はマニュアルに記載されていることもあれば、ベンダーサイトで周知されている場合もあります。
既知不具合の確認	過去に検出され判明している、コンパイラの不具合内容を確認する。	+	+	++	++	ISO 26262	既知不具合はマニュアルに記載されていることもあれば、ベンダーサイトで周知されている場合もあります。
独自仕様・機能制限の確認	検査対象コンパイラの独自仕様、制限事項を確認する。 必要であれば境界値テストをテストスイートに追加する。(後述)	+	+	++	++	ISO 26262	既知不具合が機能制限とされている場合もあります。 独自仕様が言語仕様と異なる場合は特に注意が必要です。
テストケースの作成	C言語規格にある基本的な機能を確認するテストケースを作成する。 その後、以下に挙げるパターンのテストケースを追加していくことで、検証の精度と網羅度を向上させる。 また、将来の開発プロジェクトに活用するためにも、作成したテストコードはリビジョン管理下に置く。	+	+	++	++	ISO 26262	個々のテストケースの規模(ライン数)は小さくしておくほうが、後のパターン化と不具合箇所の特定が容易になります。 また、公開されたC言語用テストスイートといったものも存在しますが、検査の範囲や量が不足がちなため、それだけで十分な検証とするのは難しいです。 C言語用テスト項目の詳細は[C言語コンパイラ用テスト項目の作成指針]を参照してください。

認定方法：ソフトウェアツールの検証

要求事項	作業内容	ASIL 要求				要求規格	備考
		A	B	C	D		
パターン化	基本的な演算を確認するテストケースを作成した後に、それを基に派生させたテストケースを作成する。	+	+	++	++	ISO 26262	以下の観点のパターンで派生させいくことで、効果的なテストケースとなります。 ・型パターン ・符号パターン ・記憶域パターン ・修飾子パターン ・ポインタパターン ・繰り返し文パターン ・最大値、最小値パターン ・ビットフィールドパターン
	・データ構造 ・アルゴリズム(ソート,探索,ハッシュなど) ・過去のプロジェクトコードからの引用	+	+	++	++	ISO 26262	言語規格書を基にしたテストケースだけでは不具合の検出には不十分のため、実際の開発時に記述されるのに近いコードでのテストが必要です。
	過去の不具合記録が存在する場合は、その再現コードをテストケースに加える。 これによりデグレードの有無とリリースノートの記述内容の確認となる。	+	+	++	++	ISO 26262	「過去の不具合記録」とは、自社で管理している過去プロジェクトで検出された不具合記録などの他に、コンパイラベンダーが公開している不具合情報とその改版履歴などを指します。
独自仕様・機能制限のテストケース作成	開発において、コンパイラの言語規格外の機能を使う必要がある場合は、それらの機能についてもテストケースを設計する。	+	+	++	++	ISO 26262	ツールの独自機能については、検証ツールや第三者検証機関での検査の範囲外となることが多いため、利用者自身で検証をおこなわなければならないケースが多くなります。
検証条件の取得	コンパイラの実際の使用状況（開発環境）を把握し、同等の条件で検証を行う。 特に以下の項目は実際の環境と合わせる必要がある。 ・機種 ・OS（SPなども含む） ・コンパイラバージョン ・パッチ ・コンパイルオプション	+	+	++	++	ISO 26262	特殊な実行環境ではメモリなども合わせる場合もあります。
検証環境の設定	検証条件より、検証環境を設定する。 実機実行の場合は検証でも実機実行か、またはシミュレータ、デバッガ使用かを必要に応じて検討する。 結果出力が保存可能な形式で得られることを確認する。 コンパイラが異常終了した場合の安全な終了方法を検討する。 実行ライセンスが必要な場合は事前に取得しておく。	+	+	++	++	ISO 26262	実機環境で実行する場合は結果出力を得られる方法を事前に確認する必要があります。
検証環境の構築	コンパイラをインストールする。 必要があればパッチをあてる。 設定された検証環境を検証マシンに作成する。	+	+	++	++	ISO 26262	修正パッチの適用が必要なケース等、インストーラの起動のみで環境構築が完結しない場合は 検査環境を明確にしてマシンによって検査環境が異なるケースが発生しないように、環境構築手順書が存在しない場合は作成する事を推奨します。
ツールバージョンの確認	検証に使用するコンパイラのバージョンが実際に使用されるバージョンと同じであることを確認する。 アセンブラ、コンパイラ、リンカについてそれぞれ確認する。	+	+	++	++	ISO 26262	ライブラリのバージョン確認が必要な場合もあります。

認定方法：ソフトウェアツールの検証

要求事項	作業内容	ASIL 要求				要求規格	備考
		A	B	C	D		
検証条件の確定	作成した検証環境にて動作確認を行う。 コンパイル時間、実行時間が想定内であることを確認する。 検証実行のための連続実行が問題なく行えることを確認する。 検証条件、検証環境、ツールバージョンに問題がないことを確認し、検証条件を確定する。	+	+	++	++	ISO 26262	
検査実施	対象のコンパイラで作成したテストケースをコンパイルし、実機またはシミュレータ上で動作させ、実行ログを記録する。	+	+	++	++	ISO 26262	コンパイラのバージョンアップなどで再検証の機会は多いので、検査実施から結果一覧の作成までを自動化しておくことが望ましいです。
不具合の検出	実行ログからコンパイラの不具合部分を検出し、不具合を再現するサンプルコードを作成する。 ※不具合かどうかの判断は、コンパイラの制限事項や実装依存について熟知した要員が行う必要がある。	+	+	++	++	ISO 26262	不具合を起こす条件（コード、メモリ配置、特定のコンパイルオプションの組み合わせなど）を明確にします。 また、正常動作したとしても実行やコンパイルに要する時間が異常な場合は開発に支障をきたすため、不具合と考えるべきです。
回避策の検討と実施	検出した不具合を基に、回避策を検討する。 ・同等の演算を行、正常動作するコード例 ・静的開発ツールなどによる異常動作するコード例の検知 ・コーディング規約などに記載し、異常動作するコード記述の周知と禁止 ・コンパイラオプションの変更	+	+	++	++	ISO 26262	
修正依頼	開発に支障をきたすレベルの不具合が検出された場合は、コンパイラベンダーに連絡し、コンパイラの修正を依頼する。					－	
レポート作成	以下の内容をツール認定レポートに記載する。 ・レポート自身の改版履歴 ・検査の目的 ・検査方法の妥当性 ・検査条件の記載 ・検査結果の一覧 ・検出した不具合の情報 ・不具合の回避策	+	+	++	++	ISO 26262	検査方法の妥当性のエビデンスとして、「ツール検証の計画」で作成した品質基準や網羅度といった指針や、テストケースの作成手法やパターン化の方針をレポートに記載します。 レポートのサンプルとして[コンパイラツール認定レポート(サンプル)]を参照してください。
再検証	ツール認定後に、コンパイラの修正や変更、コンパイルオプションの変更などを行った場合は、改めてその条件でツールの検証を実施し、ツール認定レポートを更新する。	+	+	++	++	ISO 26262	デグレードや新たな不具合の検出の可能性があります。

認定方法：安全規格に準拠した開発

要求事項	作業内容	ASIL 要求				要求規格	備考
		A	B	C	D		
安全規格の選択と 要求事項の抽出	適切な安全規格を選択し、品質確認を行うのに十分だと思われる要求事項を抽出する。	+	+	++	++	ISO 26262	ISO 26262-8:2011に記載されているように適用可能な安全規格が存在しないので既存の規格をそのまま適用するのではなく 左記のような作業が必要となります。
エビデンスの確認	上記で抽出した要求事項が満たされ適切なエビデンスが提供されている事を確認しその結果をレポートとして作成する。	+	+	++	++	ISO 26262	
設計の基準の作成・取りまとめ	不具合が入るリスクを減らす為に設計の基準の作成し、設計ガイドラインとして取りまとめる。	○	○	○	○	ISO 26262	
ハードウェア・ソフトウェアソフトウェアインターフェースの仕様の確認・取りまとめ	ソフトウェア技術者の視点からでは見落としがある可能性が高くなるので、システム及びハードウェアとソフトウェア開発の責任者といった様々な役割の人間でハードウェア・ソフトウェア間のインターフェースの確認を行う。	○	○	○	○	ISO 26262	
	上記で確認された仕様をハードウェア・ソフトウェアインターフェース仕様書としてまとめる。	○	○	○	○	ISO 26262	
設計時の検討	不具合を入るリスクを下げ検出しやすくする為に、下記の点を考慮する。 ・検証可能性 ・設計と実装可能性 ・保守性 ・ 単純さ ・ 読みやすさと理解しやすさ ・ ロバスト性 ・ 適切なソフトウェアの変更 ・ テストの容易性	○	○	○	○	ISO 26262	
	複雑になればなるほどと不具合が発生するリスクが高くなるので、高い複雑性による不具合の発生を避ける為に下記の検討する。 ・モジュール化 ・カプセル化 ・単純化	○	○	○	○	ISO 26262	
設計の検証	ISO 26262-8:2011 9節準拠の検証を行う。	○	○	○	○	ISO 26262	ISO 26262-8:2011 9節に対応した作業内容については [ISO 26262-8:2011 の検証(9 節)で要求されている作業一覧]を参照してください。
	ハードウェアインターフェース仕様及び設計仕様に対する準拠性の検証する。	○	○	○	○	ISO 26262	
設計検証のエビデンスの作成	設計仕様書の作成する。	○	○	○	○	ISO 26262	
実装の基準の作成・取りまとめ	不具合が入るリスクを減らす為に実装の基準の作成し、コーディングガイドラインとして取りまとめる。	○	○	○	○	ISO 26262	

認定方法：安全規格に準拠した開発

要求事項	作業内容	ASIL 要求				要求規格	備考
		A	B	C	D		
実装時の検討	不具合を入るリスクを下げ検出しやすくする為に、下記の点を考慮する。 ・ 単純さ ・ 読みやすさと理解しやすさ ・ ロバスト性 ・ 適切なソフトウェアの変更 ・ テストの容易性	○	○	○	○	ISO 26262	
	ハードウェアインターフェース仕様及びコーディングガイドラインに対する準拠性の検証する。	○	○	○	○	ISO 26262	
検証計画	検証の実施を余裕を持って確実に行う為に、リソース・スケジュールを明記した検証計画の作成する。	○	○	○	○	ISO 26262	
検証仕様の作成	検証環境を確定する。	○	○	○	○	ISO 26262	検証環境が実行環境に可能な限り近いことが重要です。 (異なる場合はソースとオブジェクトコード、検証環境と実行環境の相違点の分析)
	検証方法を確定する。	○	○	○	○	ISO 26262	
	妥当性確認の為に下記を検証している事を確認する。 ソフトウェアユニットが下記を達成している事の証明 ・ 設計仕様への適合 ・ ハードウェアインターフェースの仕様への適合 ・ 規定された機能 ・ 意図していない機能が存在していなこと ・ ロバスト性 ・ 機能をサポートするのに十分なリソース	○	○	○	○	ISO 26262	
	漏れが発生していない事を確認する為に検証の網羅性を評価しカバレッジの測定を行う。	○	○	○	○	ISO 26262	コンパイラの場合は言語規格の要求事項が網羅性の一つの目安となります。
	ISO 26262-8:2011 9節準拠の検証の計画・明示を行う。	○	○	○	○	ISO 26262	ISO 26262-8:2011 9節に対応した作業内容については [ISO 26262-8:2011 の検証(9 節)で要求されている作業一覧]を参照してください。
	上記で確定した情報をまとめて検証仕様書として作成する。	○	○	○	○	ISO 26262	
検証結果の作成	ISO 26262-8:2011 9節準拠の検証を実施する。	○	○	○	○	ISO 26262	ISO 26262-8:2011 9節に対応した作業内容については [ISO 26262-8:2011 の検証(9 節)で要求されている作業一覧]を参照してください。
	実施した検証結果をまとめて検証報告書として作成する。	○	○	○	○	ISO 26262	
レポート作成	以下の内容をツール認定レポートに記載する。 ・レポート自身の改版履歴 ・作成した(独自)安全規格の作成プロセス及びその妥当性 ・(独自)安全規格の内容 ・(独自)安全規格に準拠した開発を行なっている事の証明	+	+	++	++	ISO 26262	作成する安全規格はいくつかの規格を組み合わせることも可能です。 安全規格の証明には認証機関の結果を用いるなど、第三者の証明があることがの望ましいです。

ISO 26262-8:2011 の検証(9 節)で要求されている作業一覧

要求事項	作業内容	章番号	備考
予備調査	前提条件の洗い出しを行う	9.3.1	
	必要となる支援情報の調査を行う	9.3.2	
検証企画書の作成	検証実施前に問題が発生するリスクを下げる為に、計画時に下記の分析を行う ・ 適用する検証方法の妥当性 ・ 確認する作業成果物の複雑度 ・ 検証者の対象の検証に関連する以前の経験 ・ 技術使用の成熟度やその技術使用に関連するリスク	9.4.1.1	
	検証として妥当である事を担保する為に下記の分析を行う ・検証されるべき作業成果物の内容 ・検証方法 ・検証の合否基準 ・検証環境 ・検証の際に使用するツール(使用する場合) ・ 異常を検知した際に、行うべき対応 ・ コンパイラに対して変更があった際に、どのようにテストを行うのかを示した基本方針	9.4.1.2	
	上記の結果を検証企画書として作成する	9.5.1	

ISO 26262-8:2011 の検証(9 節)で要求されている作業一覧

要求事項	作業内容	章番号	備考
検証仕様書の作成	検証状況を一意にする為に各検証方法に対して下記を明確にする。 ・ チェックリストのレビューや分析方法 ・ シミュレーションシナリオ ・ テストケース、テストデータ及び検査対象	9.4.2.1	
	各テストケースで下記が明確になっている事を確認する。 ・ 固有の識別 ・ 検証すべき関連作業成果物のバージョンへの言及 ・ 前提条件と設定 ・ 入力データ(入力の順序、実際の値等) ・ 期待される挙動(出力値の許容範囲、時間挙動及び許容される挙動等)	9.4.2.2	仕様書上は必要な場合は温度等の環境条件が要求されているが、コンパイラに関しては必要ないので省略
	テストグループ(複数のテストケースで構成される単位)毎に下記の情報が明確になっている事を確認する。 ・ 検証環境 ・ 依存関係 ・リソース	9.4.2.3	
	上記の内容をまとめた検証仕様書を作成する	9.5.2	
検証報告書の作成	上記に従った検証の実施	9.4.3.1	
	検証自体の評価の為に、検証結果として下記の情報が含まれている事を確認する。 ・検証作業成果物の固有の識別 ・検証計画と検証仕様の言及 ・検証環境と使用検証ツールの設定 ・期待される結果に対する検証結果の準拠度合い ・検証の合否判定 ・(実施していないテストケースが存在している場合は)検証を行わなかった理由	9.4.3.2	仕様書上は必要な場合は検証中に使用するキャリブレーションデータが要求されているが、コンパイラに関しては必要ないので省略
	上記の内容をまとめた検証報告書を作成する。	9.5.3	