

仕様書セミナー

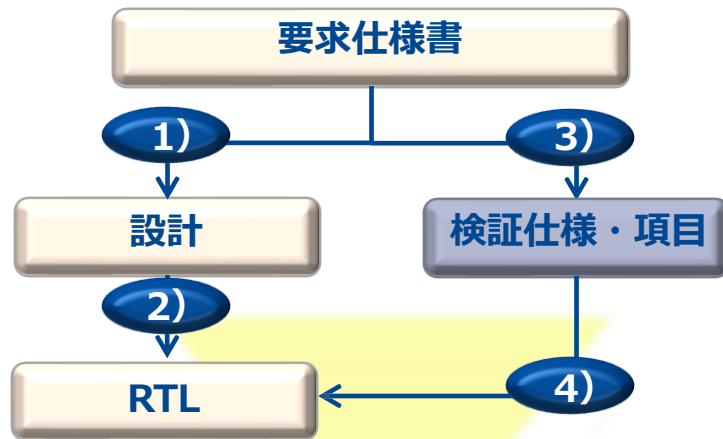
設計仕様書  
検証仕様書



vtech™

# 設計・検証における仕様書の重要性

## ■設計・検証でのリスク内在において



開発において、設計における各フェーズには「要求仕様」をベースに“等価”を意識した設計・検証が必要である。

左記より、

- 1 機能から構造への落とし込み（設計仕様書）
- 2 構造の実現化（RTL）
- 3 機能が網羅されているかの確認（検証仕様・項目）
- 4 構造 = 機能の実現性確認（検証）

上記箇所で誤解釈や実装漏れ等のリスクを孕んでおり開発期間への影響が大きいと考える。

### 【開発想定フロー】



### 【実際のフロー】

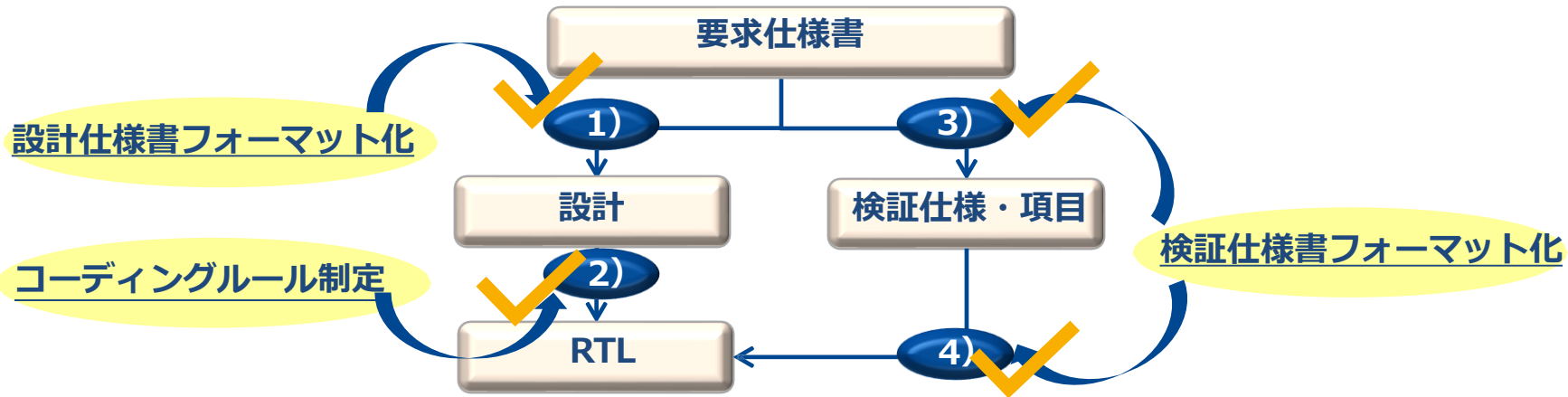


上記のように、各ポイントでの手戻りの発生から、開発期間の延長、コスト増大の可能性を秘めている。  
上記が発生する要因として、エンジニアへの依存からの作業漏れも要因である。

▶ “仕様書のフォーマット化”による作業一元化と効率化、また資産性向上が見込めると考える。

# ▶ Vtechでのフォーマット化における取組

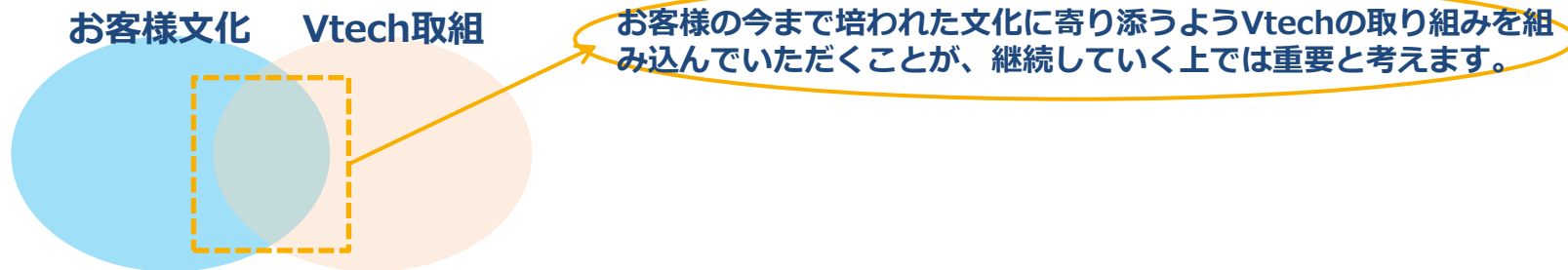
## ■ Vtechでの取り組み



Vtechとしては、過去からの500件強の実績を元に、ルール化における取り組みを実施。上記でのルールを実施した案件もあり、一様に“手戻り”の低減へとつながった実績もあります。

## ■ ルール運用におけるVtechの取り組み

設計仕様書・検証仕様書フォーマット、コーディングルールはVtechとしても、現時点での一つの解としています。ただお客様すべてに適切かは要相談と考えております。



# ▶▶ 設計仕様書・検証仕様書セミナー

マスク改訂が必要になるような不具合の50%は、要求仕様の実装漏れや仕様書の誤解が原因とされています。このような不具合をなくすには、「第三者検証（設計者以外による検証）」の実施が必須です。

長年の第三者検証の実施を通じて、「抜け・漏れ」の防止には、洗練された仕様書フォーマットの活用が有用との結論に至りました。

弊社の設計&検証サービスの一環として、仕様書フォーマットのセミナーも実施しています。仕様書の在り方について、お悩みの方は、是非、お問い合わせください。



## 検証仕様書フォーマットのセミナー

3: 検証仕様書でのフォーマット化

【検証仕様書において】  
 弊社が得意とする検証において、弊社は「検証仕様書」を作成し、検証方針にズレがないか漏れがないかを事前にお客様と話し込めるようしております。  
 よって検証におけるズレ・漏れが発生しないよう、定型的な部分等を切り分け、フォーマット化しております。

左記のように、検証における基本情報（方針・環境等）検証の実行内容、検証結果と、当資料だけで検証の目的が明確化され、検証のズレを話し込める項目を立てさせていただきます。

1 目次	4
1.1 概要	4
1.2 変更履歴	4
2 検証方針	5
2.1 適用範囲	5
2.2 適用ツール	5
2.3 検証方法	5
2.4 検証環境	5
3 検証標準	6
3.1 検証標準	6
3.2 標準項目	6
3.3 デュアルプラットフォーム検証	7
3.4 セキュリティ脆弱性/フィッシング脆弱性	7
3.4.1 CPU 依存	7
3.4.2 両プラットフォーム同時実行	7
3.4.3 OS/OS 依存	7
3.4.4 アーキテクチャ	7
4 検証結果	10
4.1 レジスタ値一覧	10
4.2 検証項目一覧	10
4.3 検証ログファイル	10
5 ソフトウェア	12
5.1 検証項目	12
5.2 検証項目	12
5.3 検証項目	12
5.4 検証項目	12
5.5 検証項目	12
5.6 検証項目	12
5.7 検証項目	12
5.8 検証項目	12
5.9 検証項目	12
5.10 検証項目	12
5.11 検証項目	12
5.12 検証項目	12
5.13 検証項目	12
5.14 検証項目	12
5.15 検証項目	12
5.16 検証項目	12
5.17 検証項目	12
5.18 検証項目	12
5.19 検証項目	12
5.20 検証項目	12
5.21 検証項目	12
5.22 検証項目	12
5.23 検証項目	12
5.24 検証項目	12
5.25 検証項目	12
5.26 検証項目	12
5.27 検証項目	12
5.28 検証項目	12
5.29 検証項目	12
5.30 検証項目	12
5.31 検証項目	12
5.32 検証項目	12
5.33 検証項目	12
5.34 検証項目	12
5.35 検証項目	12
5.36 検証項目	12
5.37 検証項目	12
5.38 検証項目	12
5.39 検証項目	12
5.40 検証項目	12
5.41 検証項目	12
5.42 検証項目	12
5.43 検証項目	12
5.44 検証項目	12
5.45 検証項目	12
5.46 検証項目	12
5.47 検証項目	12
5.48 検証項目	12
5.49 検証項目	12
5.50 検証項目	12
5.51 検証項目	12
5.52 検証項目	12
5.53 検証項目	12
5.54 検証項目	12
5.55 検証項目	12
5.56 検証項目	12
5.57 検証項目	12
5.58 検証項目	12
5.59 検証項目	12
5.60 検証項目	12
5.61 検証項目	12
5.62 検証項目	12
5.63 検証項目	12
5.64 検証項目	12
5.65 検証項目	12
5.66 検証項目	12
5.67 検証項目	12
5.68 検証項目	12
5.69 検証項目	12
5.70 検証項目	12
5.71 検証項目	12
5.72 検証項目	12
5.73 検証項目	12
5.74 検証項目	12
5.75 検証項目	12
5.76 検証項目	12
5.77 検証項目	12
5.78 検証項目	12
5.79 検証項目	12
5.80 検証項目	12
5.81 検証項目	12
5.82 検証項目	12
5.83 検証項目	12
5.84 検証項目	12
5.85 検証項目	12
5.86 検証項目	12
5.87 検証項目	12
5.88 検証項目	12
5.89 検証項目	12
5.90 検証項目	12
5.91 検証項目	12
5.92 検証項目	12
5.93 検証項目	12
5.94 検証項目	12
5.95 検証項目	12
5.96 検証項目	12
5.97 検証項目	12
5.98 検証項目	12
5.99 検証項目	12
5.100 検証項目	12



## 設計仕様書フォーマットのセミナー

1. 従来の設計仕様書における問題点

0. 序論  
 ○プロジェクトの工期における問題は「不具合」の混入と検証の「収束」にあります。

不具合混入経路分析観点

カバレッジレポート分析観点

不具合混入は「設計」時に行われます。これらの問題の多くは設計レビューをすり抜けてしまうようです。  
 本来レビューで見つかるはずの問題の多くが「検証」時に検出されることで検証期間が延び、多くの手戻りを発生させています。

一方、検証の収束における指標の一つであるカバレッジは、未踏カバレッジ要因は「コードの冗長性」及び「検証項目不足」になりますが、項目不足観点では実装時（設計時）の問題を引きずるケースが多く見られます。



Thank you!