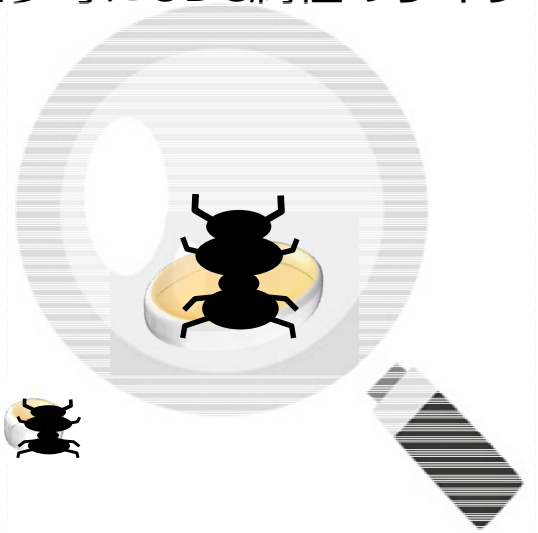
第5回の活動報告（ODC分析応用編）

ODC分析研究会　特別講義

ODC分析の属性の応用編（１）

~プロセスリファレンスモデル（PRM）を参考にODC属性のライフサイクルを考える~



2020/10/01

プロダクト品質とプロセス品質

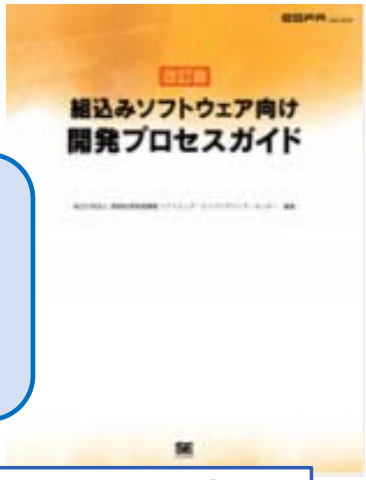
・プロダクト品質
　システムや製品、サービスそのものの品質
・プロセス品質
　プロダクトを作るときの工程や作業の品質

□仮説
　プロダクトの品質を上げるには、プロセスの品質を上げること
または、
　プロセス品質をあげると、プロダクトの品質があがる（はず*1）

はず*1）“プロセスを守る“だけでなく、プロセスの健全性も重要

プロセスリファレンスモデルとは何か？！

講義で解説したプロセスリファレンスのモデルは、「改訂版　組込みソフトウェア向け開発プロセスガイド*1」を利用しました。



『組込みソフトウェア向け　開発プロセスガイド』とは
（英語名＝ ESPR：EmbeddedSystem development Process Reference.
本文中ではES

ESPRは、組込
プラクティス

自社（自組織）のプロセスを確認するためのひな型

「
プ
い。

ODC分析をプロセス改善につなげるには、自組織のプロセスを熟知しないとどの工程やアクティビティ/タスクを改善すべきかをとらえられません。自組織の開発プロセスを振り返ってみてください。

*1　出典　独立行政法人情報処理推進機構
https://www.ipa.go.jp/sec/publish/tn07-005.html

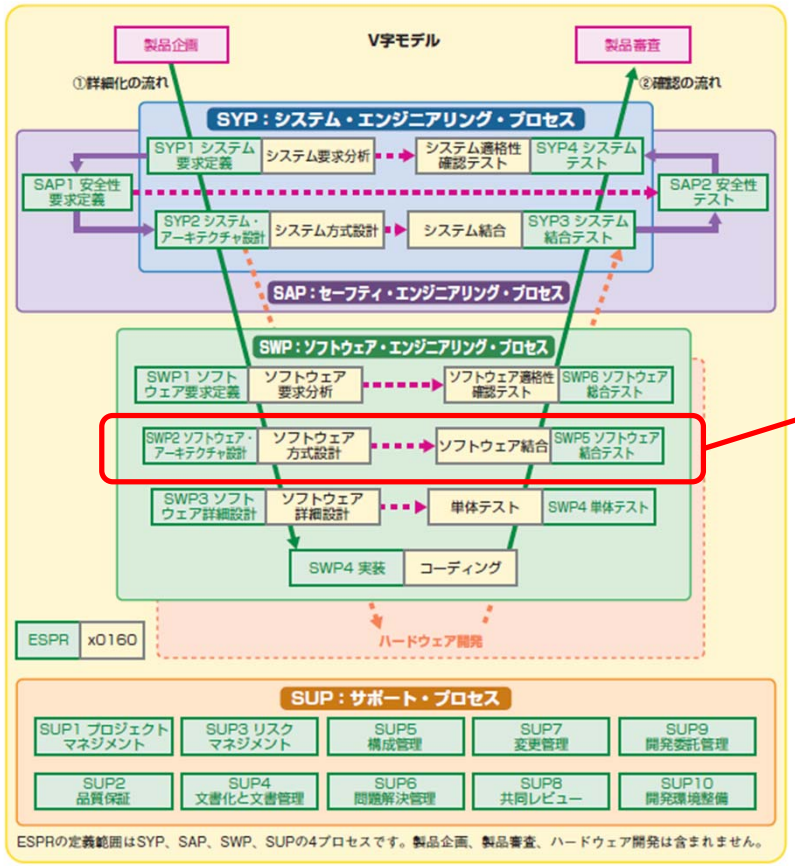
SQiP Software Quality Profession

ODC分析の属性の応用編（１）

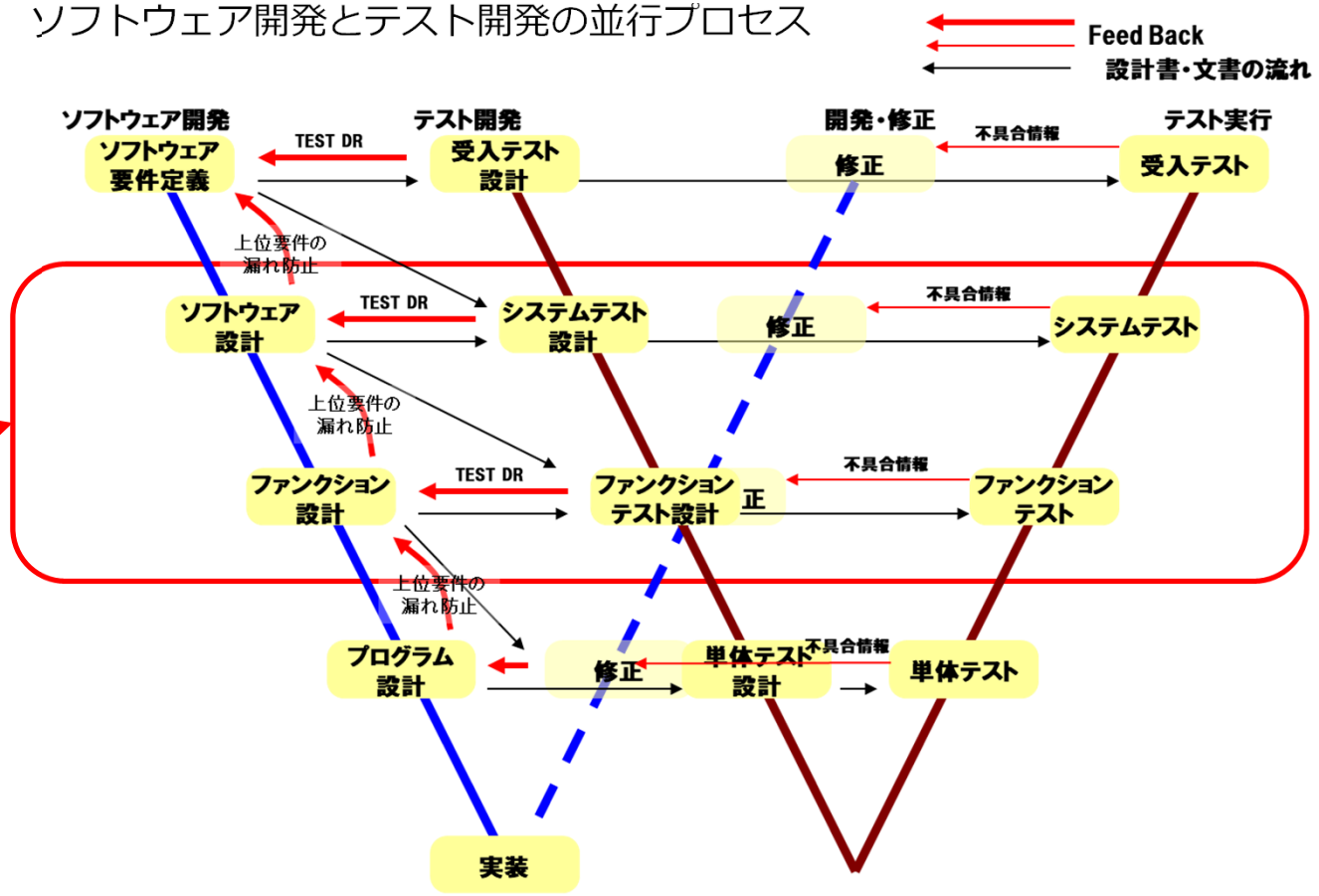
Chapter.1　ODCタイプ属性、ODCトリガー属性と
プロセスリファレンスモデル（PRM）の関係

～属性とPRMの関係性を知る～

Chapter.1　ODCタイプ属性、ODCトリガー属性とPRMの関係



ソフトウェア開発とテスト開発の並行プロセス



SQiP Software Quality Profession

Chapter.1　ODCタイプ属性、ODCトリガー属性とPRMの関係

2.1.4 インタフェースの設計

入力
(SW105) ソフトウェア要求仕様書
(SY205) システム・アーキテクチャ設計書

概要
機能ユニット間のインタフェースを設計する。

出力
(SW204) ソフトウェア・インタフェース設計書

参照情報
各ハードウェア仕様書

■実施内容

ソフトウェアを構成する機能ユニット間のインタフェースを設計する。

①メモリー・レイアウトを設計する
- メモリ全体のレイアウトを設計する。

②メモリー空間・領域を詳細化する
- ソフトウェアとして利用する各領域の詳細を決める。
  - ・領域全体の名称/構成/サイズを決める
  - ・領域内フィールドの名称/位置/形式/サイズ/意味/初期値/アクセス元など

③機能ユニット間インタフェースを設計する
- 呼び出し手順、パラメータ、復帰情報等を設計する。

④共通情報を一元化、論理値化する
- 変更に対して修正負荷の高い情報は、一元化し、かつ論理値化しておく。
  - ・テーブル・オフセット
  - ・共通定数など

■注意すべき事項

●インタフェース設計の考慮点
- 異常/例外コード、メッセージを早めに抽出し体系化し、一覧表として整理する。
- 排他的処理が必要な共通データについては更新/参照元を明確化する。
- 共通リソースについては、確保/開放タイミングを明らかにする。
- ビット構成等では拡張性を考慮する。

■プロセス：ソフトウェアエンジニアリングプロセス
■アクティビティ：ソフトウェア・アーキテクチャ設計
■タスク/サブタスク：インタフェースの設計
開発するときの活動／観点
　ソフトウェア機能間のインタフェースを決める。

✓メモリレイアウトが適切に設計されている
✓機能ユニット毎に割り振られたメモリ領域の詳細設定が適切である
✓機能ユニット間の呼び出し手順・パラメータ・復帰情報が適切である
✓異常・例外コード、メッセージが明確になっている
✓排他処理が必要な共通データの更新や参照元が適切である
✓共通リソースの確保・開放タイミングが適切である

■成果（Outcame）：ソフトウェア・インタフェース設計書

Chapter.1　ODCタイプ属性、ODCトリガー属性とPRMの関係

どうあるべきか、テストの視点を加えてみる

開発するときの観点		組込みソフトウェア向け　開発プロセスガイド（ESPR）	開発観点に対して確認する観点
インタフェースの設計	ソフトウェア機能間のインタフェースを決める	メモリレイアウトが適切に設計されている	機能処理を行えるメモリが確保されていることを確認する 処理が終了した時に、メモリを開放することを確認する
		機能ユニット毎に割り振られたメモリ領域の詳細設定が適切である	共有メモリで処理する限界値での動作を確認する。 共有メモリで処理する限界値以上の動作を確認する。
			割込みイベント時、バックグラウンドで動作する機能は動作していることを確認する 割込みイベント後の割込み消去で、割込み前の機能が復帰することを確認する

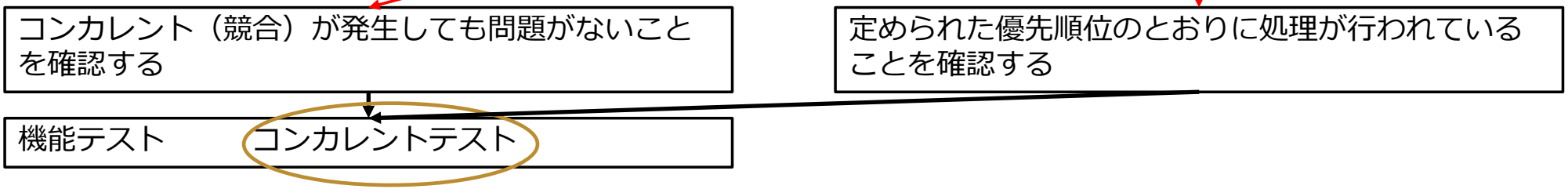
Chapter.1　ODCタイプ属性、ODCトリガー属性とPRMの関係

どうあるべきか、テストの視点を加えてみる

開発するときの観点		組込みソフトウェア向け　開発プロセスガイド（ESPR）	開発観点に対して確認する観点
インタフェースの設計	ソフトウェア機能間のインタフェースを決める	異常・例外コード、メッセージが明確になっている	異常動作時のエラーメッセージが表示されることを確認する
		排他処理が必要な共通データの更新や参照元が適切である	優先順位を考慮した排他制御の確認
		共通リソースの確保・開放タイミングが適切である	共通リソースを使用時の動作を確認する 共通リソースを同時に使用し、リソースの限界値の動作を確認する。 共通リソースを同時に使用し、リソースの限界値以上時の動作を確認する。

Chapter.1　ODCタイプ属性、ODCトリガー属性とPRMの関係

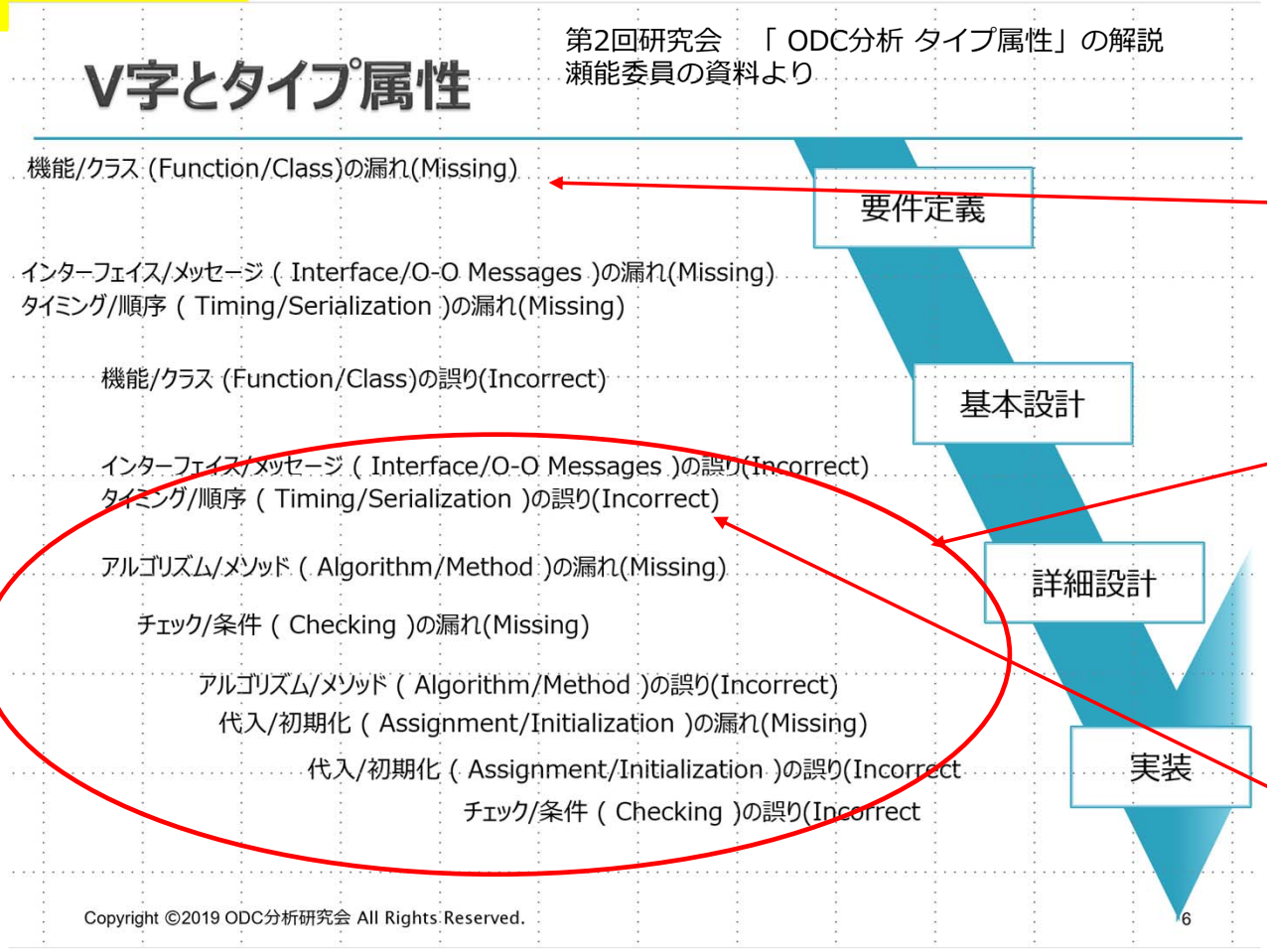
開発するときの観点		組込みソフトウェア向け　開発プロセスガイド（ESPR）	開発観点に対して確認する観点
インタフェースの設計	ソフトウェア機能間のインタフェースを決める	排他処理が必要な共通データの更新や参照元が適切である	優先順位を考慮した排他制御の確認



欠陥因子（欠陥となる要因）：③

インタフェース設計の制御ミスじゃね＾＾

Chapter.1　ODCタイプ属性、ODCトリガー属性とPRMの関係



無理やり品質特性
機能適合性　：　機能完全性
機能正確性
機能適切性

欠陥因子（欠陥となる要因）：①
ソフトウェア要求定義の機能漏れじゃね＾＾

回帰テスト

欠陥因子（欠陥となる要因）：②
ソフトウェア要求定義のタスクで想定できない部分は、
実装レベルで混入したんじゃねぇ＾＾；

機能テスト　コンカレントテスト

欠陥因子（欠陥となる要因）：③
インタフェース設計の制御ミスじゃね＾＾

SQiP Software Quality Profession

Chapter.1　ODCタイプ属性、ODCトリガー属性とPRMの関係

どんな観点を検証するのかを加えてみる

開発観点		組込みソフトウェア向け　開発プロセスガイド（ESPR）	開発観点に対して確認する観点	検証実施工程
ソフトウェア構成の設計	ソフトウェア要求分析を実現するために必要となる、ソフトウェアの構成・ソフトウェア機能分割・モジュール分割を明確にしており抽出し、ソフトウェア詳細設計に記載する内容を明確にする	実現する機能を明確にしている	実現通りの動作を機能がすることを確認する。	機能テスト
		リカバリ・データ保障等、信頼性を考慮している	リカバリ・データを作成できることを確認する リカバリ・データを保持できることを確認する。 リカバリ・データの上書時に、元のデータを破棄することを確認する リカバリ・データで以前の状態に戻せることを確認する。	信頼性テスト
		不具合追跡・デバッグ・テスト効率性等、保守性を考慮している	不具合が発生した場所の特定を容易にするLOGが出力されていることを確認する	保守性テスト
		品質確保を考慮した、機能の局所化・カプセル化を行っている	共通機能は、共通処理していることを確認する 機能間の影響を避けるために、各機能が独立していることを確認する	Not applicable
		開発効率・メモリ効率を考慮した機能の部品化・共通化を行っている	部品化した機能が動作するメモリ量を確保していることを確認する 部品化した機能が動作するメモリ量の限界値まで消費していても動作すること	Not applicable
		出荷後のソフトウェア変更による拡張性を考慮している	機能拡張時、他機能への影響が少ないことを確認する	拡張性テスト
		フェールセーフによる安全性を考慮している	異常動作時、フェールセーフが動作することを確認する フェールセーフ時、最低限の機能が動作することを確認する フェールセーフにより、異常による影響を局所的に抑えられることを確認する	安全性テスト
		ソフトウェア製作が出来る機能抽出範囲となっている		機能テスト

レビューで確認できる項目

Chapter.1　ODCタイプ属性、ODCトリガー属性とPRMの関係

どんな観点を検証するのかを加えてみる

開発観点		組込みソフトウェア向け　開発プロセスガイド（ESPR）	開発観点に対して確認する観点	検証実施工程
インタフェースの設計	ソフトウェア機能間のインタフェースを決める	メモリレイアウトが適切に設計されている	機能処理を行えるメモリが確保されていることを確認する 処理が終了した時に、メモリを開放することを確認する	資源効率性テスト
		機能ユニット毎に割り振られたメモリ領域の詳細設定が適切である	共有メモリで処理する限界値での動作を確認する。 共有メモリで処理する限界値以上の動作を確認する。	資源効率性テスト
		機能ユニット間の呼び出し手順・パラメータ・復帰情報が適切である	機能間の優先順位の確認をする	インタフェーステスト
		異常・例外コード、メッセージが明確になっている	異常動作時のエラーメッセージが表示されることを確認する	機能テスト
		排他処理が必要な共通データの更新や参照元が適切である	優先順位を考慮した排他制御の確認	機能テスト
		共通リソースの確保・開放タイミングが適切である	共通リソースを使用時の動作を確認する 共通リソースを同時に使用し、リソースの限界値の動作を確認する。 共通リソースを同時に使用し、リソースの限界値以上時の動作を確認する。	効率性テスト

Chapter.1　ODCタイプ属性、ODCトリガー属性とPRMの関係

第3回研究会　「 ODC トリガー属性」の解説
武田委員の資料より

結合/システムテストに適用するトリガー

27 / 48

	値	意味・用途
単純	基本	1つの機能をデフォルトの状態で実行した結果、欠陥を検出した場合。
	ﾊﾞﾘｴｰｼｮﾝ	1つの機能をオプション/パラメータを変更して実行した結果、欠陥を検出した場合。
	順序	複数の機能を順番に実行した結果、または単一機能を複数回実行した結果、欠陥を検出した場合。(個々の機能は正しく動作する)
	例外/復帰	出した場合。
	起動/再起動	シャットダウン、スリープ、ネットワーク遮断、電 状態から通常状態に戻る)過程をテストした結果
	HW構成	ハードウェア構成の変更に関するテストを実行 した場合。
複雑	SW構成	ハードウェア構成の変更に関するテストを実行 した場合。
	通常系	負荷テストなどある意図をもったテストを実施 問題により欠陥が検出された。(目的としたテス

手順の複雑さ

主に結合テストに適用

これではトリガーを十分に表現していない

開発観点に対してテストで確認する観点からテストタイプ（例：信頼性テスト、パフォーマンステスト、etc）との関連性を考えても十分にODC属性を表現できていない。

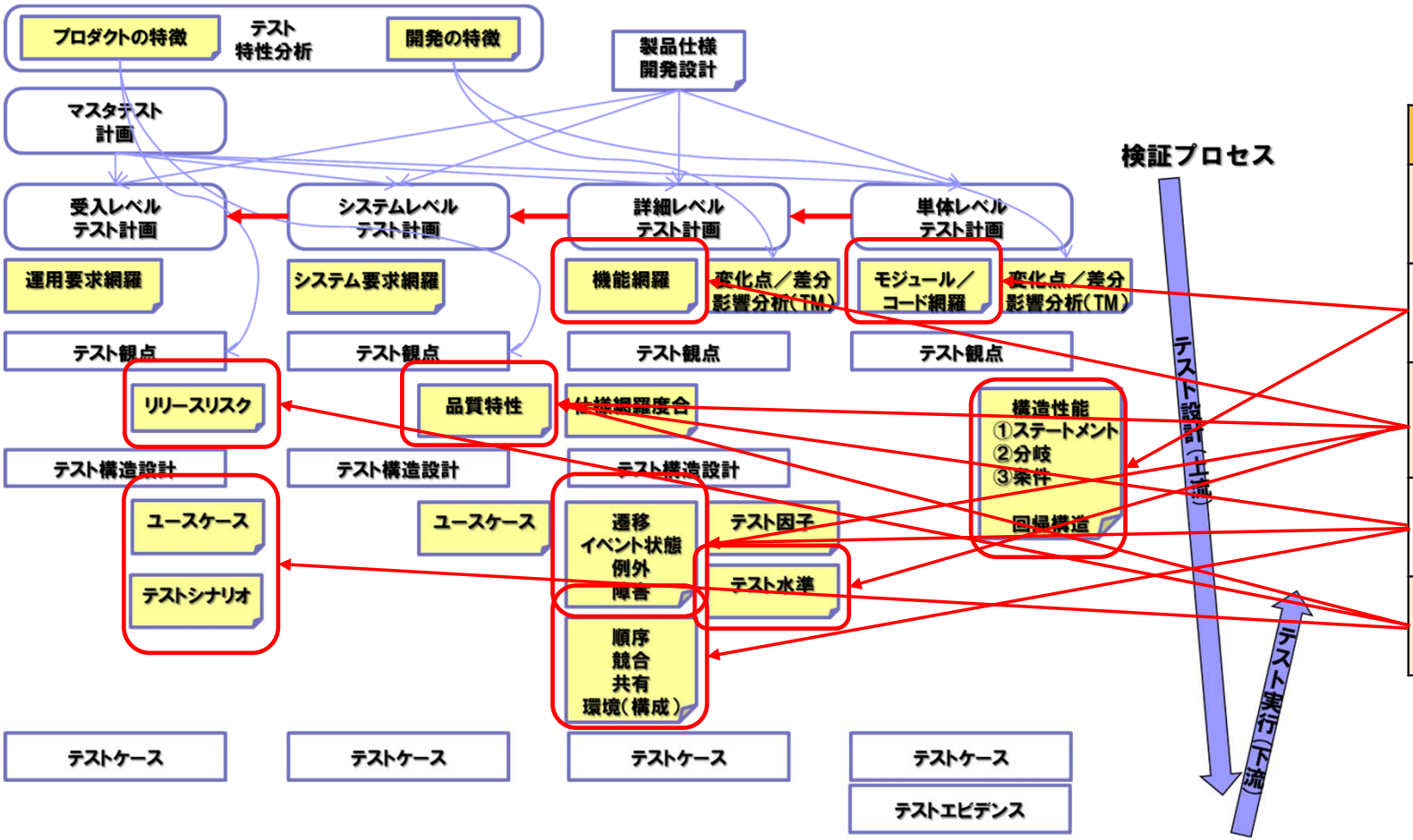
開発観点やテストベースからテスト開発のプロセスで抽出するテスト観点を検討する必要があると考える。次ページ以降のテストの考え方はその参考例です。

■プロセス：ソフトウェアエンジニアリングプロセス
■アクティビティ：ソフトウェア・アーキテクチャ設計
■タスク/サブタスク：ソフトウェア構成の設計

開発観点に対して確認する観点	検証実施工程
実現通りの動作を機能がすることを確認する。	機能テスト
リカバリ・データを作成できることを確認する リカバリ・データを保持できることを確認する。 リカバリ・データの上書時に、元のデータを破棄することを確認する リカバリ・データで以前の状態に戻せることを確認する。	信頼性テスト
部品化した機能が動作するメモリ量を確保しているこ 限界値まで消費	Not applicable
ことを確認する	拡張性テスト
ることを確認す 作することを確 響を局所的に抑えられることを確認する	安全性テスト
	機能テスト

Chapter.1　ODCタイプ属性、ODCトリガー属性とPRMの関係

テスト開発でどのようなテスト構造でテストケースが作成されているかを整理してみた



第3回研究会　「 ODC トリガー属性」の解説
武田委員の資料より

トリガーの種類	トリガーの値の数
レビュー	9
単体テスト	2
機能テスト	4
システムテスト	5
インパクト	ー

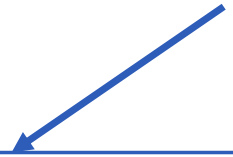
*注　レビューは、開発プロセスで重点的に実施されるものとする。

SQiP Software Quality Profession

Chapter.1　ODCタイプ属性、ODCトリガー属性とPRMの関係

■ODCタイプ属性、ODCトリガー属性とPRMの関係

✓ライフサイクルプロセスでODC属性で分類される欠陥を抑止するタスクを明確にすることで、アクティビティの成熟度を測るメトリクスにも用いることができる



レビューやテストで欠陥を検出するアクティビティでODC属性をテストやレビューの観点としてどの程度取り入れられているかを評価する。
レビューならチェックリストベースやレビュー観点、テストならテスト観点をテスト設計レビューで確認しているかをプロセス評価する。

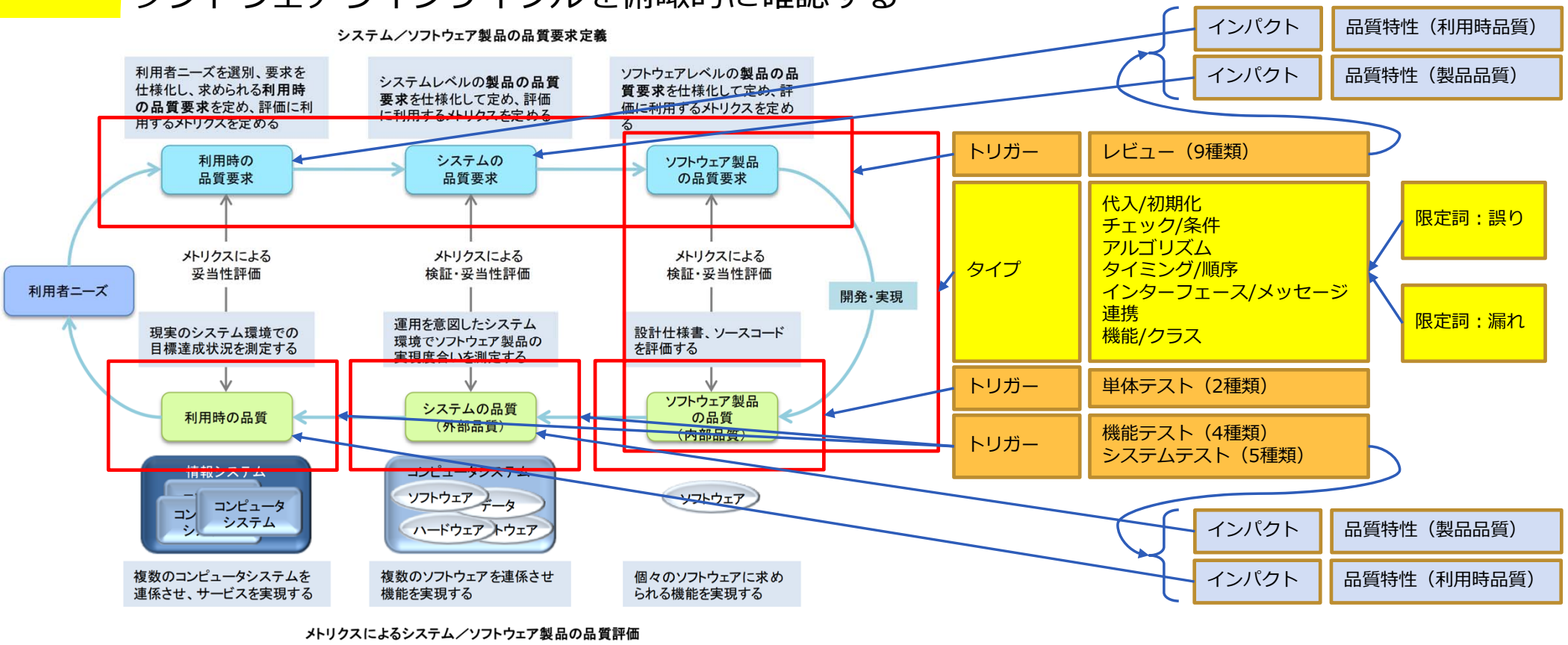
ODC分析の属性の応用編（１）

Chapter.2　ODC属性のライフサイクルの解説

～プロセスリファレンスモデルとは～

Chapter.3　ODC属性のライフサイクルの境界値による分析方法

ソフトウェアライフサイクルを俯瞰的に確認する



出典
経済産業省(以降はMETIとする)　ソフトウェアメトリクス高度化プロジェクト　プロダクト品質メトリクスWG
システム／ソフトウェア製品の品質要求定義と品質評価のためのメトリクスに関する調査報告　　発行：2011年 3月
http://www.meti.go.jp/policy/it_policy/softseibi/metrics/20110324product_metrics2010.pdf

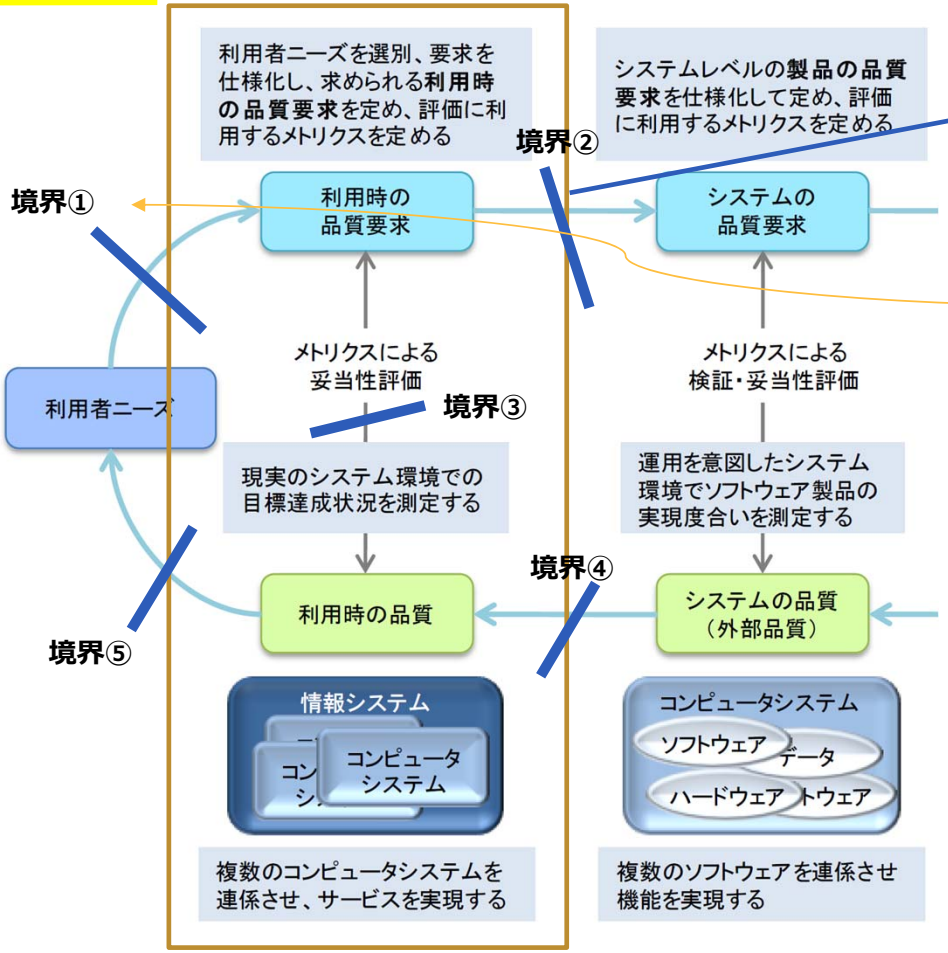
ODC分析の属性の応用編（１）

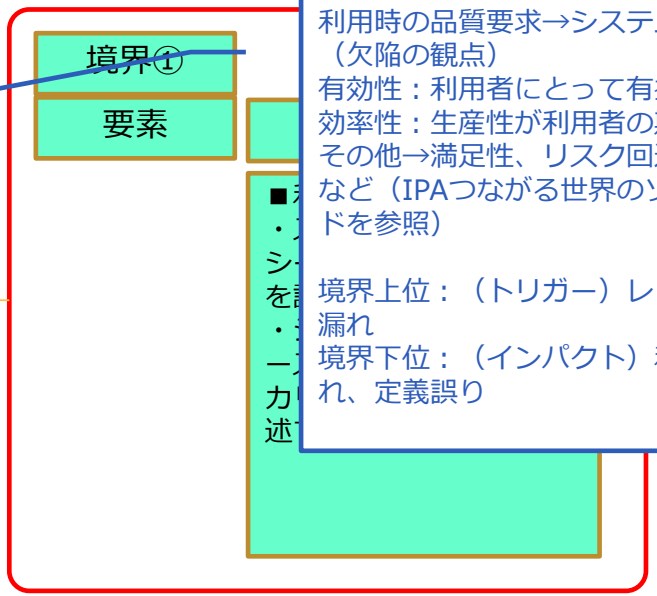
Chapter.3　ODC属性のライフサイクルの境界による分析方法

～ライフサイクルの境界で何がわかるのか～

Chapter.3　ODC属性のライフサイクルの境界値による分析方法

ライフサイクルのプロセス間を境界ととらえる

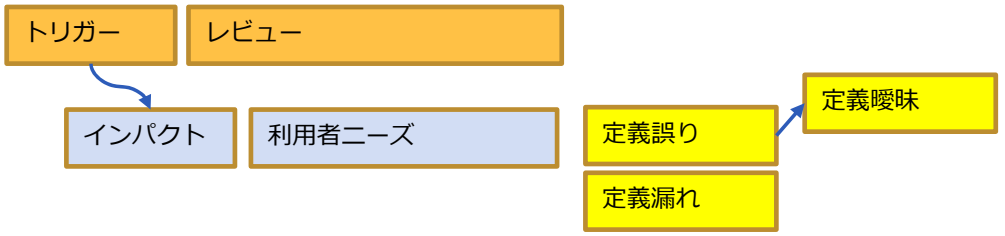




例：境界②での欠陥検出
利用時の品質要求→システムの品質要求
（欠陥の観点）
有効性：利用者にとって有益（目的達成する）か
効率性：生産性が利用者の期待通りか
その他→満足性、リスク回避性、利用状況網羅性など（IPAつながる世界のソフトウェア品質ガイドを参照）

境界上位：（トリガー）レビューのよる欠陥除去漏れ
境界下位：（インパクト）利用者ニーズの定義漏れ、定義誤り

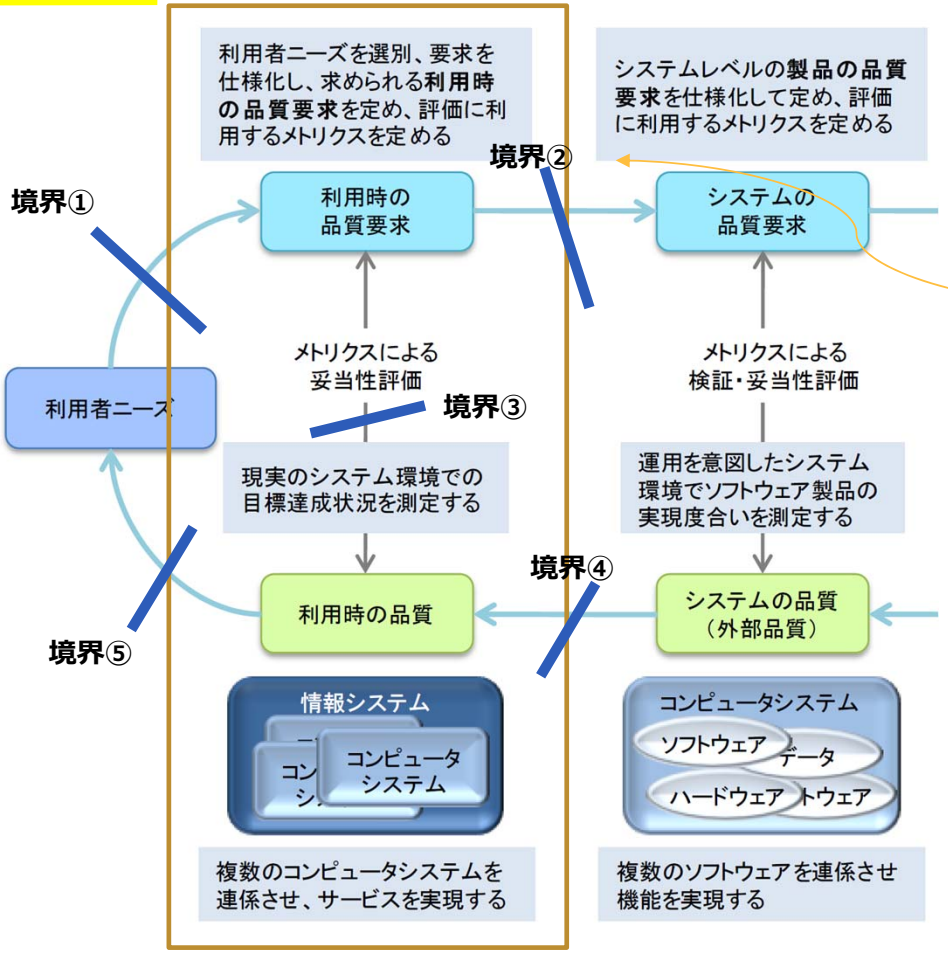
境界の欠陥モード（境界で発生する要素）

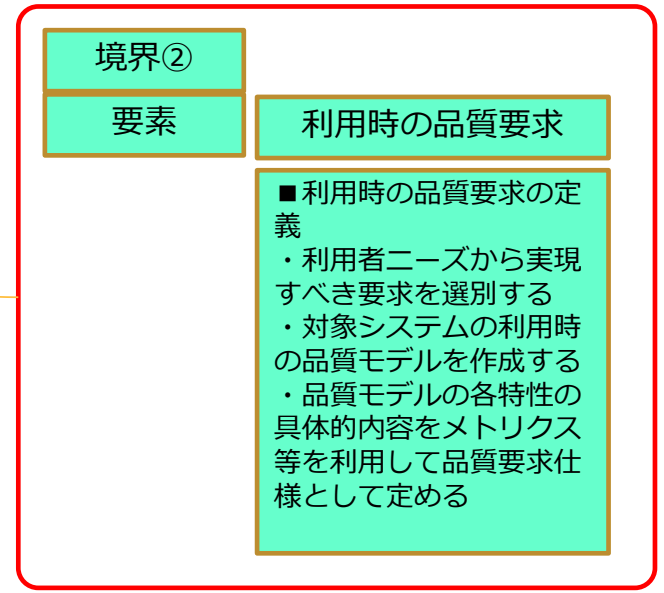


SQiP Software Quality Profession

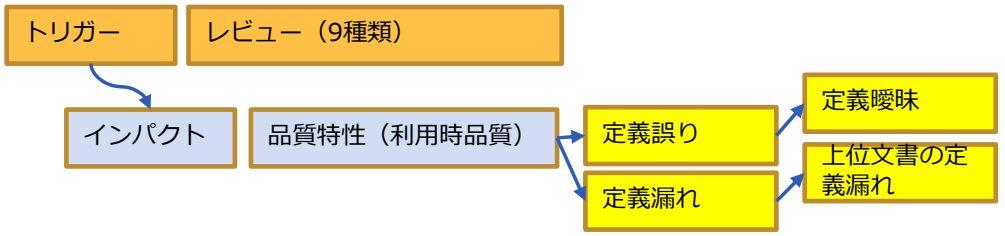
Chapter.3　ODC属性のライフサイクルの境界値による分析方法

ライフサイクルのプロセス間を境界ととらえる





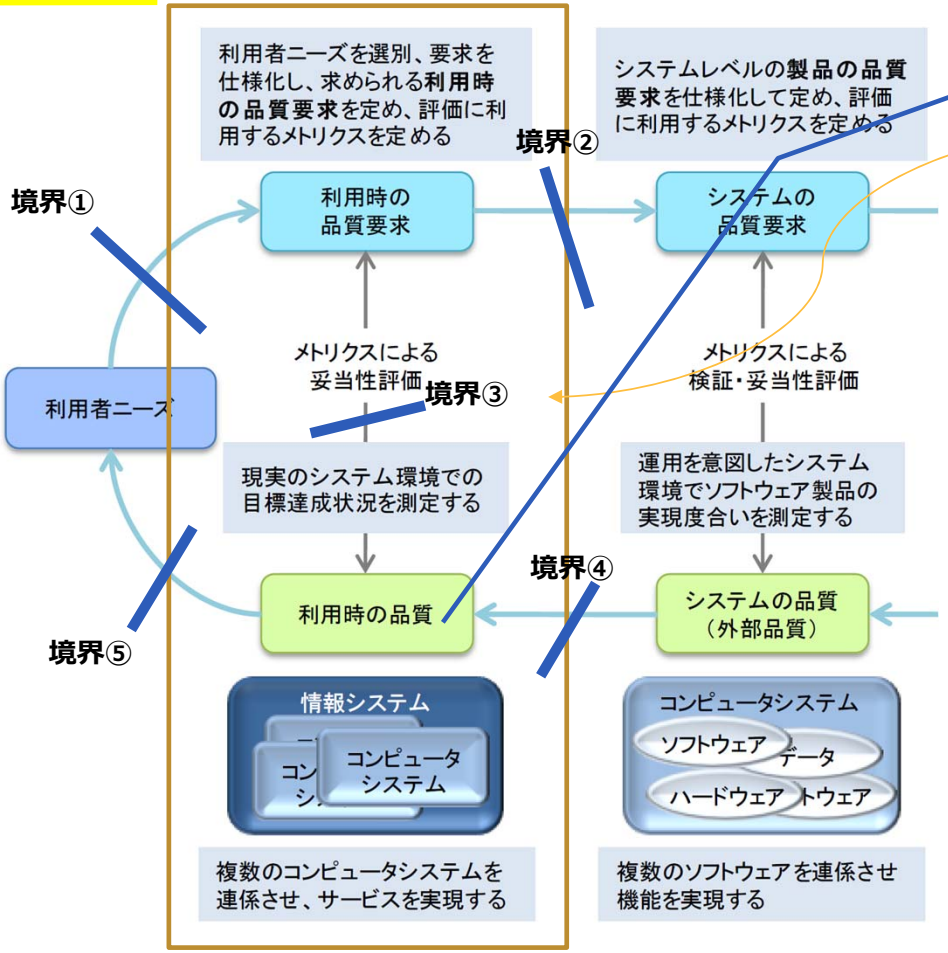
境界の欠陥モード（境界で発生する要素）



SQiP Software Quality Profession

Chapter.3　ODC属性のライフサイクルの境界値による分析方法

ライフサイクルのプロセス間を境界ととらえる



例：境界③と境界④、⑤のプロセスの課題は、
境界③は、レビュー漏れ（特に開発チームとテストチームの共通の課題）
→レビューで発見するべき問題
境界④下流側は、テストによる通常の検出
境界⑤は、テスト漏れ。境界③でない場合は、テスト開発の見直しが必要

境界の欠陥

トリガー　レビュー（9種類）

ューで発見できる
の問題でないのか？

レビュー　見逃し

トリガー　主にシステムテスト

インパクト

品質特性（利用時の品質）

品質特性（システムの品質）

定義誤り

定義曖昧

定義漏れ

上位文書の定義漏れ

タイプ

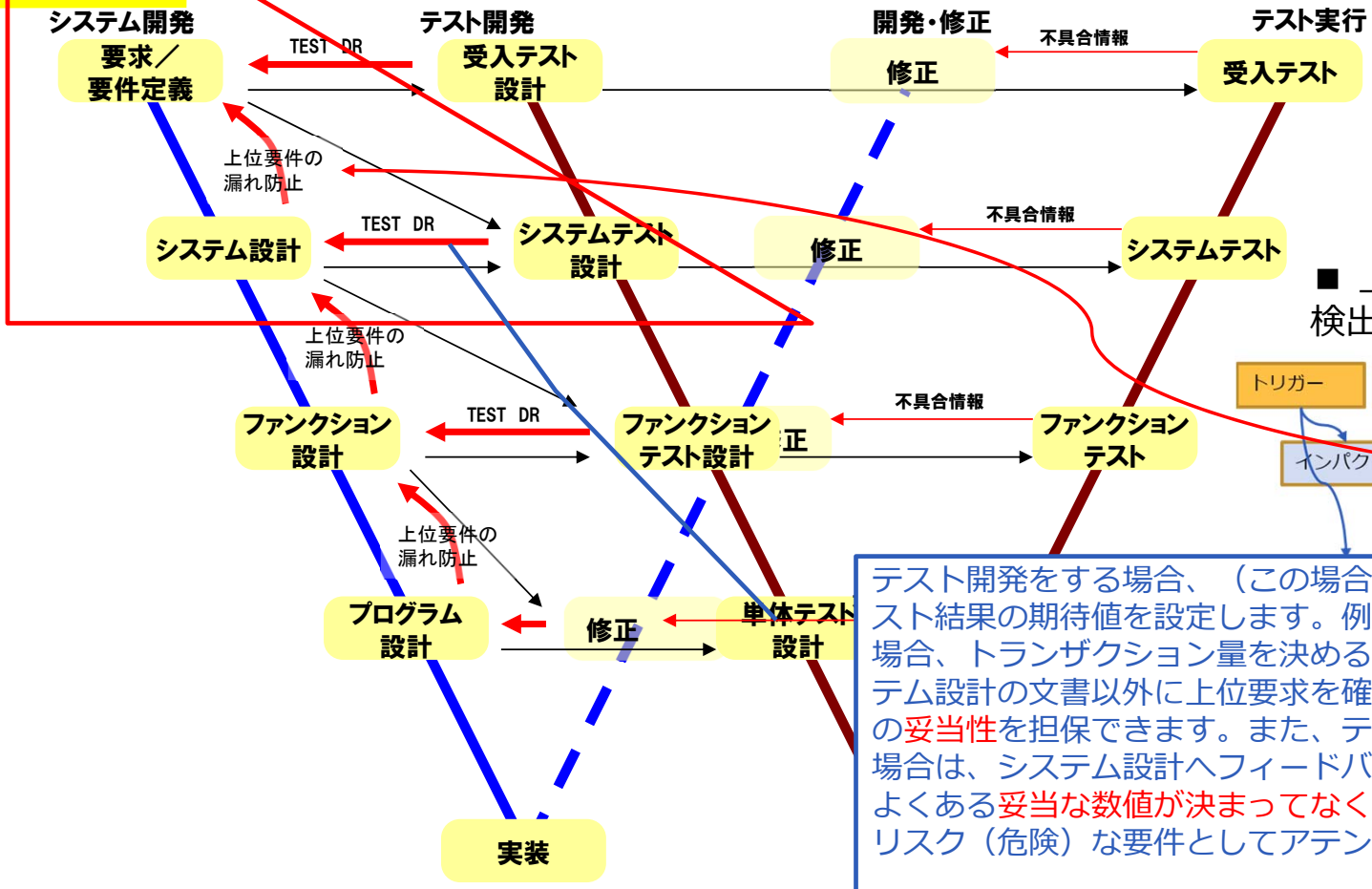
代入/初期化
チェック/条件
アルゴリズム
タイミング/順序
インターフェース/メッセージ
連携
機能/クラス

限定詞：誤り

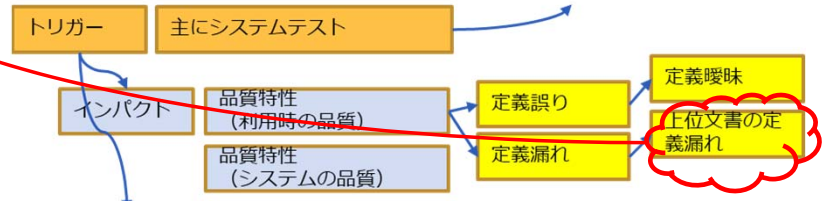
限定詞：漏れ

SQiP Software Quality Profession

Chapter.3　ODC属性のライフサイクルの境界値による分析方法

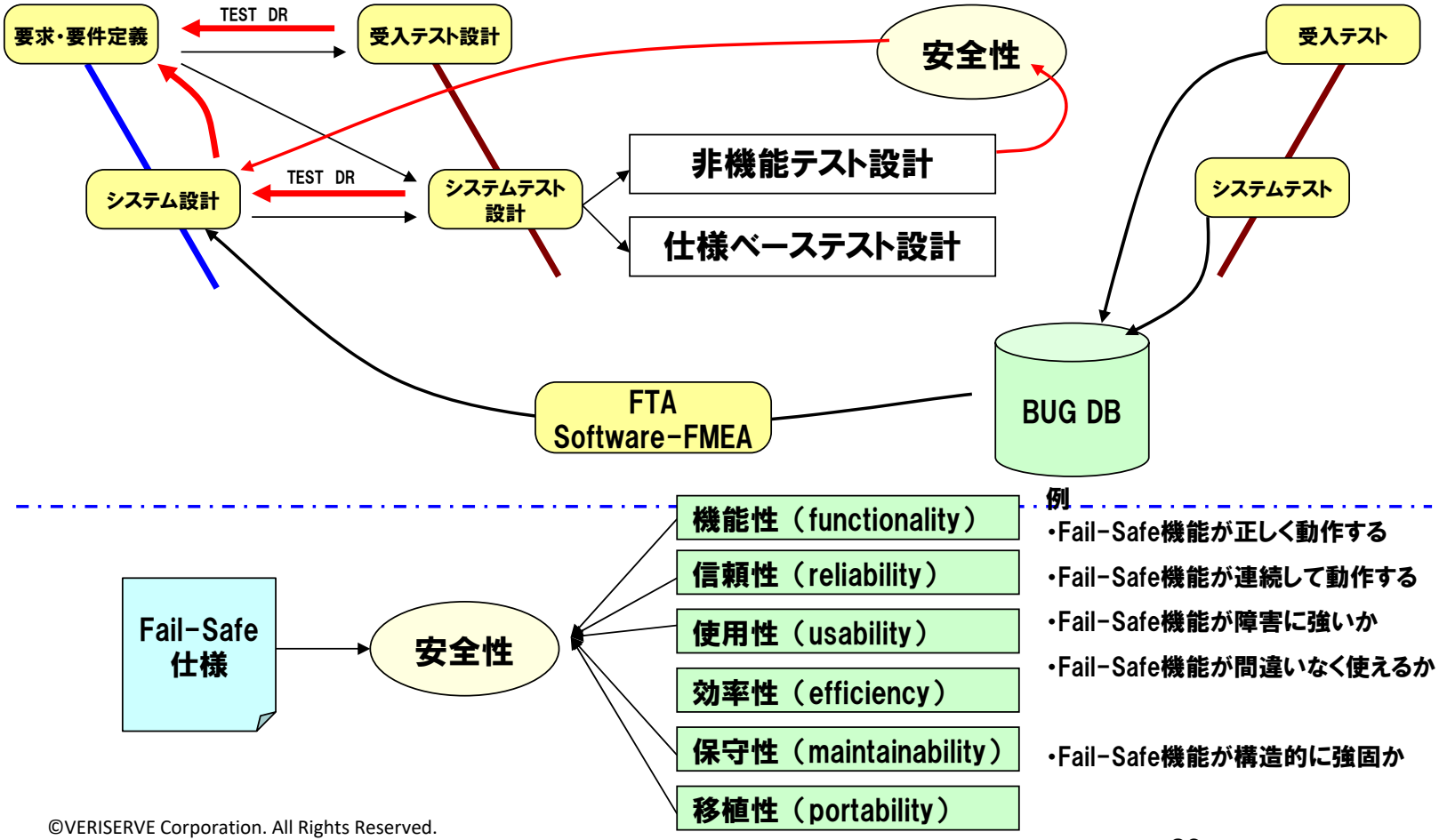


■ 上位文書の漏れは、自プロセスだけでは検出できない



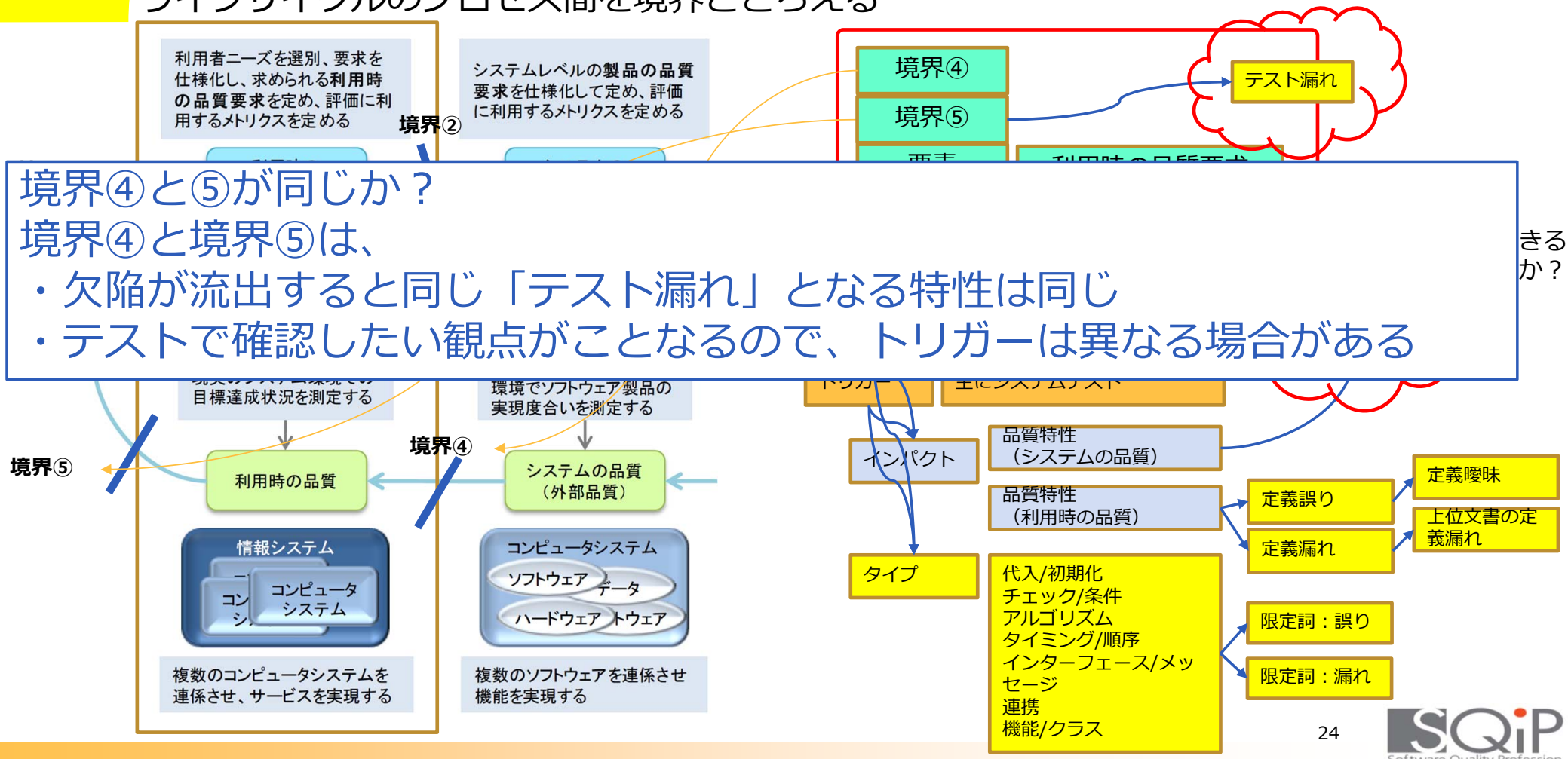
テスト開発をする場合、（この場合はシステムテストを例に）テスト結果の期待値を設定します。例えばパフォーマンステストの場合、トランザクション量を決めるなどテストベースとなるシステム設計の文書以外に上位要求を確認することで、期待値の数値の妥当性を担保できます。また、テストベースに数値情報が無い場合は、システム設計へフィードバックします。
よくある妥当な数値が決まってなく、下流に進んでしまう場合はリスク（危険）な要件としてアテンションします。

Chapter.3　ODC属性のライフサイクルの境界値による分析方法



Chapter.3　ODC属性のライフサイクルの境界値による分析方法

ライフサイクルのプロセス間を境界ととらえる



Chapter.3　ODC属性のライフサイクルの境界値による分析方法

■ODC属性のライフサイクルの境界による分析方法

✓ODC属性はライフサイクルプロセスに対して多重の意味を持たせることでプロセスのボトルネックを抽出することが可能

ODC分析をプロセス改善につなげるには、自組織のプロセスを熟知してODC属性で分類された情報がどのプロセスのアクティビティ/タスクの責務かとなぜ混入したかを認識する必要がある

Chapter. n　他のプロセスモデルへの連携

■自組織のプロセスとリレーション（連携）することで多くのプロセスでも利用可能。（アジャイルは未検証）

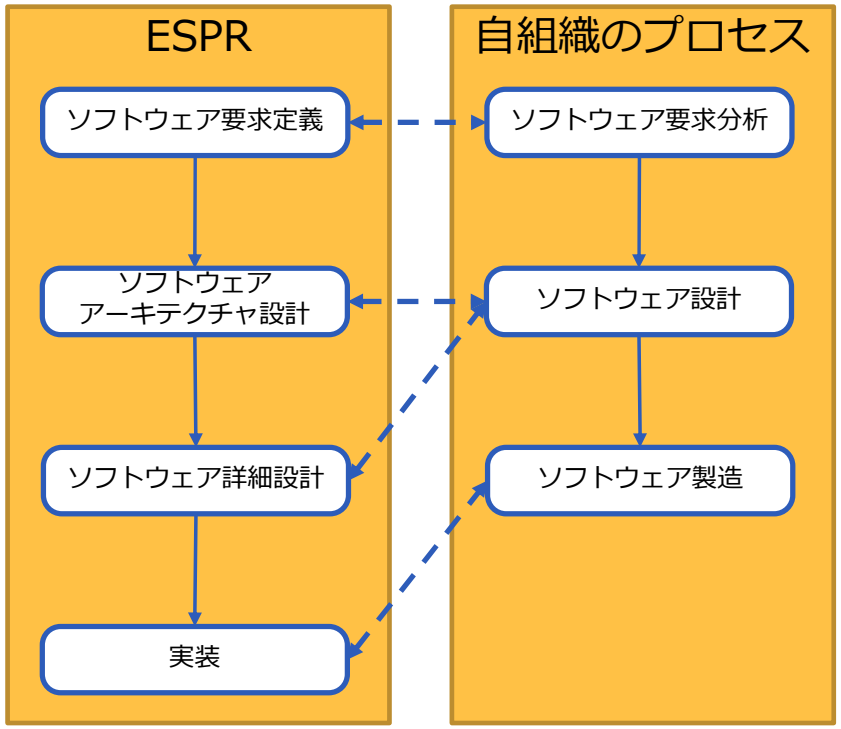


図.自組織のプロセスと対比する

■ PRMや他組織のプロセスを比較することで、自組織のプロセスを見直すことも可能

EOF