名古屋大学

JaSST東海2010

# ソフトウェア独立検証

## ～その必要性と先進事例の紹介～

名古屋大学　情報連携統括本部　情報戦略室
教授　山本修一郎



名古屋大学

# 自己紹介

- ソフトウェア開発を実践
  - NTT研究所、NTTデータ技術開発本部
- ソフトウェア工学の研究、教育
  - 要求工学、設計法、プロジェクト管理、EA
- 高信頼性システム開発手法
  - IPA　ソフトウェアエンジニアリングセンター
- 比較コミュニケーション論
  - NTTデータ　システム科学研究所
- 持続的情報連携アーキテクチャの研究開発
  - 名古屋大学　情報連携統括本部　情報戦略室

名古屋大学

# 実現した主なソフトウェア

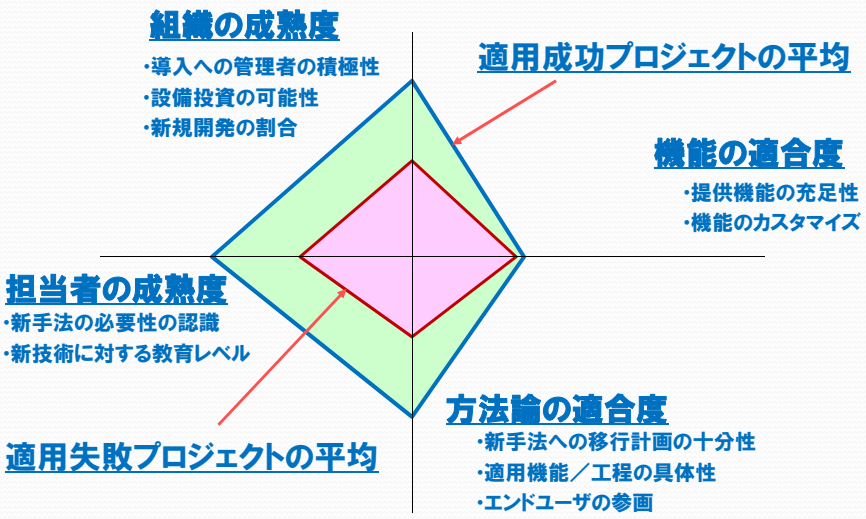
| プロジェクト | 管理面 | 技術面 |
|---|---|---|
| DIPS-APL(1980) | 開発管理、事業化 | 言語処理系，移植技術，テスト |
| SoftDA(1984) | 開発管理，社内導入 | 分析設計技術，関係情報管理，内製 |
| VGUIDE(1995) | 事業化 | 分散システム，トランザクション処理 |
| WebBASE(1996) | 実証実験、事業化 | WebDB連携，検索技術 |
| Xform(1998) | 開発管理<br>実証実験<br>標準化<br>事業会社連携 | XML，セキュア配送，情報変換言語 |
| InfoSTAGE(Σserv)<br>(1998-2004) | | XML，サービス連携，長期トランザクション処理 |
| NICE(2000-) (注) | | ICカード，セキュリティ，<br>ユビキタスコンピューティング |
| Cell computing(2003-08) | 実証実験<br>事業化<br>標準化 | PCグリッド |
| ODVPN(2002-) | | VPN，セキュリティ，デバイス認証 |
| RFID-PF(2004-06) | | RFID認証，プライバシー保護 |

(注)マルチアプリケーションICカード向け情報流通プラットホームの実用化
http://www.dbjet.jp/pub/cgi-bin/detail_pro.php?id=1042




名古屋大学

# プロジェクト特性とツール導入



名古屋大学

# 高信頼性システム開発手法の活動

| 年度 | 会合 | イベント・報告書 | 委員構成 |
|---|---|---|---|
| 2007 | 調査検討会7回 | ・高信頼性システム開発手法フォーラム<br>・高信頼ソフトウェア構築技術に関する動向調査報告書 | 産:7<br>学:5<br>官:6 |
| 2008 | 高信頼性システム技術WG 6回 | ・日中高信頼性システム検討会<br>・第7回WOCS-Workshop Of Critical Software (JAXA/IPA-SEC) | 産:13<br>学:5<br>官:5 |
| 2009 | 高信頼性システム技術WG 6回<br>PT 3回 | ・ソフトウェアジャパン2010<br>ソフトウェア開発のパラダイム・チェンジinマインド ー IPA/SECにおける新しい取組みー<br>・高信頼システム開発技術の動向(予定) | 産:16<br>学:7<br>官:8 |

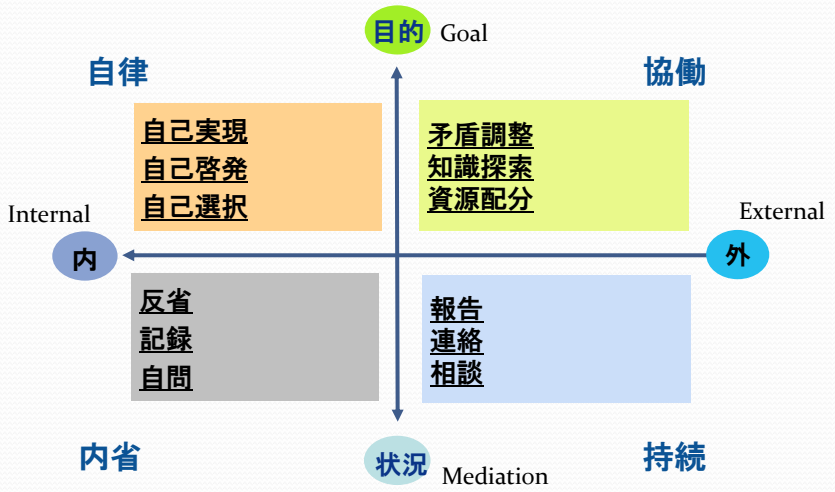
(参考)山本修一郎, 形式手法の誤解を解く, ソフトウェアジャパン2010





名古屋大学

# コミュニケーションのMIGEモデル




(参考)山本修一郎, 比較コミュニケーションモデル論に向けて, 第7回知識流通ネットワーク研究会, http://www4.atpages.jp/sigksn/conf07/index.html

名古屋大学

# 3階層情報連携アーキテクチャ

連携情報活用コミュニティ

情報連携基盤層

連携対象情報層

参考）持続的情報連携サービス分析方法論の研究課題, KBSE研究会, 2010.11.24





名古屋大学

# 主な著作

| 書籍 | 出版社 | 出版年 |
|---|---|---|
| 誰も語らなかったIT　9つの秘密 | ダイヤモンド社 | 2004 |
| 要求定義・要求仕様書の作り方 | ソフト・リサーチ・センター | 2006 |
| IT戦略デザイン | リックテレコム | 2006 |
| ゴール指向による!!システム要求管理 | ソフト・リサーチ・センター | 2007 |
| 次世代プロジェクトリーダーのための～すりあわせの技術 | ダイヤモンド社 | 2009 |
| CMCで変わる組織コミュニケーション | NTT出版 | 2010 |
| 連載要求工学 | ビジネス・コミュニケーション *www.bcm.co.jp/site/youkyu/index.html* | 2004- |

名古屋大学

# 主な議題

- なぜ独立検証なのか
- IEEE Std. 1012-2004 IV&V(独立検証確認)
- 独立検証の先進事例
- テストと開発、運用
- 相互理解モデル




名古屋大学

# なぜ独立検証なのか

- ISO26000
- 3つの理由
- 2つの実態
- 5つの課題

名古屋大学

# ISO26000　企業の社会的責任

| | |
|---|---|
| 7つの原則 | 説明責任 |
| | 透明性 |
| | 倫理的行動 |
| | 利害関係者の尊重 |
| | 法令順守 |
| | 国際行動規範の尊重 |
| | 人権尊重 |




名古屋大学

# 独立検証確認が必要な理由

- システムに欠陥がないことをテストによって完全に保証することはできない
- システムの信頼性が高いほど，重大な欠陥の発見が遅れる
- 最新の開発技術に基づいて，過去のシステム開発の妥当性が判断される

名古屋大学

# 検証確認の2つの実態

- 事故が発生したとき
- 計画した検証確認を実施していた
- 計画した検証確認を実施していなかった
- どのようにして説明するのか？




名古屋大学

# 検証確認活動を保証するには？

- 客観的な証跡がない限り，システムの信頼性をいくら当事者が主張しても信用されない
- 当事者による客観的な検証確認結果を示す証跡
- 当事者以外の第三者による客観的な検証確認(IV&V)の証跡

名古屋大学

# テストプロセス

システムとその関連生産物に対して、**具体的な証跡**を提供

a) 要求を満足すること

b) 適切な問題を解決すること

- 物理規則を正しくモデル化
- ビジネス規則を実現
- システムの前提条件を利用

c) 意図された利用とユーザニーズを満足すること

参考) IEEE Std. 829-2008 Standard for Software and System Test Documentation




名古屋大学

# 第三者検証確認の課題

- 冗長ではないのか
- どこまで検証確認するのか
- 第三者へ情報提示すると不利になるのではないのか
- 第三者検証機関の責任はどうなるのか
- 開発組織の責任はどうなるのか

名古屋大学

# IEEE Std. 1012-2004 IV&V

- テストの限界
- 説明責任





名古屋大学

# ソフトウェアV&Vとは

| 活動 | 説明 | 判断内容 |
|---|---|---|
| 検証 | 開発活動による生産物が開発活動に対する要求に適合していること | ソフトウェアとコンポーネントが，その工程の開始時点で定められた条件を満たすこと |
| 確認 | 開発されたソフトウェアが意図された利用法とユーザニーズに適合していること | ソフトウェアとコンポーネントが開発プロセスの完了時までにユーザのニーズとしての要求を満たすこと |

参考）IEEE Std. 1012-2004 Software Verification and Validation

名古屋大学

# 検証確認の証跡

| 活動 | 証跡の内容 | 時期 |
|---|---|---|
| 検証 | ①各活動に対する要求に適合すること<br>②標準，実践規約，慣例を満たすこと<br>③各活動を完了すること<br>④後続工程の開始基準を満たすこと | ライフサイクルプロセスの活動過程 |
| 確認 | ①ソフトウェア要求を満たすこと<br>②正しい問題を解決すること | ライフサイクル活動の完了時 |

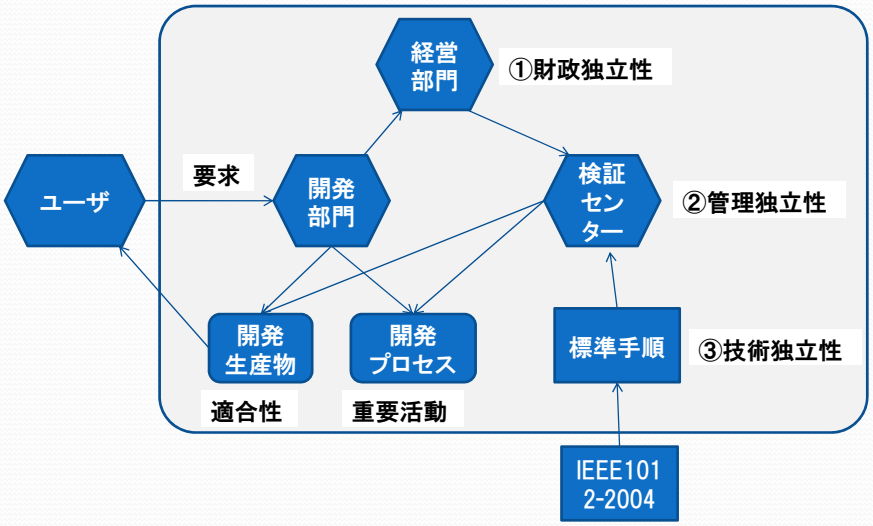
参考）IEEE Std. 1012-2004 Software Verification and Validation





名古屋大学

# IV&Vの基本概念



名古屋大学

# IV&Vの類型

管理独立性

| | | | |
|---|---|---|---|
| 厳格 | | 統合型<br>財務，管理が独立 | 正統型<br>財務，技術，管理が独立 |
| 条件的 | | 内部型<br>開発者が指揮 | 修正型<br>主契約者が管理 |
| 最小限 | 内蔵型<br>開発組織内で実施 | | |
| | 最小限 | 条件的 | 厳格 |

技術独立性

参考）IEEE Std. 1012-2004 Software Verification and Validation





名古屋大学

# ソフトウェア完全性水準

完全性水準

発生確率

| | | | | |
|---|---|---|---|---|
| 4 | | | 確実，相当 | ≧偶発 |
| 3 | | 確実，相当 | 相当，偶発 | 偶発，まれ |
| 2 | 確実，相当 | 相当，偶発 | まれ | |
| 1 | 相当≧ | 偶発，まれ | まれ | |
| | 無視可能 | 軽微 | 重大 | 破局 |

影響度

参考）IEEE Std. 1012-2004 Software Verification and Validation

名古屋大学

# ライフサイクルプロセスとV&V

| ISO/IEC 12207<br>ライフサイクルプロセス | IEEE Std. 1012-2004<br>検証確認(V&V)プロセス |
|---|---|
| 調達 | 調達V&V |
| 提供 | 提供V&V |
| 開発 | 開発V&V |
| 運用 | 運用V&V |
| 保守 | 保守V&V |
| 組織的 | V&V管理 |
| その他の支援 | |

参考) IEEE Std. 1012-2004 Software Verification and Validation





名古屋大学

# IV&Vライフサイクル

IV&V活動のプロセスには，企画，実施，報告という3段階がある

| 段階 | 説明 |
|---|---|
| 企画 | システム要求レビュ(SRR)に先行して計画立案に着手する. SRRと基本設計レビュ (PDR)の間にIV&V計画 (IVVP)を策定する |
| 実施 | IVVPをプロジェクトに提示し，IVVPに基づいて作業計画を立案する.<br>サービス開始直後に実施段階を終了する |
| 報告 | 最終技術報告として実施結果を報告し，解決した課題を文書化する |

名古屋大学

# IEEE Std. 1012-2004の要求V&V

①追跡性分析
②要求評価
③インタフェース分析
④重要性分析
⑤システムテスト計画作成
⑥受入れテスト計画作成
⑦構成管理評価
⑧危険性分析
⑨セキュリティ分析
⑩リスク分析

参考）IEEE Std. 1012-2004 Software Verification and Validation




名古屋大学

# IEEE Std. 1012-2004のテストV&V

①追跡性分析
②受入れテスト手順作成
③結合テスト実行
④システムテスト実行
⑤受入れテスト実行
⑥危険性分析
⑦セキュリティ分析
⑧リスク分析

参考）IEEE Std. 1012-2004 Software Verification and Validation

名古屋大学

# テストにおける追跡性分析

V&Vテスト計画、テスト設計、テスト項目、テスト手続き間の関係を分析

| 性質 | 検証内容 | 対象 |
|---|---|---|
| 正当性 | 追跡関係が正しいこと | V&Vテスト計画<br>テスト設計<br>テスト項目<br>テスト手続き |
| 完全性 | 追跡できること | テスト手続きからテスト計画への追跡性 |




名古屋大学

# テストV&Vの危険性、セキュリティ、リスク分析

| 分析対象 | 内容 |
|---|---|
| 危険性 | テストの仕組みが新たな危険性を持ち込まないことを検証 |
| セキュリティ | 実装システムがセキュリティリスクを増加させないことを検証 |
| リスク | リスク分析を改訂する<br>リスク解消、低減、緩和対策を提供 |

名古屋大学

# 受入れテスト手続きの作成

| 活動 | 4-3 | 2 | 1 |
|---|---|---|---|
| 受入れテスト手続きの作成 | ○ | | |
| 追跡性管理 | ○ | | |
| 開発者の受入れテスト手続きがテスト文書の目的、形式、内容に適合していること | ○ | ○ | |
| 開発者の受入れテスト手続きが要求V&V活動を満たすこと | ○ | ○ | |




名古屋大学

# V&V受入れテスト計画への要求

| 重大性 | 作成対象計画 |
|---|---|
| 4-3 | 1. 受入れV&V テストを実施<br>2. テスト結果を分析してソフトウェアがシステム要求を満たすことを検証<br>3. テスト結果がテスト計画文書のテスト追跡性によって定義されたテスト基準を満たすことを検証<br>4. 受入れテスト計画で要求された結果を文書化<br>5. 受入れV&Vテスト結果に基づいてソフトウェアがV&Vテスト受け入れ基準を満たすことを確認<br>6. 期待結果と実施結果の食い違いを文書化 |
| 2 | 調達者による受入れテスト結果に基づいて、ソフトウェアが受入れテスト基準を満たすことを検証 |
| 1 | --- |

参考）IEEE Std. 1012-2004 Software Verification and Validation

名古屋大学

# V&Vのテスト活動

| V&V活動 | 受け入れ V&Vテスト | システム V&V テスト | 結合 V&Vテスト | コンポーネント V&V テスト |
|---|---|---|---|---|
| 要求 | 計画作成 | 計画作成 | | |
| 設計 | 設計作成 | 設計作成 | 計画作成<br>設計作成 | 計画作成<br>設計作成 |
| 実装 | テスト項目作成 | テスト項目作成<br>テスト手順作成 | テスト項目作成<br>テスト手順作成 | テスト項目作成<br>テスト手順作成<br>テスト実行 |
| テスト | テスト手順作成<br>テスト実行 | テスト実行 | テスト実行 | |

参考）IEEE Std. 1012-2004 Software Verification and Validation





名古屋大学

# 要求V&V受入れテスト計画への要求

| 重大性 | 作成対象計画 | 計画の解決対象 | 計画の検証基準 | 計画の確認基準 |
|---|---|---|---|---|
| 4<br>3 | 受入れテスト計画<br>受入れテスト要求の追跡性確認計画<br>テスト文書化計画 | 運用環境の受入れ要求への適合性<br>文書の適切性 | テスト文書基準への適合性<br>受入れテスト要求の網羅性 | 期待結果への適合性<br>運用と保守の実現性 |
| 2 | -- | -- | 調達者による受入れテストが、テスト文書基準への適合性 | 調達者による受入れテストが、受入れテスト要求の網羅性<br>期待結果への適合性<br>運用と保守の実現性 |
| 1 | -- | -- | -- | -- |

参考）IEEE Std. 1012-2004 Software Verification and Validation

名古屋大学

# 要求V&Vシステムテスト計画への要求

| 重大性 | 作成対象計画 | 計画の解決対象 | 計画の検証基準 | 計画の確認基準 |
|---|---|---|---|---|
| 3-4 | システムテスト計画<br>システムテスト要求の追跡性確認計画<br>システムテスト文書化計画 | システム環境へのシステム要求の適合性<br>文書の適切性<br>境界とストレス条件での性能 | テスト文書基準への適合性<br>システム要求の網羅性 | テスト手法の適切性<br>期待結果への適合性<br>システム品質テストの実現性<br>運用と保守の実現性 |
| 1-2 | -- | -- | テスト文書基準への適合性<br>システム要求の網羅性 | テスト手法の適切性<br>期待結果への適合性<br>システム品質テストの実現性<br>運用と保守のケイパビリティ |

参考）IEEE Std. 1012-2004 Software Verification and Validation





名古屋大学

# 設計V&Vコンポーネントテスト計画への要求

| 重大性 | 作成対象計画 | 計画の解決対象 | 計画の検証基準 | 計画の確認基準 |
|---|---|---|---|---|
| 4<br>3 | テスト計画<br>設計要求の追跡性確認計画<br>テスト文書化計画 | 設計要求への適合性<br>時期、規模、正確性の評価<br>性能(協会、インタフェース、ストレス、エラー条件)<br>要求網羅性の測定 | テスト文書基準への適合性 | 要求と設計への追跡性<br>要求と設計の外部一貫性<br>単位要求間の内部一貫性<br>単位ごとの要求網羅性<br>結合とテストの実現性<br>運用と保守の実現性 |
| 2 | -- | -- | 開発者によるテストが、テスト文書基準への適合性 | 開発者によるテスト計画が、<br>要求と設計への追跡性<br>要求と設計の外部一貫性<br>単位要求間の内部一貫性<br>単位ごとの要求網羅性<br>結合とテストの実現性<br>運用と保守の実現性 |
| 1 | -- | -- | -- | -- |

参考）IEEE Std. 1012-2004 Software Verification and Validation

名古屋大学

# 設計V&V結合テスト計画への要求

| 重大性 | 作成対象計画 | 計画の解決対象 | 計画の検証基準 | 計画の確認基準 |
|---|---|---|---|---|
| 4<br>3 | テスト計画<br>設計要求の追跡性確認計画<br>テスト文書化計画 | 設計要求への適合性<br>時期、規模、正確性の評価<br>性能(協会、インタフェース、ストレス、エラー条件)<br>要求網羅性の測定 | テスト文書基準への適合性 | 要求と設計への追跡性<br>要求と設計の外部一貫性<br>単位要求間の内部一貫性<br>単位ごとの要求網羅性<br>結合とテストの実現性<br>運用と保守の実現性 |
| 2 | -- | -- | 開発者によるテストが、テスト文書基準への適合性 | 開発者によるテスト計画が、<br>要求と設計への追跡性<br>要求と設計の外部一貫性<br>単位要求間の内部一貫性<br>単位ごとの要求網羅性<br>結合とテストの実現性<br>運用と保守の実現性 |
| 1 | -- | -- | -- | -- |

参考) IEEE Std. 1012-2004 Software Verification and Validation





名古屋大学

# IEEE Std. 1012-2004の運用V&V

| 活動 | 内容 |
|---|---|
| 新規制約の評価 | システムやソフトウェア要求についての運用要求、プラットフォーム特性、運用環境などの新規制約を評価してSVVPの適用性を検証 |
| 運用手順評価 | 運用手続きがユーザ文書と矛盾せずシステム要求に適合することを検証 |
| 危険性分析 | 運用手続きと運用環境が新たな危険性をもたらさないことを検証<br>危険性分析を改訂 |
| セキュリティ分析 | 運用環境の変更に起因する新たなセキュリティリスクがないことを検証<br>時間経過による外部インタフェース、脅威、技術の変化に従う新たな未処置リスクを判定するためのセキュリティ分析の改訂 |
| リスク分析 | リスク分析を改訂する<br>リスク解消、低減、緩和対策を提供 |

名古屋大学

# IEEE Std. 1012-2004の保守V&V

①ソフトウェア検証確認計画(SVVP)改訂
②不具合評価
③重要性分析
④移行要求評価
⑤退役要求
⑥危険性分析
⑦セキュリティ分析
⑧リスク分析
⑨反復




名古屋大学

# 期待効果

①ソフトウェア異常の検出と修正の早期化
②開発プロセスと生産物のリスクに対する客観的な管理の推進
③生産性，工期，予算に対してライフサイクルプロセスの適合性
④ソフトウェアとシステム性能の影響評価の早期化
⑤ソフトウェアとシステムの適合性に対して客観的な証跡に基づく，公式検定プロセスの提供
⑥開発保守プロセス改善
⑦システム開発を統合的に分析できるモデル提供によるプロセス改善

名古屋大学

# 独立検証の先進事例

- 事例1　東証 arrowhead
- 事例2　Felica
- 事例3　JAXA





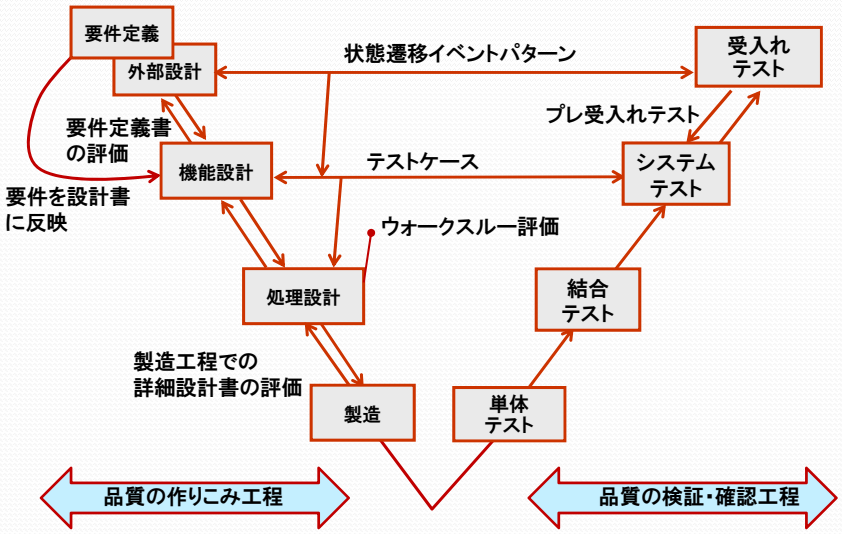
名古屋大学

# 我が国におけるIV&Vの先進事例

| 事例 | IV&V | フィードバック形態 | 形式手法 |
|---|---|---|---|
| JAXA | ■開発メーカ内部の独立組織<br>■JAXA内部の別組織 | ■検証部門から開発部門への問題内容をタイムリーにフィードバック | ■モデル検査 |
| フェリカネットワークス | ■第三者機関が検証 | ■各工程内で段階的作業による工程内フィードバック<br>■工程間フィードバック<br>■第三者による評価認証結果の開発部隊へのフィードバック | ■公的なセキュリティ評価・認証の枠組みによるVDM記述を第三者機関が検証<br>■検証結果を検証者とは別の第三者が認証 |
| 東証 | ■第三者機関が外部設計を検証 | ■工程ごとに障害発生件数を設定しておき、許容下限値に接近すると品質強化対策を実施する「リアルタイム品質管理」<br>■後続工程で先行工程の障害を積極的に検出して先行工程にその内容をフィードバック | ーーー |

名古屋大学

# フィードバック型V字モデル







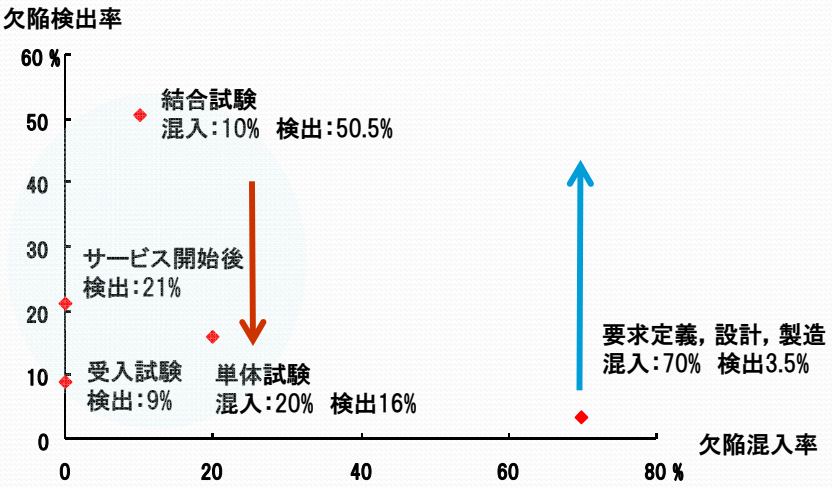

名古屋大学

# 発注品質保証プロセス

| 施策 | 説明 |
|---|---|
| 要件トレース | オークションパターンを作成し要件定義書と外部設計書の品質を確保し，要件トレースできることを確認する |
| 工程完了条件 | 次工程の準備ができていることを確認する．もし次工程に入るだけの準備ができていなければ，現工程を継続する |
| 工程内品質管理 | 工程ごとに品質目標と品質アラーム線を予め設定しておき，品質アラーム線を超えると品質対策を実施する |
| 工程完了判定 | 予め工程完了基準を定義しておき，発注者が工程完了判定を品質評価会議と稼働判定会議によって実施する |
| 詳細設計書のレビュー | ベンダーが作成した詳細設計書を発注者がレビューすることにより要件トレースを確認する |
| 品質実地検査 | サブシステムごとに品質を実地検査する |
| 納入品質の確保 | プレ受入れテストを実施し，検収前に入念に開発中のサブシステムの品質を確保する |

名古屋大学

# 上流工程における欠陥検出率



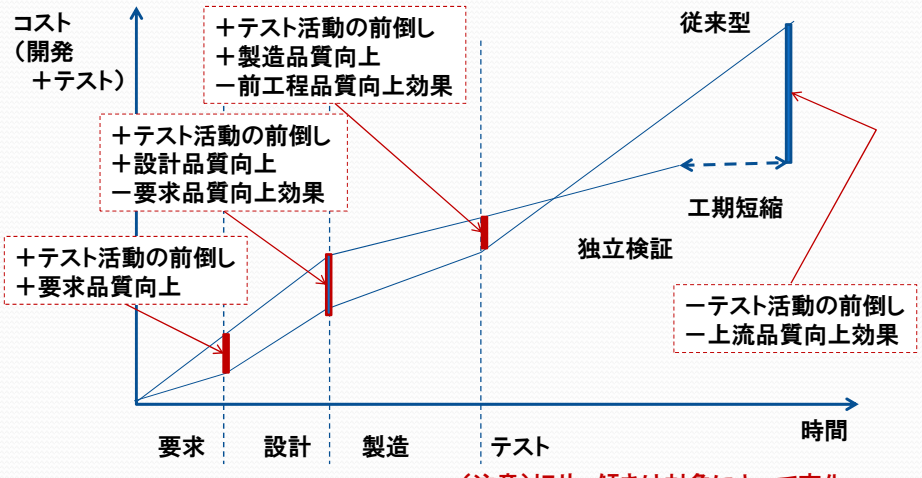

(参考)NIST, Planning report 02-3, The Economic Impact of Inadequate Infrastructure for Software Testing, May 2002





名古屋大学

# 独立検証のコスト推定(仮説)




(注意)切片, 傾きは対象によって変化

名古屋大学

# テストと開発、運用

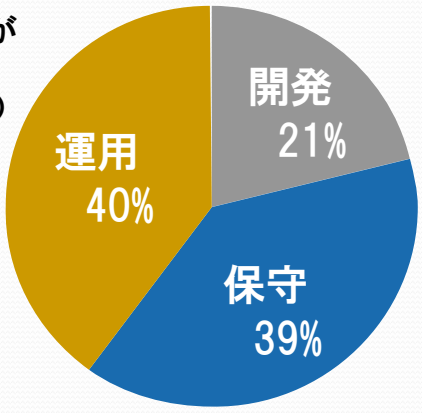
- 事例1　東証 arrowhead
- 事例2　Felica
- 事例3　JAXA




名古屋大学

# 重要インフラ障害の原因分類

社会的に問題とされマスコミが
取り上げた事例85件の分類
（2006年12月～2007年10月）




参考）経済産業省，重要インフラ情報システム信頼性研究会　報告書，2009, IPA

名古屋大学

# 運用前の要求品質の確保

【問題1】運用手順を記述する方法が明確でない

【問題2】運用手順と開発生産物とを対応付けて記述できない

【問題3】運用手順の変更管理が困難である

【原因】

運用活動の主体と対象には状況に応じた相互作用関係があるにもかかわらず，明確に定義されていない

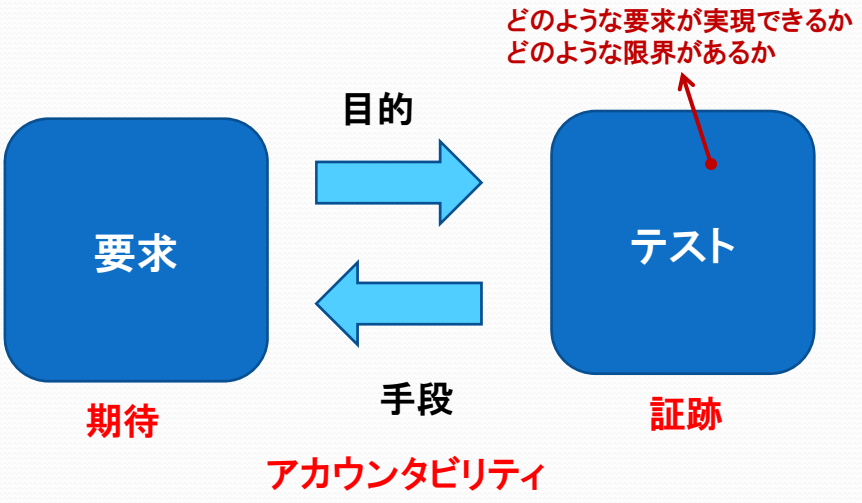
参考）山本修一郎、システム運用知識抽出法の提案、知識流通ネットワーク研究会、2010





名古屋大学

# 要求とテストの関係



名古屋大学

# 要求とテストの連携策の例

| 連携策 | 利点 | 課題 |
|---|---|---|
| 要求レビュにテスト専門家が参画 | 最終テスト段階での要求変更の発生を抑止 | 顧客の参加が重要 |
| 事業目的に対して要求仕様書の完全性と正当性をテスト | 欠陥の早期発見 | テスト専門家による事業目的の理解 |
| 要求を具体例でテスト | 曖昧な要求仕様の改善 | What if質問のスキル<br>想定ユーザの妥当性 |
| 非機能要求を測定する適切な基準を考えることでテストする | 曖昧な非機能要求の明確化 | 曖昧な非機能要求のテストは困難<br>顧客の参加が重要 |
| 副作用の影響がないことを回帰テストで確認 | 変更による影響に対してシステムが正常であることを確認 | 回帰テストのコスト |
| 入出力，事前条件，事後条件の明確化 | システムが何をしているかと何をすべきかの差を解明 | テスト専門家は答を知っているのが前提<br>要求がないと分からない |
| 探索テストによりシステムの挙動を調査 | テスト計画，テスト仕様などからなるテストウェアで要求仕様を代替可能 | 不十分な要求の下ではテスト専門家の知識に依存 |
| 要求とテストの追跡性 | テスト網羅性の向上<br>変更管理、欠陥摘出の効率化 | 追跡性の維持が困難<br>高い要求品質が前提条件 |
| 要求担当とテスト専門家のコミュニケーション | テスト専門家による推測の削減<br>低い要求品質を補完<br>テスト結果の信頼性の向上 | 要求責任者の負荷が増加<br>業務知識専門家の確保が困難<br>情報の信頼性が絶対条件 |
| テスト専門家による要求発見 | テスト可能性の向上<br>テスト工数削減と工期短縮 | 指摘内容の質の向上<br>指摘内容の優先順位 |




名古屋大学

# 要求仕様の完全性基準

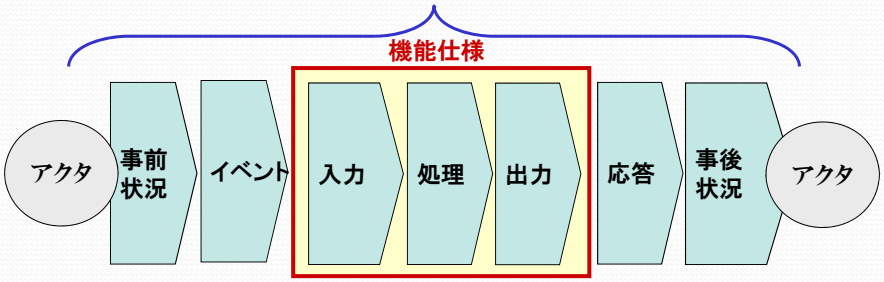
| 項目 | 内容 |
|---|---|
| 人間・コンピュータ・インタフェース | 警報：イベント，型，個数，順序構造，通知，確認，削除<br>トランザクション：構成要素，優先制御 |
| 状態 | 正常／異常状態での期待入力<br>想定外入力の考慮 |
| 入出力 | デバイス入出力：データの種別<br>操作の対称性：例えばオープンしたらクローズがある |
| イベント発生契機 | ロバスト性<br>非決定性<br>値・タイミング：前提条件，間隔，容量・負荷 |
| 出力イベント | 環境の許容量，データの寿命，遅れ |
| 入出力イベントの関係 | 応答性 |
| 状態間の関係 | 到達性，回帰性，可逆性，優先性，ロバスト性 |

参考）Leveson, N.G., SAFEWARE – System Safety and Computers, Addison Wesley, 1995

名古屋大学

# 要求記述の構成要素

曖昧性：範囲，内容，関係の多義性・不明性




アクタが どういうときに

何を 契機として

どんな入力 に対して

システムが 何を出力す ると

どんなこと が発生して

アクタに 何が起き るのか

観測対象　観測事象　観測応答

参考)要求の曖昧さ，http://www.bcm.co.jp/site/youkyu/youkyu38.html





名古屋大学

# Teradine社における要求とテストとの追跡性の実証分析

- 要求工学国際会議2010優秀論文
- 追跡性についての技術者評価の差異分析
  - 要求とテスト項目の関係は比較的明確
  - システム要求とサブシステム要求の関係は曖昧
  - システム要求は多様な情報源との関係が曖昧
- 検証テスト文書：要求IDを明示
- 要求追跡性のデグレード原因
  - 追跡関係の曖昧性
    - 強い関係：判断者が異動すると意味が変化
    - 弱い関係：多すぎると意味がない
  - 要求変化
  - 追跡性の保守が困難

追跡性の評価

| 技術者 | 有効性 | 適用性 |
|---|---|---|
| 設計検証者 | 80% | 100% |
| システム開発者 | 70% | 50% |
| サブシステム開発者 | 25% | 60% |

要求追跡表

| 定義要求 | 準拠要求 |
|---|---|
| RQ.Inst.Waveform | RQ.Inst.PgmSamples |
| なし | RQ.Inst.PgmRate |

テスト項目表

| テストID | TC.Inst.RateErr |
|---|---|
| 記述 | サンプルエラーメッセージをテスト |
| 要求ID | RQ.Inst.PgmSamples<br>RQ.Inst.PgmRate |
| 手順 | 1.既定範囲外のサンプルレートを設定<br>2.エラーを観察 |

名古屋大学

## 時間を考慮した利用モデルによるテスト手法の適用事例

- ISO 26262(2011年制定予定)で厳密な要求仕様が要請
- Audi AG開発での適用事例
  - ECU :350状態, 42遷移, 37テスト項目
  - Energy 管理：250状態, 310遷移
- TUM: Timed Usage Model
  - 状態，遷移，刺激(イベント)，遷移確率，滞留時間(状態，遷移)
- TUMプロセス
  - 要求仕様からTUMモデルを作成
  - TUMモデルからテスト項目を自動生成
  - テスト項目をEXAMテストベンチで実行してテスト結果を報告
  - EXAM: Extended Automation Method




名古屋大学

## シーケンスベースソフトウェア仕様記述法

- システム境界を識別
- すべての刺激と応答のシーケンスをシステム境界について列挙
  - 刺激に対してシステムが応答
  - 刺激には入力が対応
  - 応答には出力が対応
  - 刺激と応答の組(u->r)の簡約規則：将来の応答結果の等価性
  - 不可能イベント
    ①環境が生成できない
    ②システムで観測できない

(参考)Stacy J. Prowell and Jesse H. Poore, Foundations of Sequence-Based Software Specification, IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 29, NO. 5, pp.417-429, MAY 2003

名古屋大学

# 相互理解モデルに向けて

- テストと，BABOK：ビジネスアナリシス知識体系
- 運用要求知識
- 要求変更，コンポネント調達，要求監視




名古屋大学

# BABOKのステークホルダ

| ステークホルダ | 関係 | 対象 |
|---|---|---|
| ■ビジネス分析者 | 分析 | ビジネス |
| □顧客 | 利用 | |
| □スポンサー | ニーズ | ビジネス<br>ソリューション |
| ■分野専門家 | トピック | |
| □エンドユーザ | 使用、参加 | ソリューション |
| ■実装専門家<br>①開発専門家<br>②組織管理専門家<br>③システムアーキテクト<br>④訓練専門家<br>⑤ユーザビリティ専門家 | 構築<br>①開発<br>②組織による受容<br>③コンポネント分解<br>④エンドユーザによる受容<br>⑤操作性 | |
| ■プロジェクトマネジャ | 管理 | |
| ■テスト専門家 | 検証 | |
| □規制者 | 標準の定義，施行 | |
| □サプライヤー | 提供 | |

参考）BABOK 2.0, IIBA 日本支部, ビジネスアナリシス知識体系ガイド, 2009

名古屋大学

# BABOK 2.0の知識領域とタスク

| BABOK 2.0 知識領域 | 知識領域の用途 | 作業 |
|---|---|---|
| ビジネス分析<br>計画監視 | ビジネス分析活動を定義 | ①ビジネス分析接近計画②ステークホルダ分析主導<br>③ビジネス分析活動計画④ビジネス分析コミュニケーション計画⑤要求管理プロセス計画⑥ビジネス分析有効性管理報告 |
| 要求管理<br>コミュニケーション | 要求矛盾，課題，変更管理とスコープでステークホルダと合意形成 | ①ソリューションと要求のスコープ管理②要求追跡性管理③再利用のための要求保守<br>④要求パッケージ準備⑤要求コミュニケーション |
| エンタープライズ分析 | ビジネスニーズを選択，洗練，明確化しソリューションスコープを定義 | ①ビジネスニーズ定義②ビジネスニーズ充足上のギャップ決定③ソリューション接近法決定<br>④ソリューションスコープ定義⑤ビジネスケース開発 |
| 抽出 | ステークホルダのニーズを抽出し，その正当性，完全性を確認 | ①抽出準備②抽出活動指揮③抽出結果の文書化<br>④抽出結果確認 |
| 要求分析 | ビジネスとステークホルダのニーズを満足するソリューションを定義 | ①要求優先順位付け②要求体系化③要求仕様化<br>④前提と制約の決定⑤要求検証⑥要求確認 |
| ソリューション<br>評価確認 | 提案ソリューションとビジネスニーズとの適合性を評価，問題点抽出改善案を提示 | ①提案ソリューション評価②要求割付③組織準備判断④移行要求判断⑤ソリューション確認<br>⑥ソリューション有効性評価 |
| 基礎能力 | ビジネス分析を適切に実行する能力 | ①分析思考と問題解決②倫理等の行動特性<br>③ビジネス領域知識④コミュニケーション能力<br>⑤グループ活動能力⑥オフィスソフトウェア知識 |

参考)BABOK 2.0, IIBA 日本支部, ビジネスアナリシス知識体系ガイド, 2009





名古屋大学

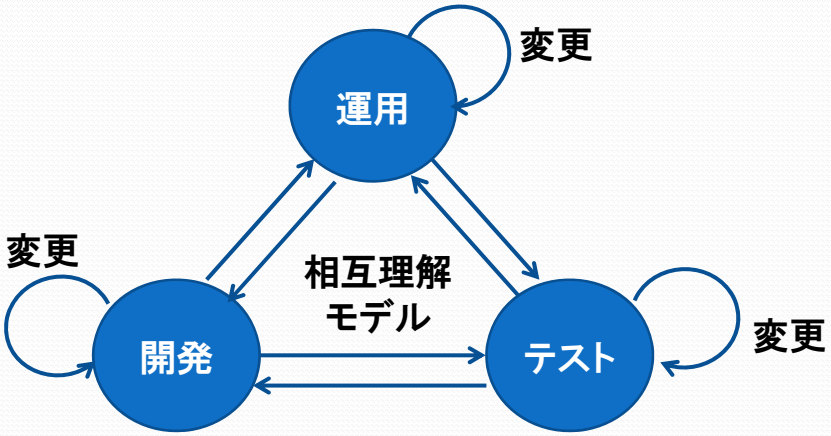
# BABOK 2.0の知識領域と技法

| BABOK 2.0 知識領域 | 技法 |
|---|---|
| ビジネス分析計画監視 | ①決定分析②プロセスモデリング③構造化ウォークスルー④受け入れ基準と評価基準の定義⑤ブレーンストーミング⑥インタビュー⑦組織モデリング⑧要求ワークショップ⑨リスク分析⑩シナリオとユースケース⑪ユーザーストーリ⑫スコープモデリング⑬調査とアンケート⑭見積もり⑮機能分解⑯問題のトラッキング⑰教訓プロセス⑱メトリクスとKPI⑲根本原因分析<br>■RACIマトリクス■ステークホルダマップ■差異分析 |
| 抽出 | ①ブレーンストーミング②文書分析③焦点グループ④インタフェース識別⑤インタビュ⑥観察⑦プロトタイピング⑧要求ワークショップ⑨調査とアンケート⑩問題トラッキング |
| 要求管理コミュニケーション | ①問題トラッキング②要求ワークショップ③構造化ウォークスルー<br>■ベースライン化■サインオフ■カバレッジマトリック■要求の文書化■ベンダー選定のための要求 |
| エンタープライズ分析 | ①ベンチマーク②ブレーンストーミング③ビジネスルール分析④焦点グループ⑤機能分解⑥根本原因分析⑦文書分析⑧SWOT分析⑨ベンチマーク⑩決定分析⑪見積もり⑫インタフェース分析⑬スコープモデリング⑮ユーザストーリ⑯決定分析⑰メトリクスとKPI⑱リスク分析<br>■フィージビリティ分析■ステートメント |
| 要求分析 | ①決定分析②リスク分析③ビジネスルール分析④データフロー図⑤データモデリング⑥機能分解⑦組織モデリング⑧プロセスモデリング⑨シナリオとユースケース⑩スコープモデリング⑪ユーザーストーリ⑫受け入れ基準と評価基準の定義⑬データディクショナリと用語集⑭インタフェース分析⑮メトリクスとKPI⑯非機能要求⑰シーケンス図⑱状態遷移図⑲問題トラッキング⑳リスク分析⑳プロトタイピング⑳構造化ウォークスルー<br>■MoSCoW分析■時間制限／予算制限■投票■チェックリスト |
| ソリューション評価確認 | ①受け入れ基準と評価基準の定義②決定分析③ベンダのアセスメント④ビジネスルール分析⑤機能分解⑥プロセスモデリング⑦シナリオとユースケース⑧データフロー図⑨プロセスモデリング⑩焦点グループ⑪インタビュ⑫調査とアンケート⑬組織モデリング⑭問題トラッキング⑮リスク分析⑯SWOT分析⑰データモデリング⑱根本原因分析<br>■フォースフィールド分析 |

参考)BABOK 2.0, IIBA 日本支部, ビジネスアナリシス知識体系ガイド, 2009

名古屋大学

# TODサイクル







名古屋大学

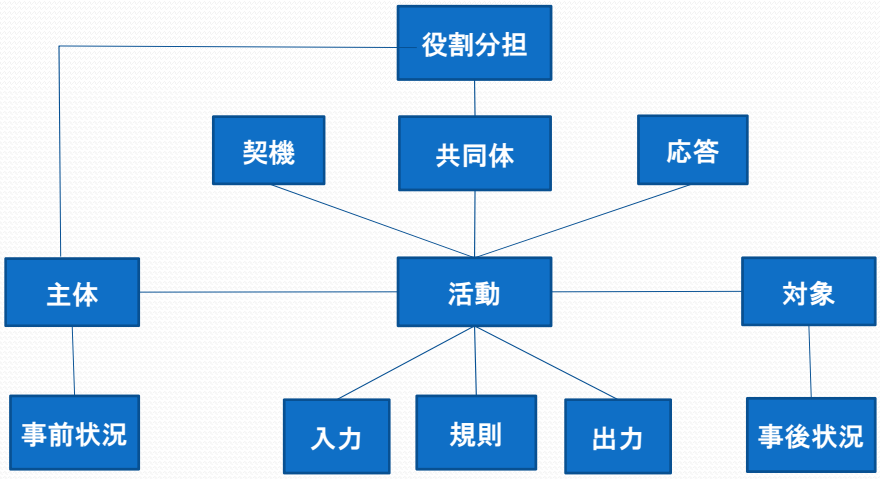
# 比較モデル論

| モデル | アクタ | ゴール | 事前／事後状況 | 契機・応答 | 入出力 | 機能 | 規則 | 役割分担 |
|---|---|---|---|---|---|---|---|---|
| データフロー | ○ | | | | ○ | ○ | | |
| 状態遷移 | | | ○ | ○ | ○ | ○ | | |
| アクタ関係 | ○ | ○ | | | | | | |
| アクタ状況 | ○ | ○ | ○ | ○ | ○ | ○ | | |
| ユースケース | | | | ○ | ○ | ○ | | |
| シナリオ | | | ○ | ○ | ○ | ○ | | |
| CATWOE | ○ | ○ | | | | ○ | ○ | |
| 活動理論 | ○ | ○ | | | | | ○ | ○ |
| CORE | ○ | | | ○ | ○ | ○ | | |
| アクタ相互作用 | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ |

（参考）山本修一郎、システム運用知識抽出法の提案、知識流通ネットワーク研究会、2010
http://www4.atpages.jp/sigksn/conf07/index.html

名古屋大学

# 運用要求の基本構造




参考）山本修一郎, システム運用知識抽出法の提案, 2010





名古屋大学

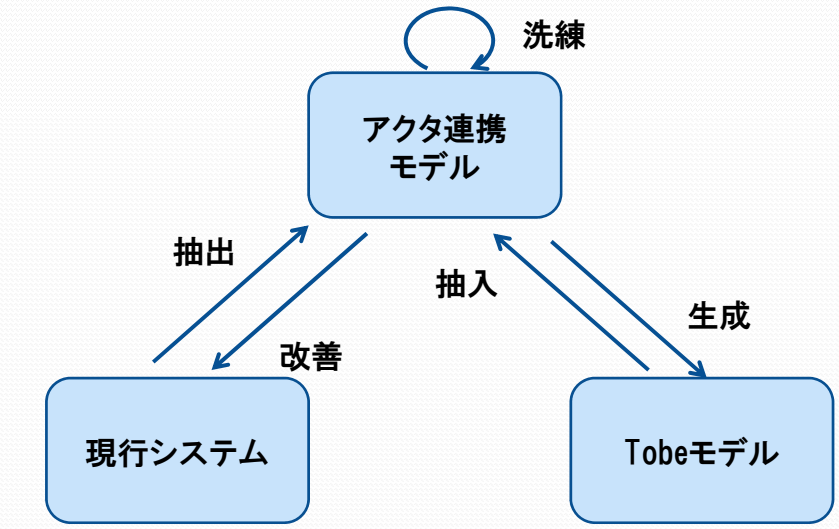
# 運用要求定義票

| 要求ID | | | | |
|---|---|---|---|---|
| 主体 | 契機 | 運用手順 | 応答 | 対象 |
| | | | | |
| 事前状況 | | 入力 | 出力 | 事後状況 |
| | | | | |
| 運用規則 | | 関係者 | 役割分担 | |
| | | | | |

参考）山本修一郎、システム運用知識抽出法の提案、知識流通ネットワーク研究会、2010

名古屋大学

# アクタ連携モデル




参考）山本修一郎、システム運用知識抽出法の提案、知識流通ネットワーク研究会、2010





名古屋大学

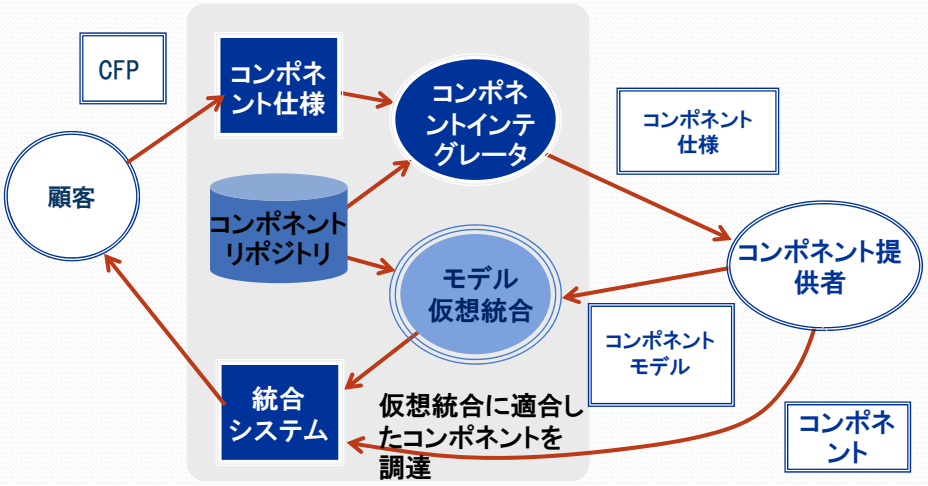
# サービス要求水準

| 水準 | 分類 | サービス環境 | サービス運用者 | サービスコンポネント | サービス確認 |
|---|---|---|---|---|---|
| 0 | 管理 | 管理計画 | | | 点検結果 |
| 1 | 目的 | 前提，制約 | 責任分担 | 要求，制約 | 充足性評価 |
| 2 | 設計原則 | インタフェース | 作業分担 | 機能分担 | 分担性評価 |
| 3 | アーキテクチャ | 環境モデル | 運用モデル | 機能モデル | モデル確認 |
| 4 | 設計仕様 | I/F設計 | HCI設計 | サービス仕様 | サービステスト |
| 5 | 物理仕様 | | 物理設計 | コード | テスト |
| 6 | 運用 | 監査手順 | 運用手順 | 障害報告 | 運用性確認 |

参考）Israel Navarro, Nancy Leveson, Kristina Lunqvist, Semantic decoupling: reducing the impact of requirements change, Requirements Engineering, Springer, 03 June 2010

名古屋大学

# モデルベースコンポネント調達




(参考)Lewis, B., Architectural Computer System Model-Based Engineering with AADL, Sept. 2008, SEI





名古屋大学

# 4階層要求監視フレームワーク

| 階層 | 名称 | 説明 |
|---|---|---|
| 1 | 応用層 | デバグ，検証，確認，業務活動監視，進化 |
| 2 | 表示層 | 表示 |
| 3 | モデル層 | 診断モデル，ユーザモデル，モデルリポジトリ |
| 4 | イベント層 | イベント・リポジトリ<br>イベント・フィルタ<br>イベント・ソース<br>イベント(ソフトウェア，CPUなど) |

ツール

参考) William Robinson, A roadmap for comprehensive requierments monitoring, IEEE Computer, May, 2010, pp.64-72.

名古屋大学

# まとめ

- なぜ独立検証なのか
- IEEE Std. 1012-2004 IV&V(独立検証確認)
- 独立検証の先進事例
- テストと開発、運用
- 相互理解モデル




名古屋大学

# 参考文献


1. IEEE Std. 1012-2004 Software Verification and Validation
2. IEEE Std. 829-2008 Standard for Software and System Test Documentation
3. NIST, Planning report 02-3, The Economic Impact of Inadequate Infrastructure for Software Testing, May 2002
4. Israel Navarro, Nancy Leveson, Kristina Lunqvist, Semantic decoupling: reducing the impact of requirements change, Requirements Engineering, Springer, 03 June 2010
5. Leveson, N.G., SAFEWARE – System Safety and Computers, Addison Wesley, 1995
6. **経済産業省，重要インフラ情報システム信頼性研究会　報告書，**2009, IPA
7. **山本修一郎、独立検証確認と形式手法がもたらすソフトウェア開発プロセスの改革**, IPA SEC Journal, 2010
8. **宇宙航空研究開発機構，~究極の高品質ソフトウェア開発プロセスをめざして~ベストプラクティス調査報告書，経済産業省，プロセス改善研究部会，**2007, http://*sec.ipa.go.jp/reports/20070514/JAXA.pdf*
9. **栗田太郎，モバイル**FeliCa**のソフトウェア開発における品質確保のための構造と実践～抽象度の制御やコミュニケーションの活性化に向けて，情報処理学会デジタルプラクティス，**Vol.1, No.3, 2010, http://www.ipsj.or.jp/15dp/Vol1/No3/dp0103.html
10. **田倉聡史，上流工程での品質確保のための発注者責任，**SEC Journal, Vol.6, No.2, 2010
11. *M.Panis,* Successful Deployment of Requirements Traceability in a Commercial Engineering Organization...Really , RE2010, pp.303-307, 2010
12. *S.Siegl, K-S.Hielscher, R.German ,* Model Based Requirements Analysis and Testing of Automotive Systems with Timed Usage Models, RE2010, pp.345-350, 2010
13. Stacy J. Prowell and Jesse H. Poore, Foundations of Sequence-Based Software Specification, IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 29, NO. 5, pp.417-429, MAY 2003
14. **山本修一郎、システム運用知識抽出法の提案、知識流通ネットワーク研究会、**2010
15. BABOK 2.0, IIBA **日本支部, ビジネスアナリシス知識体系ガイド**, 2009
16. William Robinson, A roadmap for comprehensive requirements monitoring, IEEE Computer, May, 2010, pp.64-72.
17. Lewis, B., Architectural Computer System Model-Based Engineering with AADL, Sept. 2008, SEI