講演時に使用したスライドの
一部を掲載しています

ソフトウェア品質概論
ソフトウェア品質の基本中の基本の概念を学ぶ

東洋大学経営学部　野中　誠

2014年7月31日

人間中心設計推進機構セミナー
「利用品質の理解を深めるためにソフトウェア品質を学ぶ」

野 中　　誠

- **所属**
  - 東洋大学 経営学部 経営学科　教授
- **背景**
  - 工業経営／経営システム工学，ソフトウェア工学，品質マネジメント
- **主な学外活動**
  - 日本科学技術連盟 SQiPソフトウェア品質委員会 運営委員長
  - IPA/SEC 高信頼化定量管理部会主査(『ソフトウェア開発データ白書』)
  - 組込みシステム技術協会実装品質強化WG メンバー
  - 日本SPIコンソーシアム 外部理事
  - 国立情報学研究所 特任研究員（トップエスイー講座，メトリクス講義担当）
- **主な著書**
  - 野中・小池・小室著『データ指向のソフトウェア品質マネジメント』日科技連出版社（2012）
    2013年度日経品質管理文献賞受賞
  - 野中・鷲崎訳『演習で学ぶ ソフトウェアメトリクスの基礎』日経BP社（2009）
  - SQuBOK策定部会編著『ソフトウェア品質知識体系ガイドーSQuBOK　Guideー』オーム社（2007）
- **研究・教育**
  - ソフトウェア品質マネジメント(メトリクスを中心に)
  - 情報システムと経営の関わり

品質にしっかりと取り組めば，組織は賢く，強く，幸せになれる

- **品質にしっかりと取り組む**
  - ソフトウェアを通じて顧客に提供する価値を考える
  - その価値を提供し続けるために，組織的に必要な活動を考える
- **必要な活動をデザインする**
  - 顧客に価値を提供できていることを保証するプロセスを確立する
  - 価値提供のスピードを加速させるようプロセスを改善する
- **組織が賢く，強くなる**
  - 価値提供の結果に学ぶ
  - 失敗に学ぶ
  - 自社独自の経験に学ぶ
  - 学びを積み重ねることで，組織は賢く，強くなる
  - 賢く，強い組織は幸せになる

ソフトウェア品質を取り巻く状況

- **情報システムとソフトウェアの重要性の高まり**
  - 経済・社会のインフラとして
  - 製品の競争力（機能，性能，特徴，魅力）の源泉として
- **情報システムとソフトウェアが占めるGDP貢献度の大きさ**
  - 日本の名目GDPは**約472兆円**（2012年度） 出所：内閣府
  - 情報サービス産業は，売上高**約20兆円**（2012年度） 出所：経済産業省
  - それ以外の産業に属する企業内にもソフトウェア組織がある
  - ⇒ 情報システムとソフトウェアの実質的なGDP貢献度は大きい
- **ソフトウェア開発組織の責務**
  - 顧客満足に結びつく品質の良いソフトウェアを開発する
  - 出荷後バグの少ないソフトウェアを開発する
- **ソフトウェアによってもたらされる価値の重要性**
  - 顧客満足のために，どのような価値を創り出すか
  - ビジネスへの貢献のために，どのような価値を創り出すか

（内閣府）http://www.esri.cao.go.jp/jp/sna/menu.html
（経済産業省）http://www.meti.go.jp/committee/sankoushin/jouhoukeizai/jinzai/001_s01_00.pdf

ソフトウェアと顧客満足：身近な例

コンビニ設置コピー機での証明写真プリント

- 自分で撮ったお気に入りの画像データを使える
- 履歴書用写真が4カット**200円**
- 証明写真ボックス → 履歴書用写真が6カット**700円**
  - 個人的な経験上，気に入った写真が得られた試しがない
- コンビニのコピー機で写真を印刷すると1枚**30円**
- 少しのソフトウェア処理を加えたことで**価値**が生まれる

利用者も満足 / コピー機事業者も満足 / コンビニも満足

ソフトウェアとビジネス価値

事例：コマツのKOMTRAX

- 世界で稼動する**約30万台**の建設機械の情報が本社で即座に分かる
  位置 / 走行距離 / エンジンON･OFF / 稼働状況 / 燃料 / 油温 / etc
- **いま**の詳細なデータに基づく経営判断･戦略構想
- 継続的なデータ収集がもたらす競争優位性

事例：日本調剤 ～ 製薬会社向け情報提供ビジネスの創出

- 全国約470の調剤薬局で扱う処方箋 ～ 年間**1000万枚**以上
- 処方箋データを分析して製薬会社に提供する新ビジネスを創出
  - どの薬がどの地域で多く処方されているのか？
  - それは，患者の年齢層によってばらつきがあるのか？
  - 後発薬(ジェネリック医薬品)と先発薬のどちらを選んだのか？ など

ソフトウェア品質とは ～ 国際規格，著名な定義

- **明示的ニーズと暗黙のニーズを満たす（stated needs / implied needs）**
  - 特定の条件で利用する場合の，明示的ニーズ又は暗黙のニーズを満たすための ソフトウェア製品の能力（JIS X 25000:2010(ISO/IEC 25000:2005)）
  - 本来備わっている特性の集まりが，要求事項(※)を満たす程度 (JIS Q 9000:2006(ISO 9000:2005)) ※要求事項 … 明示 / 暗黙的に了解 / 義務として要求されているニーズまたは期待
  - 「ニーズに関わる対象の特徴の全体像」飯塚 悦功
- **立場によってニーズは異なる / ニーズは時間とともに変化する**
  - 「品質は誰かにとっての価値である」Gerald M. Weinberg ⇒ ユーザやシステム管理者など，立場によって価値は異なる(品質の相対性)
  - 「システムが本稼動するとき，どこまで真のニーズにあっているかということ」J. Martin ⇒ ある時点で満足を得られたとしても，時間の経過と共にユーザニーズは変化し， 満足度は低下する(タイムリーな提供の必要性)
- **信頼性と機能適合性に絞って捉える**
  - 「 ソフトウェアの停止や，不正確な出力結果の原因となる欠陥が残存している度合い」 Capers Jones
- **開発標準を満たす**
  - 「機能および性能に関する明示的な要求事項，明確に文書化された開発標準，および職業的に開発が行われた全てのソフトに期待される暗黙の特性に対する適合」R. S. Pressman

ソフトウェア分野における「要求」と「ニーズ」

要求 (requirements)

1. 問題解決または目標達成のためにステークホルダーが必要とする条件(Condition)または能力(capability)
2. 契約，企画，仕様，規則などをみたすためにシステムやシステムコンポーネントが満たしていなければならない条件または能力
3. 上記1．または2．にあるような条件または能力を文書化したもの

- 条件：現状(As-is)から目標(To-be)に到達するために必要なもの
- 能力：問題解決のための手順や，それを実現するシステム機能等
- IEEE 610では「ユーザー」だが，BABOKでは「ステークホルダー」

ニーズ

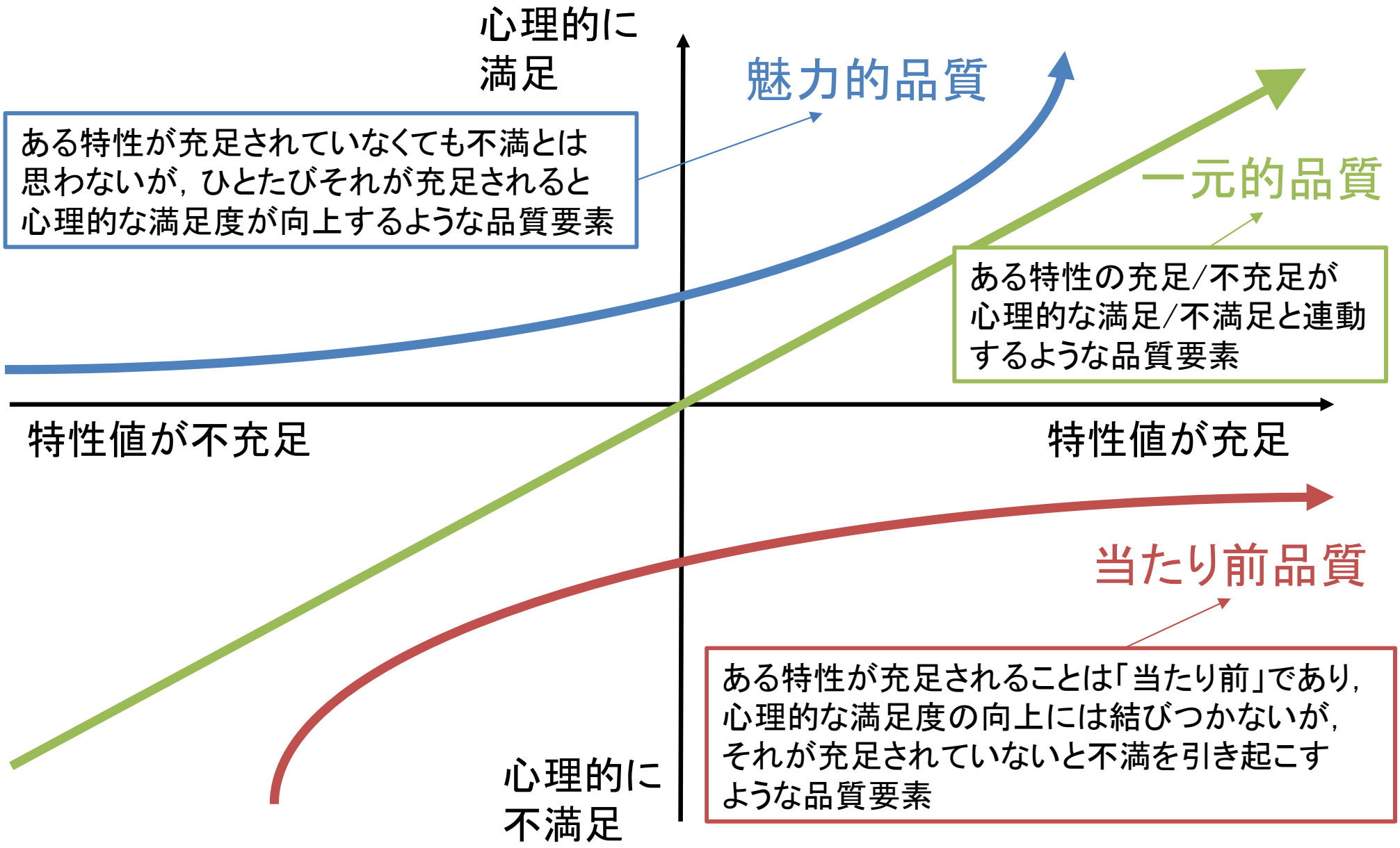
1. 製品やサービスを利用した結果に対する利用者の期待（東 基衛）

IEEE Std 610.12™-1990 (R2002), IEEE Standard Glossary of Software Engineering Terminology
BABOK (Business Analysis Body Of Knowledge)

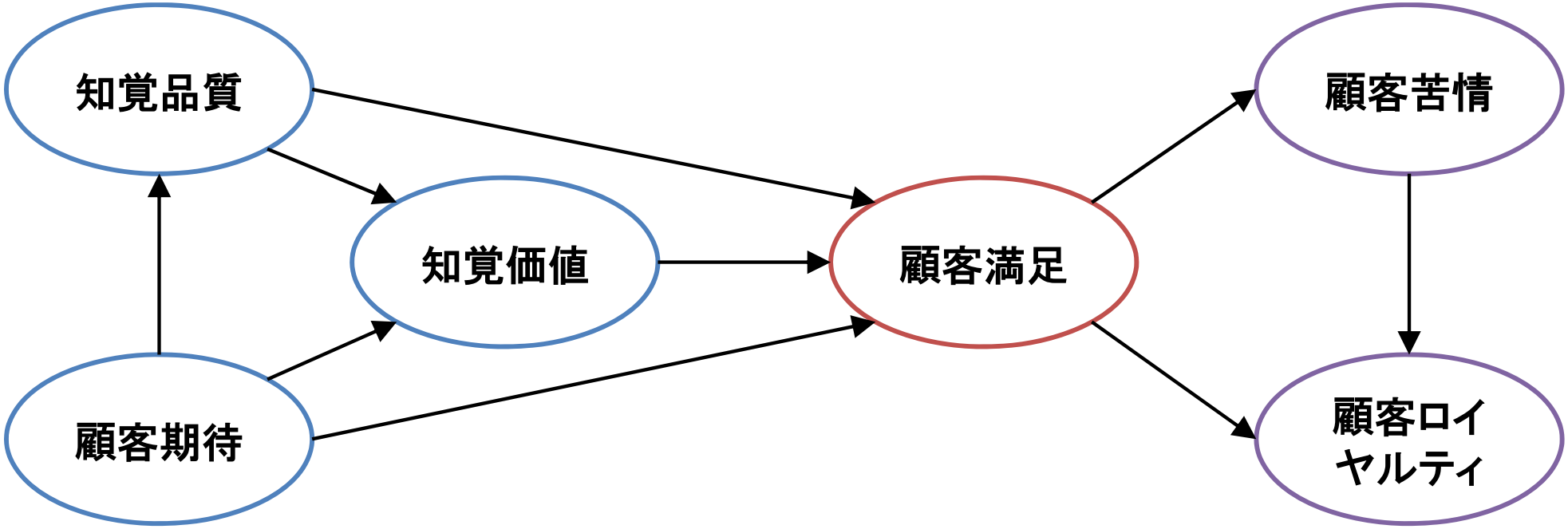
品質とは ～ 日本で培われてきた考え方

- **徹底的な消費者指向で考える**
  - 品質の良し悪しは顧客の満足で決まる(提供する側の評価で決まるものではない)
  - 素人の消費者や顧客の判断が，専門家である提供側の判断よりも上位に位置づけられる
    ⇒ 品質論の原点は「相対的認識」であり，他人に認められてはじめて評価が定まる
- **品質を「総合的」に考える**
  - 品質概念の拡大
    - 1950年代，TQC (Total Quality Control) という概念が米国から日本に持ち込まれた
      ⇒ 元々のTQCの意味：品質部門の仕事だったQCを，全部門の品質スタッフの仕事に
    - 日本における拡大解釈：①全部門 / ②全階層(トップ～現場) / ③品質概念の拡大
  - 総合的な質の概念 (石川馨)
    - 狭義の品質：製品の品質
    - 広義の質：仕事の質，サービスの質，情報の質，工程の質，部門の質，人の質，
      システムの質，会社の質など ⇒ 日本的品質管理へ大きな影響
- **特性値の充足状況と心理的満足を分けて考える (狩野紀昭)**
    - 当たり前品質要素
    - 魅力的品質要素

当たり前品質と魅力的品質



顧客満足に関わる因果関係モデル
ACSI 顧客満足モデル American Customer Satisfaction Index



- **顧客期待** … 理想期待または事前期待。総合的な期待，ニーズ充足の期待，信頼性の期待で構成される。
- **知覚品質** … 利用経験に基づく評価。総合的な評価，ニーズ充足の評価，信頼性の評価で構成される。
- **知覚価値** … 価格に対する品質の評価。
- **顧客満足** … 顧客の期待と，その達成度に対する顧客の知覚の差によって生じる感情。
  総合的な満足度，期待との比較，理想との比較で構成される。

- **顧客苦情** … 公式または非公式に不満を言ったことがあるか否か。
- **顧客ロイヤルティ** … 顧客がその企業から継続的に製品やサービスの提供を受けたいと望む顧客の意思。

参考：日本生産性本部 顧客満足度調査(JCSI) webサイトに類似のモデルが示されています
http://consul.jpc-net.jp/jcsi/

消費者にとっての価値の階層

下位価値が確立されてはじめて上位価値が得られる

ブランドが発信するノスタルジー，ファンタジー，ブランドの歴史への憧れや共感度，自己のライフスタイルへの共感度などによって構成される価値。

ブランドに「品質や機能以外のストーリーを付加する」もの。

五感を通して楽しい消費経験を提供する価値。

製品や広告，販促物に感じる魅力や好感度など。

製品の購買・消費時に利便性を提供する価値。

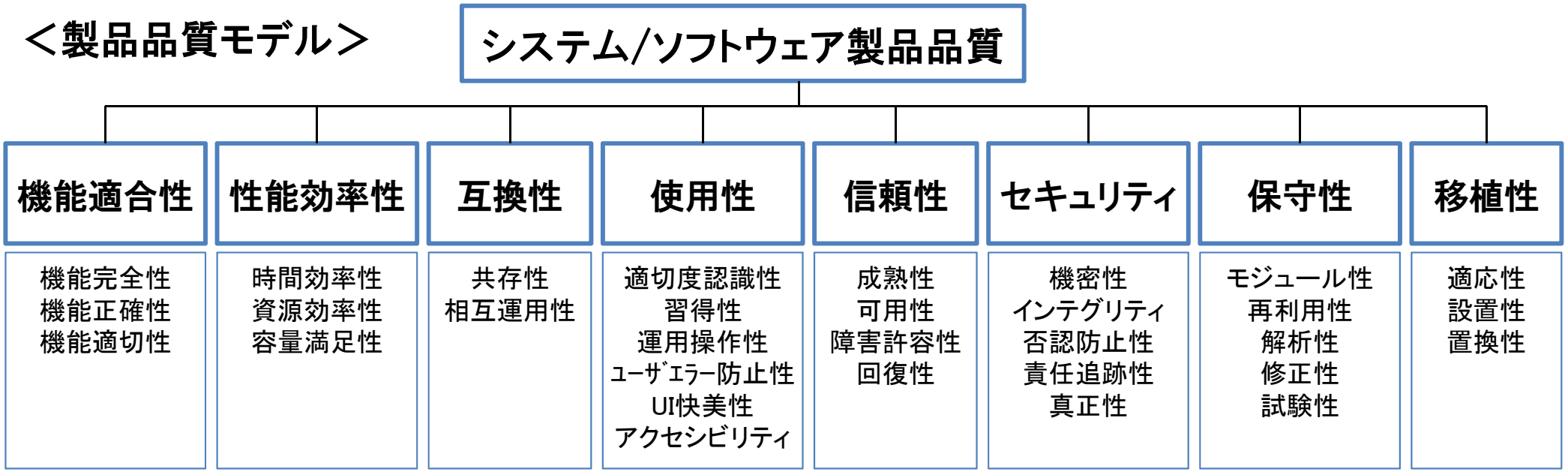
価格や使用しやすいパッケージの工夫など。

製品の物理的機能性。品質そのもの，また製品の優良度など。

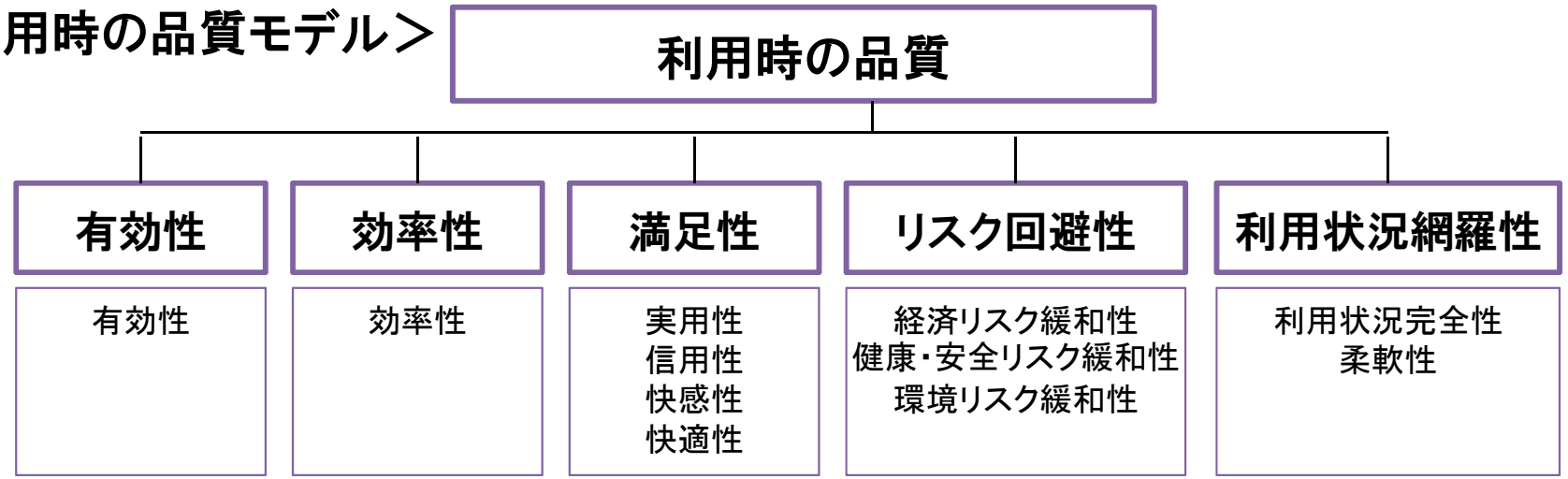
和田充夫（2002）『ブランド価値共創』同文舘出版

ソフトウェアの品質モデル ISO/IEC 25010:2011（JIS X 25010:2013）

＜製品品質モデル＞



＜利用時の品質モデル＞



※ ISO/IEC 25010の品質モデルは，ISO/IEC 9126-1:2001の品質モデルの改訂版である
※ ISO/IEC 250xxの一連の規格は，SQuaRE（Systems and software Quality Requirements and Evaluation）シリーズと呼ばれる

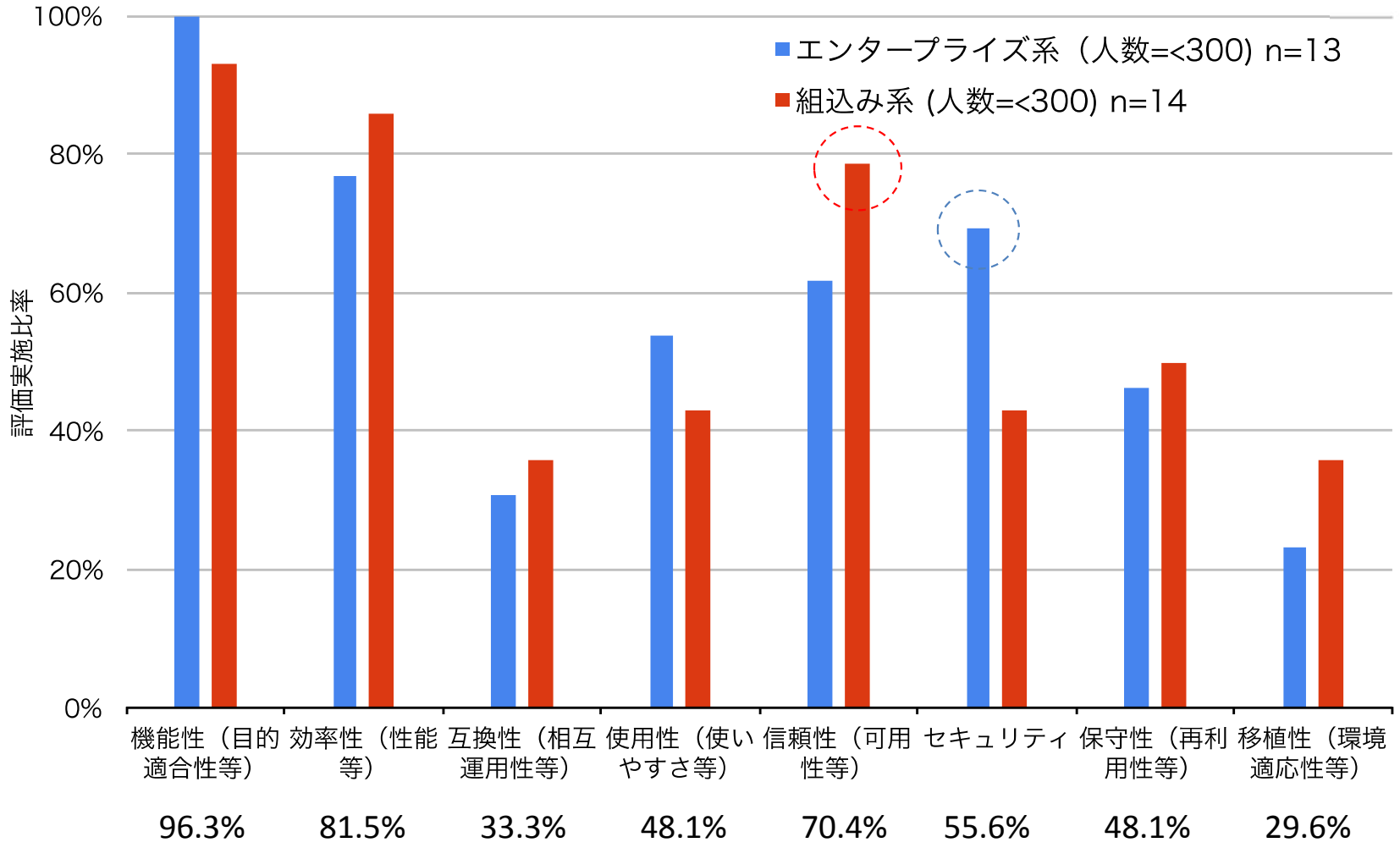
システム/ソフトウェア製品品質の特性 ISO/IEC 25010:2011（JIS X 25010:2013）

品質特性	概要	品質副特性
機能適合性	暗黙的・明示的ニーズを満たす機能が提供されているか	機能完全性，機能正確性，機能適切性
性能効率性	時間やCPUなど資源の使用量が効率的か	時間効率性，資源効率性，容量満足性
互換性	他製品と共存したり情報交換したりできるか	共存性，相互運用性
使用性	利用者のタスクが，効率的に，効果的に，満足して行われるか	適切度認識性，習得性，運用操作性，ユーザエラー防止性，ユーザインタフェース快美性，アクセシビリティ
信頼性	所与の条件，所与の期間において要求機能が遂行できているか	成熟性，可用性，障害許容性（耐故障性），回復性
セキュリティ	不当なアクセスに対して情報やデータが保護されるか	機密性，インテグリティ，否認防止性，責任追跡性，真正性
保守性	保守による製品の修正を効率的で効果的に行えるか	モジュール性，再利用性，解析性，修正性，試験性
移植性	異なる環境へと効率的に移動させることができるか	適応性，設置性，置換性

開発部門

開発部門が評価している品質特性（開発対象別）

SQiP2012 実態調査

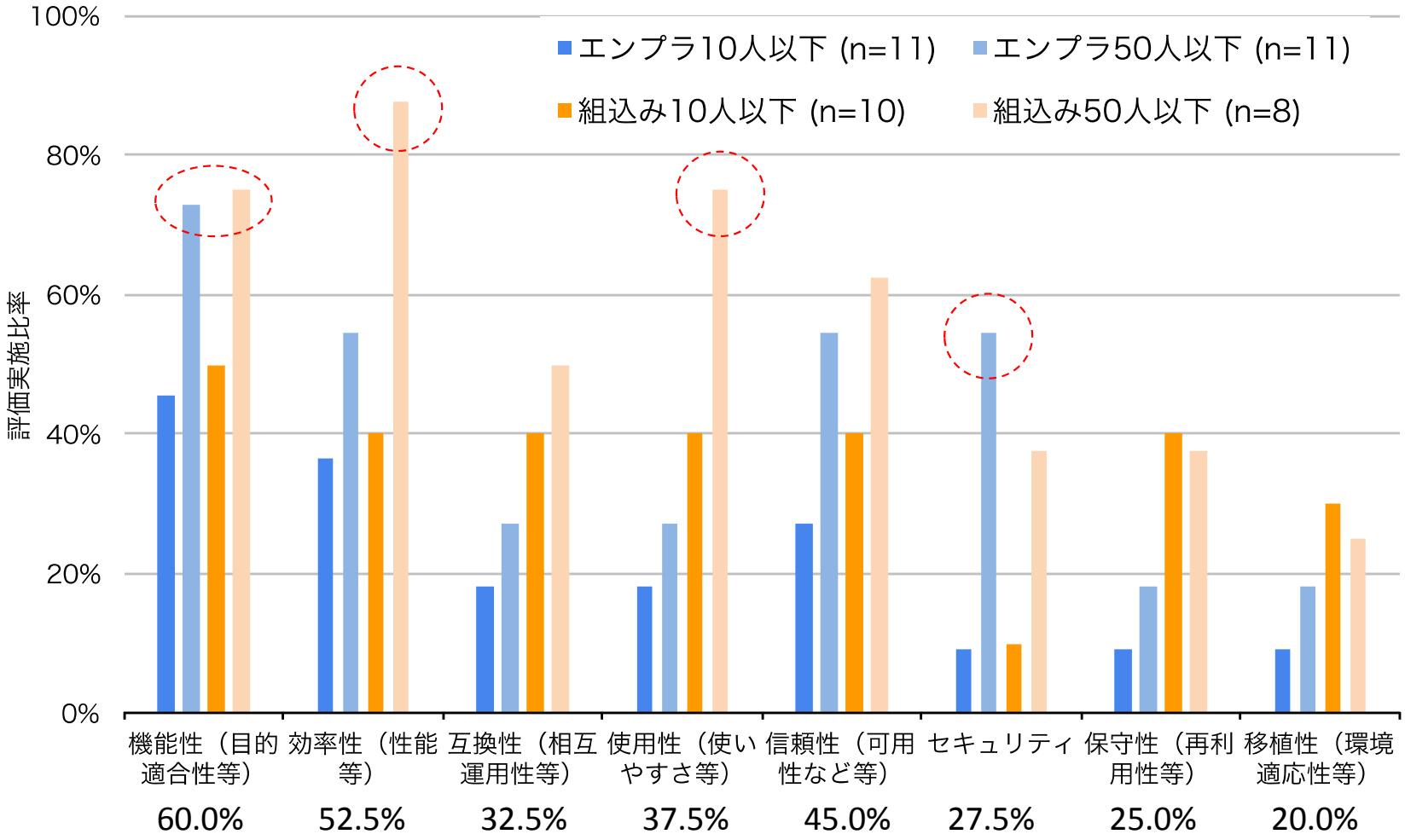


・組込み系：信頼性の評価実施比率が高いが、セキュリティの評価実施比率は低い
・エンタープライズ系：セキュリティの評価実施比率が高い

品質部門

最終成果物について、品質部門として評価している品質特性（開発対象別）（品質専任者数別）

SQiP2012 実態調査



- 機能性の評価実施比率が高く、効率性、信頼性の評価実施比率も比較的高い
- 組込み・50人以下は、効率性、機能性、使用性の評価実施比率が高い
- エンプラ・50人以下は、機能性、信頼性、セキュリティの評価実施比率が高い

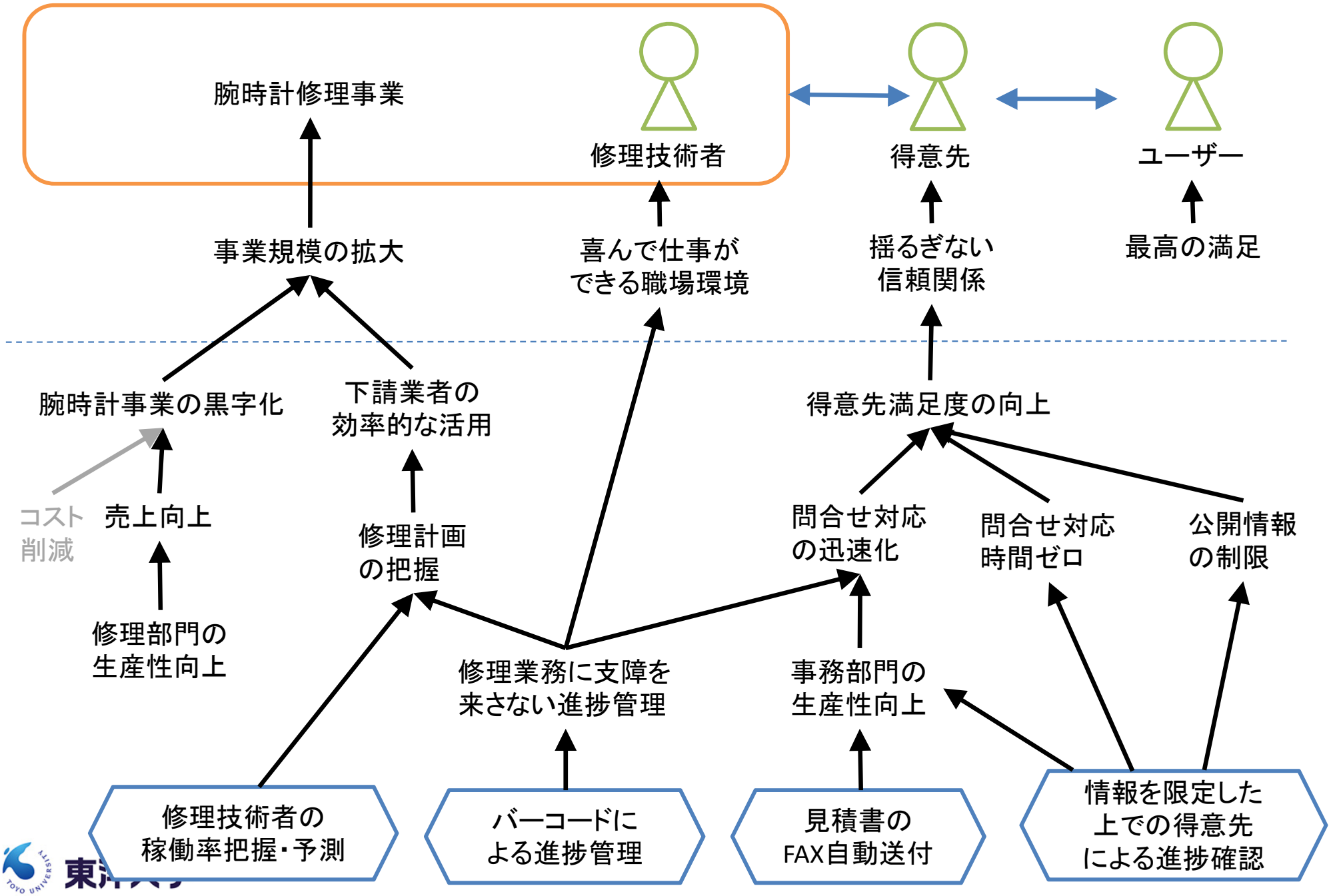
品質要求は非機能要求の一部分

要求の分類

- **機能要求**
- **非機能要求**
  - 品質要求
    - 機能適合性，性能効率性，互換性，使用性，信頼性，
      セキュリティ，保守性，移植性（ISO/IEC 25010 SQuaREモデル）
  - 制約
    - アーキテクチャ制約　… 配備，分散
    - 開発制約　… コスト，納期，使用技術，委託先
  - 法令遵守

非機能要求
- **機能以外の要求**
- **品質要求と制約からなる**

参考：要求獲得プロセスで使用するゴールモデルの例



品質を追求すれば生産性は後からついてくる

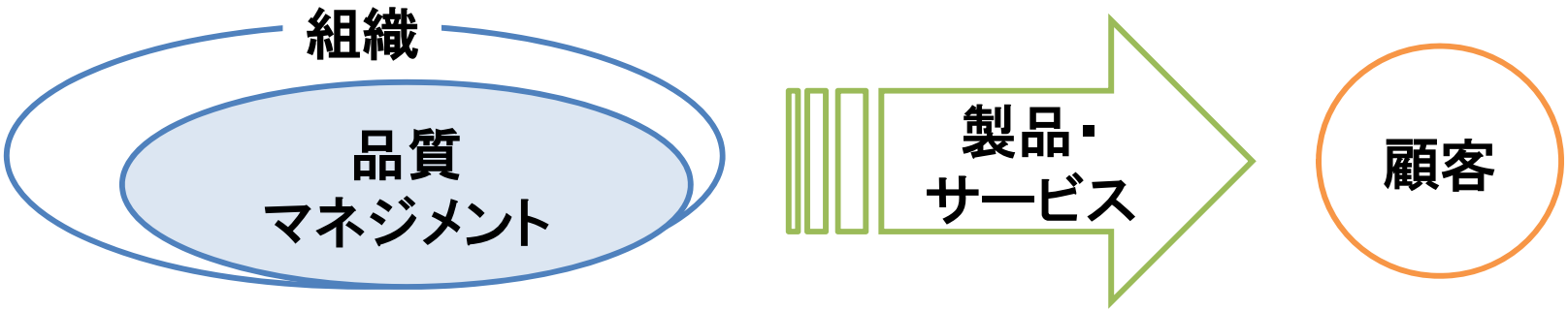
- **QCD(品質，コスト，納期)はプロジェクト管理の基本**
  - コストと納期 → 目標値が明確であり，現状を定量的に把握できる
  - 品質 → 目標値が明確でなく，現状を評価することも難しい
    → レビューやテストでの欠陥摘出数は定量的に把握できるが，ソフトウェアの残存欠陥数は推定値でしかない

  ⇒ 把握しやすいコストと納期が優先され，品質が疎かになりがち
- **品質を作り込めば，コストと納期は達成できる**
  - 品質不良による後戻りや修正作業が，コストと納期に影響を与える
  - 最初から品質を向上させれば，コストと納期を達成できる
- **品質向上には技術が必要**
  - ソフトウェア品質技術を活用し，品質向上を達成する

品質を良くしようという，強い思いと覚悟で取り組む必要がある

品質のマネジメントと主たる管理対象

- **品質のマネジメントとは**
  - 品質のよい製品･サービスを提供するために組織を指揮し，管理すること
  - 幅広い顧客に長期的に満足してもらえるための必須要件



- **管理対象**
  - 結果系（検査重視）
    - 基準を満たした，好ましい状態の結果が得られたか？
    - 悪いものを外に出さないことを基本とする
  - 要因系（プロセス重視）
    - プロセスは維持された状態にあるか？
    - 結果を良くするために，プロセスに対して処置をする
    - 品質の良いものを創り出すプロセス作りを基本とする

プロセスと結果を常に監視し，プロセスに対して処置をする

ハードウェアと比較したソフトウェアの特徴と，管理のポイント

- **物理的特性のない「情報」のため，目に見えない**
  - 仕様書やソースコードを見ることはできるが，膨大であり，理解に時間がかかる
  - ⇒ プロダクトとプロセスの特性を測定し，ソフトウェアを定量的に把握する
- **設計の自由度が高く，物理化学法則や空間的制約に縛られない**
  - 設計解の候補は無数にあり，悪い設計を選んでもプログラムは一応実行できる
  - ⇒ 自由度を抑制するための技法やパターンを適用し，管理する
- **開発終盤でも変更できる**
  - 一部分を書き換えて再コンパイルすることができてしまう（容易であると誤解される）
  - ⇒ 開発序盤で把握できる変更要求なら，仕様化とレビューの時点で対処する
  - ⇒ 副作用が生じないよう，構成を管理し，影響範囲を見極め，変更を管理する
- **人間の知的作業が中心である**
  - 人間的要因に成果が左右されたり，誤解や思い込みで欠陥を混入してしまったりする
  - ⇒ モチベーションやチームワークなどの人間的要因に注意を払う
  - ⇒ 欠陥を蓄積し，その傾向を把握し，再発防止と未然防止に取り組む

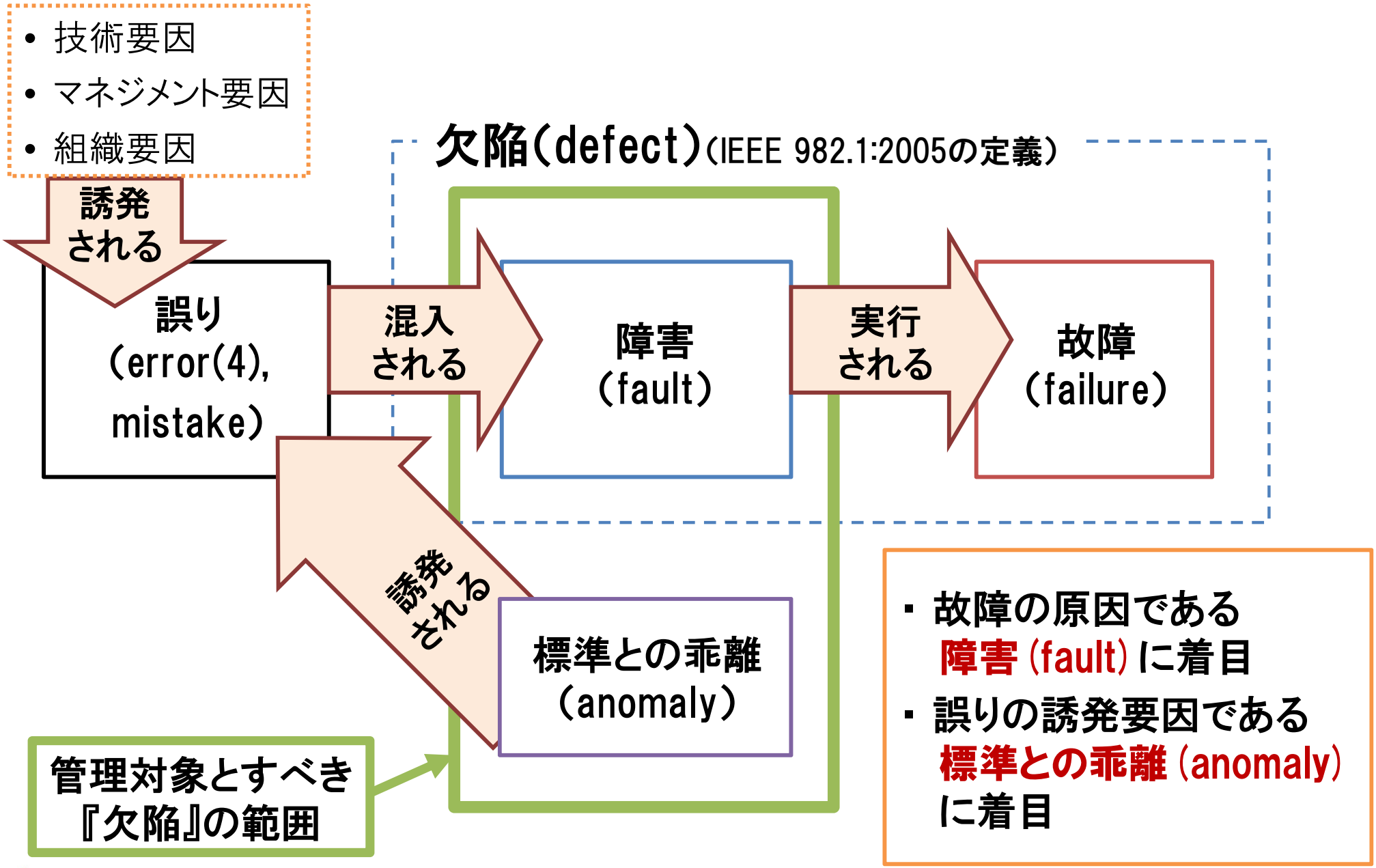
プロセス品質とプロダクト品質の両面で改善していくことが重要

欠陥について：標準的な用語の定義（IEEE標準に基づく）

用語	定義（筆者訳）	出典のIEEE標準
欠陥 (defect)	障害(fault)または故障(failure)のいずれかを言及できる一般的用語。	982.1-2005
障害 (fault)	(1) ハードウェアデバイスまたはコンポーネントの欠陥(defect)。 (2) コンピュータープログラム内の，不正確なステップ，プロセス，またはデータ定義。一般には，エラー(error)やバグ(bug)をこの意味に用いる。	610.12-1990(R2002)
故障 (failure)	システムまたはコンポーネントが，指定された性能要求の範囲内で要求された機能を果たせないこと。	610.12-1990(R2002)
エラー，誤り (error)	(1) 正しい結果と，観測された結果の相違。狭義のerror。 (2) 不正確なステップ，プロセス，またはデータ定義。狭義のfault。 (3) 不正確な結果。狭義のfailure。 (4) 不正確な結果を生み出した人間の行為。狭義のmistake。	610.12-1990(R2002)
不正 (anomaly)	要求仕様，設計文書，ユーザ文書，標準，または誰かの認識もしくは体験など，これらに基づく期待から乖離した状態。	1044-1993(R2002) 1028-2008
	文書において，またはソフトウェアもしくはシステムの運用において観測された，期待から乖離したものすべて。ただし，期待は，検証済みのソフトウェア製品，リファレンス文書，または指定された振る舞いを記述したその他の情報源に基づく。	829-2008

人の誤り(mistake) ―<混入>→ 原因箇所(fault) ―<実行>→ 現象(failure)

何を「欠陥」として捉えるのか？



欠陥の分類

機能性欠陥
(functional defects)

障害 (fault)に相当

大規模欠陥	機能性の欠如，大幅な追加･修正が必要
サポート	ライブラリの欠陥
タイミング	マルチスレッド環境で生じる問題
チェック	妥当性確認ミス，不正な値の検出ミス
リソース	データ，変数，リソース初期化，解放などの誤り
ロジック	比較演算子，制御フロー，計算などの誤り
インタフェース	ライブラリ，デバイス，DB，OS等とのインタフェース不整合

発展性欠陥
(evolvability defects)

標準との乖離
(anomaly)に相当

構造	コード構造
視覚的表現	コードの読みやすさ
ドキュメンテーション	コードの説明

コードインスペクション指摘の
約70%は，発展性欠陥を検出
(Mäntylä, 2009)

- **欠陥の定量的管理の前提として，欠陥の分類を考えておく必要がある**
- **欠陥除去のフロントローディングを議論するときは，機能性欠陥に着目する**
- **将来の機能性欠陥につながる発展性欠陥は，早い段階で芽を摘んでおく**
- **欠陥の分類に着目して，プロジェクトの状況を把握する**

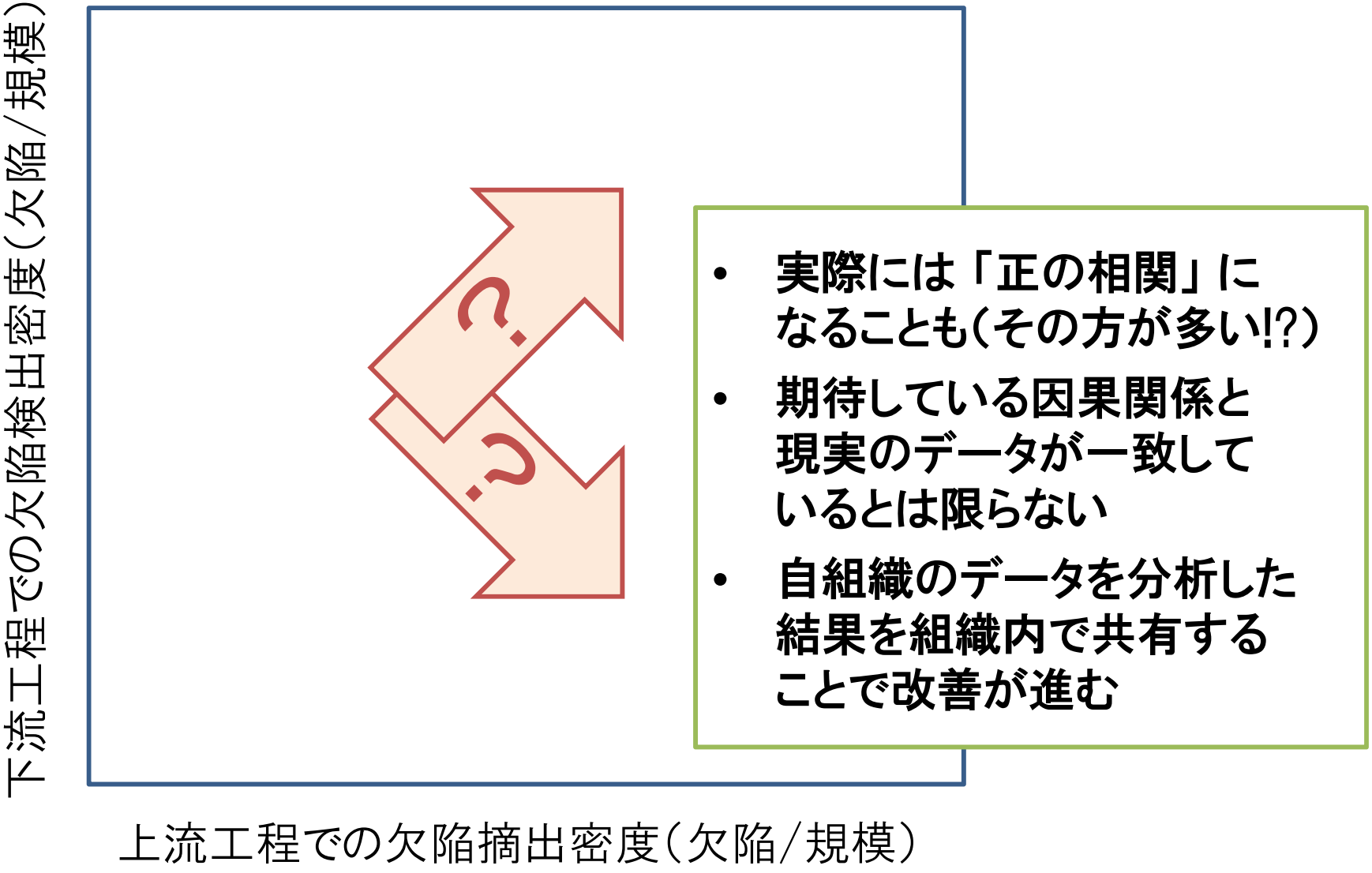
Mäntylä, M. V. and Lassenius, C., “What Types of Defects Are Really Discovered in Code Reviews?,” *IEEE Trans. Softw. Eng.*, vol. 35, no. 3, pp. 430-448, 2009.

品質データ分析は改善の推進力

- **「事実に基づく管理」は品質管理の基本原則である**
- **ソフトウェア開発における「事実にもとづく管理」は容易でない**
  - データの測定・収集プロセスが，**人に大きく依存**している
  - 目的と整合性のあるメトリクス分析をし続けることが難しい
- **ソフトウェア開発において「事実に基づく管理」を進めることが，中・長期的には品質向上の有効な施策である**
  - **品質向上（または悪化）のメカニズム**を解明する
  - データが示す**客観的事実の力**を活用し，改善の推進力とする
  - データを手がかりに原因を特定し，アクションをとり，効果を確認する

  そして，
  - 再発防止のためにプロセスを改善し，失敗プロジェクトを撲滅する
  - 顧客価値の向上につながる活動に注力し，顧客の満足度を高める

データが示す客観的事実が改善の推進力になる例：上流での欠陥摘出と下流での欠陥検出の関係は？



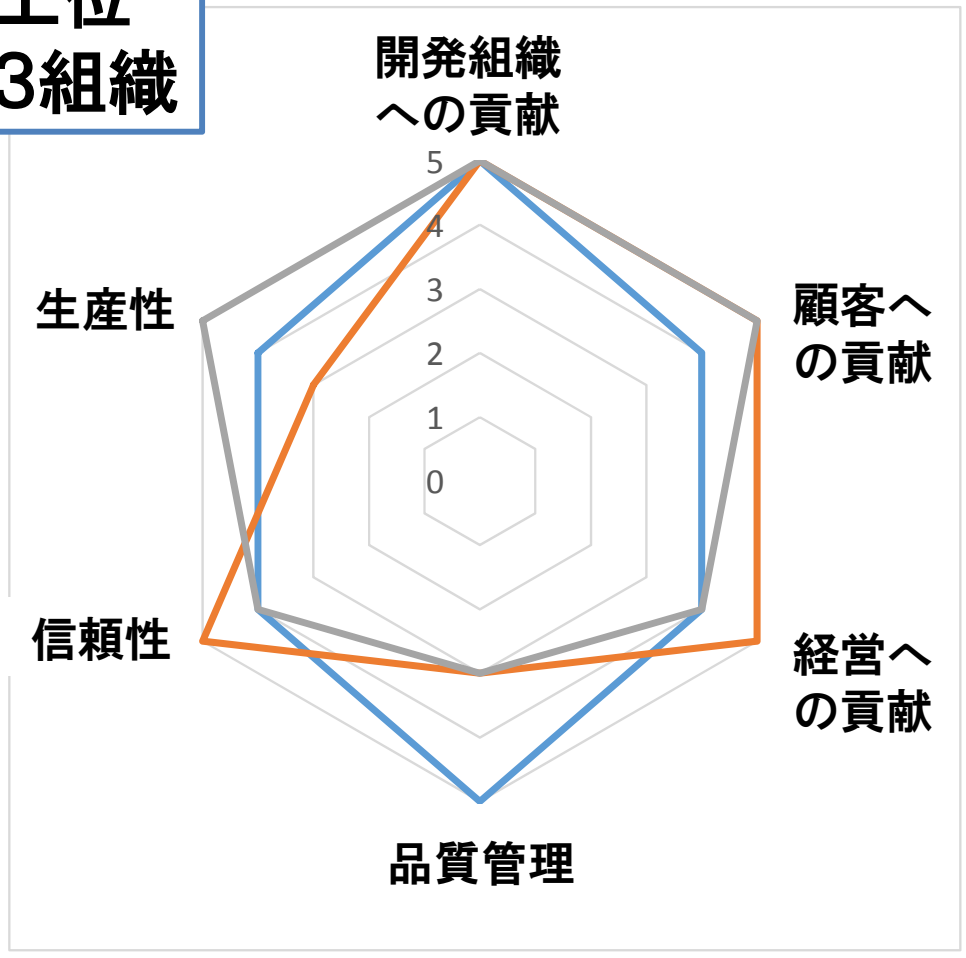
- 実際には「正の相関」になることも（その方が多い!?）
- 期待している因果関係と現実のデータが一致しているとは限らない
- 自組織のデータを分析した結果を組織内で共有することで改善が進む

ソフトウェア品質部門の品質管理能力(の自己評価)

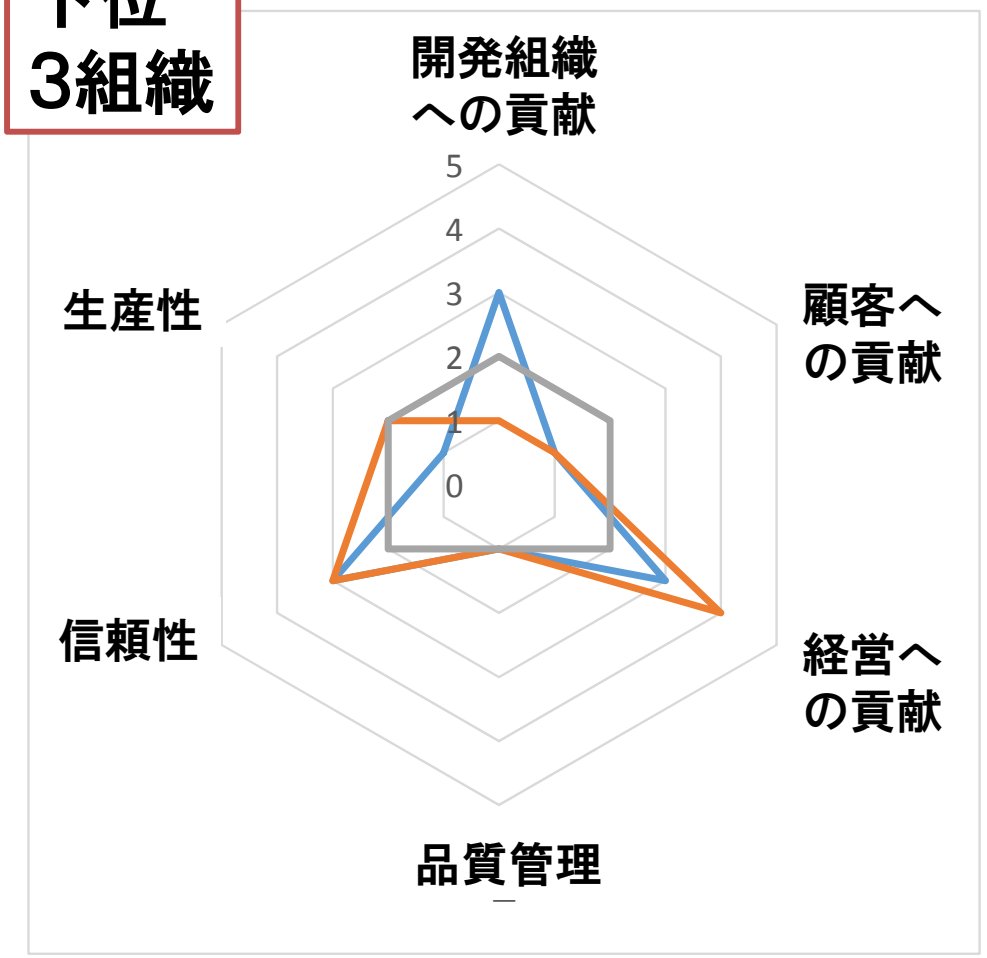
SQiP2013 実態調査

（N=42）

上位3組織



下位3組織



自己評価の高い組織と低い組織では,『面積』に大きな開きがある

http://www.juse.jp/sqip/symposium/2013/program/files/happyou_shiryou_E2-1.pdf

上位10組織 vs 下位9組織の比較

（N=42）

SQiP2013 実態調査

比較項目	上位1/4の組織		下位1/4の組織	
	実施率	ミッション認識率	実施率	ミッション認識率
完了済プロジェクトの実績データの収集と分析	0.90	0.80	0.33	0.33
定量的な評価基準・判断基準の策定	0.90	0.60	0.56	0.22
進行中プロジェクトのモニタリング	0.90	0.60	0.50	0.13
モニタリング結果の分析とアクション	0.90	0.60	0.25	0.13
不具合の収集と原因分析	0.80	0.60	0.50	0.25

自己評価の高い組織においては，定量的品質管理を軸としたこれらの品質改善活動を品質部門のミッションと位置づけ，ルーチンとして実施している

http://www.juse.jp/sqip/symposium/2013/program/files/happyou_shiryou_E2-1.pdf

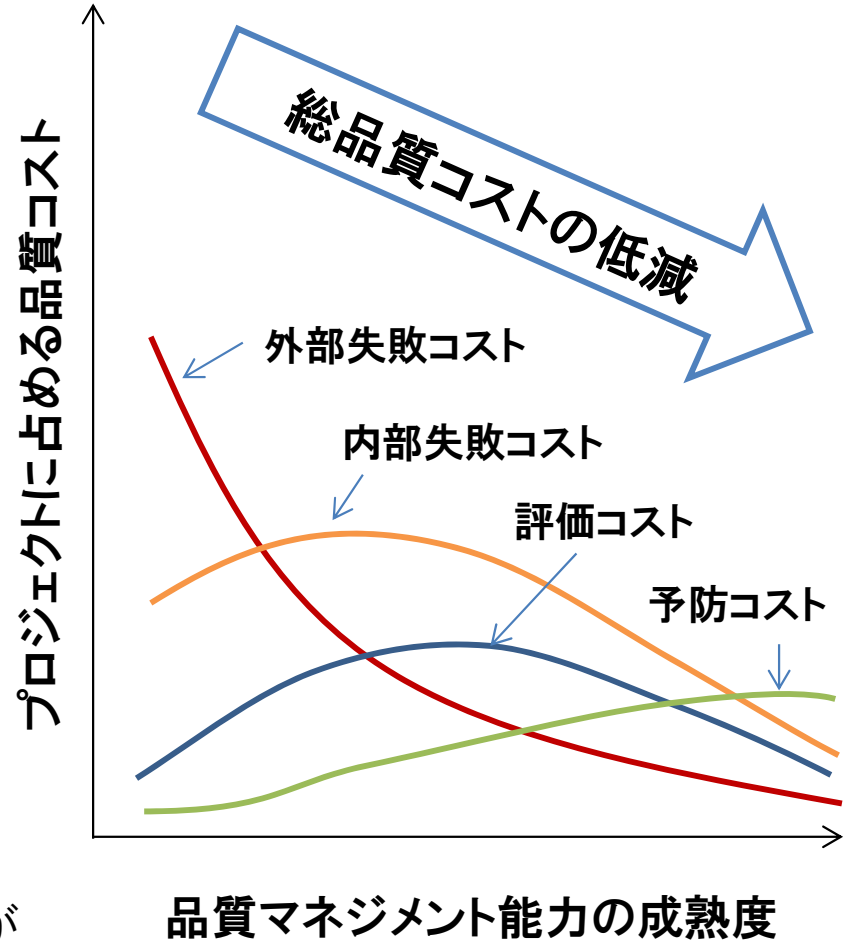
ソフトウェアの定量的品質管理で使用される典型的な管理指標

用途	メトリクス	意味	主な利用方法
テストのモニタリング リリース判定	未解決不具合数	検出不具合数と解決済不具合数の差	テストとデバッグの進捗を評価
	不具合検出率	単位テスト工数当たりのテスト検出不具合数	テストでの欠陥検出の十分性，効率性を評価 （不具合（failure）ではなく欠陥（fault）を見る場合もある）
	累積テスト検出不具合数	検出不具合数の累積値	ソフトウェアの信頼度を評価
	消化済みテストケース数	消化テストケース数の累積値	テストの進捗を評価
	テスト検出欠陥数/KLOC	単位規模当たりのテスト検出欠陥数	テストでの欠陥検出の十分性を評価
	テスト工数/KLOC	単位規模当たりのテスト工数	テスト工数の十分性を評価
	テストケース密度	単位規模当たりのテストケース数	テストケースの十分性を評価
上流工程のモニタリング	レビュー工数密度	単位レビュー工数当たりの欠陥摘出数	レビューの欠陥検出の十分性，効率性を評価
	レビュー指摘密度	単位規模当たりの欠陥摘出数	レビューの欠陥検出の十分性，有効性を評価 （上流工程のレビューをまとめて評価する場合もある）
	レビュー工数/KLOC	単位規模当たりのレビュー工数	レビュー工数の十分性を評価
	レビュー工数/開発工数	開発工数に対するレビュー工数の比率	レビュー工数の十分性を評価
	上流工程での欠陥除去比率	上流工程での混入欠陥数の予測値に対するレビューでの欠陥摘出数の合計	上流工程（例：コードレビュー終了まで）の欠陥摘出の有効性を評価
	レビュー速度	単位レビュー時間当たりの成果物規模	レビューの効率性，適切性を評価
	開発工数/KLOC	単位規模当たりの開発作業工数	開発作業への投入工数の十分性を評価
事後分析	リリース後欠陥密度	単位規模当たりの，リリース後に発見された欠陥数	開発プロセス全体の品質保証の十分性を評価
	工程の欠陥除去比率	ある欠陥除去工程の開始時の残存欠陥数に対する，当該工程での欠陥摘出数	欠陥摘出工程（レビューとテスト）の欠陥検出の有効性を評価
	プロセス全体の欠陥除去比率	全欠陥数に対する，開発プロセスのすべての欠陥除去工程での欠陥摘出数合計	開発プロセス全体の欠陥摘出の有効性を評価

品質にしっかりと取り組めば，コストは下がり，スピードは向上する

品質マネジメント効果の発現順序

1. **外部失敗コストの低下**
   - テストを重視し，テスト状況を把握し，外部失敗を内部失敗へと振り分ける
2. **内部失敗コストの低下**
   - レビューを重視し，レビュー状況を把握し，欠陥を早期に除去できるようにする
3. **欠陥除去プロセスの効率向上**
   - レビューとテストの効率向上に努める
   - リスクベースで活動の重点化を図る
   - プロセス改善，知識蓄積，教育に投資する
4. **総品質コストの低減・安定化**
   - リスクベースによるレビューとテストの「間引き」が機能不全に陥らないよう，さらなる効率化を図る



- **一定の品質コストはつねに必要，しかし，その効率向上を追求し続ける必要がある**
- **安易な品質コストの削減は，取り返しのつかないところまで組織能力を衰退させる**

実績データの収集・分析：市場流出欠陥のベンチマークデータ

日・米・欧・印のパフォーマンス比較（Cusumano *et. al.*, 2003）

項目	インド	日本	米国	欧州他	合計 or 平均
調査対象のプロジェクト数（件）	24	27	31	22	104（合計）
新規コード行数（KLOC, 中央値）	209	469	270	436	374
出荷後の欠陥密度（中央値）	0.263	**0.020**	0.400	0.225	0.150

IPA/SEC『ソフトウェア開発データ白書2012-2013』

	新規開発 (N = 520)	改良開発 (N = 349)
出荷後不具合 / KLOC（中央値）	0.016	0.000
出荷後不具合 / KLOC（平均値）	0.106	0.123

- この指標を用いて組織横断で比較するのは「眉唾」に思うかもしれない
- 測定方法がほぼ揃った同一組織において，開発の結果をデータで把握し，その分布や変化を知ることが，改善サイクルを回していく第一歩

Cusumano, M., *et. al.*, "Software Development Worldwide: The State of the Practice," *IEEE Software*, Nov/Dec 2003, pp.28-34.

品質にしっかりと取り組めば，コストは下がり，スピードは向上する

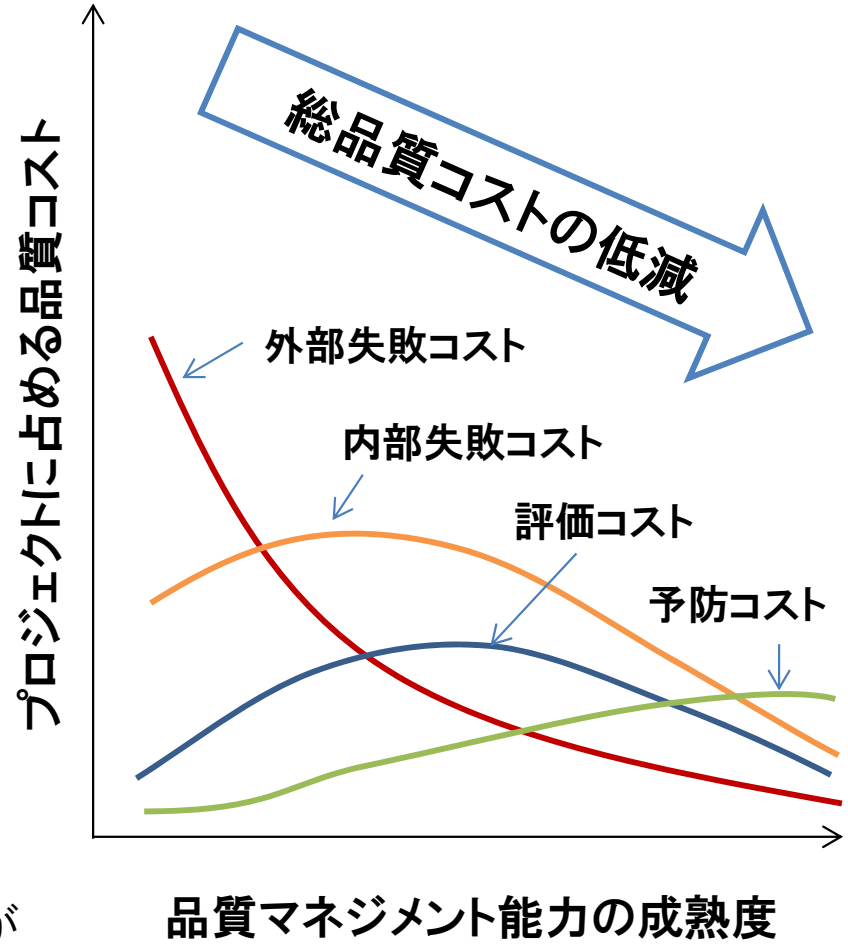
品質マネジメント効果の発現順序

1. **外部失敗コストの低下**
   - テストを重視し，テスト状況を把握し，外部失敗を内部失敗へと振り分ける

2. **内部失敗コストの低下**
   - レビューを重視し，レビュー状況を把握し，欠陥を早期に除去できるようにする

3. **欠陥除去プロセスの効率向上**
   - レビューとテストの効率向上に努める
   - リスクベースで活動の重点化を図る
   - プロセス改善，知識蓄積，教育に投資する

4. **総品質コストの低減・安定化**
   - リスクベースによるレビューとテストの「間引き」が機能不全に陥らないよう，さらなる効率化を図る



- **一定の品質コストはつねに必要，しかし，その効率向上を追求し続ける必要がある**
- **安易な品質コストの削減は，取り返しのつかないところまで組織能力を衰退させる**

欠陥に学び，再発防止策を考える

入居者の死亡，1カ月気付かず 横浜の高齢者用住宅

横浜市の生活援助員派遣サービス付き高齢者用住宅で，一人暮らしの60代男性が部屋の中で病死したまま約一カ月間も放置されていたことが分かった。

在宅中でも外からの施錠で「不在」と表示されるというシステムの不十分さと，業務委託先の市福祉サービス協会や生活援助員の安否確認の不足などが原因としている。

① AM8:25　水道12時間未使用センサーが作動
　　警備会社に通知

② AM8:38　警備員が安否を確認
　　（高齢者は寝過ごしただけ）

③ **警備員が外から施錠**

④ AM10:00　生活援助員は「不在」表示を確認
　　安否確認せず

⑤ 生活援助員は，その後も週二回「不在」表示 を確認するが安否確認せず

⑥ 17日後，生活援助員が市の福祉サービス協会本部に経緯を報告

⑦ さらに14日後，親族に連絡して家に入る許可を得て，死亡を確認

（2006/2/10 新聞記事より）

高齢者用住宅の事例に学び，再発防止策を考える：レビューチェックリストとして何を考えるべきだったか？

フェイルセーフ（安全性）

ソフトウェアで認識した状態と，観測対象の実際の状態とのあいだにズレが生じたときに，安全側に倒れるようにシステムが設計されているか

振舞いの妥当性

状態とイベントの各々の組合せについて，ソフトウェアの振舞いは妥当か

利用シナリオの網羅性

想定できるすべての利用シナリオについて，ソフトウェアの振舞いの妥当性を検討したか

- 欠陥に学び，チェックリストというかたちで形式知化する
- チェックリストを維持管理し，開発対象に応じて項目を絞る

チェックリストを起点に組織学習と知識創造を促進する

①レビュー等のプロセスを通じて，
個人の暗黙知をグループの暗黙知とする

②欠陥に学び，
その暗黙知を形式知化する

④体系的な形式知を内面化した上で，
新たな暗黙知を個人が創造する

③個々の形式知から
体系的な形式知を創造する

Nonaka, I. and Takeuchi, H., *The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation*, Oxford University (1995)
野中郁次郎・竹内弘高『知識創造企業』東洋経済新報社 (1996).

高齢者用住宅の事例に学び，再発防止策を考える：一段上のシステムレベルで考慮すべきチェックリスト

フェイルセーフ（安全性）

ソフトウェアで認識した状態と，観測対象の実際の状態とのあいだにズレが生じたときに，安全側に倒れるように**システム全体**で設計されているか

機能の識別・実現

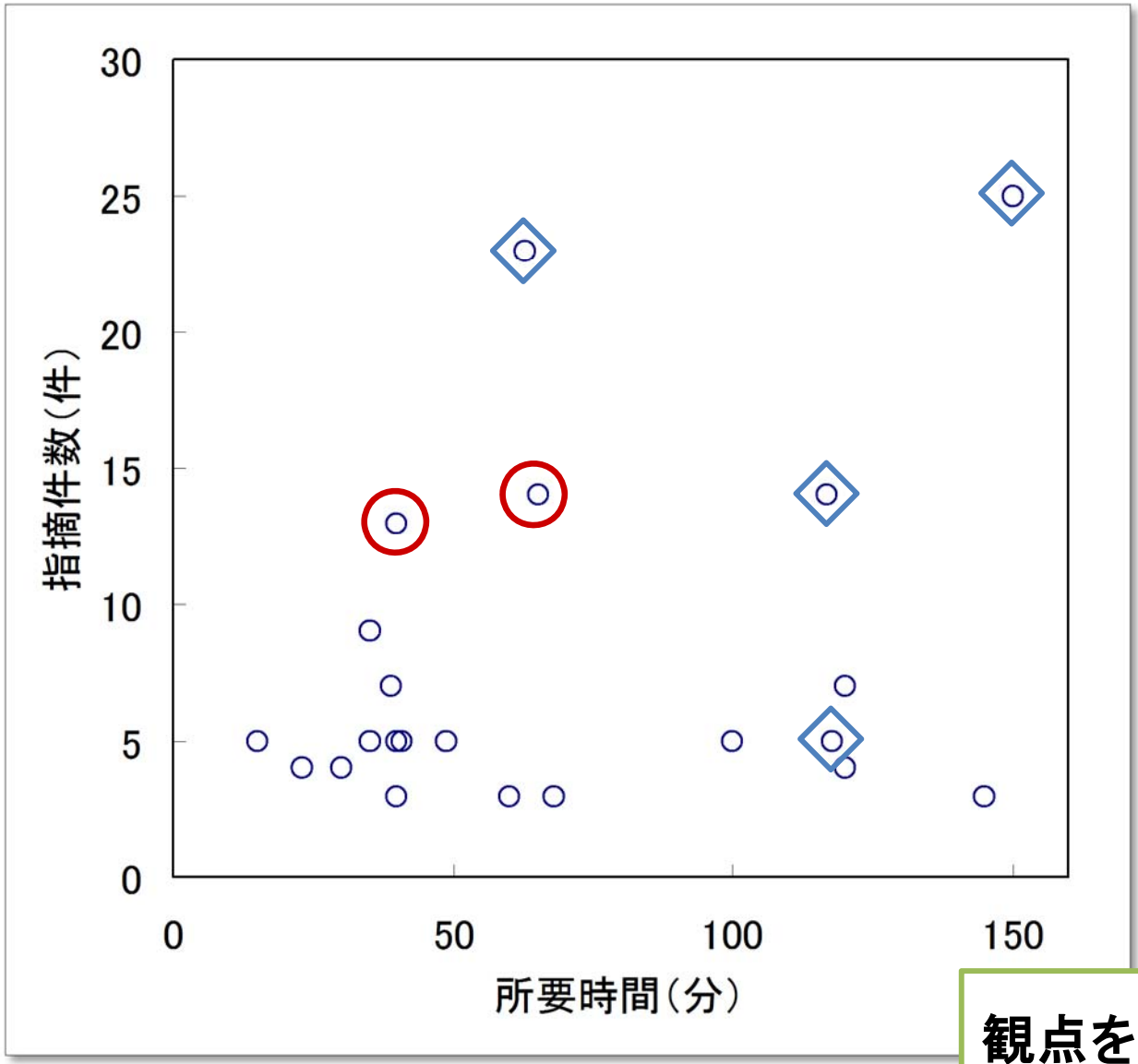
ソフトウェアで認識した状態と，観測対象の実際の状態とのあいだにズレが生じたときに，これを補正する**機能**を**システムのどこか**で実現しているか

機能の合理的な配置

その機能はどの**システム階層**に配置するのが合理的か（コスト等を考えて）

ソフトウェアだけではなく，情報システムだけではなく，
ビジネスシステムを含めた全体で，機能とその合理的な配置をデザインする

レビューでの「成果物の読み方」と指摘件数



レビュー観点と
シナリオを明示

レビュー観点を明示

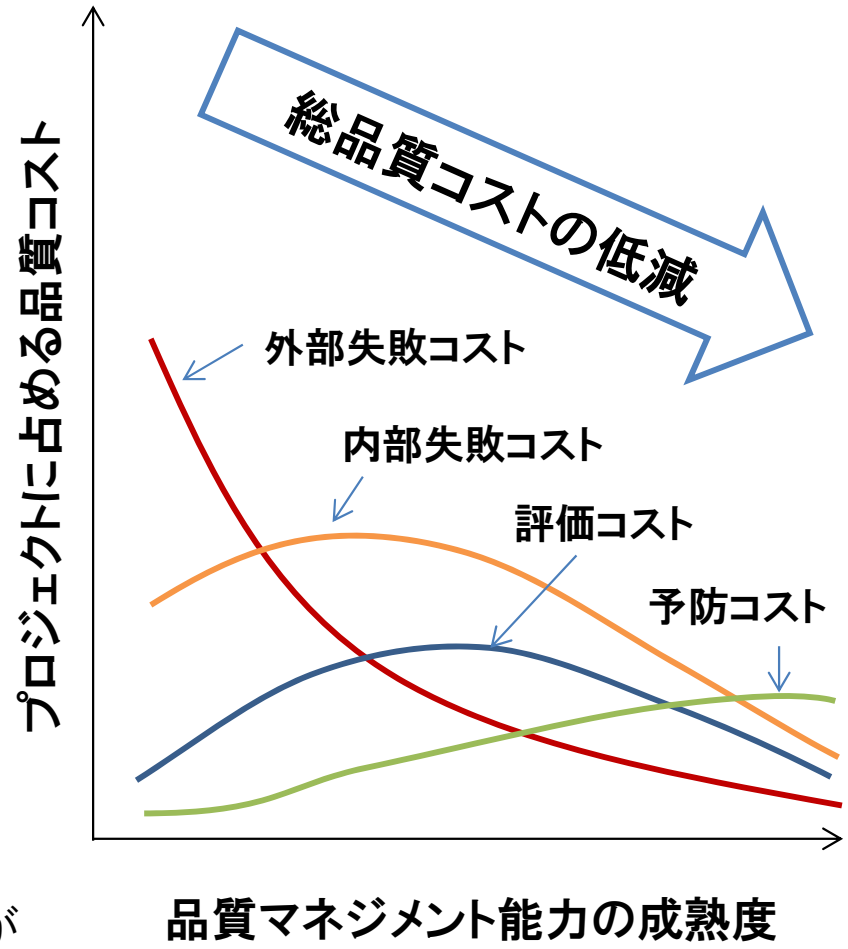
注）指摘件数は，不適合な指摘もすべて計上している

観点を明示したレビューアは，
多くの欠陥・課題を指摘をしている

品質にしっかりと取り組めば，コストは下がり，スピードは向上する

品質マネジメント効果の発現順序

1. **外部失敗コストの低下**
   - テストを重視し，テスト状況を把握し，外部失敗を内部失敗へと振り分ける
2. **内部失敗コストの低下**
   - レビューを重視し，レビュー状況を把握し，欠陥を早期に除去できるようにする
3. **欠陥除去プロセスの効率向上**
   - レビューとテストの効率向上に努める
   - リスクベースで活動の重点化を図る
   - プロセス改善，知識蓄積，教育に投資する
4. **総品質コストの低減・安定化**
   - リスクベースによるレビューとテストの「間引き」が機能不全に陥らないよう，さらなる効率化を図る



- **一定の品質コストはつねに必要，しかし，その効率向上を追求し続ける必要がある**
- **安易な品質コストの削減は，取り返しのつかないところまで組織能力を衰退させる**

「この工程の欠陥除去能力を高めましょう」とだけ言って／言われて，正しく対処できるだろうか？

IPA/SEC「重要インフラ情報システム信頼性研究会」報告書(2009)に見る，
再発防止のための指示　http://sec.ipa.go.jp/reports/20090409.html

- 重要インフラ情報システムの障害事例の収集(約100事例)
- 原因の分析 (ソフトウェア欠陥 ／ 運用ミス ／ …)
- 問題構造の分析 (障害事例の当事者ではないが，専門家が時間をかけて議論)
- 再発防止のポイントを分類・整理

例:「複数の経験者によるレビューを徹底し，仕様，プログラムなどを充分レビューする」

現場では，「徹底」「充分」とだけ言われても困る(…という人もいる)

重点評価／改善すべき箇所を特定できるソフトウェア品質技術が必要

プロダクトメトリクスにも目を向けよう

- **スタッフ部門が着目するのは，多くの場合，プロセスメトリクス**
  - 例）レビュー工数，レビュー指摘欠陥数，テスト密度，…
  - プロセスに着目すること自体は悪くなく，むしろ推奨すべき
    → プロセスを制御してプロダクト品質を制御するのは品質管理の基本
  - しかし，プロセスだけを見て，**プロダクトを見ないのはいけない**
    → レビュー工数が足りない，レビュー指摘数が足りない，…という指摘は，「どの部分に着目して対処すべきか」の解にはならない
    → プロセスメトリクスは，**分解能に限界**がある

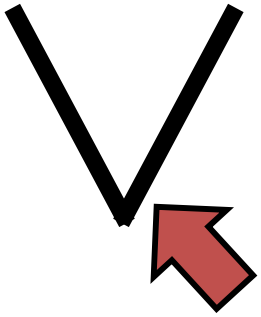
- **ソフトウェア開発技術の道理を踏まえた改善を推進しよう**
  - **凝集度，結合度，複雑度**などは，70年代，80年代から言われ続けているソフトウェアエンジニアリングの基本
  - これらの特性を測定するメトリクスは90年代までに数多く示されていて，ツール化されているものも多い
  - 保守性を疎かにしていると，後でたいへんな**ツケ**となって回ってくる

欠陥の60～80%は，20%のモジュールに存在する

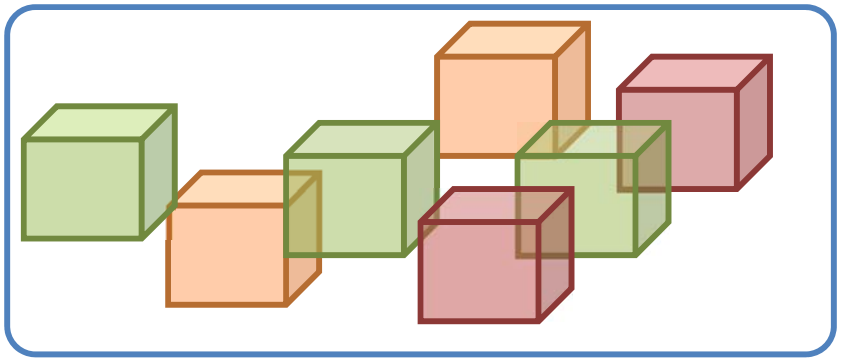
- DOS/V OSにおける約500個のモジュールのうち，20%のモジュールでエラーの約80%を検出した (Endres 1975)
- Fault-prone性が高いと予測された上位20%のモジュールに，全フォールトの60%が含まれていた (Ohlsson 1996)
- 再作業の80%は20%の欠陥によるものだった (Boehm 2000)
- 通信スイッチシステムに含まれる20%のモジュールに，60%のフォールトが含まれていた (Fenton 2000)
- 20%のモジュールに，運用時に発見されたフォールトの80%が含まれていた (Fenton 2000)
- 通信システムにおける欠陥の63～70%は，20%のモジュールに含まれていた (Andersson 2007)
- GNU Compiler Collectionのプロジェクトで，80%のフォールトが20%のファイルに含まれていた (Hamill 2009)

欠陥がありそうな「約20%」モジュールを特定したい
→ fault-proneモジュール予測

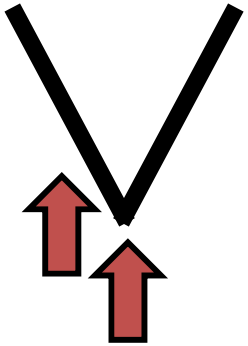
欠陥のありそうなモジュールを特定する：fault-proneモジュール予測



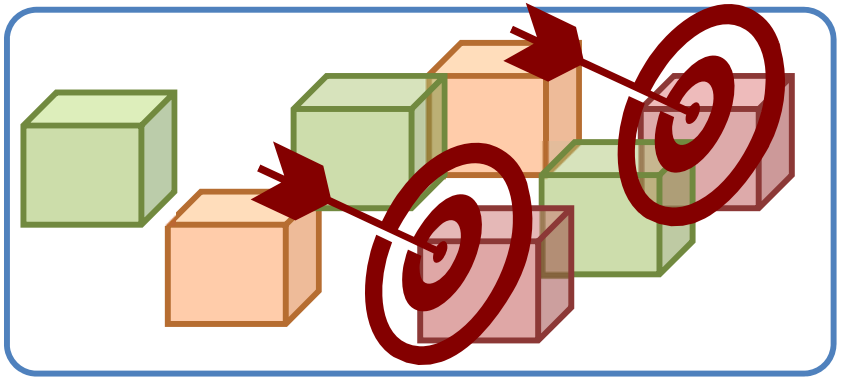
ソフトウェア開発プロセス
（V字モデル）のコーディング／
単体テストが終わった時点で，

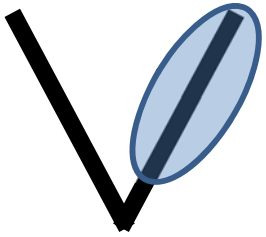


欠陥がありそうな度合いによって
モジュールを色分けして，



ソースコード等から測定可能な
プロダクトメトリクスや
プロセスメトリクスを用いて，





機能テスト／システムテストで
欠陥が見つかりそうなモジュールを，

優先的／重点的にレビューしたり
テストしたりすることで，
レビューやテストの効率／効果を
高めたい

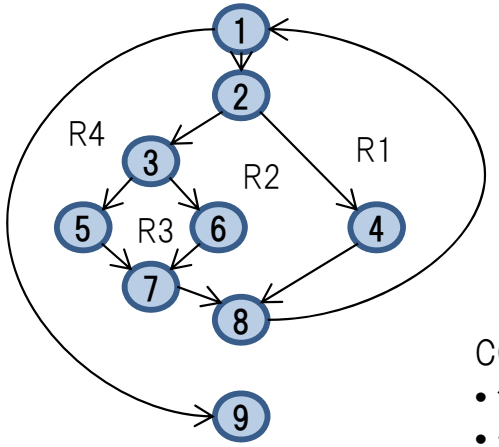
Twitter4Jのコードメトリクスと，欠陥発生モジュールの対応分析によって得られた知見

- コード行数が**380行以上**という大きいクラスは3個あり，これらはすべてリリース後に欠陥修正があった**（欠陥修正率3/3 = 1.0）**。
- **凝集度が欠如していない**クラスは46個あり，このうちリリース後に欠陥修正があったのは15個であった**（欠陥修正率15/46 = 0.33）**。
- 一方，**凝集度が欠如した**クラスは53個あり，このうちリリース後に欠陥修正があったのは42個であった**（欠陥修正率42/53 = 0.79）**。
- 凝集度が欠如していたクラス53個について，**結合度**の値が**中央値の5以下**では**欠陥修正率が17/28 = 0.61**であったのに対して，**中央値より大きい場合**では**欠陥修正率が25/25 = 1.0**であった。

レビューやテストの重点化を進めるにあたって，
このようなデータから得られた客観性の高い知見は
プロセス改善の推進力になる

コードの複雑度，結合度，凝集度の欠如，応答メソッド数

WMC ＝ Σ 各メソッドのサイクロマチック複雑度

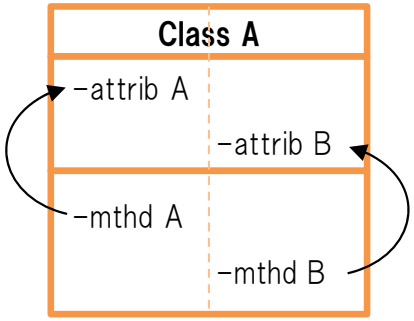


CC = 4

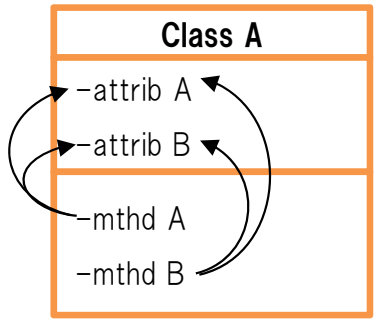
CC (Cyclomatic Complexity)
- 領域の数
- 辺の数 － ノード数 ＋ 2

LCOM ＝属性に対してクラス内メソッドが網羅的にアクセスしていない度合い

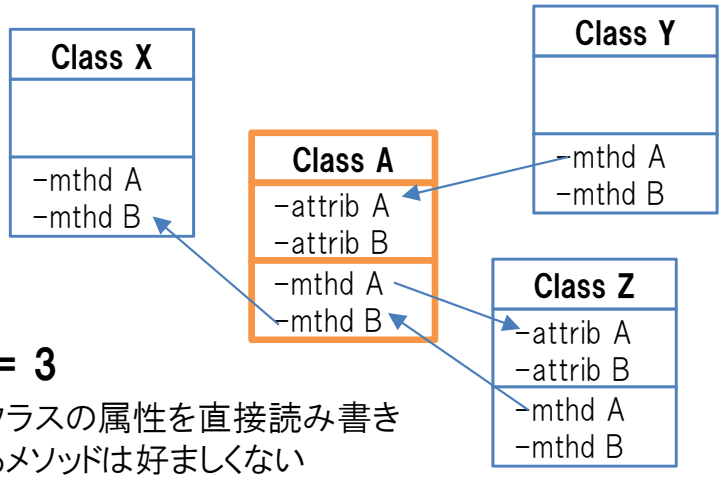
LCOM = 1



LCOM = 0



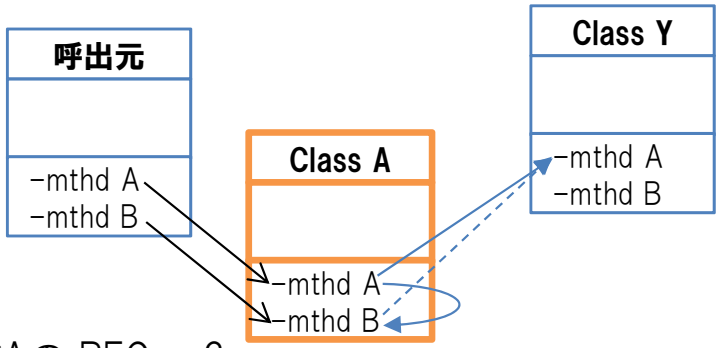
CBO ＝ 結合関係にある他クラスの数



CBO = 3

※ 他クラスの属性を直接読み書きするメソッドは好ましくない

RFC ＝ 応答に用いるメソッド呼出しの種類の数



RFC = 3
- メソッドAの RFC = 2
- メソッドBの RFC = 1 → Class Yのmthd Aはメソッドでカウント済，クラス単位で集約するときは数えない

*"I'd rather be **vaguely right** than **precisely wrong**."*

精密に誤るよりも，漠然と正しくありたい

– John Maynard Keynes

- 私たちのメトリクス分析は **精密に誤って**いないか？（自戒の念も込めて）
  - 理論モデルの洗練が目的になっていないか
  - 現場にとって分かりにくい分析結果やモデルになっていないか
  - その分析結果やモデルは品質向上に貢献するのか
- **漠然と正しい**情報が得られているか？
  - 意思決定の役に立つレベルの精度が得られているか
  - 正しい意思決定に役立つ，正しい方向感の情報を生み出しているか

再掲：品質にしっかりと取り組めば，組織は賢く，強く，幸せになれる

- **品質にしっかりと取り組む**
  - ソフトウェアを通じて顧客に提供する価値を考える
  - その価値を提供し続けるために，組織的に必要な活動を考える
- **必要な活動をデザインする**
  - 顧客に価値を提供できていることを保証するプロセスを確立する
  - 価値提供のスピードを加速させるようプロセスを改善する
- **組織が賢く，強くなる**
  - 価値提供の結果に学ぶ
  - 失敗に学ぶ
  - 自社独自の経験に学ぶ
  - 学びを積み重ねることで，組織は賢く，強くなる
  - 賢く，強い組織は幸せになる