要求管理を確実に行う7つの技法

2008.2.22

ソフトウェアプロセスエンジニアリング（株）

三宅　和之

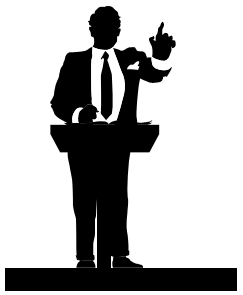
Software Process Engineering Inc.

自己紹介

■ 三宅 和之（みやけ かずゆき）

信託銀行のクオンツアナリストからコンサルティング
会社での要求アナリストへ転身。2003年SPEIを設立。
現在は、要求管理・プロジェクト管理を専門分野として
開発プロジェクトの支援に邁進中。

- ◆ソフトウェアプロセスエンジニアリング（株） 代表取締役
- ◆RaQuest開発アドバイザー
- ◆グローバルナレッジ「ソフトウェア要求管理」講師
- ◆PMI認定PMP



要求管理の現状

○：重要性は少しずつ認識されてきた

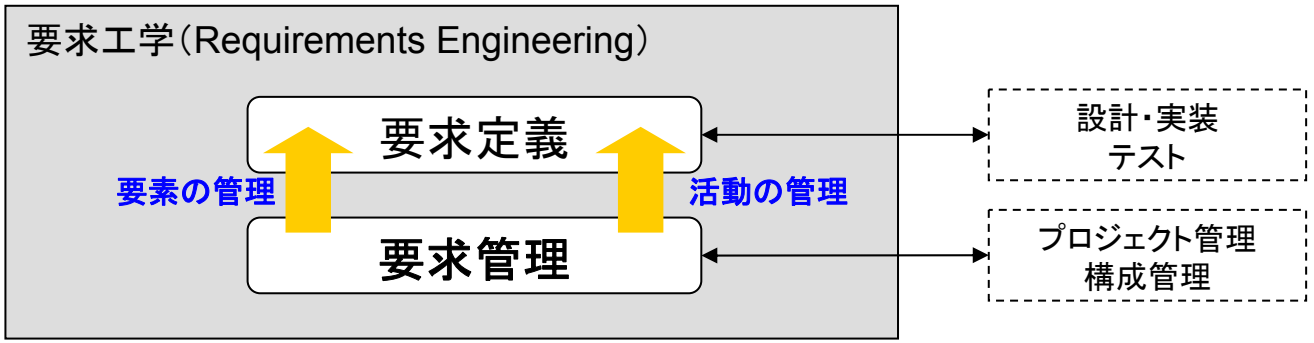
△：プロジェクトの現場には浸透していない

■ なぜ浸透しないか

- “要求管理”の概念が曖昧
- 解決するための技法が分からない

要求管理とは

- “要求工学”に含まれる分野
  - 要求定義
    - 要求を収集し、仕様化する活動
    - 設計や実装などエンジニアリング作業と密接に関連
  - 要求管理
    - 要求で扱う要素とその活動を管理する活動
    - プロジェクト管理や構成管理などの活動と密接に関連



要求管理の技法

要求管理にもベストプラクティス・技法は存在する

- 経験も重要だが、工学的なアプローチも有効
- 特に要求の”要素”を管理する分野は、技法を適用し易い

要求管理における主要な課題と対応する技法

課題	解決するための技法
要求の蓄積	技法1: リポジトリの利用
要求の分類	技法2: 要求パッケージの利用
要求の論理検証	技法3: トレーサビリティの活用
要求の暗黙知	技法4: 要求属性の活用
要求の状態	技法5: 要求のステータス管理
要求スコープ	技法6: 要求のベースライン管理
変更への対応	技法7: 変更要求管理

課題1:要求の蓄積

典型的な事象

- 一時的な記録方法の利用
  - 紙、スプレッドシート、ワードプロセッサなど
- ルールの無いデータ管理
  - ファイルサーバに保管する程度

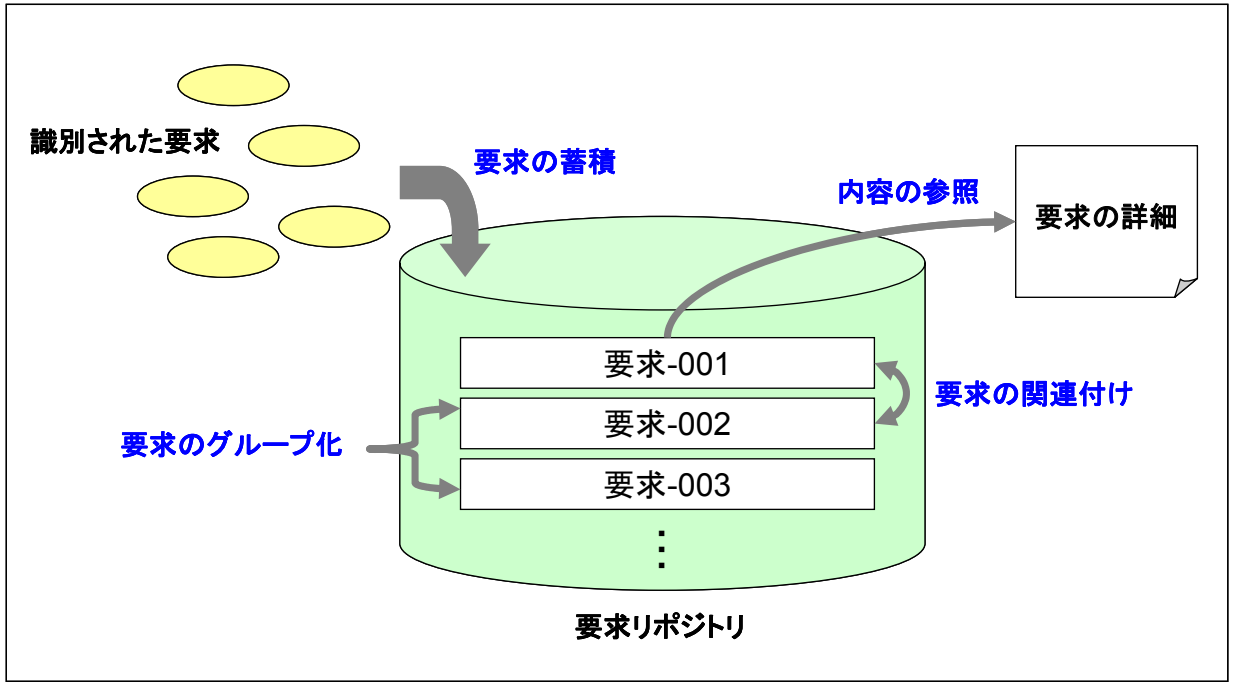
起こりうる問題

- 要求が埋もれてしまう
- 要求をまとまった単位で扱うことができない

◆技法1:リポジトリの利用

- 要求の要素をリポジトリに蓄積する
  - 要求を一意に識別可能
  - 様々な切り口で要求を参照し、更新できる

→要求を工学的に管理するための基盤



運用上のポイント:リポジトリの利用

- ✓ 要求工学に基づくリポジトリ構造を利用する
  - 要求の抽象度・分類を考慮した構造

  →技法2と関連
- ✓ リポジトリ管理をルール化する
  - 要求を参照・更新する際は、必ずリポジトリを経由する
- ✓ できればツールの利用を推奨
  - リポジトリ構造の定義に労力をかけるべきではない

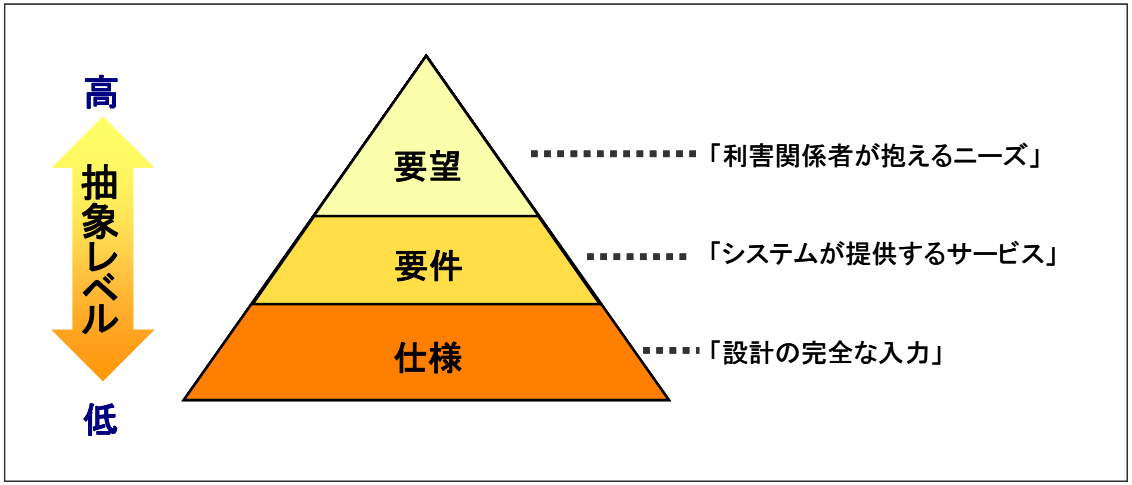
課題2：要求の分類

- 典型的な事象
  - 抽象度がバラバラ
  - 特定の分野に偏った要求の収集
- 起こりうる問題
  - 何を詳細化するべきか分からなくなる
  - 要求が漏れてしまう

概念：要求の抽象度と分類

■ 要求の抽象度

- 要望
- 要件
- 仕様



■ 要求の分類

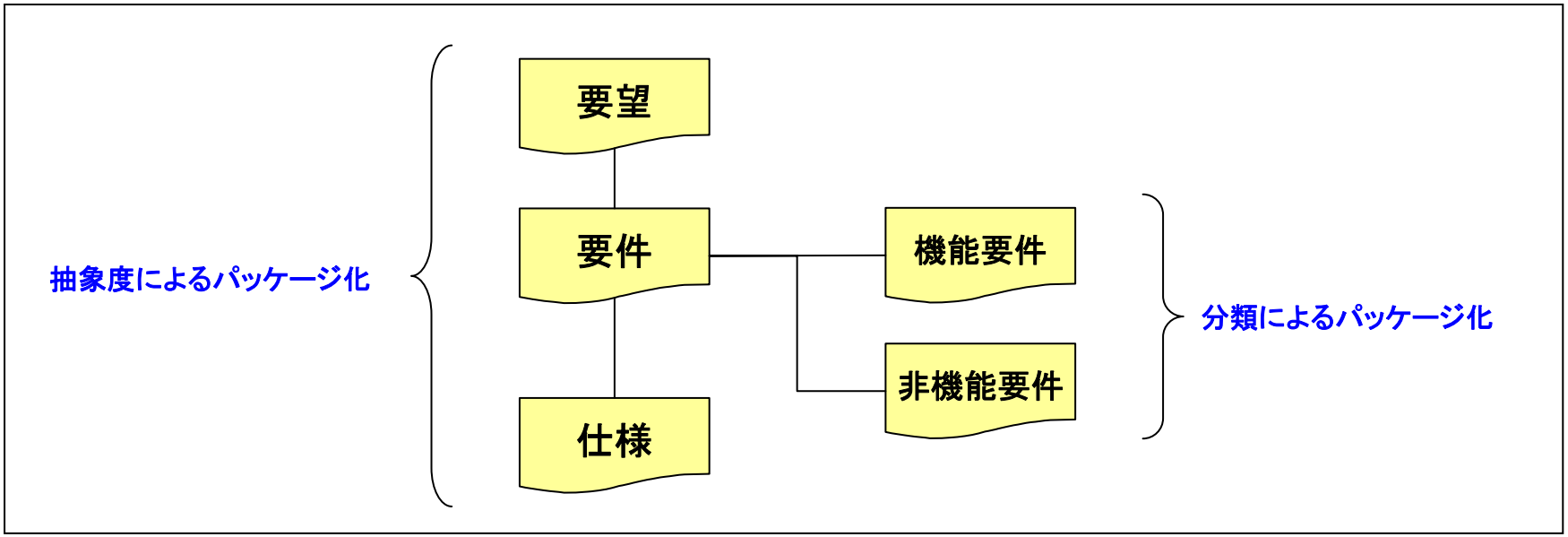
- 機能要求
- 非機能要求

機能要求	「機能」	ユーザにとって有益な機能性を提供するために システムが行わなければならない行動
機能外要求	「ユーザビリティ」	ユーザインターフェースの統一性、習得のしやすさ、 ヘルプ・サポート文書の必要性に関する要求
	「性能」	応答時間、スループット、キャパシティ、 資源効率性に関する要求
	「信頼性」	利用可能性、可用性、データの正確性、 システムの安全性、データ保護に関する要求
	「セキュリティ」	機密性、保全性、セキュリティ可用性に関する要求
	「保守性」	変更のしやすさ、移植性、再利用性、 ローカライズ可能性に関する要求
	「運用性」	運用のしやすさ、監視のしやすさに関する要求
	「システム制約」	技術的制約、法律・規定への準拠に関する制約事項

◆技法2：要求パッケージの活用

- 要求を抽象度と分類でパッケージ化（グループ化）する
  - 要求のリポジトリ構造が定義される
    - 要求を蓄積・検証しやすい単位で保持できる
    - 漏れを防ぎ、整合性を確保できる

→要求を管理するための基本構造



運用上のポイント:要求パッケージの活用

- ✓ 抽象度を意識したプロセスの運営
  - 今、どの抽象度に注目すべき局面かを意識する
    - 要望？、要件？、仕様？
- ✓ 既存の分類モデルを活用する
  - **FURPS+**
    - Robert Gradyによる要求の分類
    - F: 機能、U: 操作性、R: 信頼性、P: 性能、S: 保守性、+:その他
  - **ISO9126 (JIS X 0129-1)**
    - ISO/IEC 9126「ソフトウェア製品の品質」による分類
    - ・機能性　・信頼性　・使用性　・効率性　・保守性　・移植性

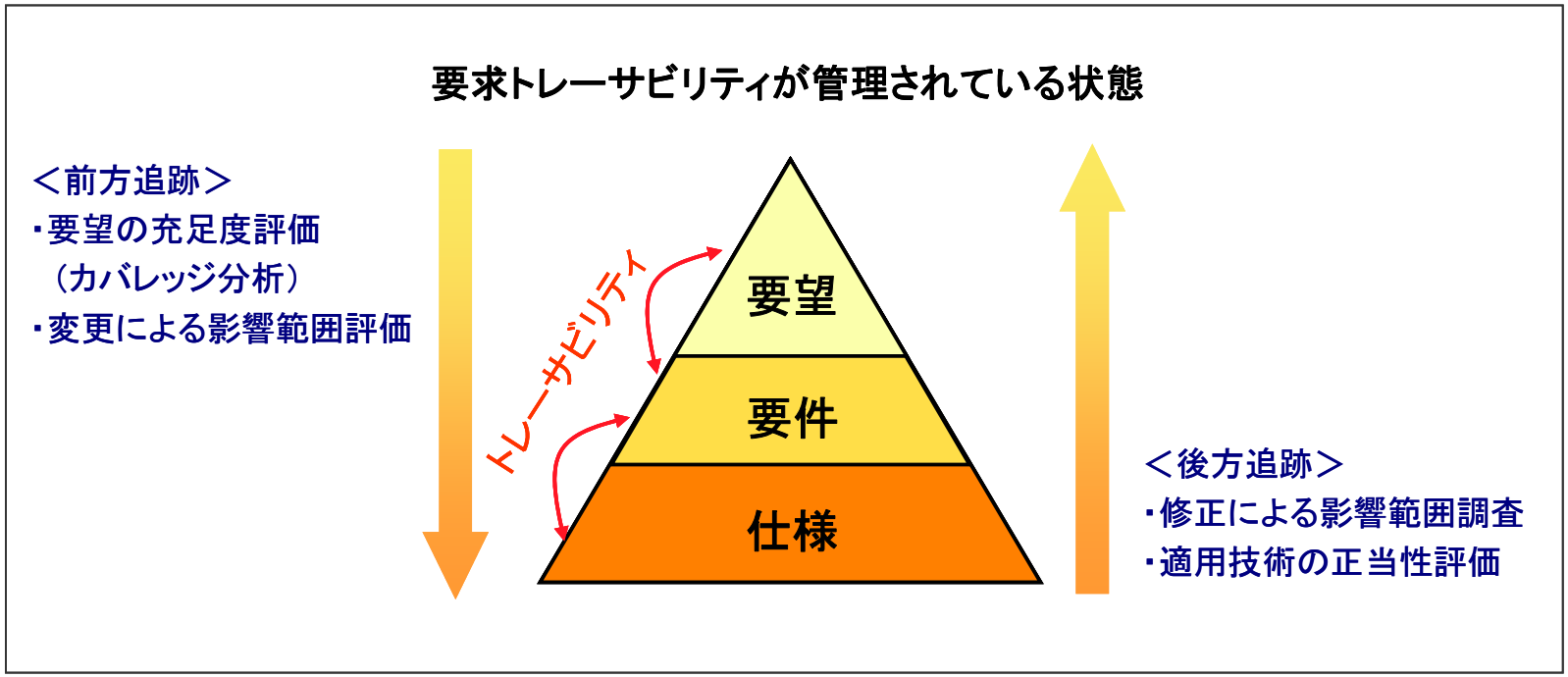
課題3：要求の論理検証

- 典型的な事象
  - 要求の充足度が確認できない
  - 変更による影響範囲が特定できない

- 起こりうる問題
  - 要求の漏れに気づかないままプロジェクトが進んでしまう
  - 変更による検証コストが高くつく

◆技法3:トレーサビリティの活用

- 要求の要素間にトレーサビリティを与える
  - “カバレッジ分析”による完全性の検証が可能となる
  - 要求の変更や修正に伴う”影響分析”が容易になる



運用上のポイント:トレーサビリティの活用

- ✓ 要求の収集段階からトレーサビリティを付与する
  - 後で付与するのは困難
- ✓ 要求パッケージと併用するのが効果的
  - カバレッジ分析では、異なる抽象度間でトレーサビリティが存在することが前提条件

課題4:要求の暗黙知

典型的な事象

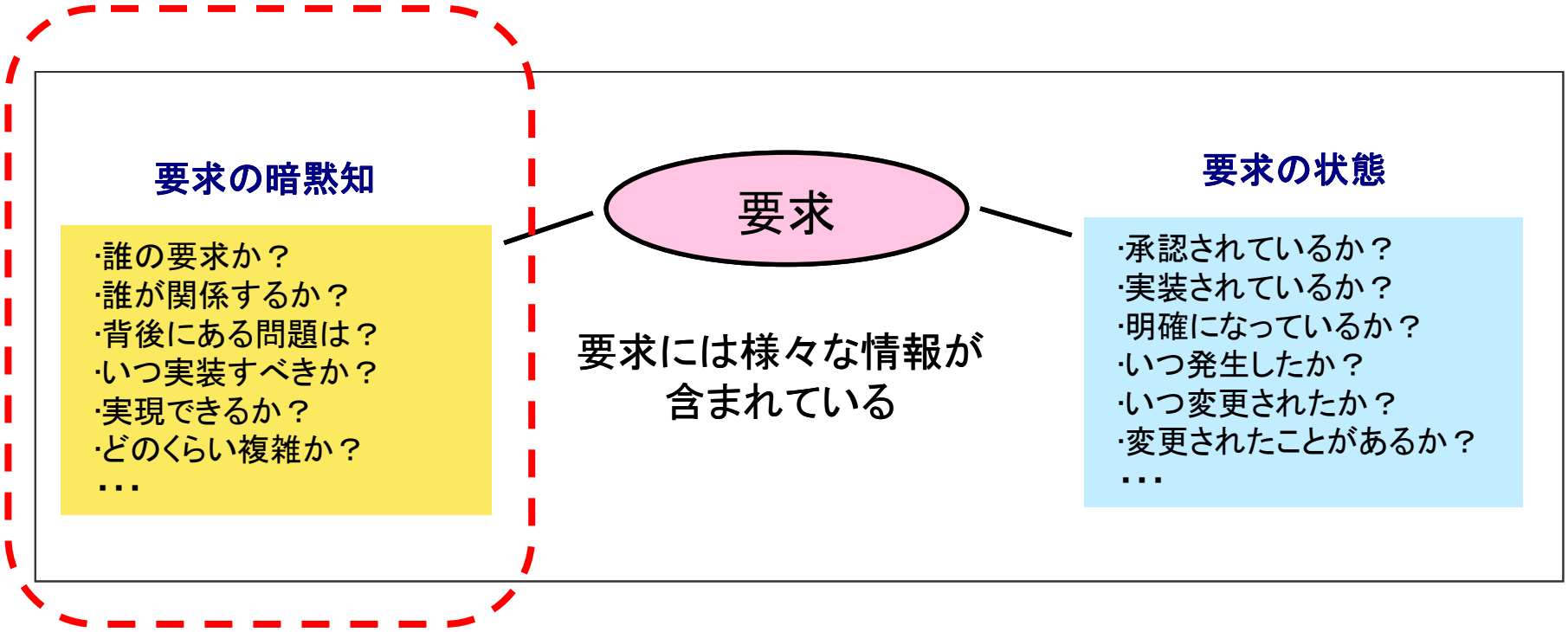
- 要求に対する関係者の認識が埋もれている
  - 誰の要求か？ 重要度は？ 実現可能か？
- 要求に付随する情報が残らない
  - 要求の本文以外にも伝えたいことはあったはず

起こりうる問題

- 客観的な評価ができない
- 要求の整理ができない

◆技法4:要求属性の活用

- 個々の要求に暗黙知を表現できる属性を与える
  - 優先度や難易度を客観的に評価できる
  - 様々な角度で要求を整理できる



運用上のポイント：要求属性の活用

- ✓ 要求の定義と同時に属性を付与する
  - 識別された時点の意図や情報を残す
- ✓ 管理すべき要求属性を決めておく
  - 「優先度」
    - 要求の相対的な重要度・必要タイミング
  - 「難易度」
    - 要求を実現する際の技術的リスク
  - 「コスト（作業量）」
    - 要求の規模・複雑性
  - 「要求安定性」
    - 要求が変化する可能性

課題5：要求の状態

典型的な事象

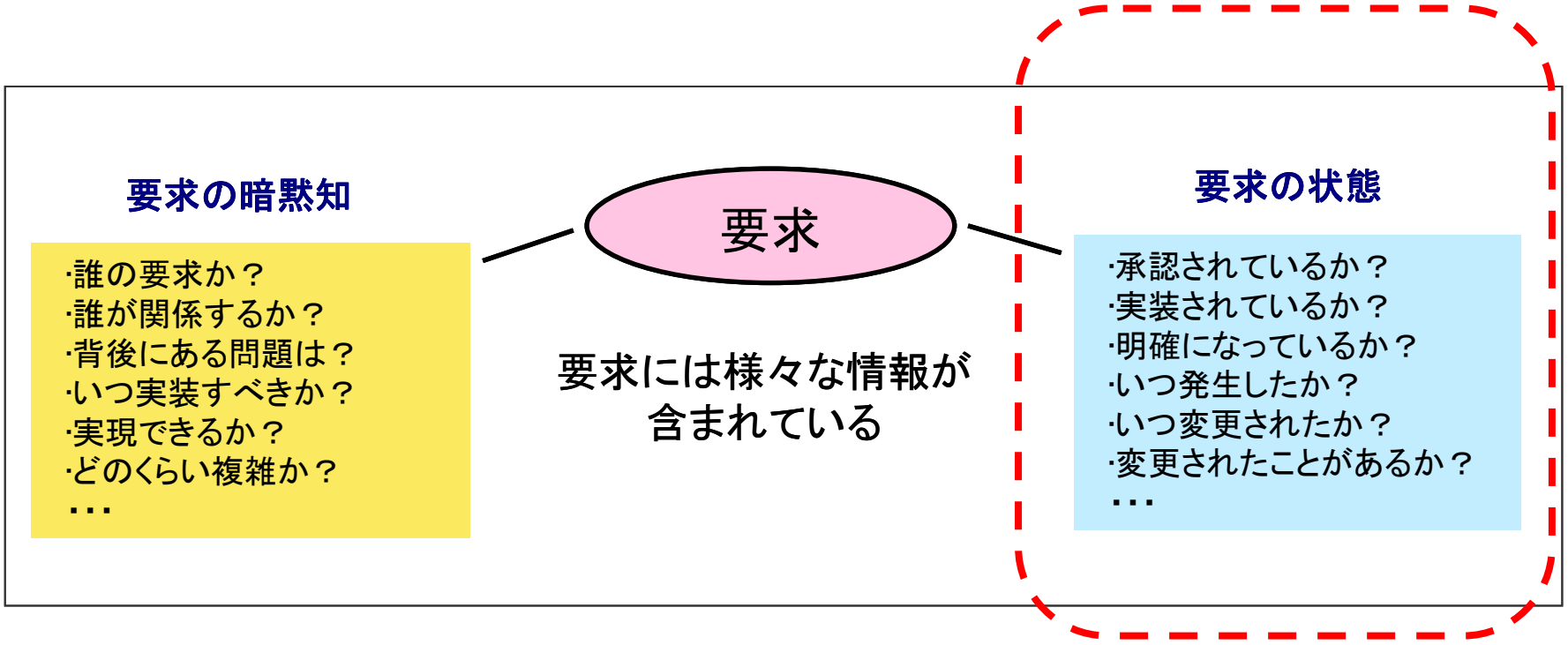
- 要求がどう扱われているか不明
- 古い情報を参照してしまう

起こりうる問題

- プロジェクトの進捗を把握できなくなる
  - どの要求が実現されているのか？
- 単純ミスを誘発してしまう
  - 異なるバージョン・リビジョンの要求に基づく実装

◆技法5：要求のステータス管理

- 要求に状態を表す属性を与える
  - 要求と成果物の整合性を確保する
  - 要求からプロジェクトの状況を正しく把握できる



運用上のポイント：要求のステータス管理

- ✓どのような”状態”が必要かを事前に定義する
  - プロジェクト運営ルール（承認など）との関係を考慮
- ✓システマティックな運用を行う
  - 運用を継続できるよう、手間がかからない方法で
  - 典型的なステータス
    - 「状態」
      - 要求がプロジェクトでどう扱われているかを表現
    - 「バージョン」
      - 製品のバージョンとの関連づけに利用
    - 「リビジョン」
      - 個々の要素の履歴管理に利用

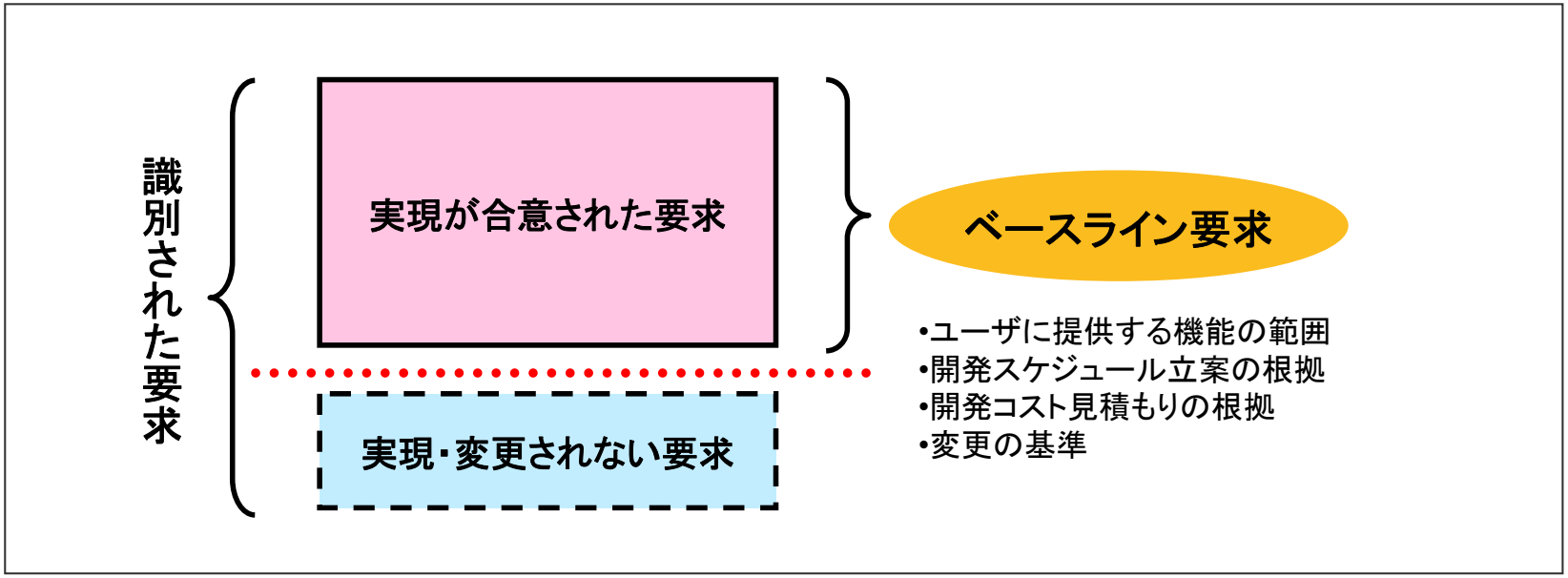
課題6：要求のスコープ

- 典型的な事象
  - 開発範囲についての認識がずれている
  - 何を変更とするかの基準が無い
- 起こりうる問題
  - プロジェクトの基準が曖昧となる
    - ユーザに提供する機能
    - スケジュール、コスト
    - 変更の基準
  - ベンダーや開発チームとの関係悪化

◆技法6：要求のベースライン管理

- 実現対象の要求セットを「ベースライン要求」として識別する
  - 何を設計・実装すべきかの基準が明確になる
  - 変更可否の判断基準が定義される

→プロジェクトのあらゆる基準を提供



運用上のポイント：要求のベースライン管理

- ✓ ベースラインを可視化し、関係者の合意を得る
  - 形だけのベースラインにならないように
- ✓ 開発ライフサイクル中、継続的にベースラインを監視（保守）する
  - 開発中
  - リリース後

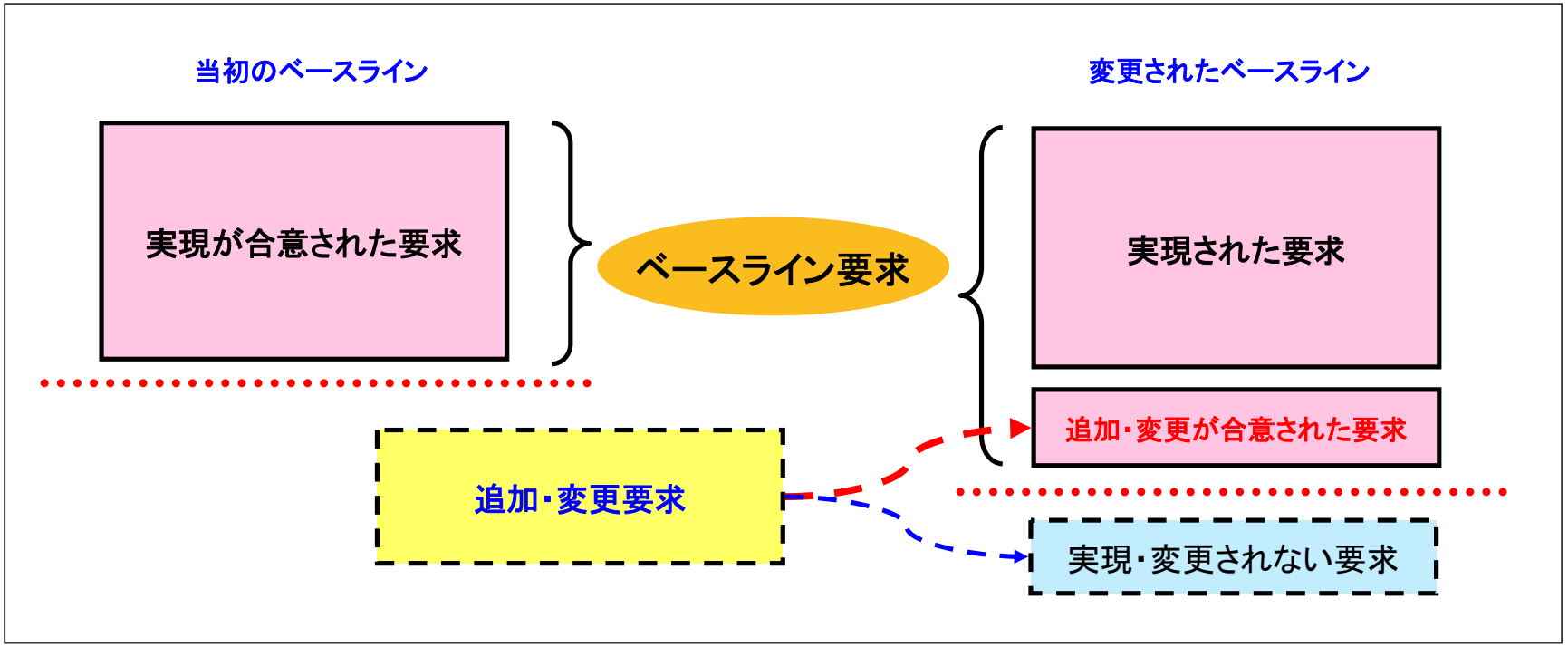
課題7：変更への対応

- 典型的な現象
  - 変更かどうかを区別できない
  - 何が変更されたか残されていない
- 起こりうる問題
  - 際限なく変更を受け入れてしまう
  - ベースラインが形骸化する

◆技法7：変更要求管理

■ 変更要求を区別して扱う

- ルールに基づく変更管理をサポートする
- ベースラインの管理を維持できる



運用上のポイント：変更要求管理

- ✓ 変更管理のルールを運用できる環境
  - 合意された要求は自由に変更できない環境
  - 決められたルート・手続きで変更が処理される環境
- ✓ ベースライン管理・プロジェクト管理との併用
  - 変更前後の差分が識別できるように
    - 要求の差分
    - コスト・スケジュールの差分

要求管理の実践に向けて

- 要求の持つ特性を理解する
  - 要求工学
- プラクティスを少しずつ取り入れる
  - 7つの技法から
- 環境を整備する
  - ツールの活用
  - 運用プロセスの定義

→プロジェクトでの実践と改善を繰り返す

Software Process Engineering Inc.

www.spei.jp