OLYMPUS

Your Vision, Our Future

プロジェクトをゴールの見えないマラソンとしないために

2006年10月13日

オリンパス株式会社　IT改革推進部
岩見　好博

Opto Digital Technology

問題提起

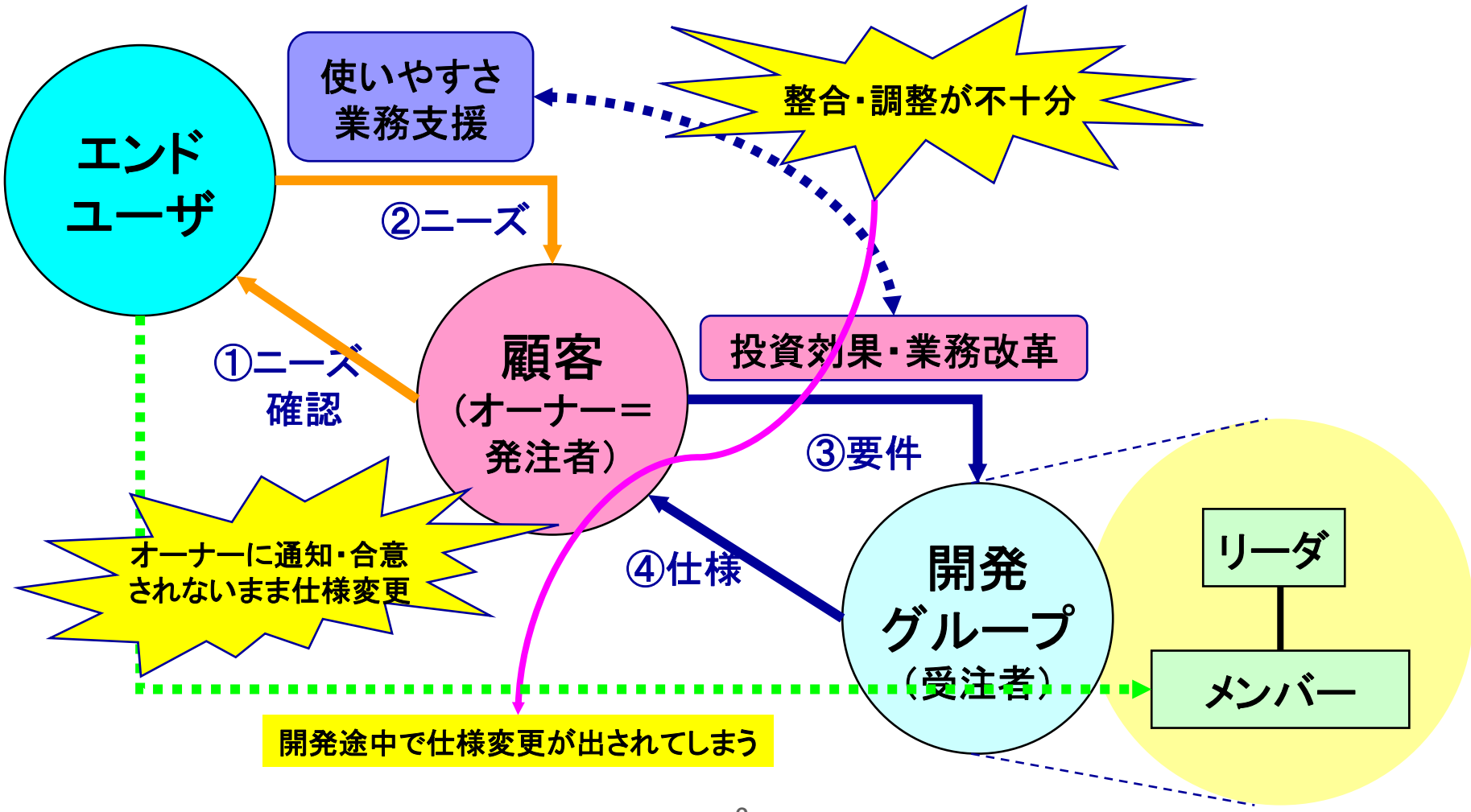
- あるプロマネの言葉
  - 「ソフトウェア開発プロジェクトは、まるでゴールの見えないマラソンを走っているみたいだ。ゴールに近づいたと思ったらゴールが逃げていく、あるいは道が曲がりくねっている」

- デスマーチ・プロジェクトの特徴（症状）
  - 要件の確定が遅れたため見切り発車したが、あとで修正に追われる羽目に
  - 納期を延ばすたびに、なぜか要件が増えていく
  - テストをやればやるほど不具合が見つかり、収束しない
  - 現場に「笑い」がない

1. 要件確定遅れで見切り発車、後で修正に追われる

原因	対処法
アサインされた要員を遊ばすわけにいかないので、設計に手をつけてしまう	●要員の配属時期を適正化する。そのため、仮の規模見積り（TSPでのConceptual Design）を行って初期計画を作り、これをベースに交渉する
仮置き仕様の確度が低い Fast Trackingには相当のスキルが要る	●要件の合意がない以上、仮置き仕様は確定に伴って当然変更しなければならなくなる。まず合意を得た要件の仕様化から段階的に始める
利用部門、顧客が要件確定での役割や責任をよく理解していない また、理解させる努力が足らない	●確定時期が遅れることによるスケジュール全体への影響をはっきり言う。これにも、前述の初期計画が役立つ

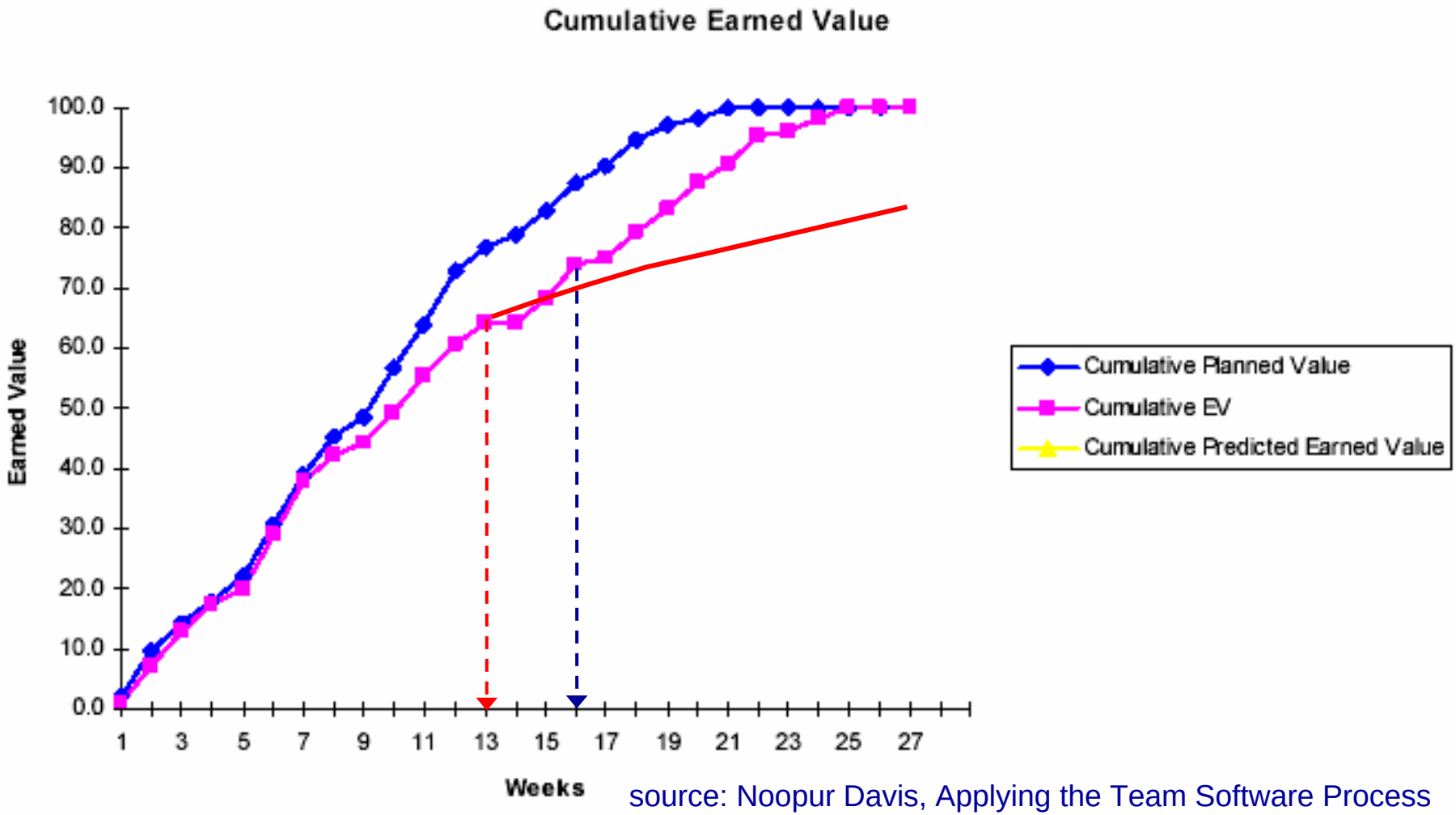
1. 要件確定の難しさ



2. 納期を延ばすたびに、なぜか要件が増えていく

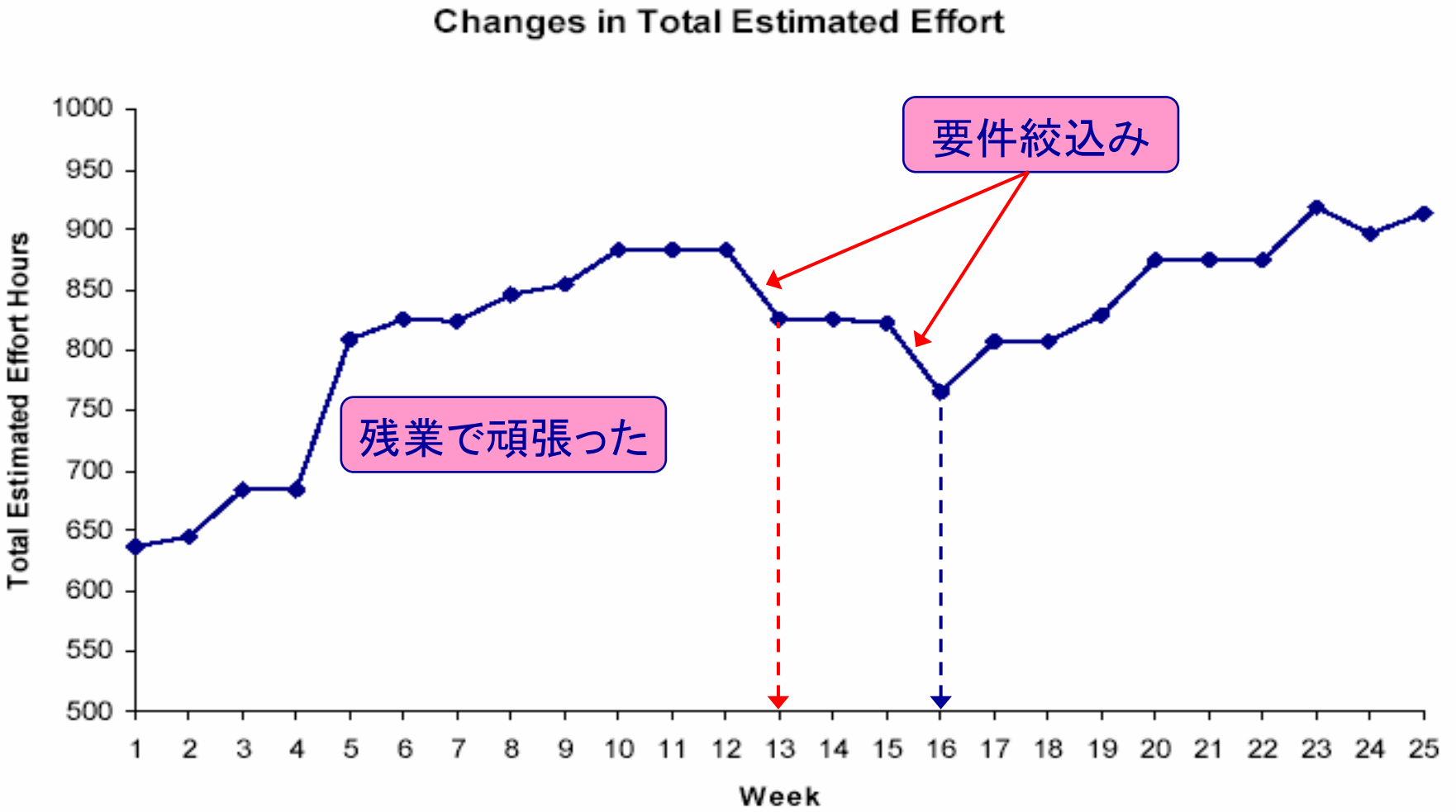
原因	対処法
日々の進捗が把握できず、納期間際になるまで遅れが表面化しない。納期間際に「遅れます」と言う羽目に	●開発者自身に作業計画を作ってもらい日々の進捗を把握する ●少なくとも週次で、進捗をステークホルダーに知らせる その際、遅れがあればその対処策を必ず書く
その結果、納期延期の見返りに、新規要件を受け入れてしまう。ただでさえ、厳しい工数がますます足りなくなる。最悪の場合、この連続となって開発現場のやる気がなくなり、製品そのものが完成しないことも	●要件を初めから絞り込んでおく ●顧客に優先順位を必ず決めてもらう （優先度の低い要件は自動的に延期）

2. Earned Valueによる進捗追跡



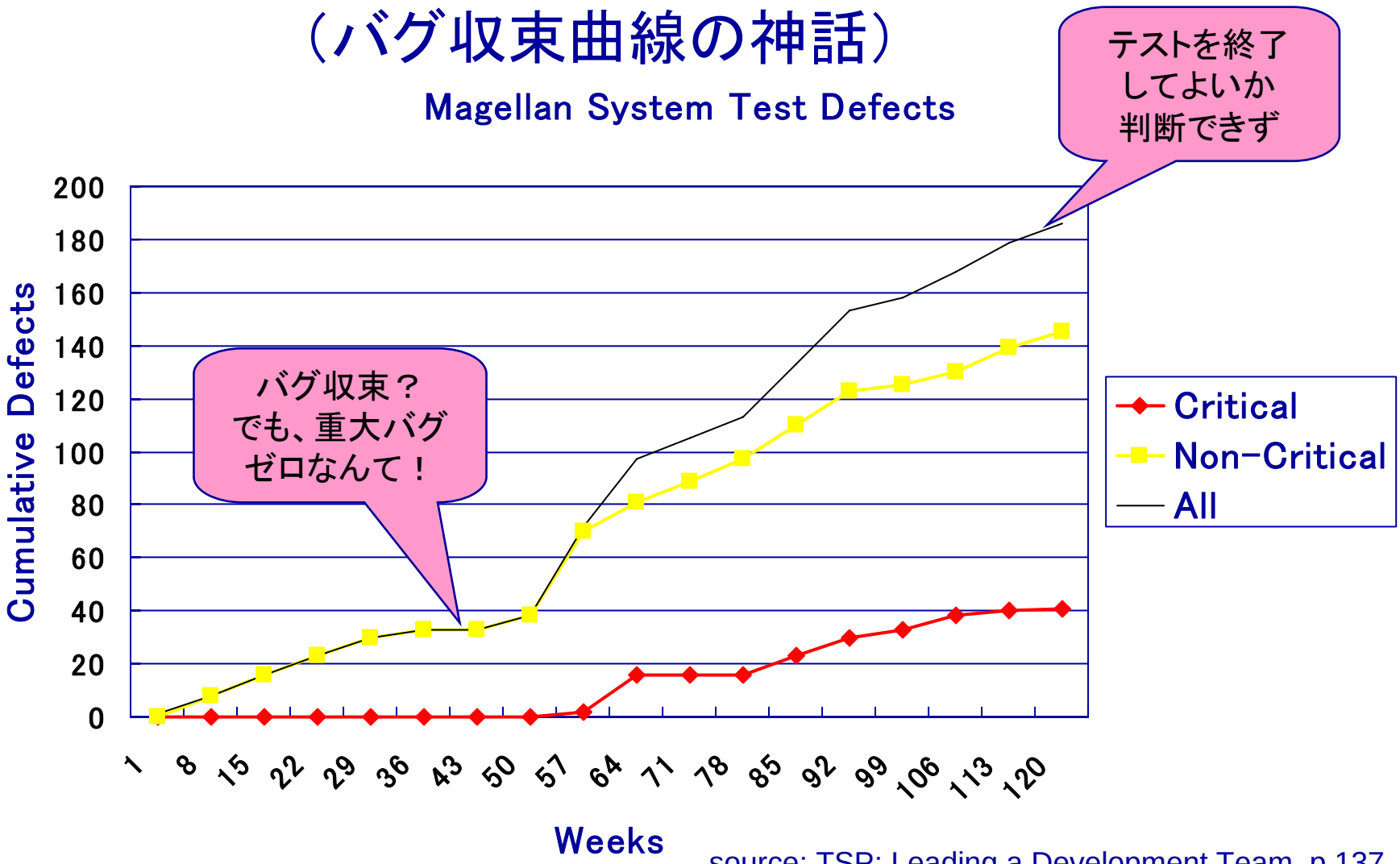
source: Noopur Davis, Applying the Team Software Process

2. 要件の絞り込み例



source: Noopur Davis, Applying the Team Software Process

3. テストをやるほど不具合が見つかり、収束しない（バグ収束曲線の神話）



source: TSP: Leading a Development Team, p.137

3. テストをやるほど不具合が見つかり、収束しない

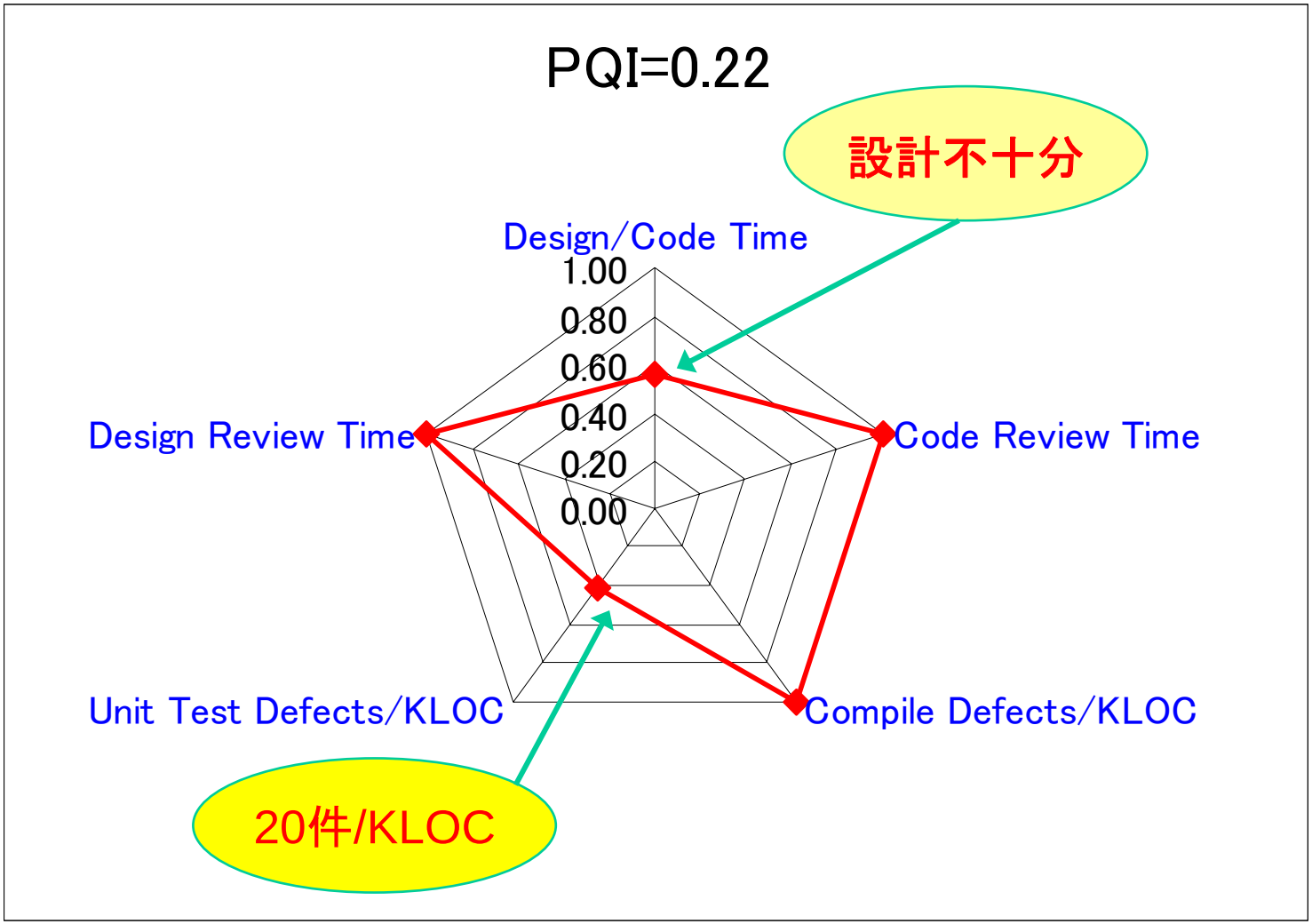
原因	対処法
設計せずにいきなりコーディングとの開発プラクティスのため、ソフトウェアの品質が悪すぎる	●適切な設計レビュー、コードレビューのやり方を教育、指導する。ただ、「レビューをやれ」では逆効果に ●レビュー工程でも欠陥記録を取る。テスト工程から記録する組織が意外に多い
決められた設計レビュー、コードレビューをしていない。あるいはやっていても形骸化していて効果がない	●品質目標を決める。ただしレビュー指摘件数N件以上との目標は誤った行動（例えば、些細な指摘で件数を増やす）を引き起こすので要注意 ●設計時間とレビュー時間の比率とか、尺度を工夫する（PSPのPQI）

3. Process Quality Index (PQI)

- PQI判定基準：0.0 ～ 1.0の範囲
  - 設計／コーディング時間 = 設計時間／コーディング時間
  - 設計レビュー時間 = 2＊設計レビュー時間／設計時間
  - コーディングレビュー時間 =
    2＊コーディングレビュー時間／コーディング時間
  - コンパイル欠陥数／ KLOC = 20／（10＋コンパイル欠陥数／ KLOC）
  - ユニットテスト欠陥数／ KLOC = 10／（5＋ユニットテスト欠陥数／ KLOC）
- PQI は上記5つの値の積である
- PQIの値が0.4以上のプログラムは、ユニットテスト以降欠陥が見つからなかった、と来歴データは示している

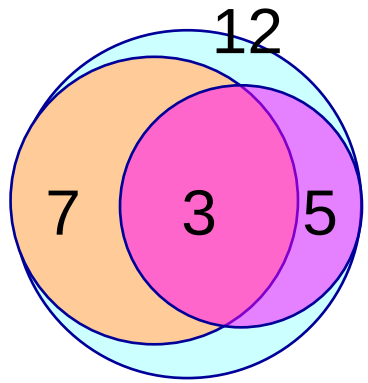
Source: PSP: A Self-Improvement Process for Software Engineers
by W.S. Humphrey, p.150-151

3. PQIの例



3. 残存欠陥数の推定

- テストフェーズに入ると残存欠陥数を推定できない
- 複数メンバーで設計レビュー、コードレビューを行うと推定できる（標本再捕法）
  - レビューアAが見つけた欠陥　　A: 7
    レビューアBが見つけた欠陥　　B: 5
    両方が見つけた欠陥　　C: 3
  - 残存欠陥数　　= (A × B)/C-(A+B-C)
    = (7 × 5)/3-(7+5-3)　=3



- 前提条件
  - 全レビューアが全体をレビューすること
  - 欠陥発見数がそれなりに多いこと
  - レビューアの欠陥摘出率が70%以上であること

まとめ

- デスマーチ・プロジェクトにしないためには
  - 要件の合意がない限り、設計に着手しない
  - 要件確定での利用部門、顧客の役割を理解してもらう
  - 要件の優先順位づけ
  - 確実な進捗管理と、ステークホルダーへの定期的報告
  - 設計作業に十分時間をかける（コーディングで設計しない）
  - レビューで欠陥をできるだけ前工程で除去する
  - プロセス品質を管理する

ご清聴ありがとうございました