

組込みソフトウェア開発における 品質向上の勧め **[バグ管理手法編]**

独立行政法人情報処理推進機構 技術本部
ソフトウェア・エンジニアリング・センター

紹介者： 柏 祐太郎

- 石尾 渉 【オムロン オートモーティブエレクトロニクス株式会社】
- 植武 信弘 【株式会社日立情報制御ソリューションズ】
- 岡本 孝之 【株式会社富士通ソフトウェアテクノロジーズ】
- 千葉 正史 【三菱電機株式会社】
- 十山 圭介 【IPA/SEC（株式会社日立製作所）】
- 西野 敦士 【ルネサス エレクトロニクス株式会社】
- 三橋 二彩子 【日本電気株式会社】
- 三原 幸博 【IPA/SEC 組込み系プロジェクト（アルパイン株式会社）】
- 宮本 貴之 【一般社団法人 TERAS（キャッツ株式会社）】
- 村山 耕一 【イーソル株式会社】
- 山上 宣彦 【東芝ソリューション株式会社】



1. バグ管理と本書について
2. バグに関連する用語について
3. バグ管理プロセス
4. バグ管理内容と管理項目
5. バグカウントの指針
6. バグの分析

Column:

大規模開発における留意点

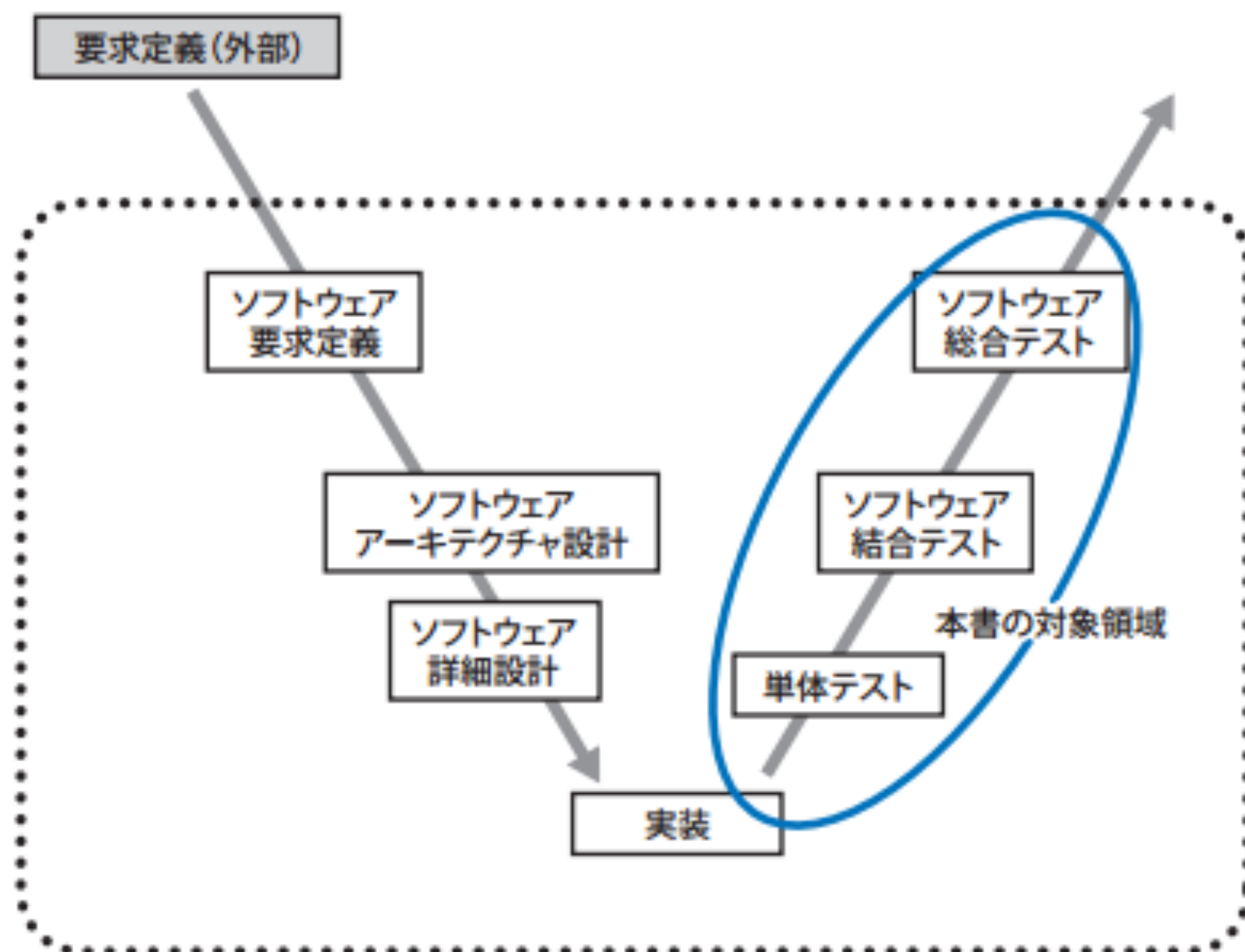
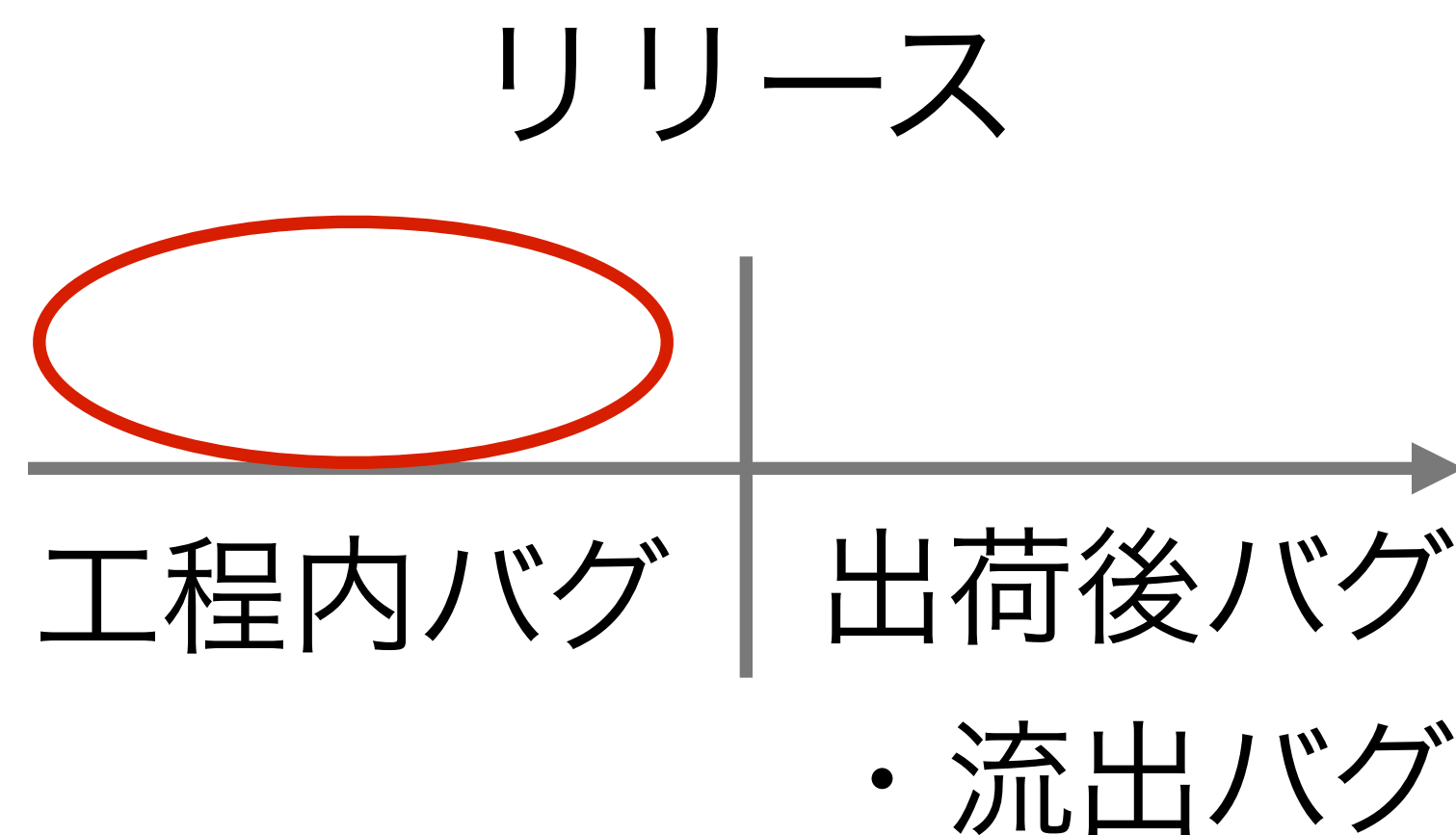
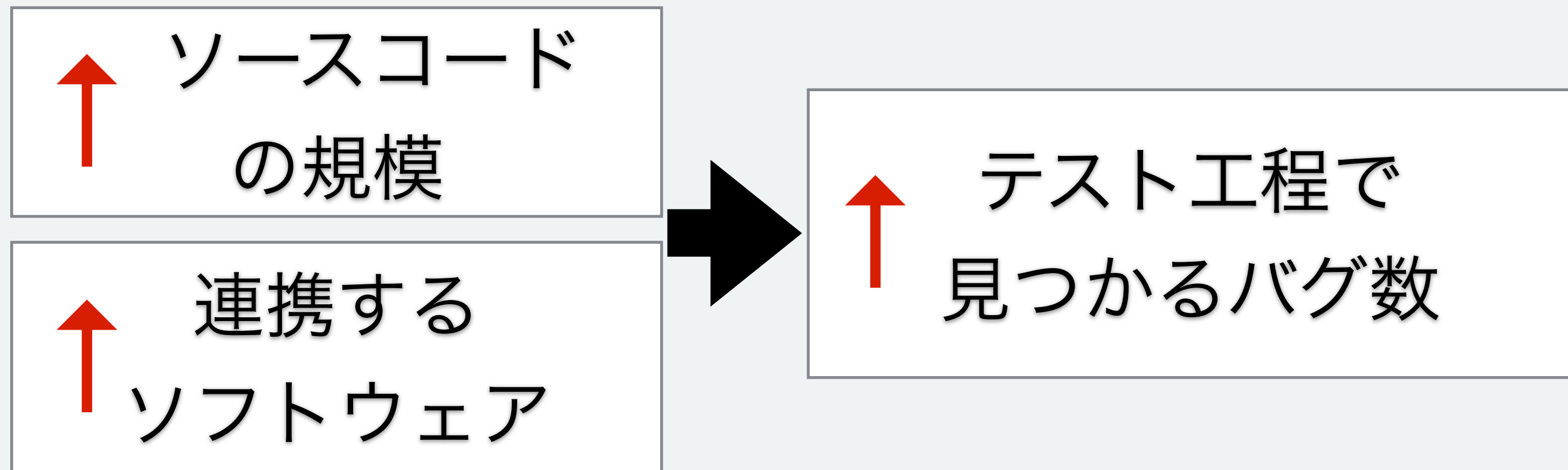


図 1.1 ソフトウェア開発のV字モデルと本書の対象領域



バグ管理の背景と目的



■ 目的

- バグの修正
- 製品リリース時に残存するバグの有無の把握
- バグの検出状況によるソフトウェア品質の推定
- バグの分析によるソフトウェア開発のカイゼン



1. バグ管理と本書について
2. バグに関連する用語について
3. バグ管理プロセス
4. バグ管理内容と管理項目
5. バグカウントの指針
- 6. バグの分析**

Column:

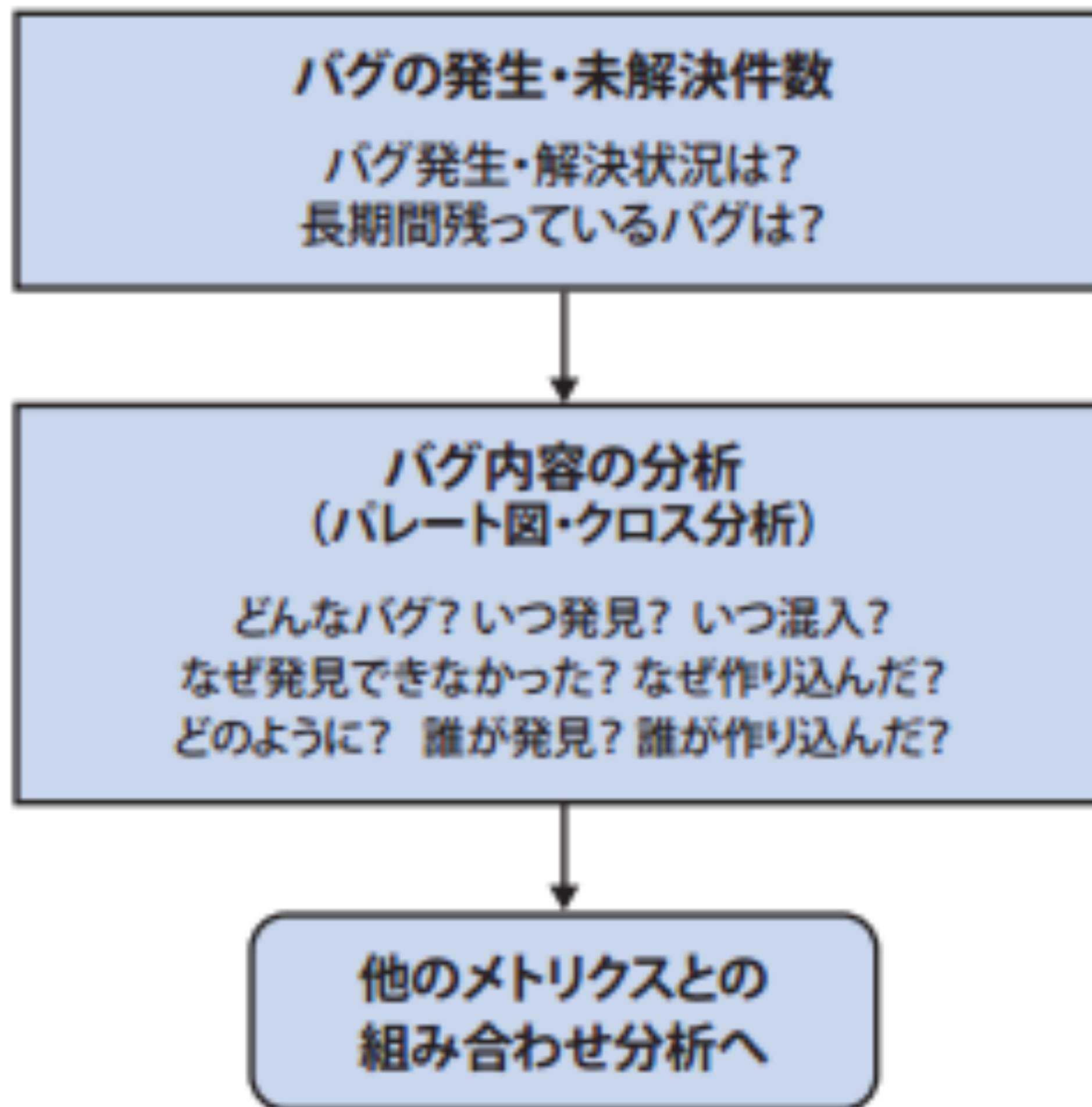
大規模開発における留意点

■ バグの検出状況によるソフトウェア品質の推定

- 現工程作業の制御
 - テスト計画の見直し
 - 前工程の再実施
- 後工程・最終品質の予測

■ バグの分析によるソフトウェア開発のカイゼン

- 振り返りによる問題の特定と対策
 - 作り込み要因
 - 前工程で発見できなかった要因

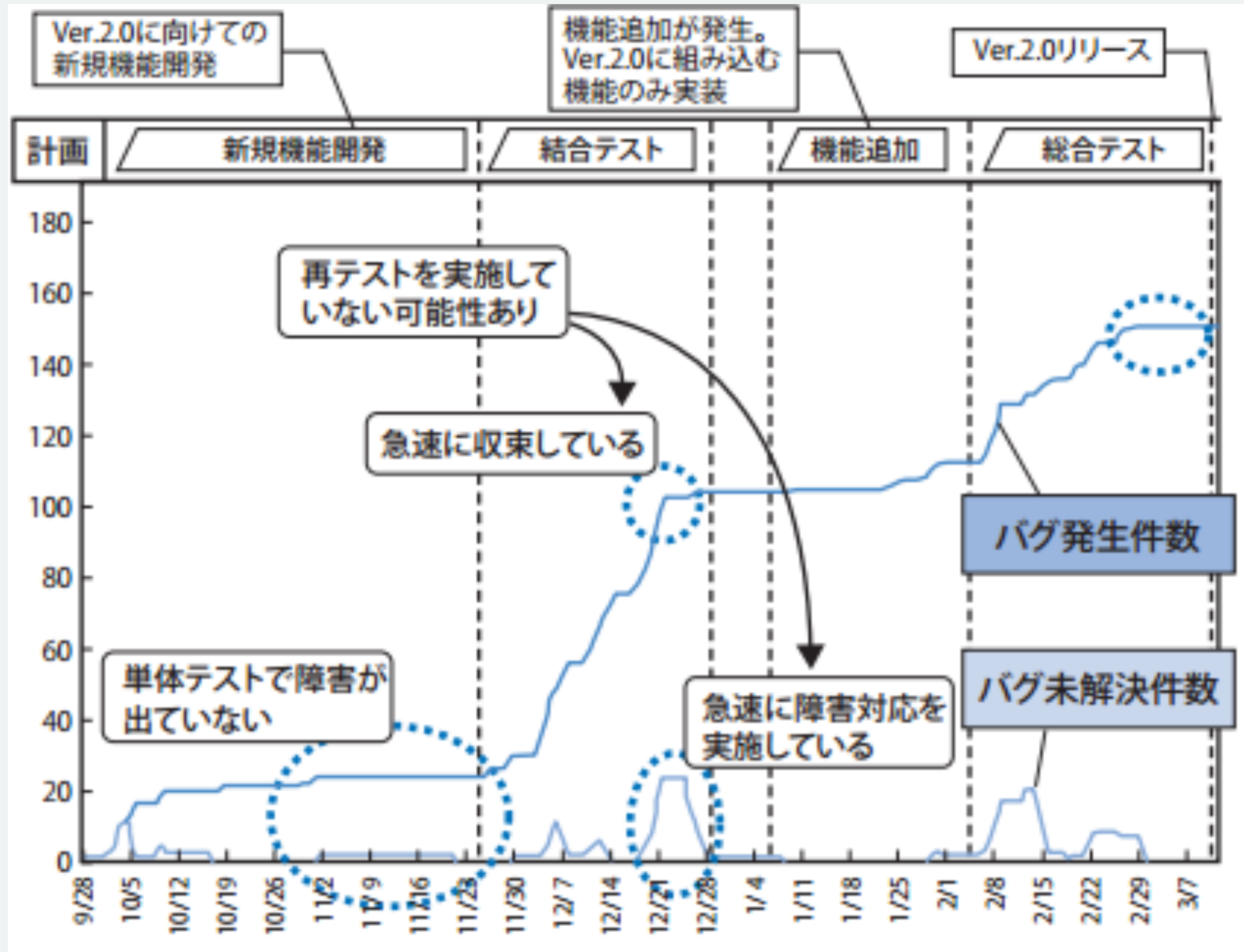


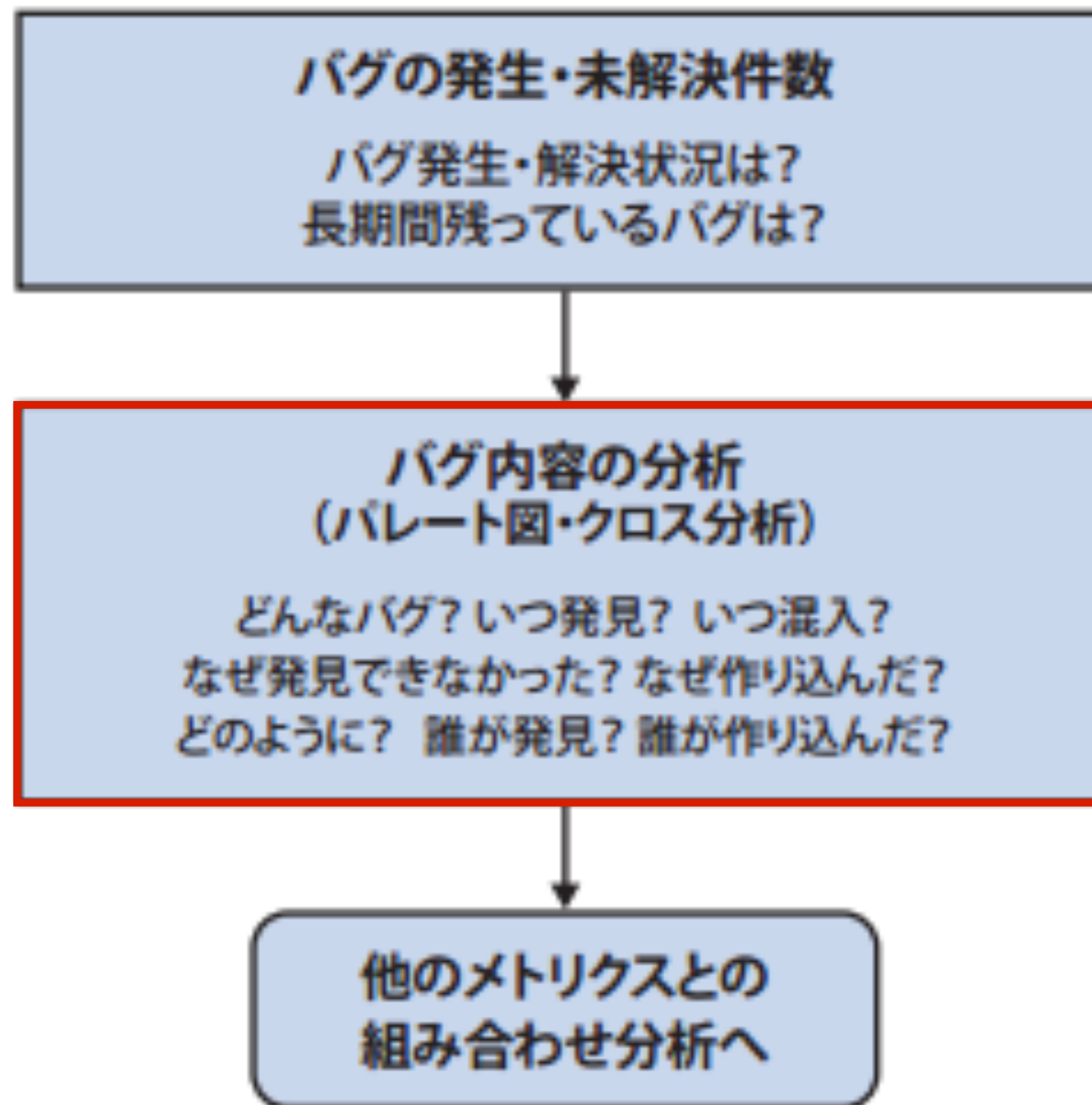
■ システム全体の分析

- 全体の推移状況から、**変化点**（急激な変化や停滞を起こしている点）を見つけ出します
- 変化点を見つける際には、工程・マイルストーンを考慮することで見つけやすくなります
 - 「テスト工程の終了間際で多くのバグが発生」
 - 「テスト工程の終了間際にもかかわらず未解決のバグが残っている」

バグ発生件数・未解決件数の分析

10

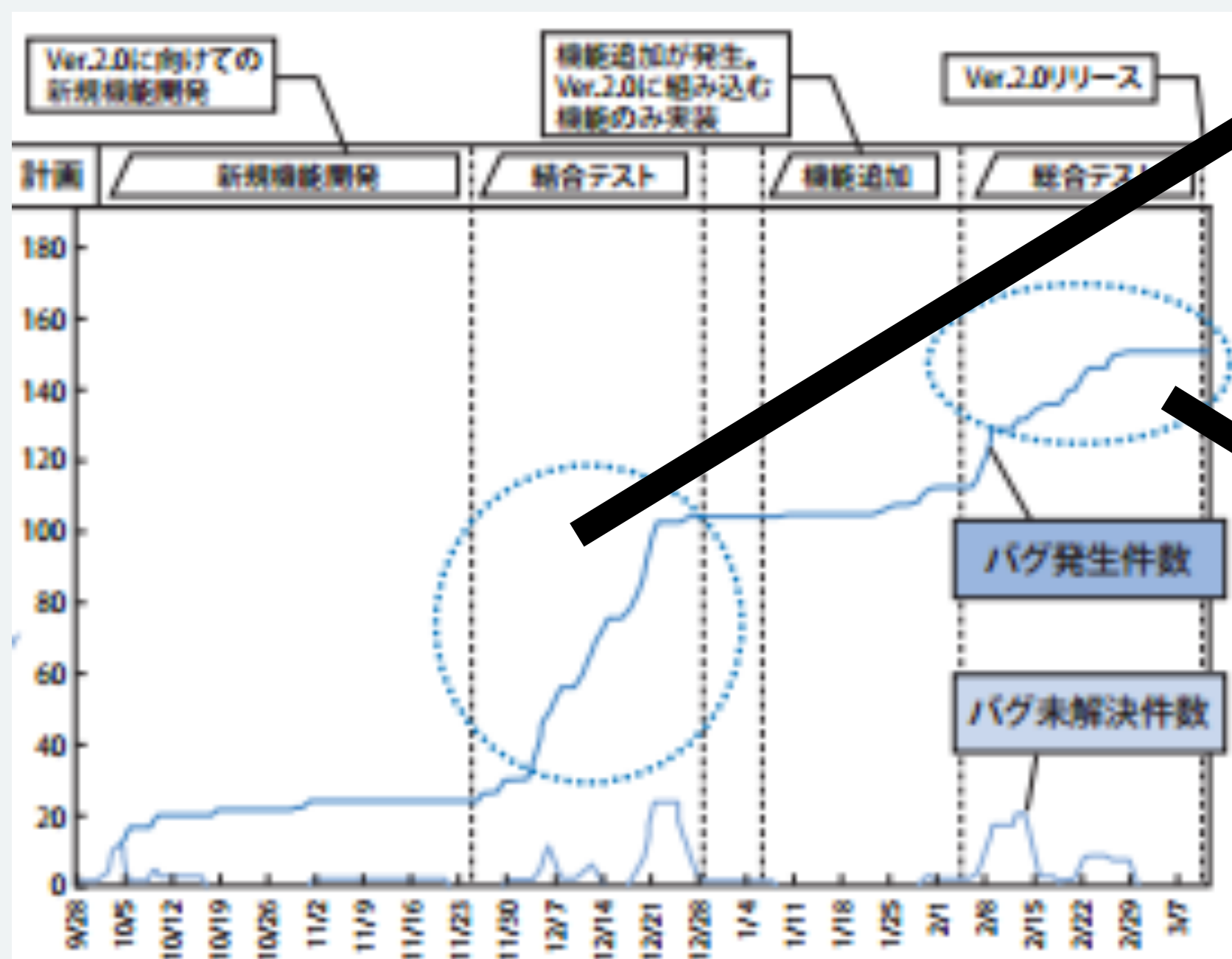




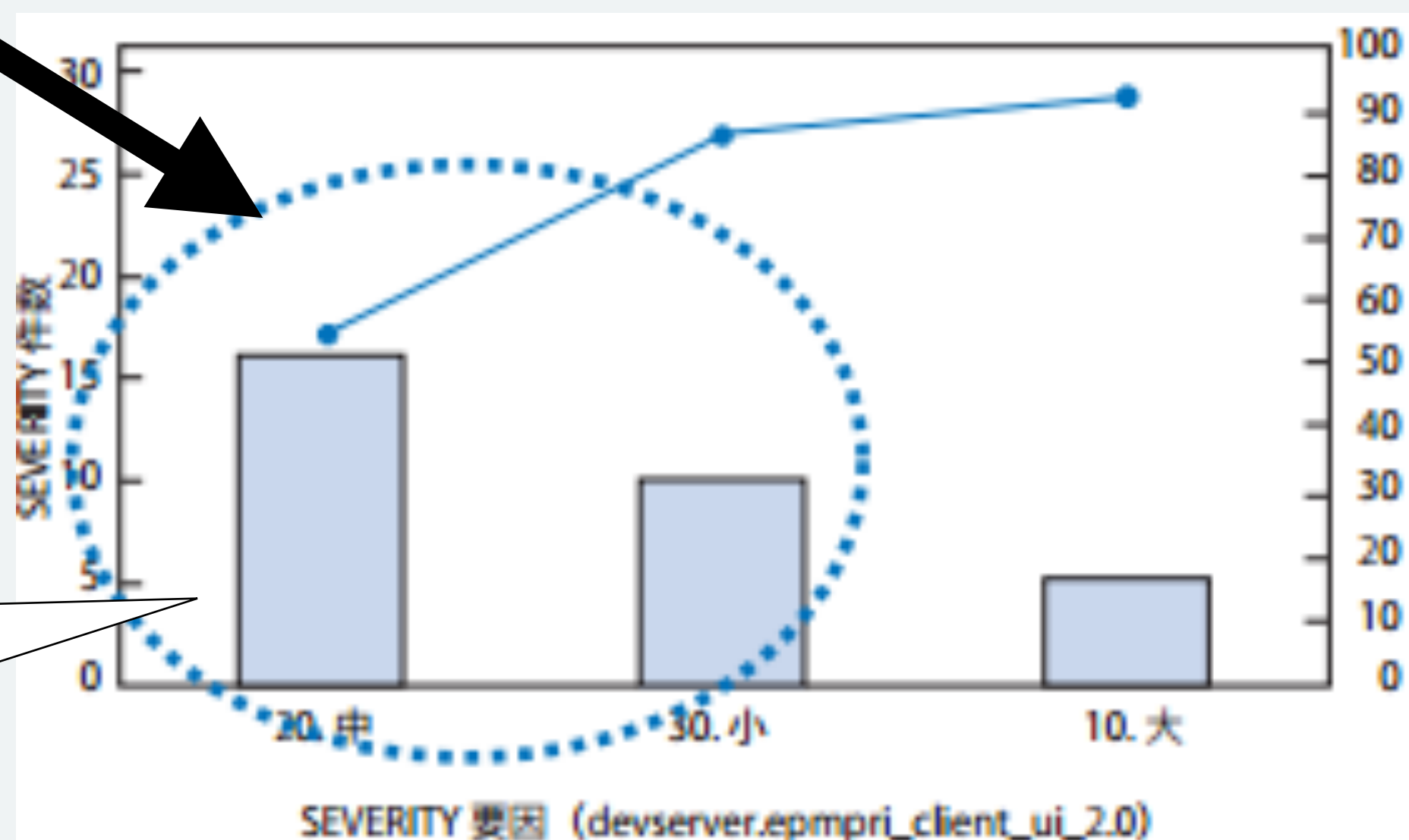
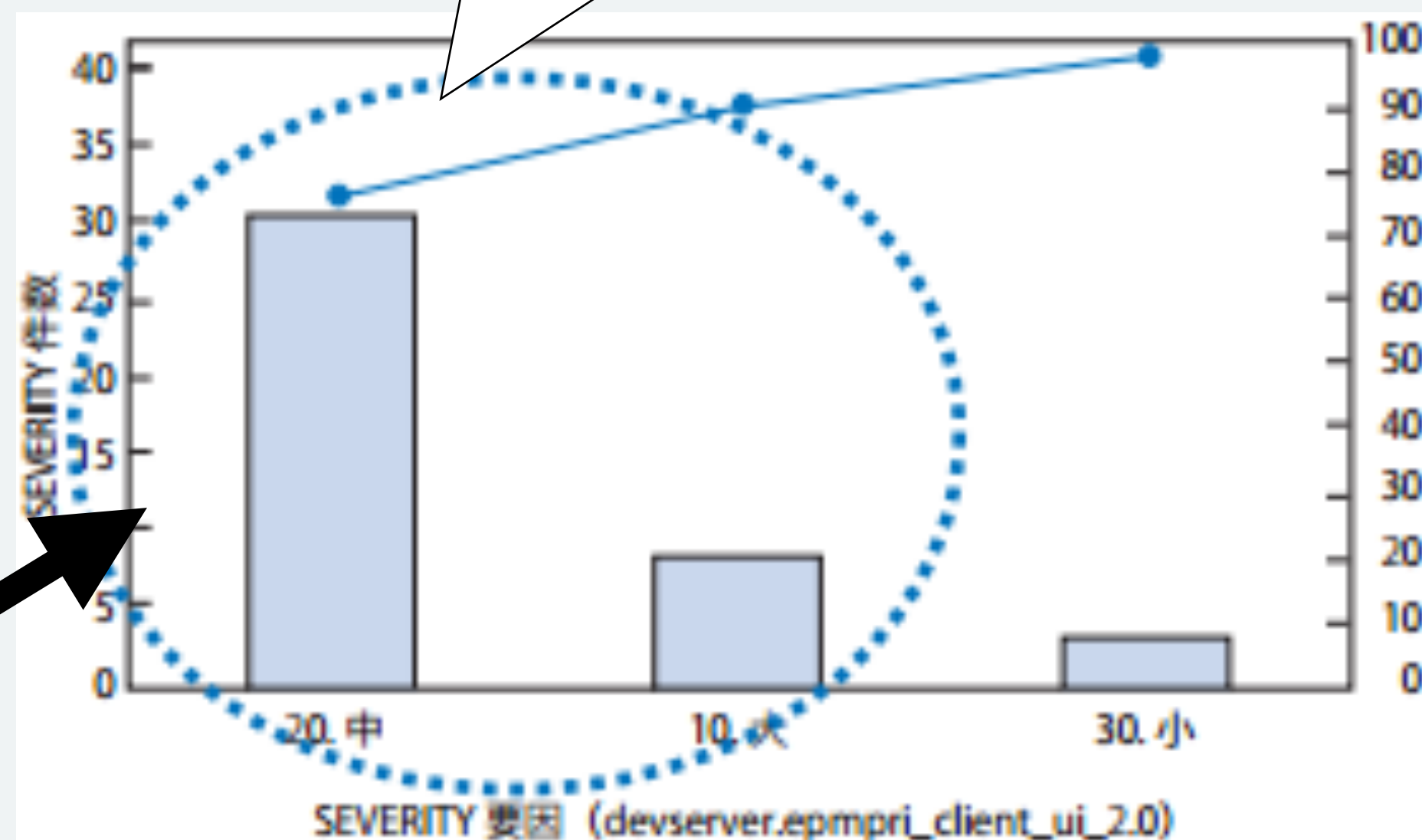
バグ内容の分析

12

■ 重要度の分析



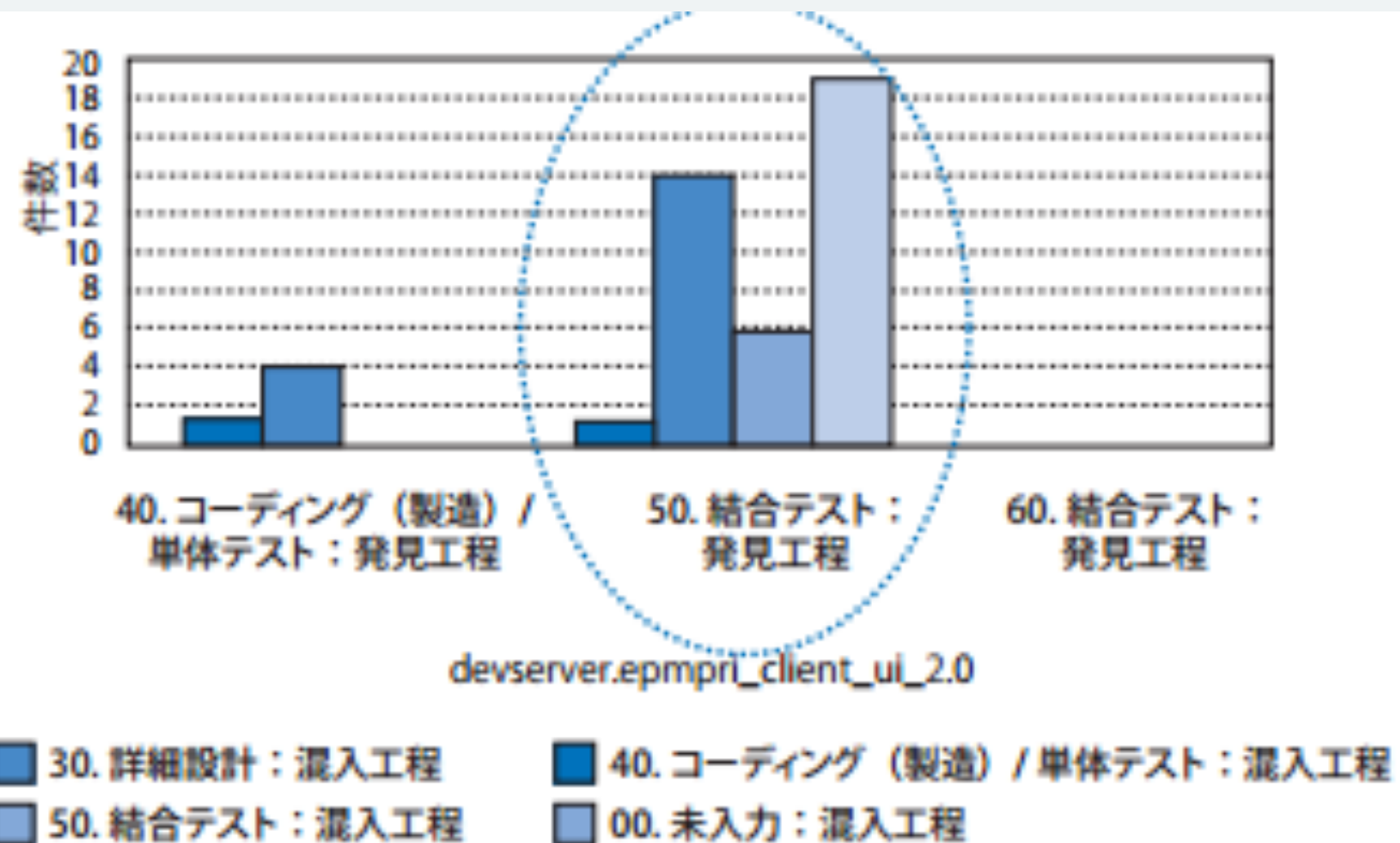
重大なバグが多く発生
ソースコードへの影響も大きい



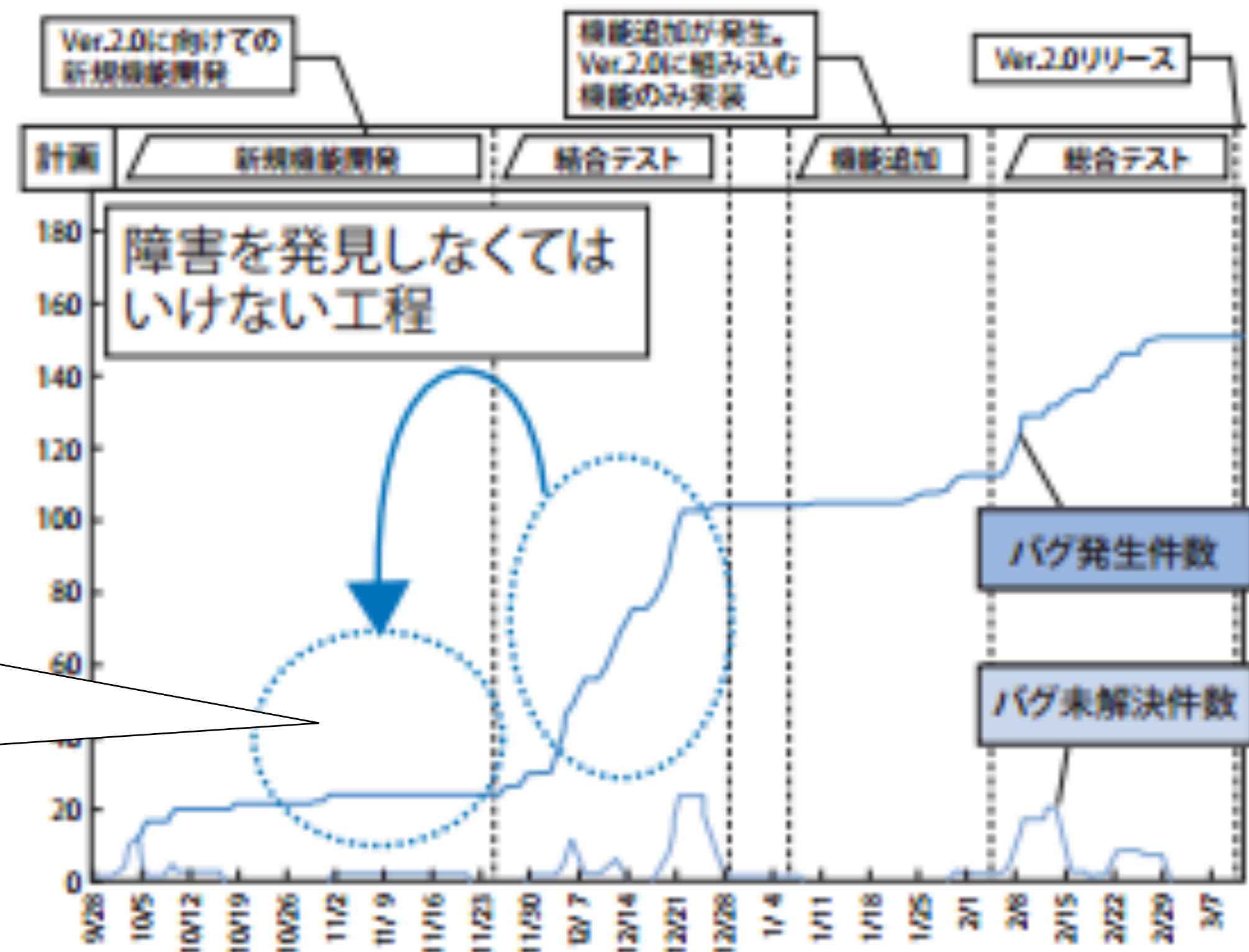
重大なバグは少数だけ発生
ソースコードへの影響は小さい

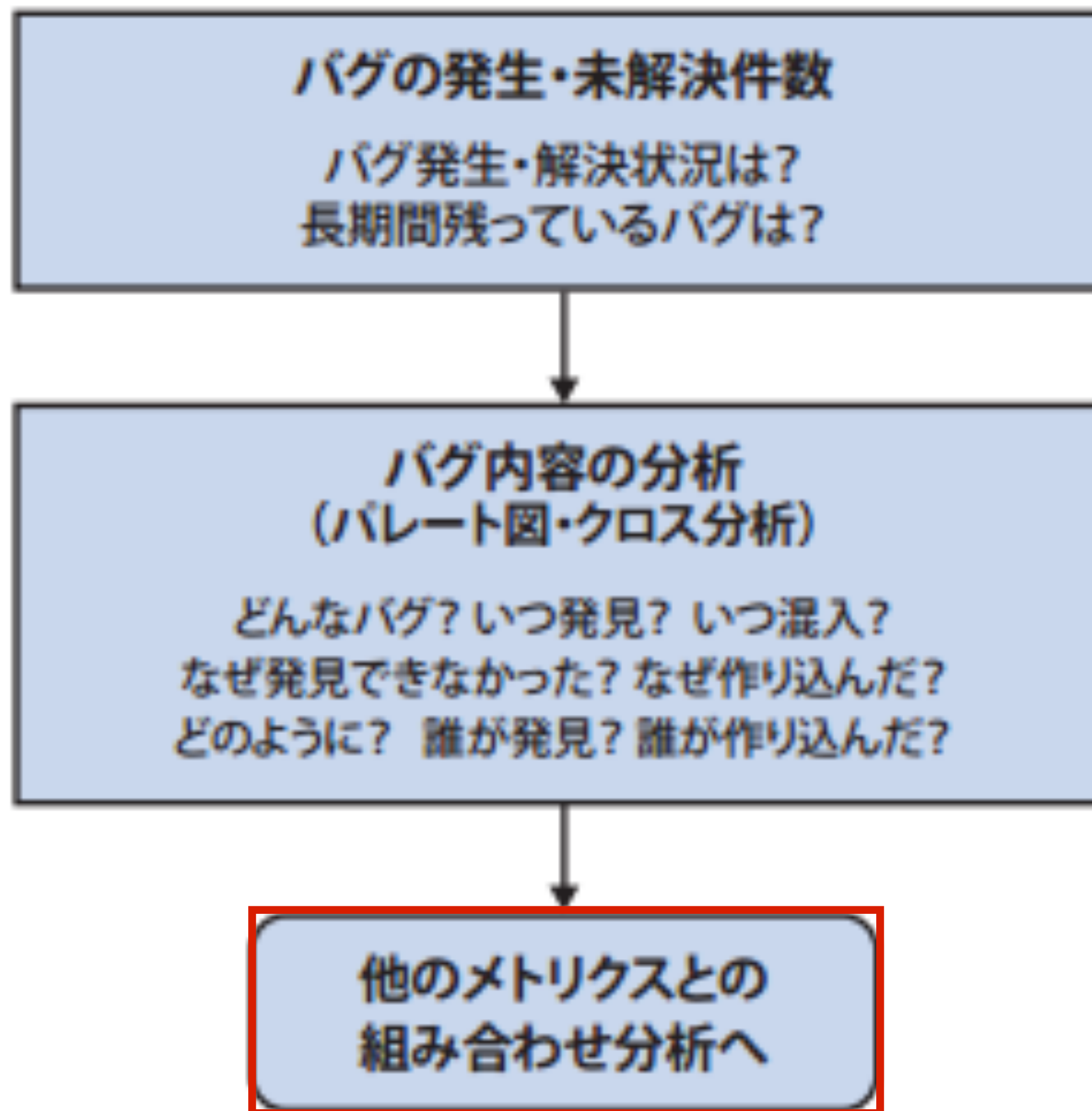
バグの内容の分析

■ 混入工程と発見工程の分析



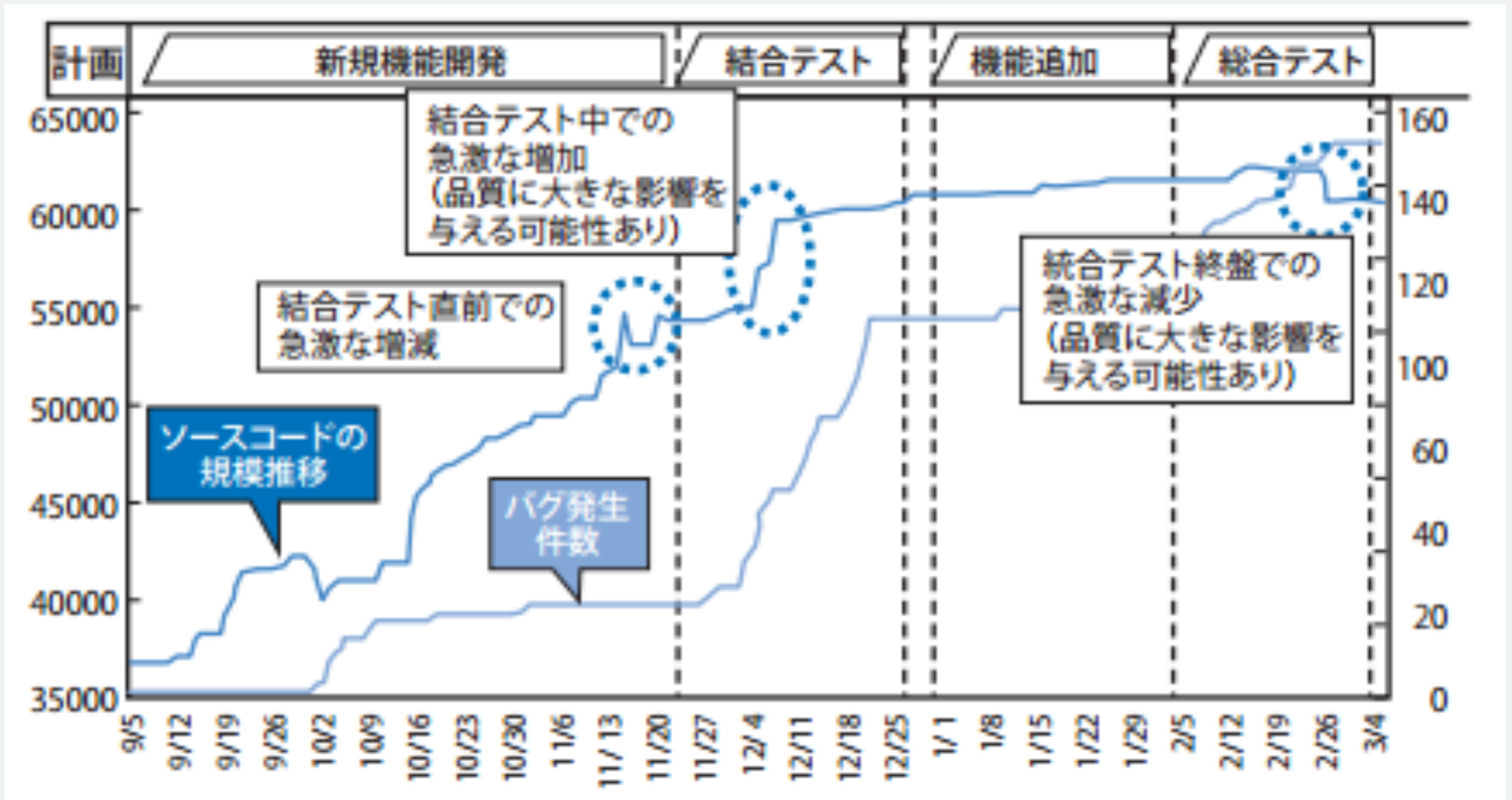
本来は障害の停滞を認知して
この工程を改善する必要あり





バグ発生件数とコード行数の組合せ分析

15





1. バグ管理と本書について
2. バグに関連する用語について
3. バグ管理プロセス
4. バグ管理内容と管理項目
5. バグカウントの指針
6. バグの分析

Column:

大規模開発における留意点

大規模開発における留意点

■ 不具合への対応

- 検出バグの修正においては、お客様も参画する修正審議グループによる審議を行う（トリアージミーティング）

体制：

- テスト担当チーム（機能単位、機能間連携、サブシステム間連携テスト）
- テスト管理者
- 修正審議グループ（開発責任者、運用責任者、お客様）
- 構成管理チーム
- テスト支援チーム（バグ票管理ツールの運用チーム）

大規模開発における留意点

■ 重複バグの発生を抑制する

- 原因究明にかけた貴重な労力が無駄遣いとなる
 - 重複バグがバグ全体の数十パーセントも占めると、納期・開発費に与える影響は致命的

テストにおける対策：

- 複合機能テスト（製品全体のテスト）は、各機能のテストの結果、品質が熟成したと判断できるまで開始しない
- ランダムテストや耐久テスト、実用テストなどは、並行度合いをむやみに上げて行わない

大規模開発における留意点

■ ブロッッキングバグは最優先に集中して駆除する

数万から数十万件に及ぶテスト項目を効率よくこなすためには、リセットやフリーズといった、テストの続行を困難にするバグをいち早く発見して解決する必要がある

解決策：

- 早期に発見するためのテスト戦略
- 最速で解決するための方法

早期に発見するためのテスト戦略

■ 外部機能（利用者から見た機能）から

- これまでの経験に基づいて重大バグが発生しやすいと判明している機能から先にテストする

■ 内部機能（実装するための機能）から

- リソースを扱う（ロックや競合が発生しやすい）部分から先にテストする

■ 変更箇所の多いソフトウェアから

- 差分開発で変更箇所の多いソフトウェアから先にテストする。

最速で解決するための戦略

■ バグのプロファイリング

- バグに重要度（システムに与える影響の度合いでランク付けする）と優先度（処置の優先度をランク付けする）を付けて管理する

■ スワットチームによる支援

- テストと並行して、テストの優先度を高くしたソフトウェアをハイレベルのエンジニアで構成するスワットチームが集中的にレビューする
- 重要度の高いバグの解決に際しては、発見された原因と対策案に対するスワットチームによるレビューを必須とする

SEC BOOKS

組込みソフトウェア開発における 品質向上の勧め [テスト編～事例集～]

独立行政法人情報処理推進機構 技術本部 ソフトウェア・エンジニアリング・センター 編



復習：バグとは

■ 障害 (fault : 誤り)

計算機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の定義

■ 故障 (failure : 不十分, 機能停止)

要求された機能を遂行する製品的能力が尽きる状態

■ 欠陥 (defect : 不足・欠点)

要求や仕様に合致せず不完全であること, または不足していること. 修復か取り替えが必要である状態