2007年10月31日 於：SPI Japan2007

EASE Project

プロジェクト管理支援のための実証データ計測と分析

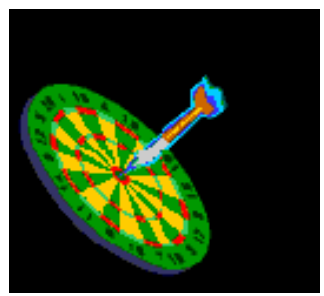
奈良先端科学技術大学院大学
情報科学研究科
門田暁人
www.empirical.jp

EASEプロジェクト

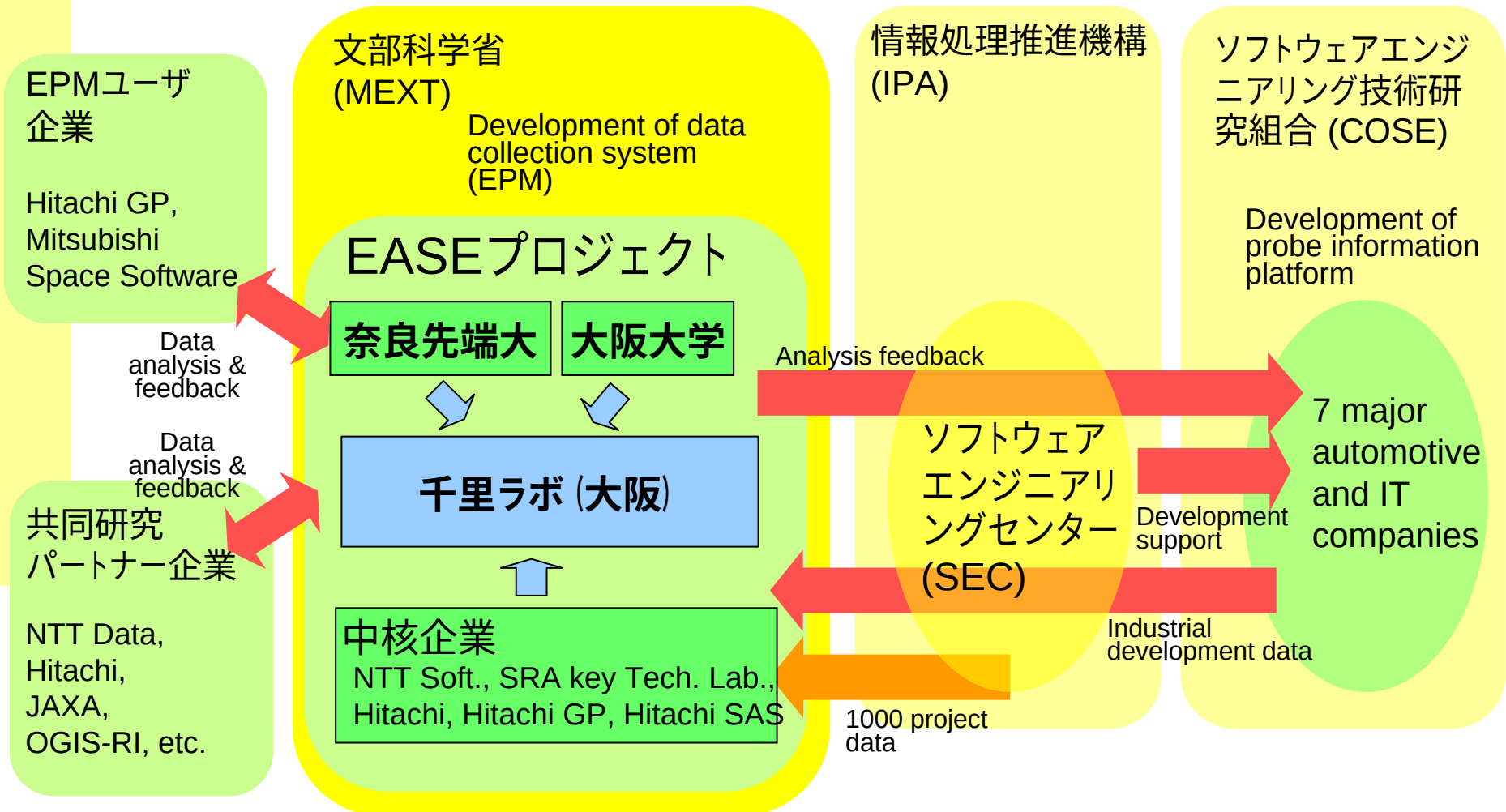
- Empirical Approach to Software Engineering.
- 文部科学省リーディングプロジェクト
- 平成１５年度から５年計画で実施中
- 主要組織
  - 奈良先端科学技術大学院大学，大阪大学．
  - ＮＴＴソフトウェア，日立製作所，日立公共システム，ＳＲＡ先端技術研究所

EASEプロジェクトの目標

- 定量的データに基づくソフトウェア開発技術の実現
  - 定量的データ収集システムの構築
    - ソフトウェア開発データの自動収集.
    - プロセス (プロジェクトの現状) のリアルタイム把握.
  - 分析したデータに基づいて開発作業を支援するシステムの構築
    - プロジェクトや組織の壁を越えた, エンピリカルデータやその分析による知見の蓄積.
    - 環境適用先の組織のソフトウェア開発における生産性. 信頼性の向上.
- 定量的データに基づくソフトウェア開発技術の現場への普及, 促進



EASEにおける産学連携



EASEにおける研究テーマ

EASE Project

- プロジェクトマネージメント支援
  - 定量データによるプロジェクトモニタリング (EPM)
- 見積もり支援
  - 協調フィルタリングによる工数見積もり
  - 信頼性予測
- コーディング・保守支援
  - コードクローン分析
  - 流用モジュールの可視化
- データ計測支援
  - EPDG (Electric Process Data Guidebook)
  - データ品質診断
- 過去プロジェクト評価・プロセス改善支援
  - マイクロプロセス分析
  - 障害データのマイニングによるルール発見
  - 生産性の要因分析
  - 工期の厳しさの要因分析

EPMによるプロジェクト管理支援

ソフトウェア開発

構成管理
CVS

メール
Mailman

障害管理
GNATS

EPM
(Empirical Project Monitor)

1日ごとのプロジェクトサマリ
- **ソースコード総行数**
- **総ファイル数**
- **変更者がN人以上のファイル数**
- **M行以上修正されたファイル数**
- **障害発生数 (種類毎, 累積)**
- **対処済み障害数**
- **障害滞留時間　　など**

インプロセス分析

1つの開発プロセスを観察・計測し，開発中に生じている現象を明確にする.

IPAによる機能強化版EPM：http://sec.ipa.go.jp/tool/epm.php

プロジェクト管理支援のためのインプロセス分析

EASE Project

- インプロセス分析とは
  - １つの開発プロセスを観察･計測し，開発中に生じている現象を明確にする．
  - 客観的な（自動計測可能な）データを利用することで，
    - 自己申告よりも正確な状況把握
    - リスクや問題の早期発見
    - 問題の原因究明
    - データ共有による，共通認識の形成やモチベーションの向上

    を目指す．

ＥＰＭ適用企業（共同研究契約締結企業）

- ＮＴＴソフトウェア
- ＳＲＡ先端技術研究所
- 日立システムアンドサービス
- 日立公共システムエンジニアリング
- 住商情報システム
- 三菱スペース・ソフトウェア
- ＪＦＥシステムズ
- サイバー創研
- 横河電機
- ソフトウェアエンジニアリング技術研究組合
  - 日本電気，トヨタ自動車，デンソー，日立製作所，富士通，松下電器産業，ＮＴＴデータ
- 日本電子計算

（共同研究契約日順）

インプロセス分析の観点

EASE Project

- 3つの観点
  - 規模
  - 変更
  - 障害 (バグ)
- 分析のアプローチ
  - 時系列の分析
  - モジュール (機能) ごとの分析
  - 開発者や工数との関係

規模に関するインプロセス分析

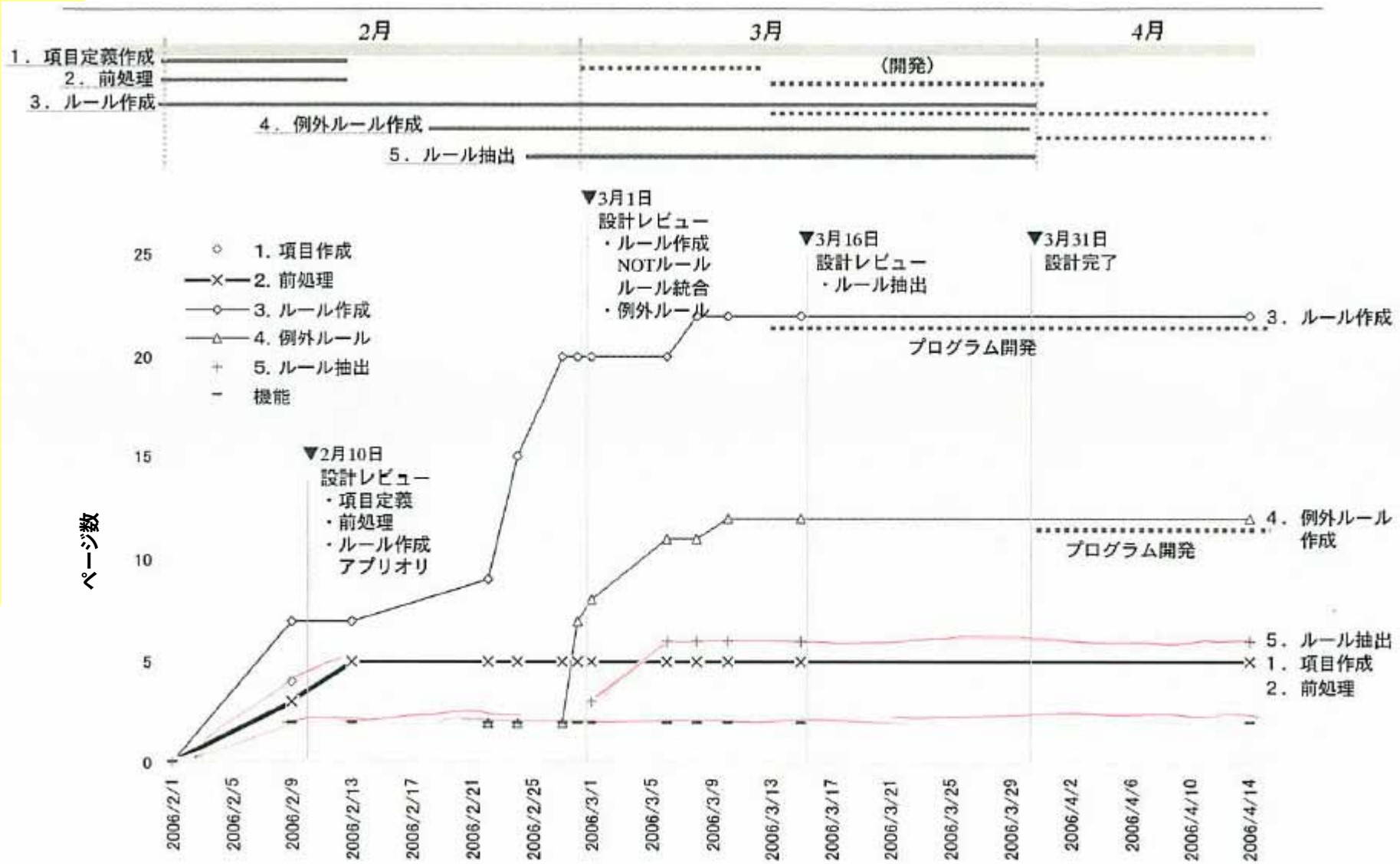
EASE Project

- 規模推移
  - 要求仕様書，設計書のページ数
  - プログラム行数
- コードクローン推移

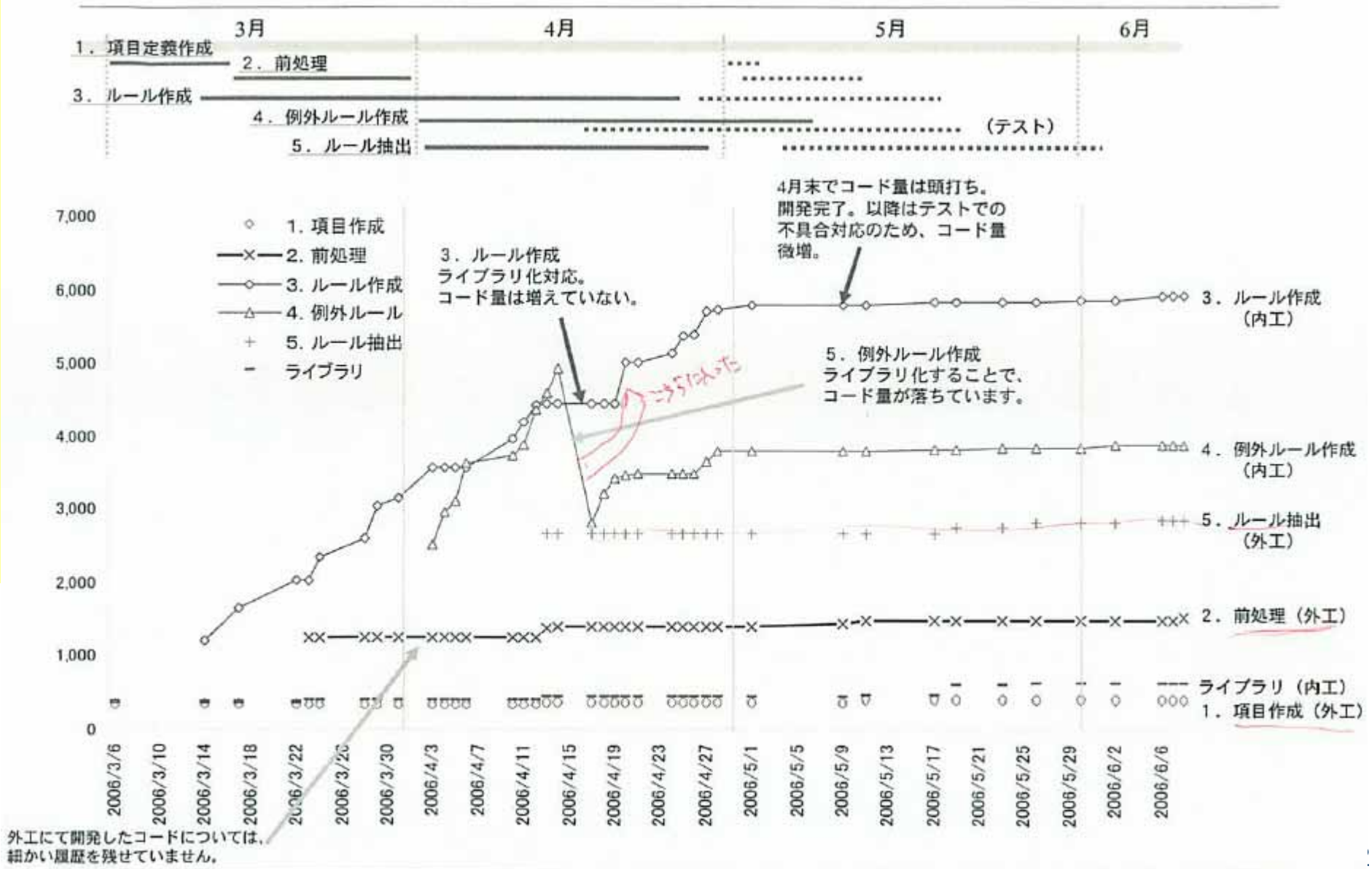
規模の推移は，開発の進捗を概ね反映している．

- 急激な規模の変化，規模の停滞は，何らかのトラブルの兆候の可能性あり
- テスト工程において，いつまでも規模が増え続けるのも要注意

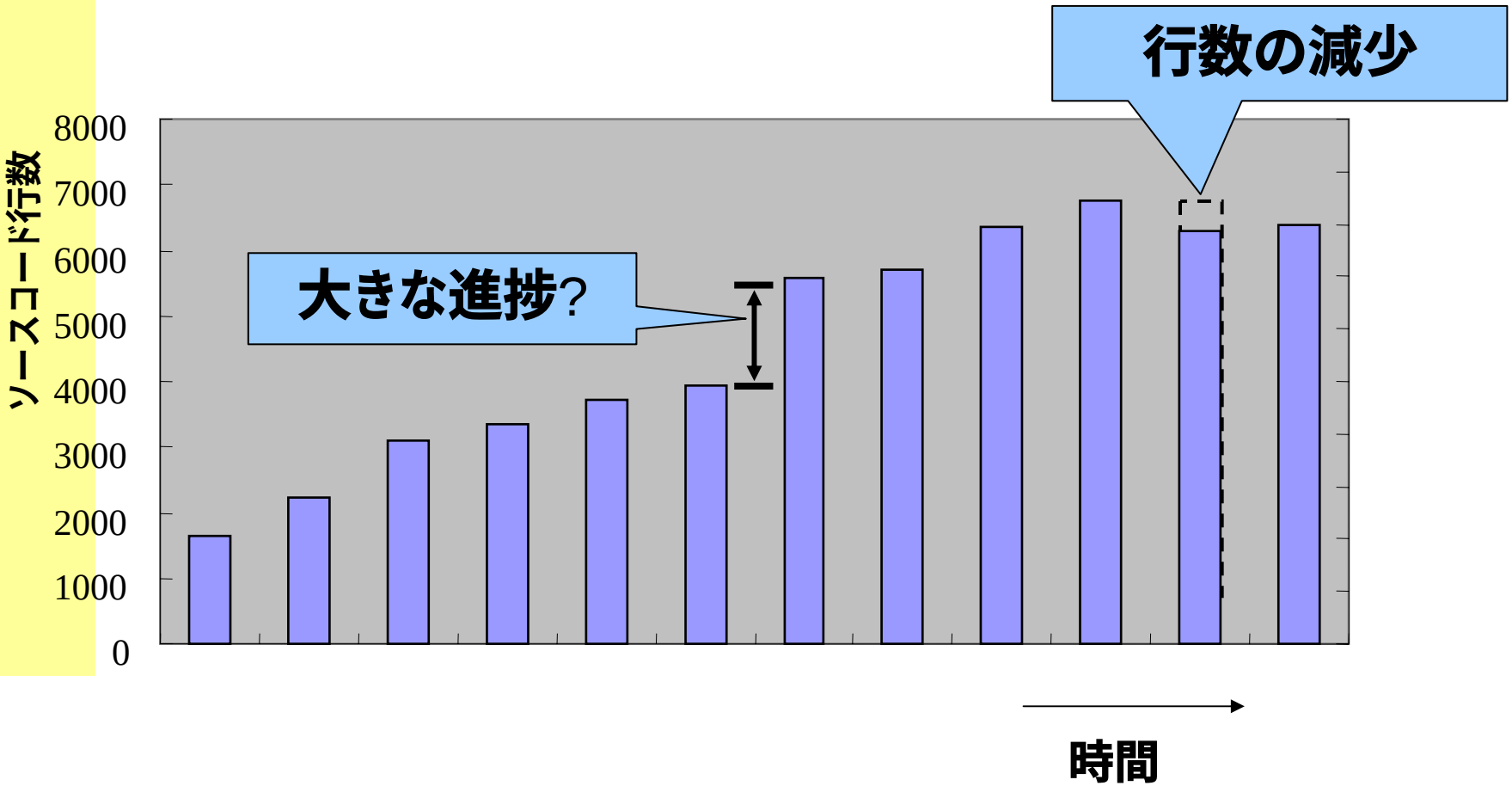
設計書の規模推移の例



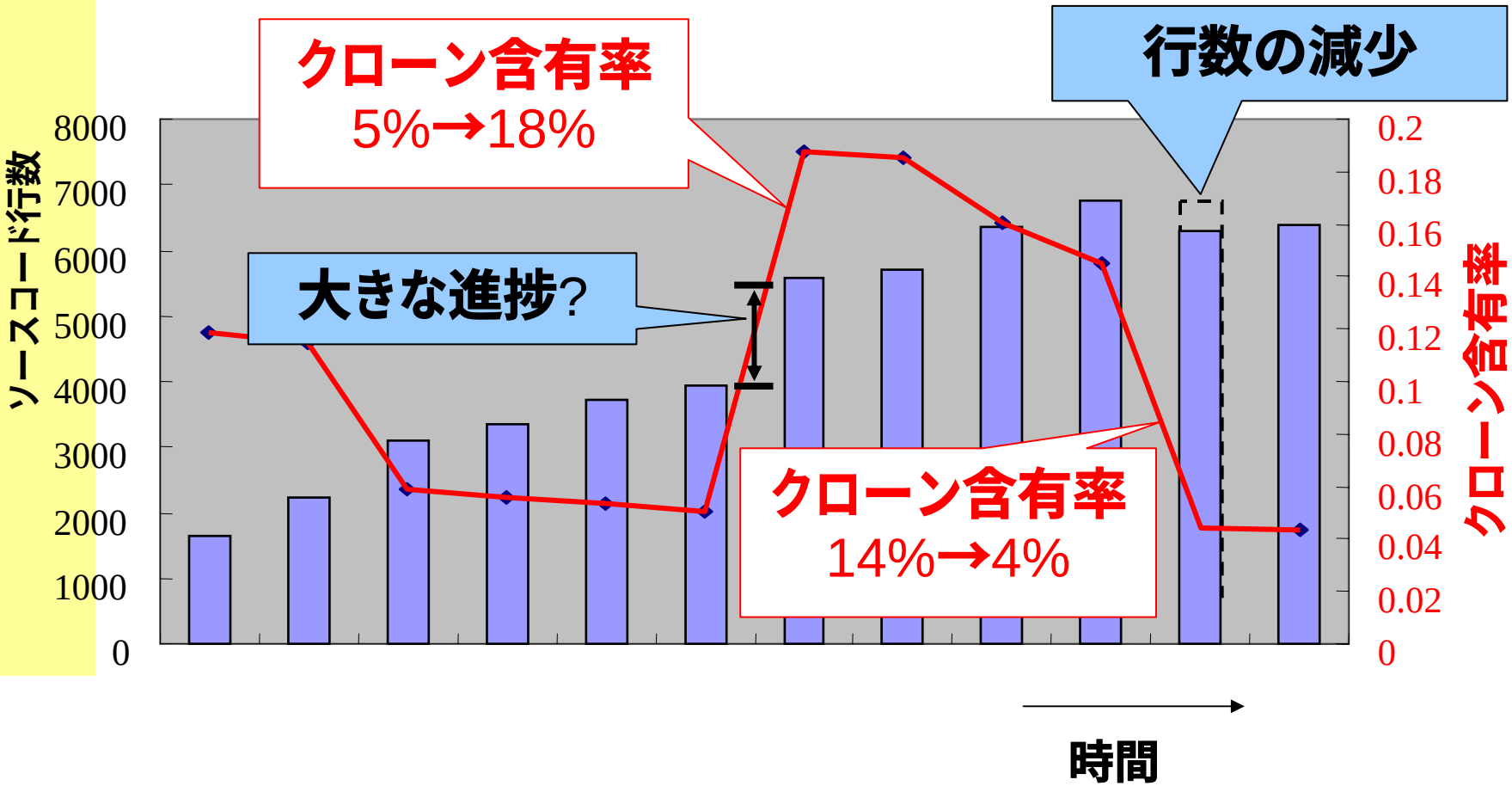
ソースコード行数の推移（1）



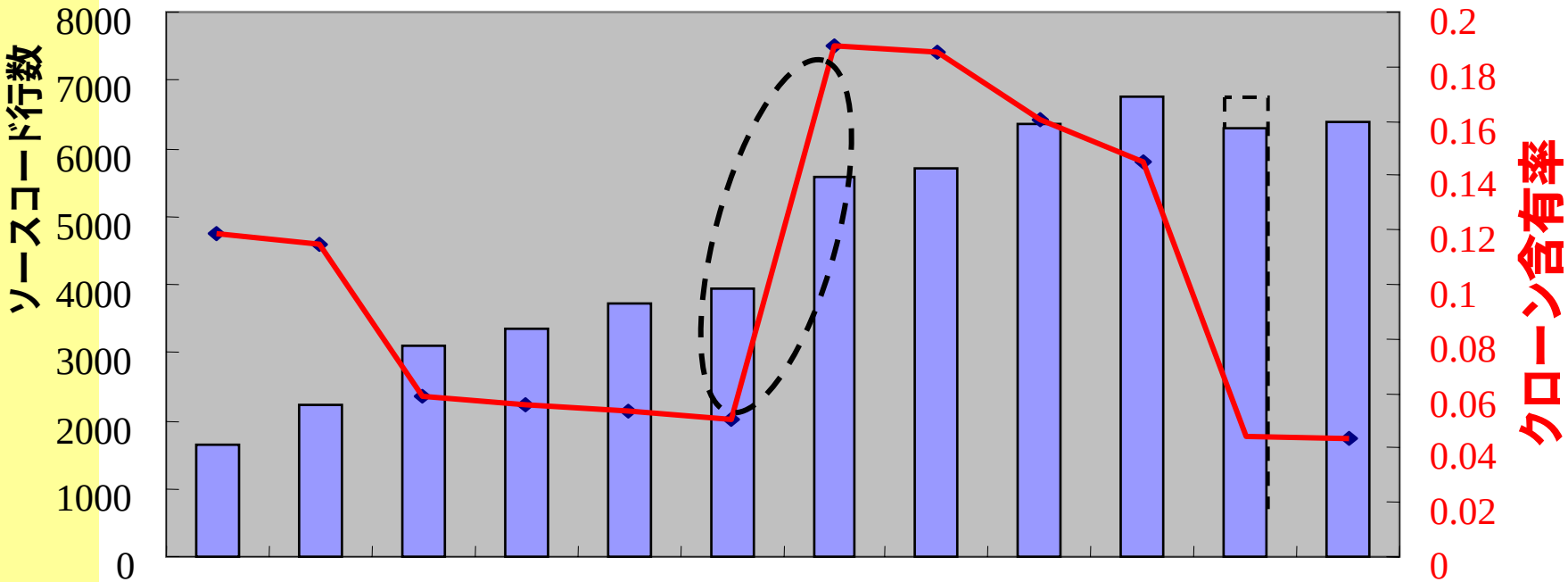
ソースコード行数の推移（2）



コードクローン (重複コード) 含有率の推移

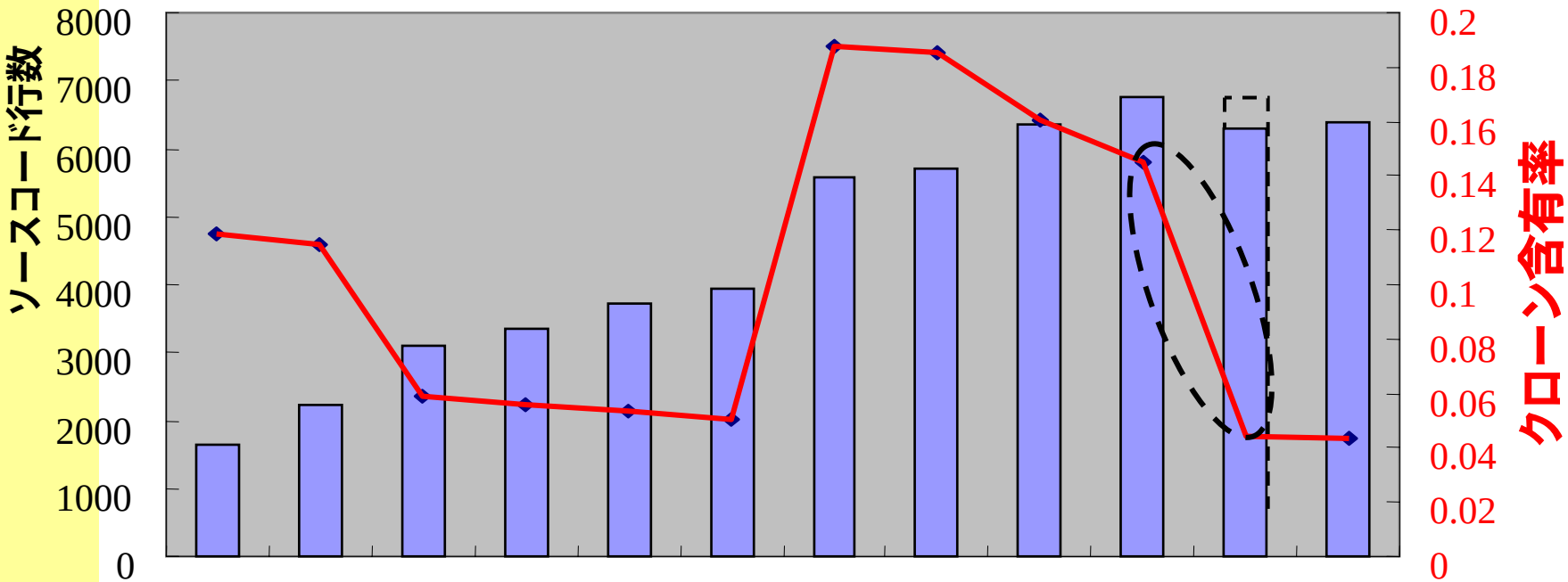


より詳細な分析（1）



ファイル	クローン含有率の変化	
charproc.c	0	→ 7.6%
mml2mid.c	0	→ 8.3%
mmlproc.c	9.3%	→ 12.5%
note.c	4.8%	→ 30.0%

より詳細な分析（2）



ファイル	クローン含有率の変化	
charproc.c	4.6% →	4.5%
mml2mid.c	0 →	0
mmlproc.c	19.9% →	9.1%
note.c	26.4% →	0

変更 (書き換え) に関するインプロセス分析

EASE Project

- 変更量 (追加行数, 削除行数)
- 変更量の多いファイル
- 変更範囲の推移
- 変更者の人数
- 変更者の多いファイル

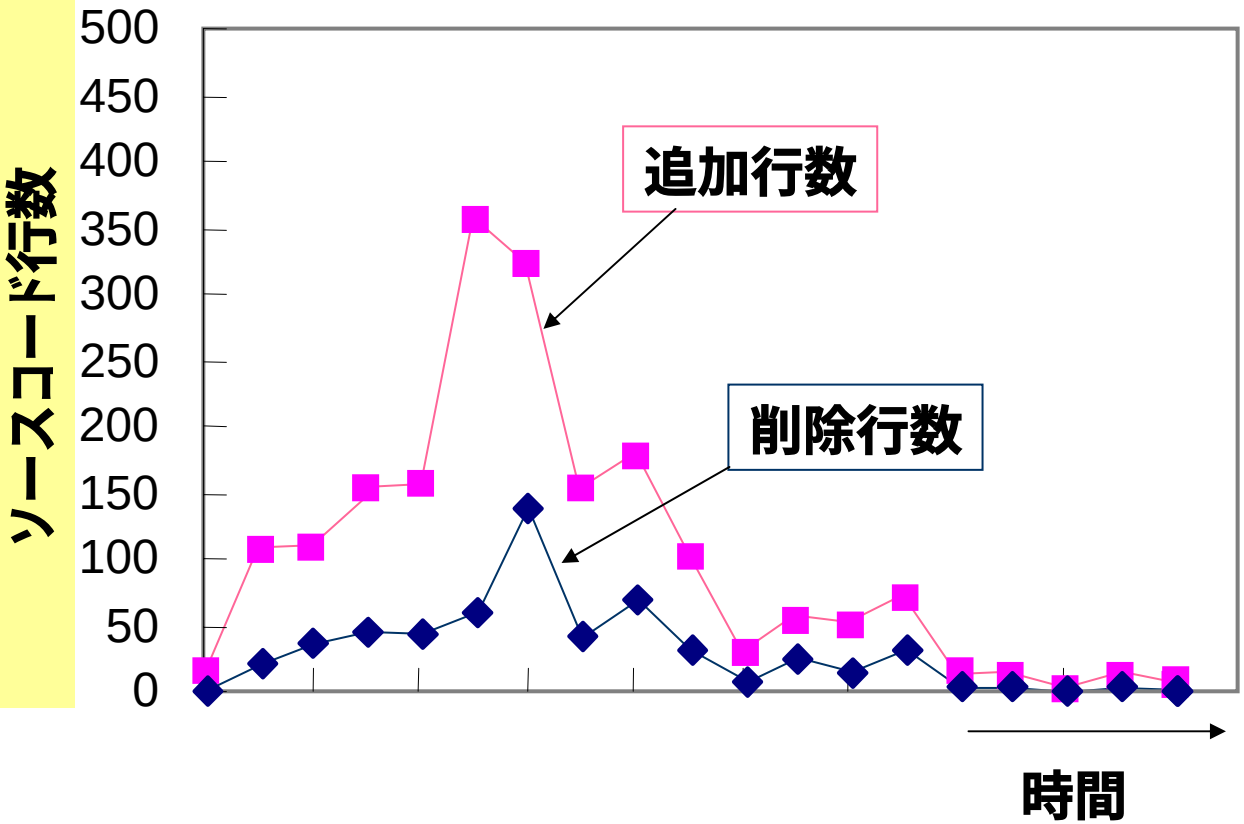
大きな変更は, トラブルの兆候の可能性あり

- 前工程での不備が後工程での変更として表れる.

多人数による変更も要注意

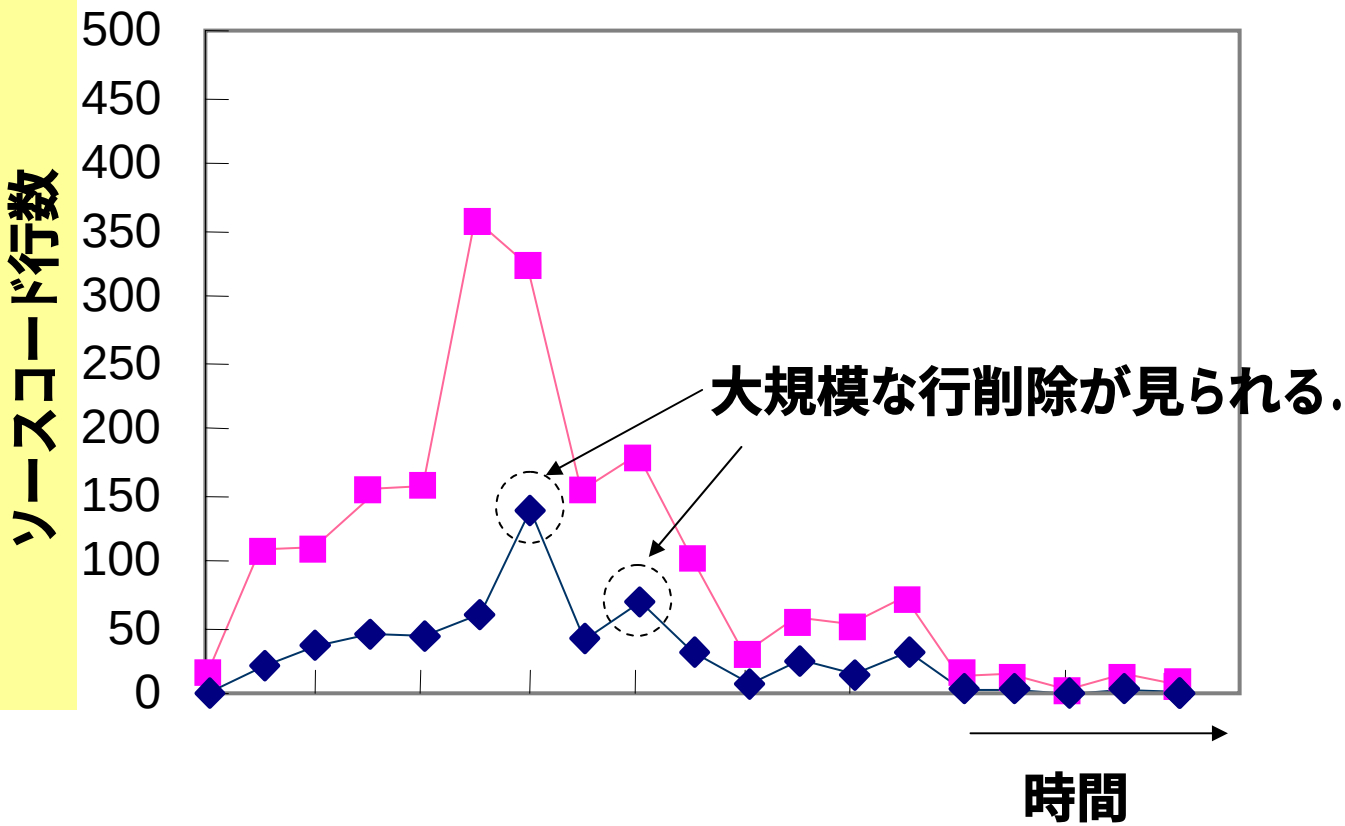
- 人の割り当てがおかしい or 設計に問題がある可能性がある.

追加行数と削除行数（1）

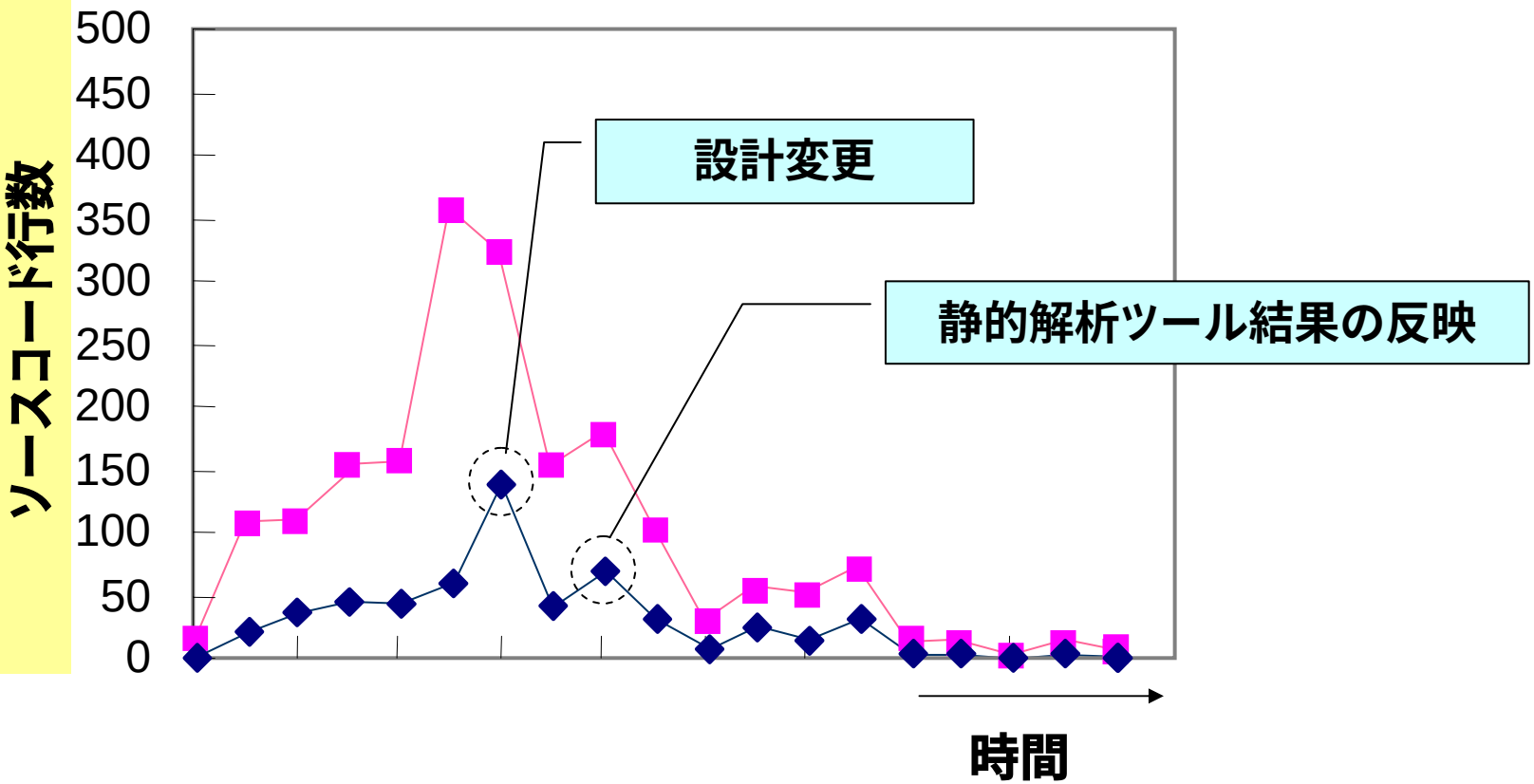


- CVSやsubversionなどの，バージョン管理ツールの履歴データから計測可能
- 大規模な行削除は要注意　→設計変更の疑いあり
  →信頼性の面でも要注意

追加行数と削除行数（2）



追加行数と削除行数（3）

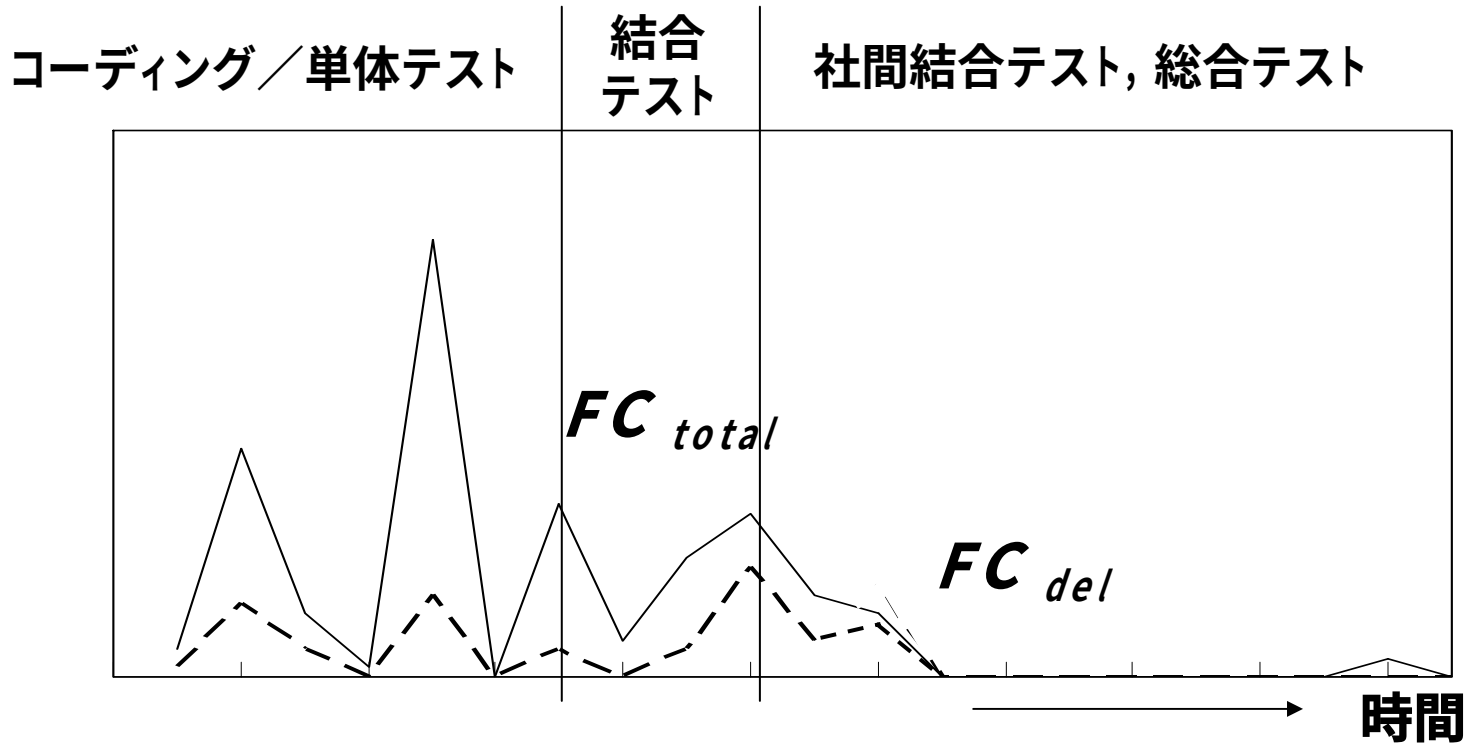


- データだけからは，「大規模な行削除」が行われた原因までは分からないので，追加調査を行う必要がある．
- 大規模な行削除＝問題発生，とは限らないが，追加調査をする価値はある．

削除行数≧１０行のファイル割合の推移

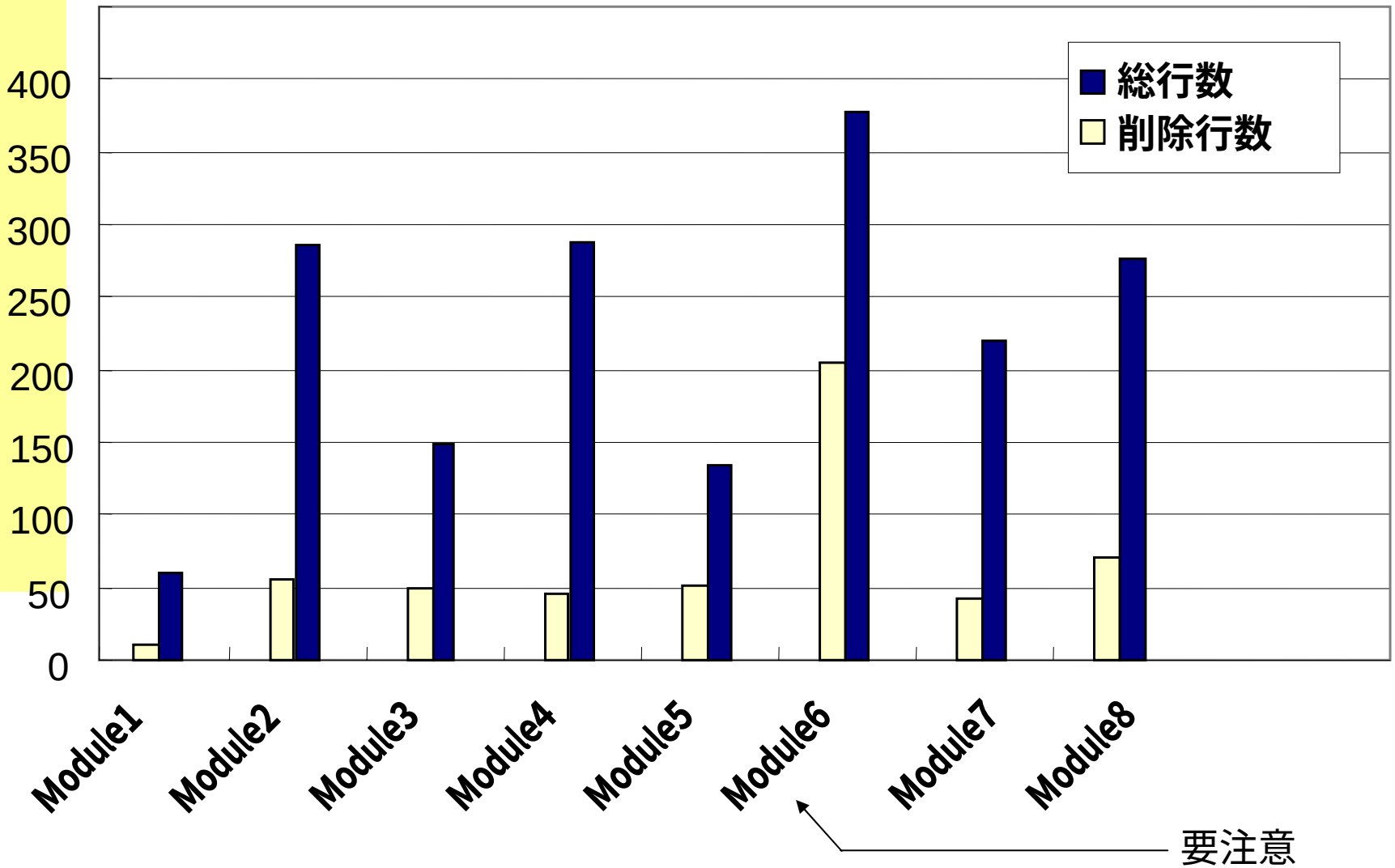
EASE Project

FCtotal　：CVSにチェックインされたファイル数（1週間あたり）
FCdel　　：そのうち削除行数≧10行のファイル数



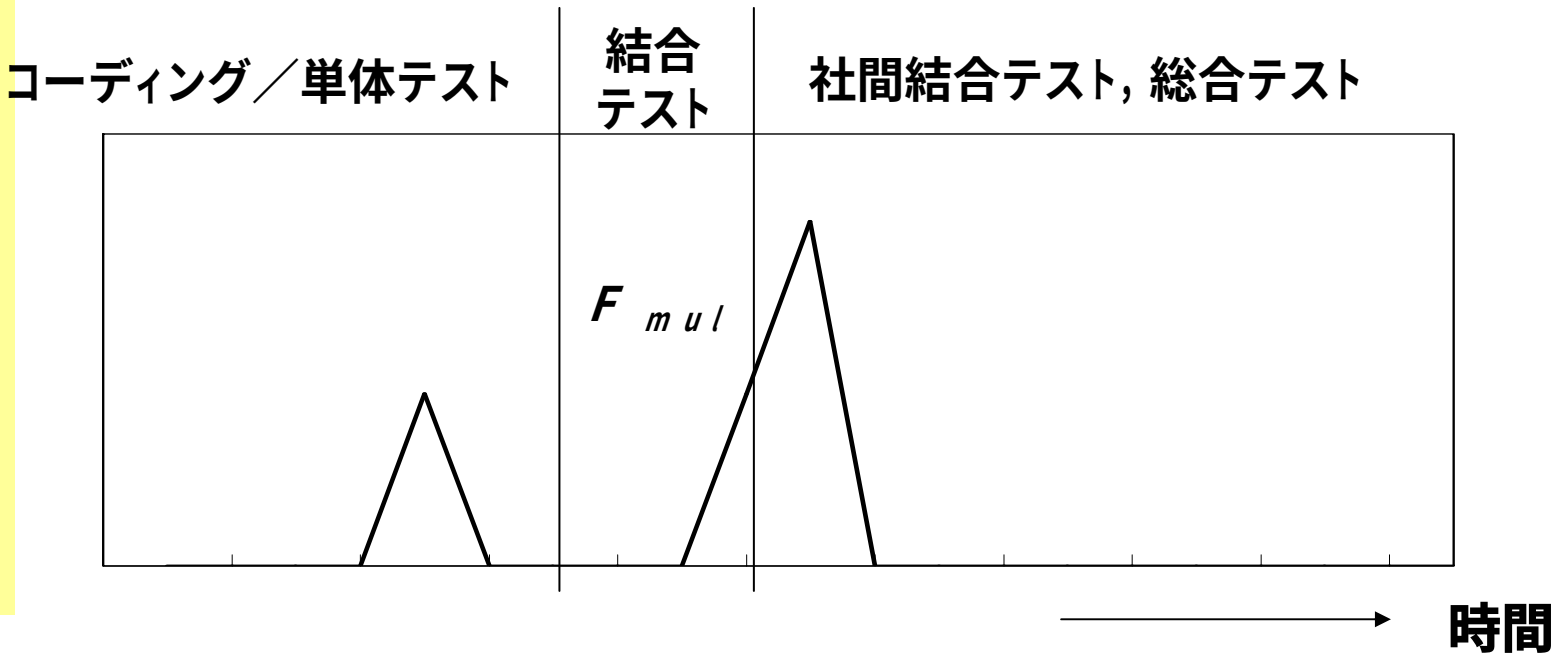
- 行削除の範囲を表しているといえる．
- FCdelが大きい＝プログラム全体にわたって大規模な変更が起こっている．

モジュール (機能) ごとの削除行数



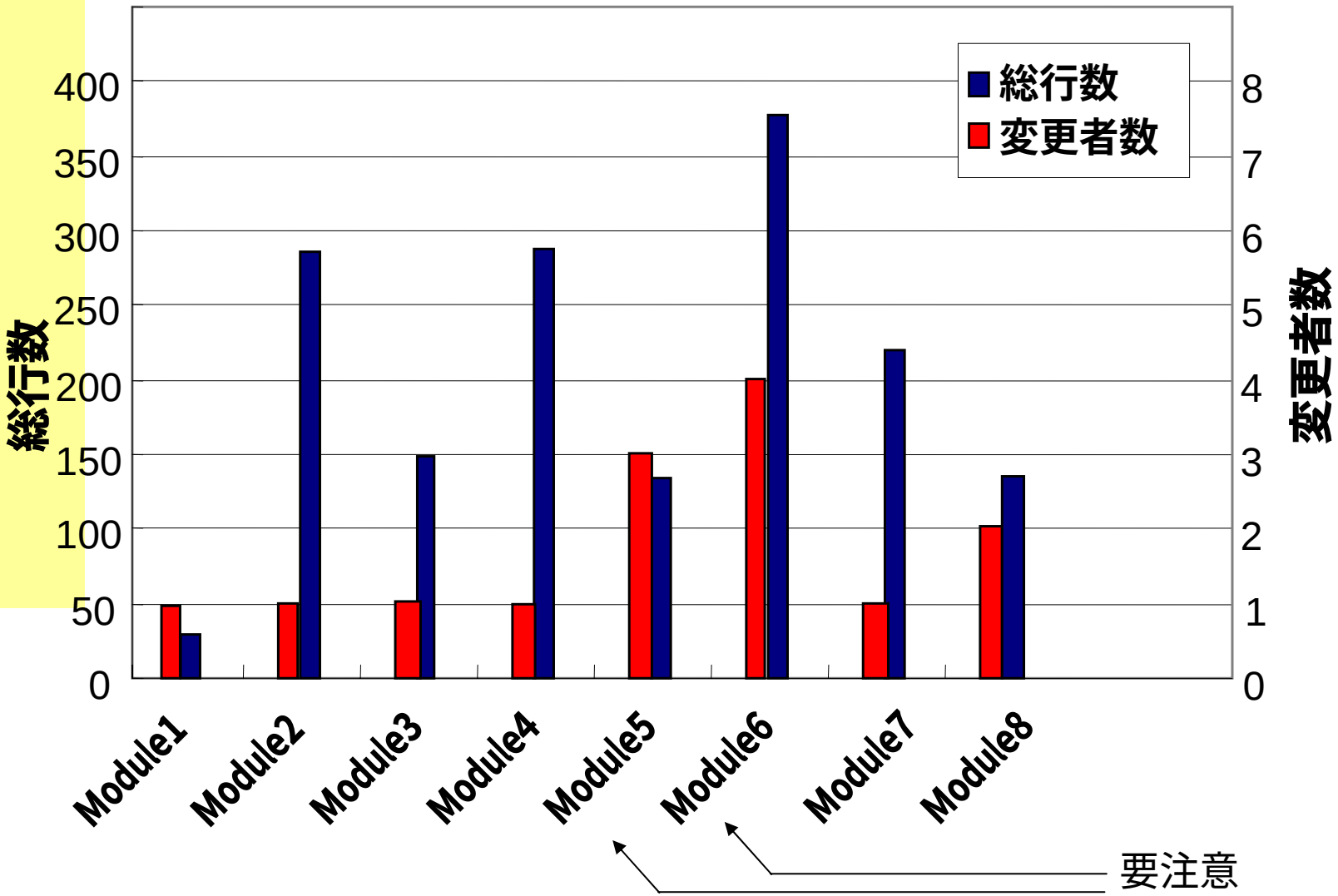
ファイル変更者に関する尺度

Fmul：2人以上が変更したファイル数（1週間毎）



- Fmulの値が大きくなる＝
  - 開発者が交代した．or
  - 開発者の割り当てに問題あり．or
  - モジュール設計に問題あり．

モジュール (機能) ごとのファイル変更者数



障害（バグ）に関するインプロセス分析

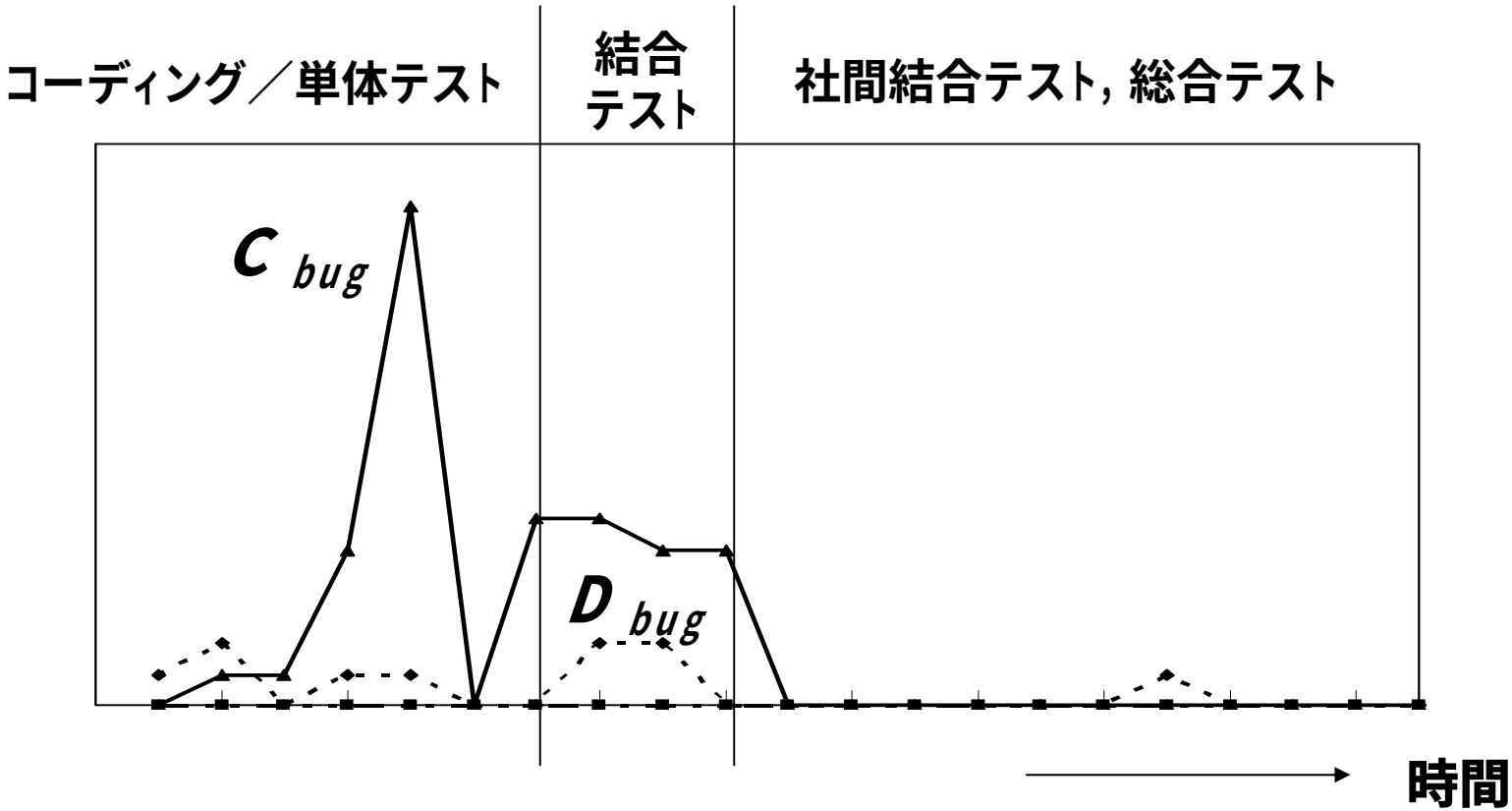
- 障害発生数，未解決障害数，障害の平均未解決時間
- テストの進捗との関係
- 障害の種類
- 混入工程ー発見工程の関係の分析
- レビュー工数と障害の関係

障害の分析は，開発プロセスの評価･改善の基本となる．

- 多すぎる障害や，特定の種類に偏った障害は
  開発プロセスに問題がある可能性がある．
- 少なすぎる障害も要注意である．

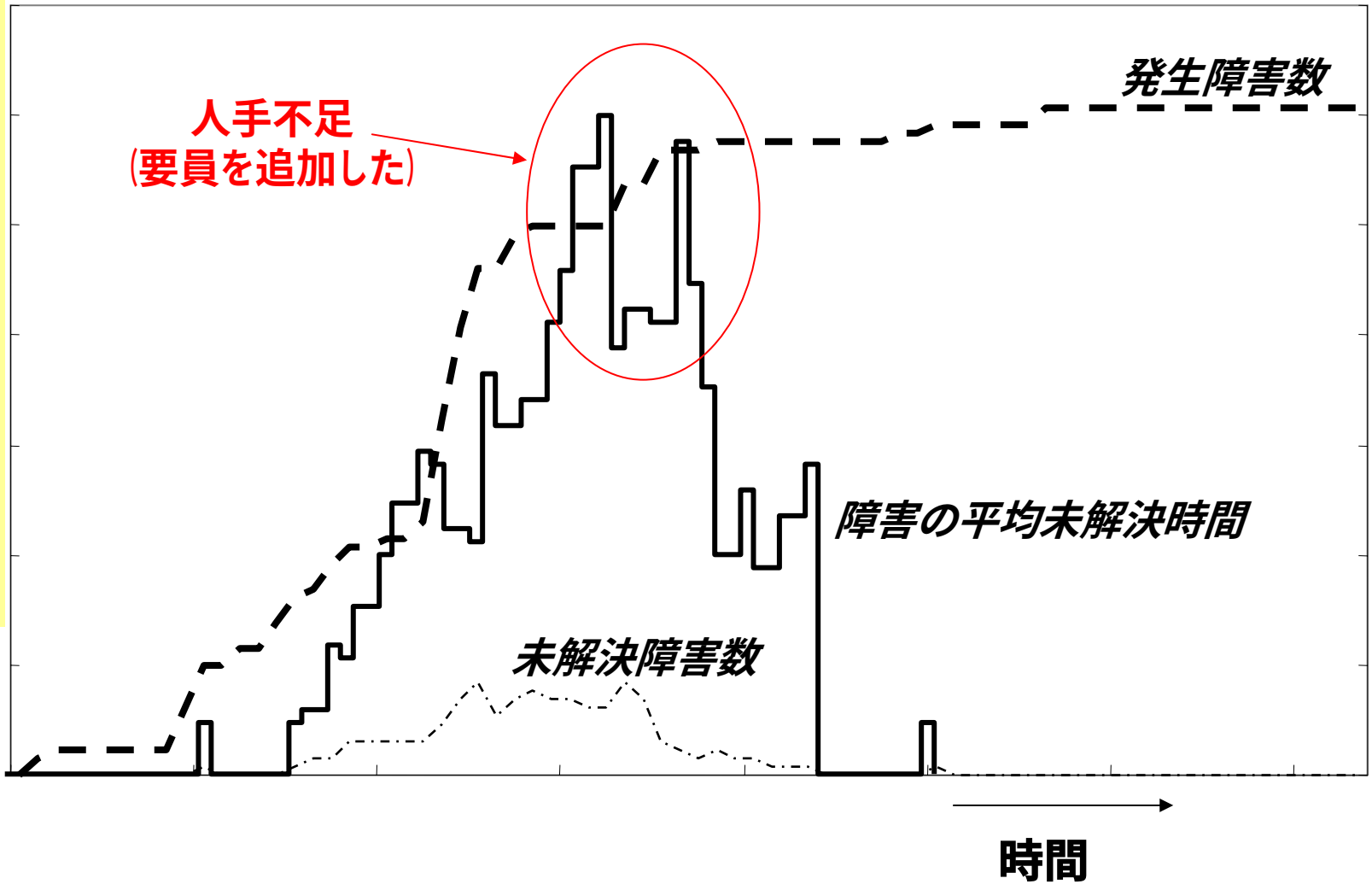
障害の推移

Cbug　：コーディングバグ数（1週間あたり）
Dbug　：設計バグ数（1週間あたり）

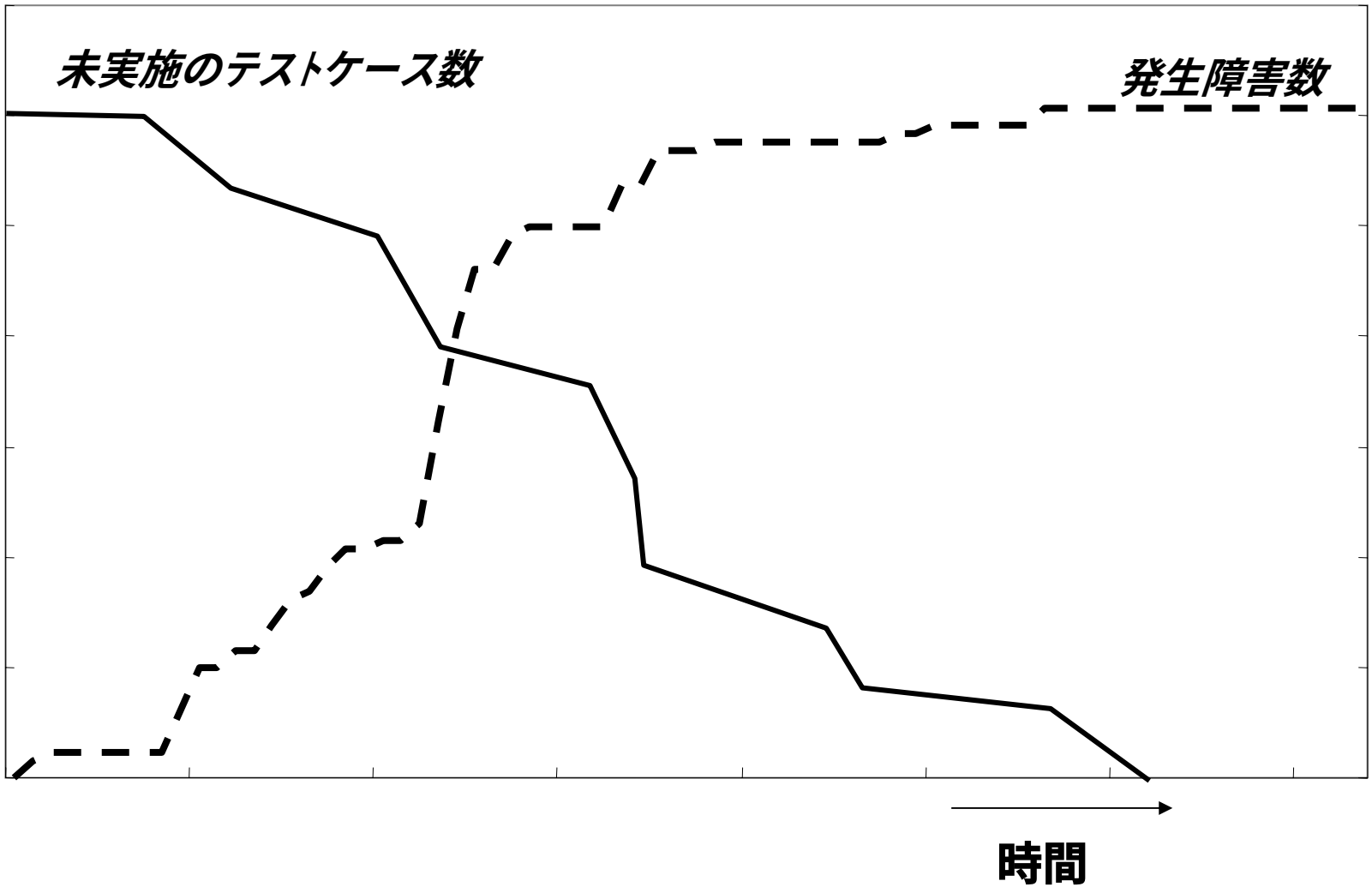


- 設計バグが後工程で発見されると要注意
  - 設計レビューが不十分な可能性あり

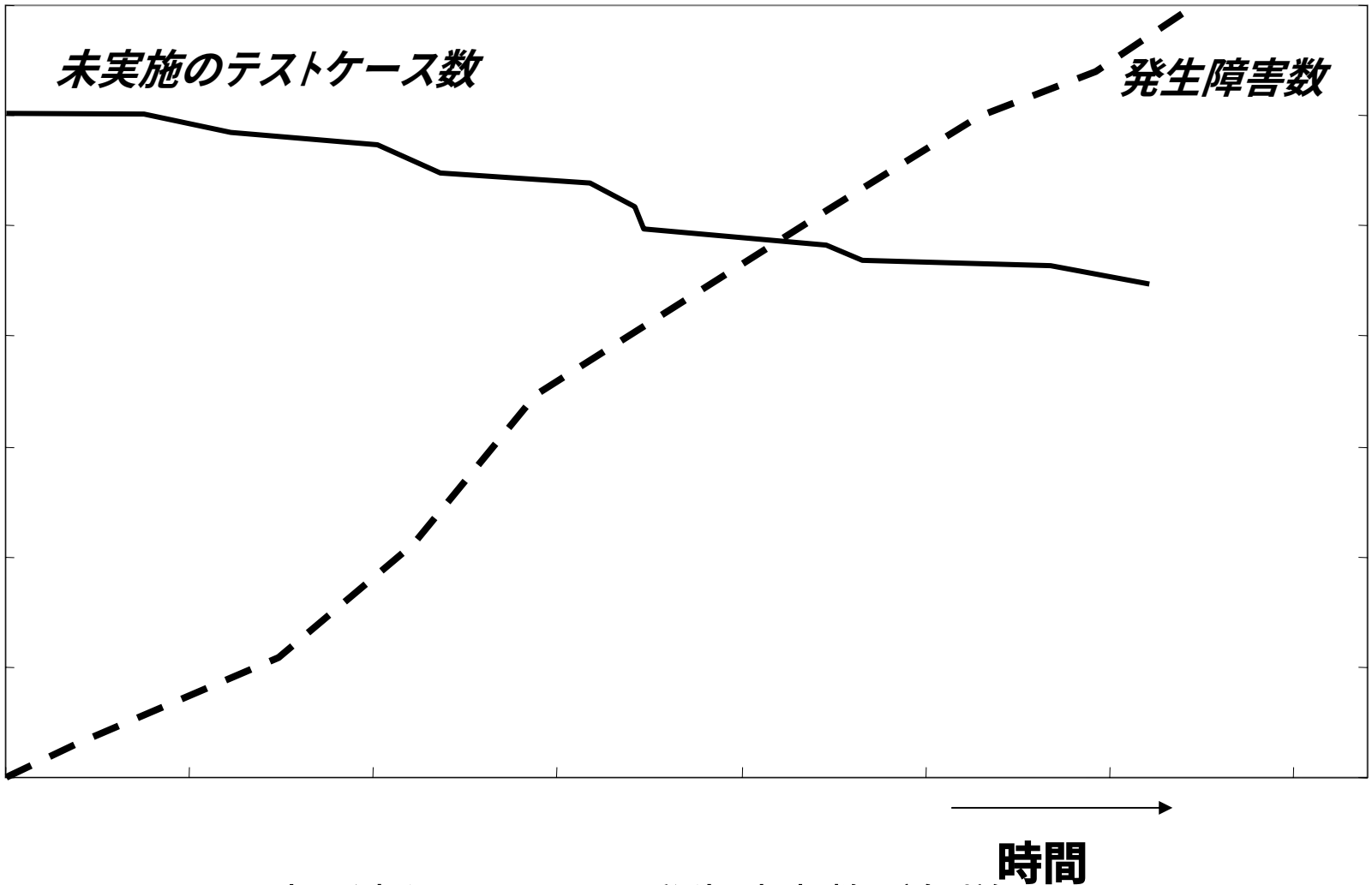
障害対応の推移



テストの進捗と障害の関係（1）

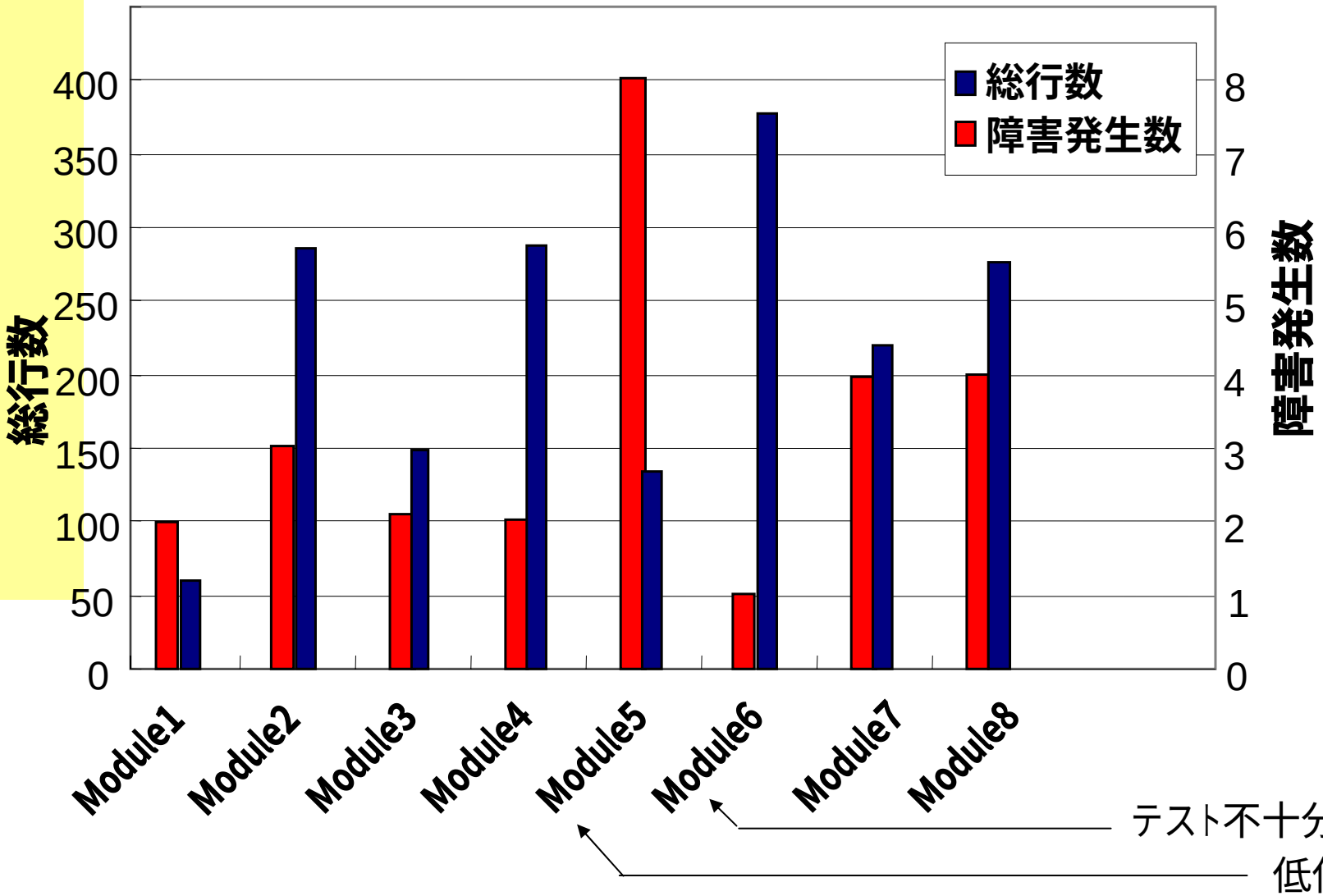


テストの進捗と障害の関係 (2)



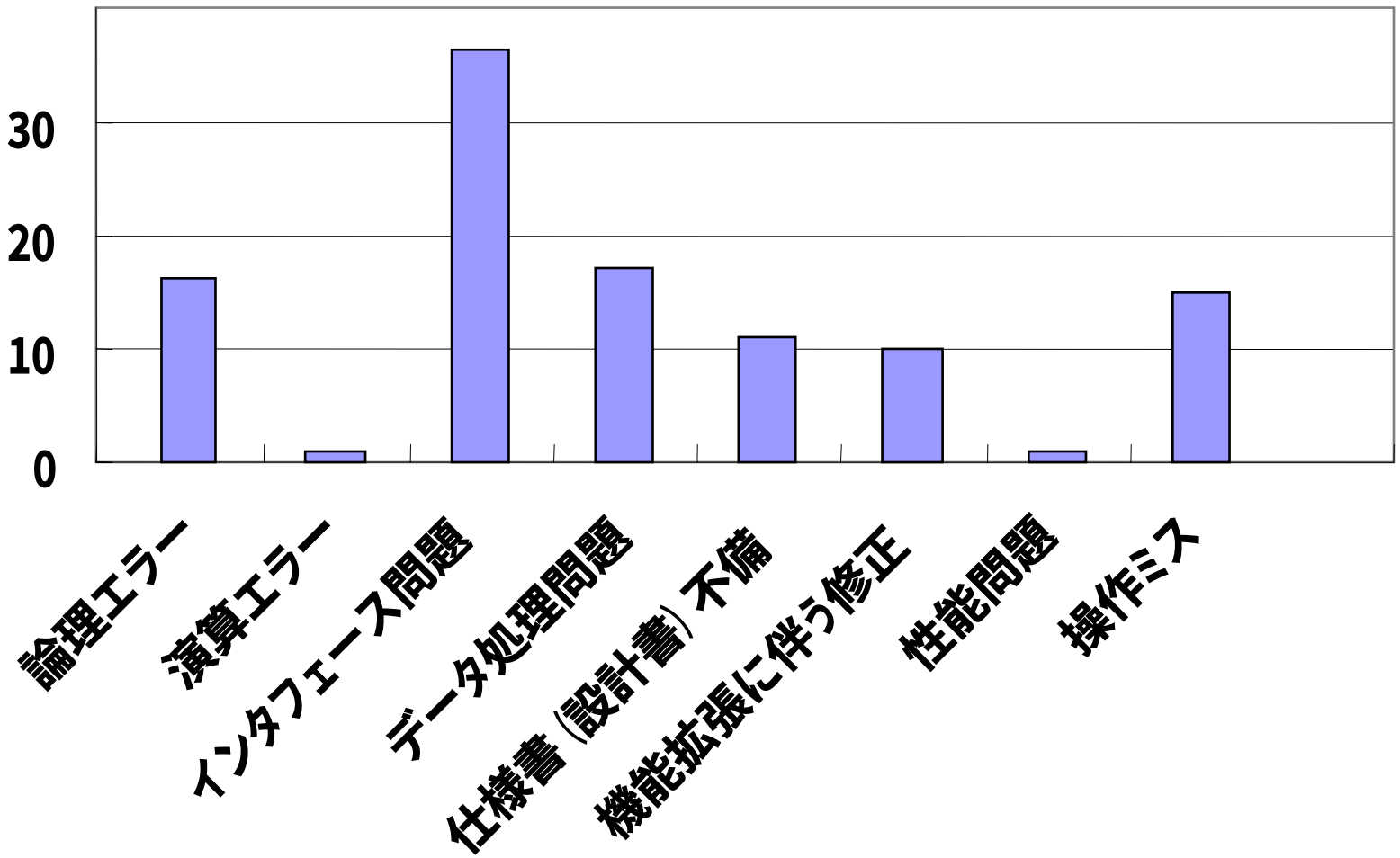
- テストケースを少し消化しただけで発生障害数が急増している.

モジュール (機能) ごとの障害発生数

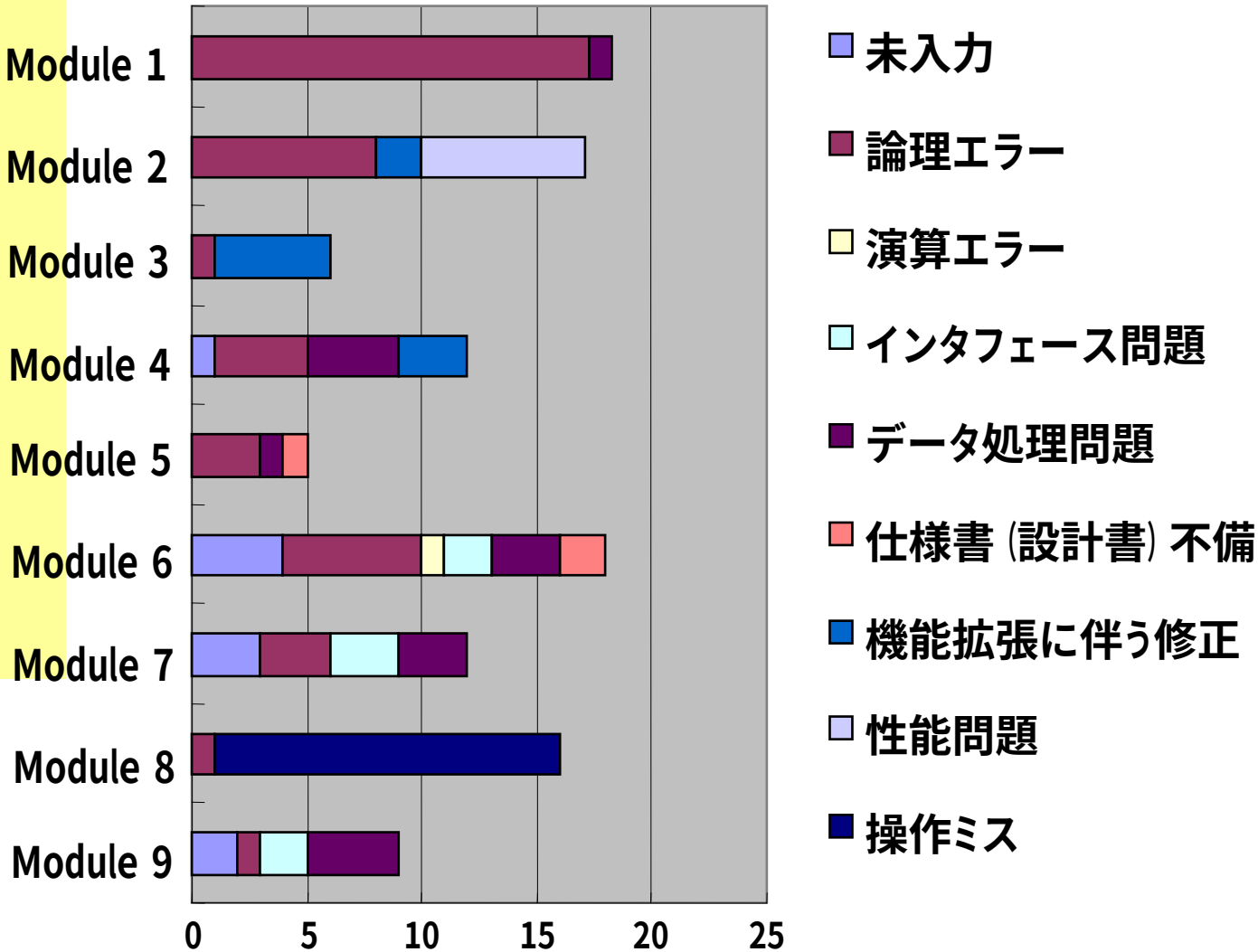


障害の原因ごとの集計

EASE Project



モジュール (機能) ごとの障害原因分析



障害の混入工程－発見工程

EASE Project

障害の混入工程	合計	障害の発見工程 未入力	要件定義	基本設計	詳細設計	単体テスト	結合テスト	総合テスト	運用
要件定義	0								
基本設計	0								
詳細設計	6					6			
コーディング/単体テスト	82	11				74	5		
結合テスト	14						13	1	
総合テスト	0								
運用	0								
合計	102	11	0	0	0	80	18	1	0

障害の混入工程ー発見工程 (モジュール毎)

EASE Project

障害の発見工程

障害の混入工程	Module1	Module2	Module3		Module4	Module5		Module6
	単体テスト	単体テスト	単体テスト	結合テスト	単体テスト	単体テスト	結合テスト	結合テスト
要件定義								
基本設計								
詳細設計			2		1			
コーディング (製造) / 単体テスト	13	10	3	2	4	4	1	
結合テスト							6	6
総合テスト								
運用								

レビュー工数と障害の関係（1）

EASE Project

	設計書 ページ数	レビュー 工数(人時)	発見 障害数	障害密度 ページ当たり	障害密度 工数当たり
Module 1	32	12	6	0.188	0.500
Module 2	24	8	3	0.125	0.375
Module 3	49	10	2	0.041	0.200
Module 4	33	12	3	0.091	0.250
Module 5	12	4	1	0.083	0.250
Module 6	21	6	7	0.333	1.167

- レビュー密度
- テストケースを少し消化しただけで発生障害数が急増している.

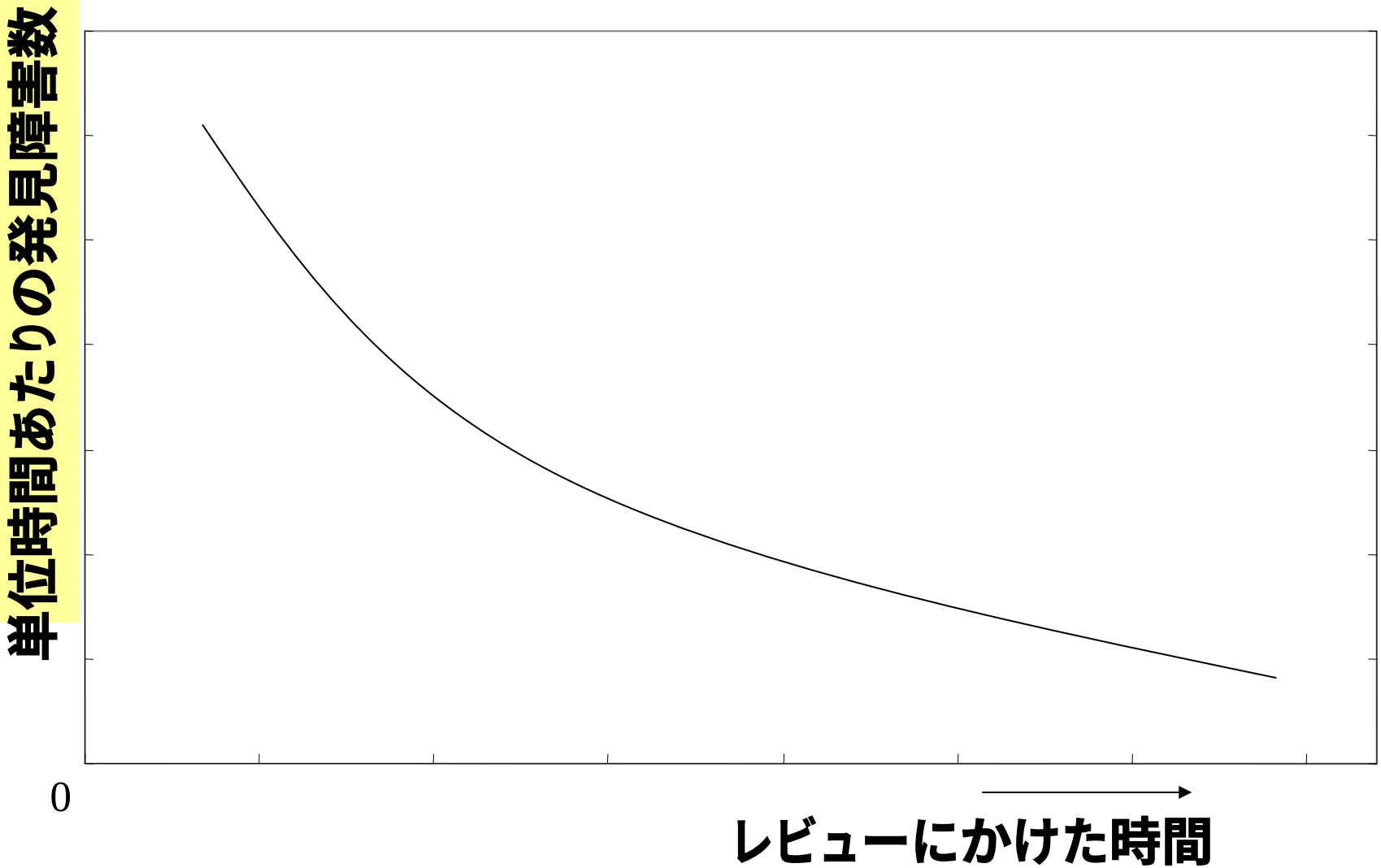
レビュー工数と障害の関係(2)

EASE Project

	設計書 ページ数	レビュー 工数(人時)	発見 障害数	障害密度 ページ当たり	レビュー 密度
Module 1	32	12	6	0.115	0.375
Module 2	24	8	3	0.125	0.333
Module 3	49	10	2	0.041	0.204
Module 4	33	12	3	0.091	0.364
Module 5	12	4	1	0.083	0.333
Module 6	21	6	7	0.333	0.286

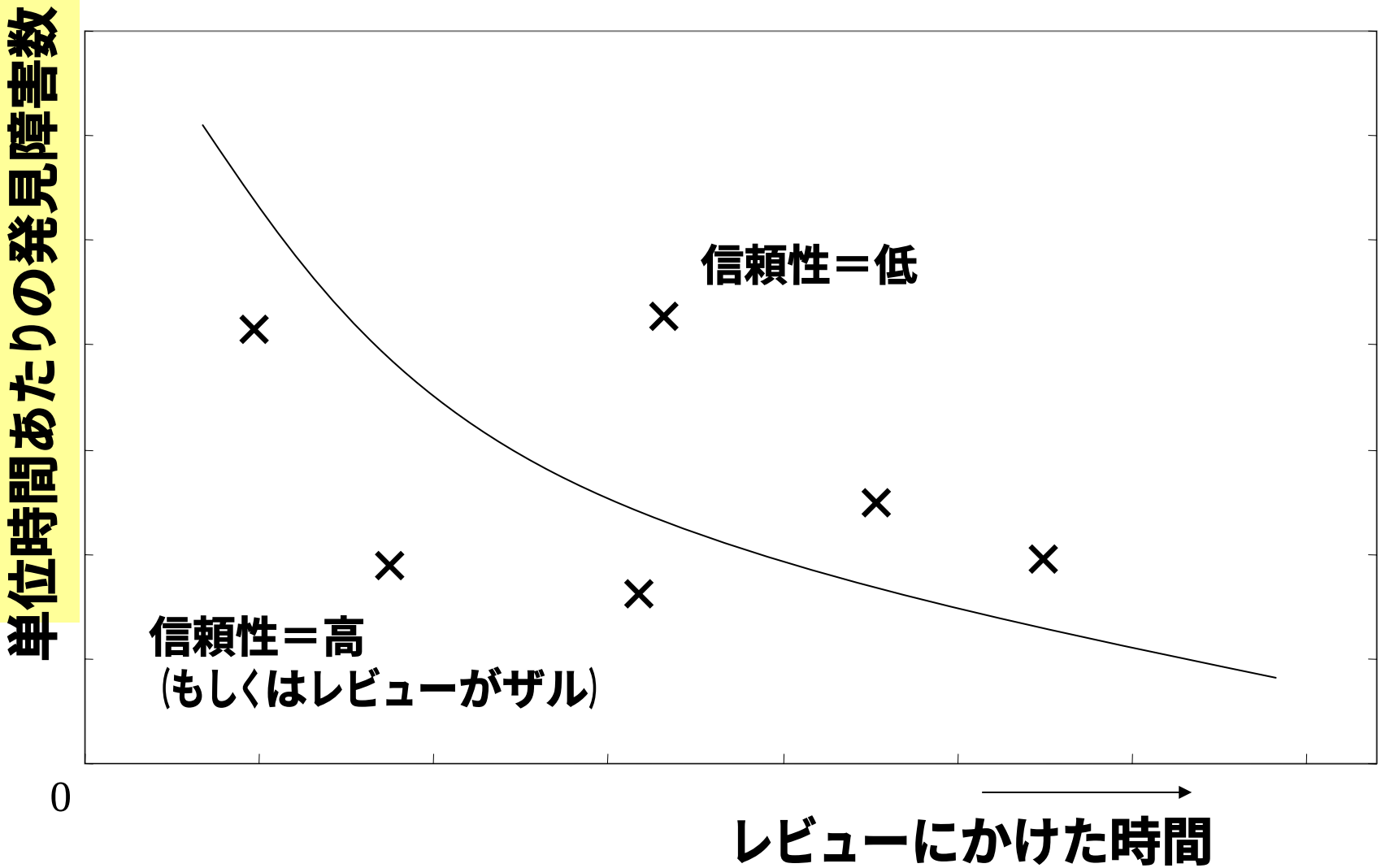
- レビュー密度＝設計書１ページあたりにかけたレビュー工数
- モジュール3はレビュー工数が足りないので，障害密度が低いからといって信頼性が高いとは言い切れない．

レビュー工数と障害の関係（3）



参考：Robert B. Grady, "Practical software metrics for project management and process improvement", Prentice Hall, 1992.

レビュー工数と障害の関係（4）



まとめ

- 3つの観点
  - 規模
    - 開発の進捗の把握
  - 変更
    - 変更による開発遅延や障害への影響の把握
  - 障害 (バグ)
    - 開発プロセス・プロダクトの評価のベースとなる.
    - プロセス改善の手がかり