JaSST 2013 Tokyo – Project Fabre

# 過失に着目した欠陥のモデリング

## ーバグ分析はなぜうまくいかないのか？ー

Nobuhiro Hosokawa, Yasuharu Nishi,

Aya Ureshino, Makoto Nonaka, Yukiko Hara

# Project Fabre セッション コンテンツ



Copyright © 2013 Project Fabre All Rights Reserved.

# 1. Project Fabre メンバー

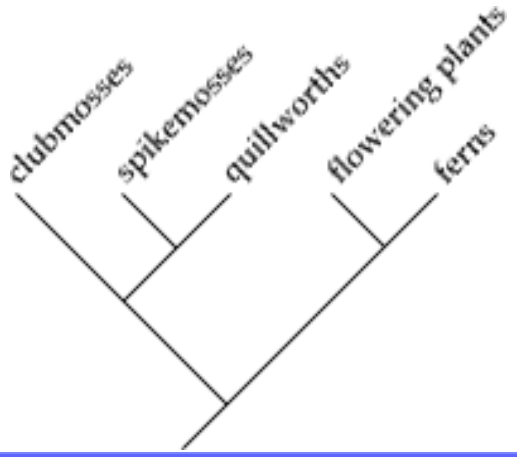
細川 宣啓（日本IBM）
< CARVIN@jp.ibm.com >

野中誠（東洋大学）
< nonaka-m@toyo.jp >

西 康晴（電気通信大学）
< Yasuharu.Nishi@uec.ac.jp >

原 佑貴子（日本IBM）
< HARAYUKI@jp.ibm.com >

嬉野 綾（ワークスアプリケーションズ）
< ureshino@worksap.co.jp >




Copyright © 2013 Project Fabre All Rights Reserved.

## 2. バグ前提社会

# 「バグのない社会」を我々は目指すのか？

- 人は全知全能の神ではない
- 脳はバグるし、人はよく間違える
- 全然間違えない、失敗しない日はありますか？
- お客様の求めるものの多様化・複雑化と
  ソフトウェアの進化・複雑化が加速度的に発生
- 止めるべきは、バグそのものではなく、ビジネス上の実害

Copyright © 2013 Project Fabre All Rights Reserved.

## 2. バグ前提社会

# 「バグ前提社会」での考え方

- 開発エンジニアは、どのようにしてバグを埋め込む罠にはまったのか？
  開発プロセスで何があったのか？
- 品質エンジニアは、どのようにしてバグを見逃す罠にはまったのか？
  テストプロセスで何があったのか？
- 間違えたくないが、間違えない開発やテストはできない
- 同じバグを繰り返さないよう、モデル化が必要

Copyright © 2013 Project Fabre All Rights Reserved.

## 3. Project Fabreの役割

- ソフトウェアの欠陥そのものに関する研究
- ソフトウェアの欠陥を「悪さの知識」と位置づけ
- 欠陥を汎用的な欠陥マスタとして共有／蓄積
- 欠陥を伝承・移転可能にし、学術界・産業界で資産化
- 欠陥に関する構造等を定義
- 欠陥エンジニアリングという技術分野を確立
- 欠陥に強いエンジニアの育成に貢献

（ http://aster.or.jp/business.html#fabre より）

Copyright © 2013 Project Fabre All Rights Reserved.

# 4. バグ分析のレベル

**バグ分析自体を「実施してない」**
*バグを検出して除去すればよい、分析などしたくない。*

**バグ分析をしても「効かない」**
*バグ分析を実施しても、あまり効果が得られない。*

**ある程度効くことは理解している。効能を「体感済」**
*バグ分析を実施し、効果あり。でも、同じバグに遭遇。*

Copyright © 2013 Project Fabre All Rights Reserved.

# 5. 本日のポイント

ー 「バグ分析をしていない、うまくいっていない」
⇒うまく分析するためのコツを伝授

ー 「バグ分析はやって、ある程度効果はあったが、
プロジェクトやリリース毎に同じようなバグは出る」
⇒モデル化で、一歩進んだバグ分析へ

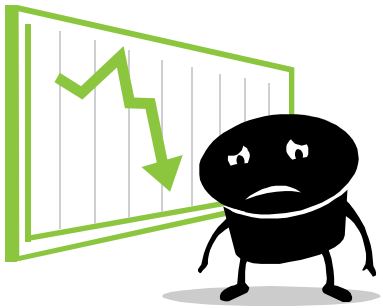
ー 「欠陥（ライフサイクル）のモデル化」
⇒バグが埋め込まれた時、どんな状況だったのか？
バグを防げなかったのは、どんな状況だったからなのか？

Copyright © 2013 Project Fabre All Rights Reserved.

# 6. バグ分析の失敗要因

Q.バグ分析は、なぜ失敗するのか？

- 本当にバグを分析しているか？
- バグ分析そのものが目的になっていないか？
- 事実確認が甘すぎないか？
- 「なぜなぜをやるとよい」と聞いたから
- 失敗事例が共有されていない



Copyright © 2013 Project Fabre All Rights Reserved.

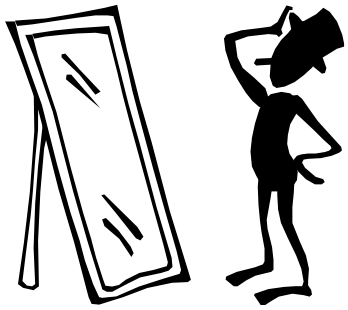
# 6. バグ分析の失敗要因

## A.バグを偏った側面で捕らえている。

- 人、設計、プロジェクト、プロセス、手法等「だけ」に答えを求めている。原因と対策が表と裏「だけ」になっている。

  例）人→人、設計→設計

- なぜなぜの方向が違う。
- コントロール出来る範囲を考えていない。



バグ分析の本質は、バグの存在を確かめ、バグの資産化のため、獲得・記録・抽象化・保存・移転する点にあります。バグそのものと周辺の状況を捕らえ、原因分析や対策立案をしなければ、同じようなバグが再発します。

Copyright © 2013 Project Fabre All Rights Reserved.

# 7.「なぜなぜ分析」の注意点と「なになに分析」の勧め

## 「なぜなぜ分析」の注意点

- 他者に対して問い詰め、否定形になりやすい
  - ー なぜバグを出してしまったんだ？
- 一つの答えに帰結しやすい
  - ー複数、複雑なものに対処しづらい
- バイアスにより、新たな発想が生まれにくい
  - ー限られた範囲の原因追求や自分自身への問いかけには有効

## 「なになに分析」の勧め

- ポジティブに考えやすい
  - ーなにがどこで起こったの？その前には、なにかあったの？
- バグ票の必須要素を含む
  - ー現象・期待結果・起動経路を明確化
- 可能性が広がる、選択肢が想像しやすくなる
  - ー俯瞰的・多角的に観察しやすい

Copyright © 2013 Project Fabre All Rights Reserved.

# 8. バグ分析のアンチパターン（原因編）

- 現象と原因がごっちゃになったバグ票を使っている
  ⇒「なぜなぜ」以前の問題
- うっかりしてた、しっかりやってない、きちんとできてなかった
  ⇒たまたまであれば、その時どんな状況だったの？
- 決めつけ、犯人捜し、うやむや
  ⇒本気で解決する気ありますか？
- 製品設計の問題
  ⇒設計のどこでどういう問題が起きている？
- テスト漏れ
  ⇒何がどこで漏れた？テスト網羅された状態とは何？

Copyright © 2013 Project Fabre All Rights Reserved.

# 8. バグ分析のアンチパターン(対策編)

- すでに項目数が多いチェックリストへチェック追加
  ⇒どこまで増やせる？トレードオフ考えている？
- 全体会議で周知(注意喚起)する
  ⇒来月の新規配属、異動してきた人には伝わる？
- 教育・研修の充実
  ⇒範囲は？対象者は？起こすのは新人だけ？
  資源は確保できる？
- 文化醸成・属人化排除・エライ人投入
  ⇒コントロールできる問題？

Copyright © 2013 Project Fabre All Rights Reserved. 

# ９．バグ分析のまずまず良いパターン（原因編）

－ 「どんなバグを減らしたいか？」
⇒バグ分析の目的の共有・合意、優先順位付け

－ 担当者・関係者へヒアリングして、状況を明らかにする
⇒関係する情報の質と網羅率を上げる

－ ありがちだが実は不適切なことから離れて考える
⇒原因追求は「なぜなぜ」、対策は「なになに」

－ 自責（自分、自チーム）と他責（他人、他チーム）の両方で考える
⇒開発プロセスで埋め込んだ原因と
テストプロセスで発見できなかった原因

Copyright © 2013 Project Fabre All Rights Reserved. 

# 9. バグ分析のまずまず良いパターン(対策編)

- 多くの仮説を立てる(あらゆる方法を想像)
  ⇒あとの選択肢を広げる
- 分けて考える
  ⇒実害の影響度、コントロール可能範囲、除去難易度、テスト難易度、コストのかからない方法など
- 実行しやすい範囲や対象から始める
  ⇒自分のチーム、近所のチーム
- 合わせ技
  ⇒注意喚起と継続的解決、アナログとシステム化、短期と中長期、開発フェーズとテストフェーズ

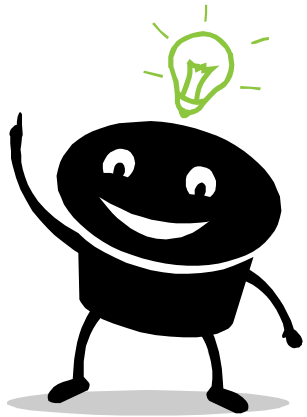
Copyright © 2013 Project Fabre All Rights Reserved.

# 10. バグ分析ができない事例の特徴

多くの場合、下記ができていない事例が多い
→できるようにすれば、解決へ近づける

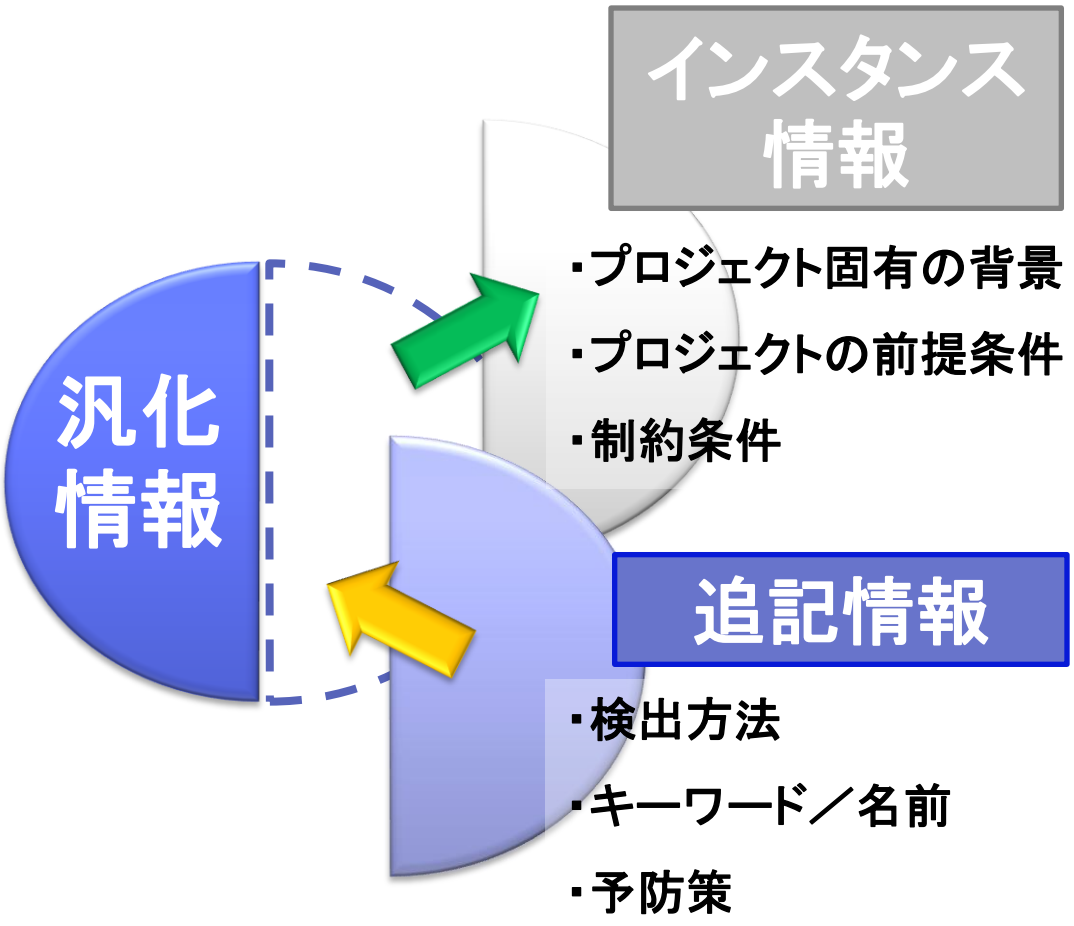
- バグを楽しむ
- バグを表現する
- バグを比較する
- バグを移転する
- バグを蓄積する
- バグを資産とみなす

**過失に着目！**



Copyright © 2013 Project Fabre All Rights Reserved.

# 参考：昨年のワークショップ資料．インスタンス情報の切り離しと追記情報




「固有」部分を切り出し

汎化された情報

所見を追記して欠陥マスターDBへ

Copyright © 2013 Project Fabre All Rights Reserved.

# 参考：昨年のワークショップ資料．欠陥の情報分類：#1「病理学」風の整理

医学の視点で、定義と用途を重視した永続的・長期的な情報整理を行います。

## 欠陥の情報

- **定義・基礎情報**
  1. 名前・別名
  2. 欠陥の定義・分類
  3. 一般的な推定原因と発生確率
  4. 併発症・合併症
  5. 特記事項
- **検出情報**
  1. どのように検出するか？
  2. 何をキーワードに検索すれば、欠陥がある場所を特定／狭めることができるか？
  3. 定性情報？定量情報？
- **兆候情報**
  1. どうやったらその欠陥に気付くか？
  2. 何をきっかけにすると、欠陥兆候を捉えることができるか？
  3. 定性情報？定量情報？
- **除去方法・予防方法**
  1. 関連する欠陥・連鎖欠陥
  2. 除去時の注意点
  3. 副作用・除去リスク
  4. 再発予防方法

Copyright © 2013 Project Fabre All Rights Reserved.

# 11. バグ分析（具体例）ー例題のバグ票ー

## 「なぜなぜ」分析

- Currencyを使わなかった

⇒**Currencyの使い方を知らなかった**

⇒Currencyを正しく使えるように教育する

## 「なになに」分析

- Currencyを使わなかった

⇒**Currencyの使い方を知っていたが、その時は使わなかった**

⇒既存ソースがdoubleだったので、派生開発時にdoubleを使った
⇒該当部分修正に加え、静的チェックで新規発生を防ぐ、他の浮動小数点型の定義は後に修正

Copyright © 2013 Project Fabre All Rights Reserved.

# 12. 欠陥モデリング(定義)

**誘発因子 (Induction Trigger)**

成果物の中に含まれる、人間の思考の誤りを誘発する“トリガー”となる要素のこと。
誘発因子が存在すれば、開発者能力・経験・技術力と関係なく過失が引き起こされやすくなる。

**過失因子 (Negligence Factor)**

人間の思考や判断の誤りそのもののこと。
欠陥は過失因子の集合(=連続)として生み出される。

**増幅因子 (Amplifier)**

過失の連鎖を助長し、欠陥の混入確率を増幅させる要素。
多くは定量的に測定可能である。外乱・環境特性ともいう。

**欠陥 (Defect)**

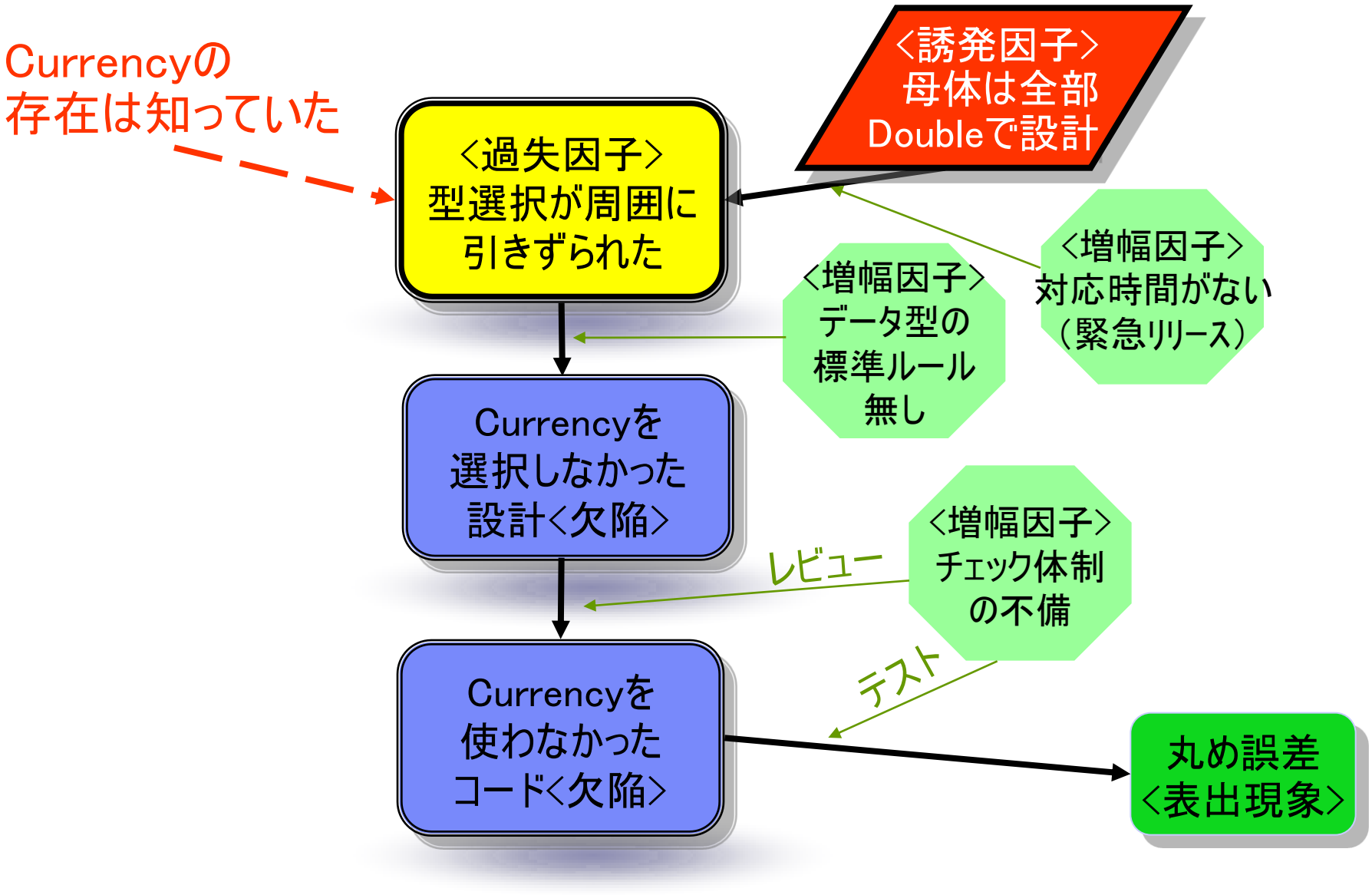
成果物に含まれた、人間の思考の過ちが具現・表出化したもの。不具合・障害等の「現象」を発生させる。

**表出現象 (Incident)**

欠陥によって引き起こされる不具合・障害。
多くは定量的に測定／加算可能。

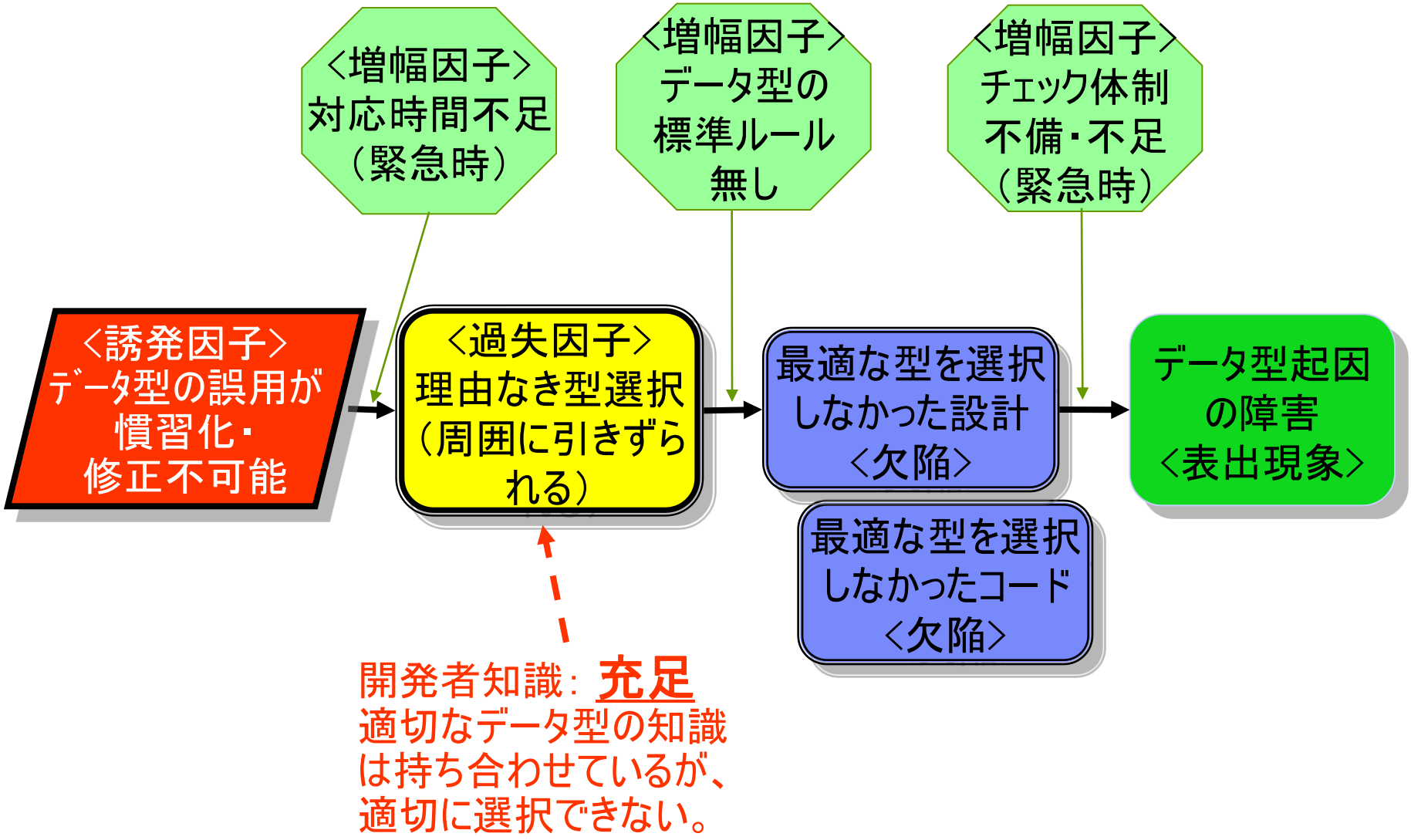
Copyright © 2013 Project Fabre All Rights Reserved.

# 13. 欠陥モデリング(具体例)ー例題のバグ票ー



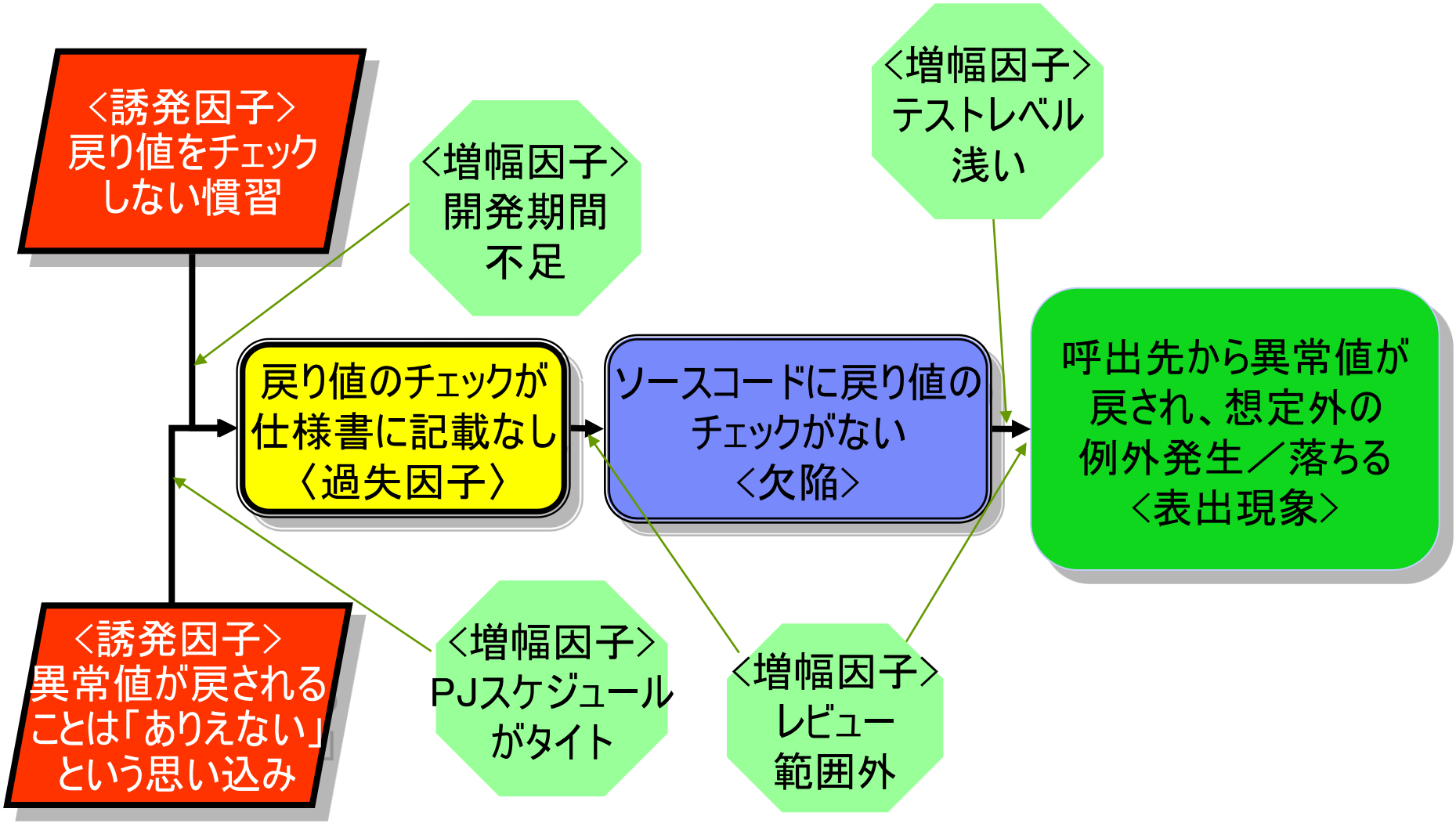

Copyright © 2013 Project Fabre All Rights Reserved. 

# 13. 欠陥モデリング（具体例）ー例題のバグ票を汎用化ー




Copyright © 2013 Project Fabre All Rights Reserved. 

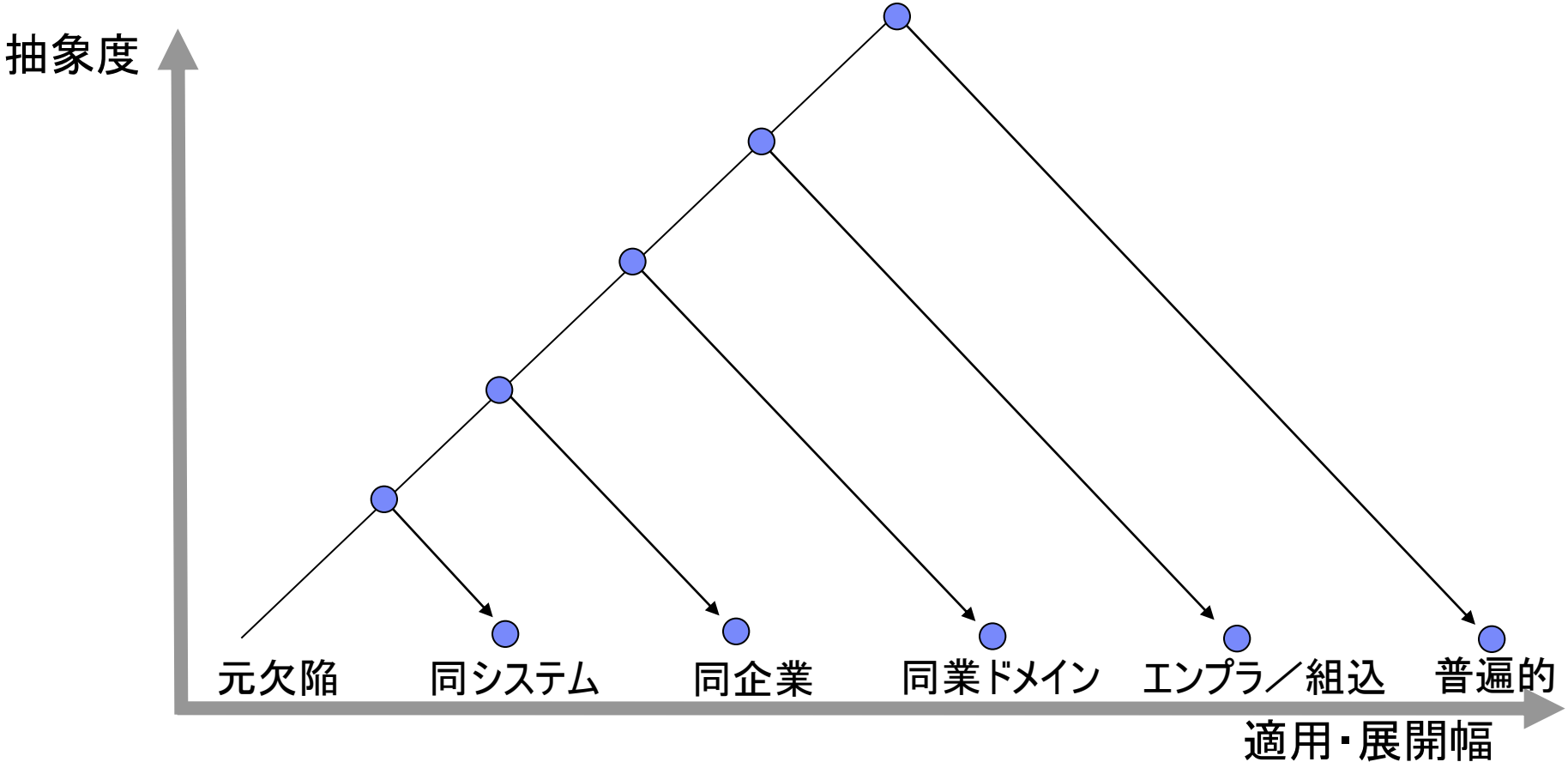
# 13. 欠陥モデリング（具体例）ーおまけー




Copyright © 2013 Project Fabre All Rights Reserved. 

# 14. 欠陥の抽象度と展開幅

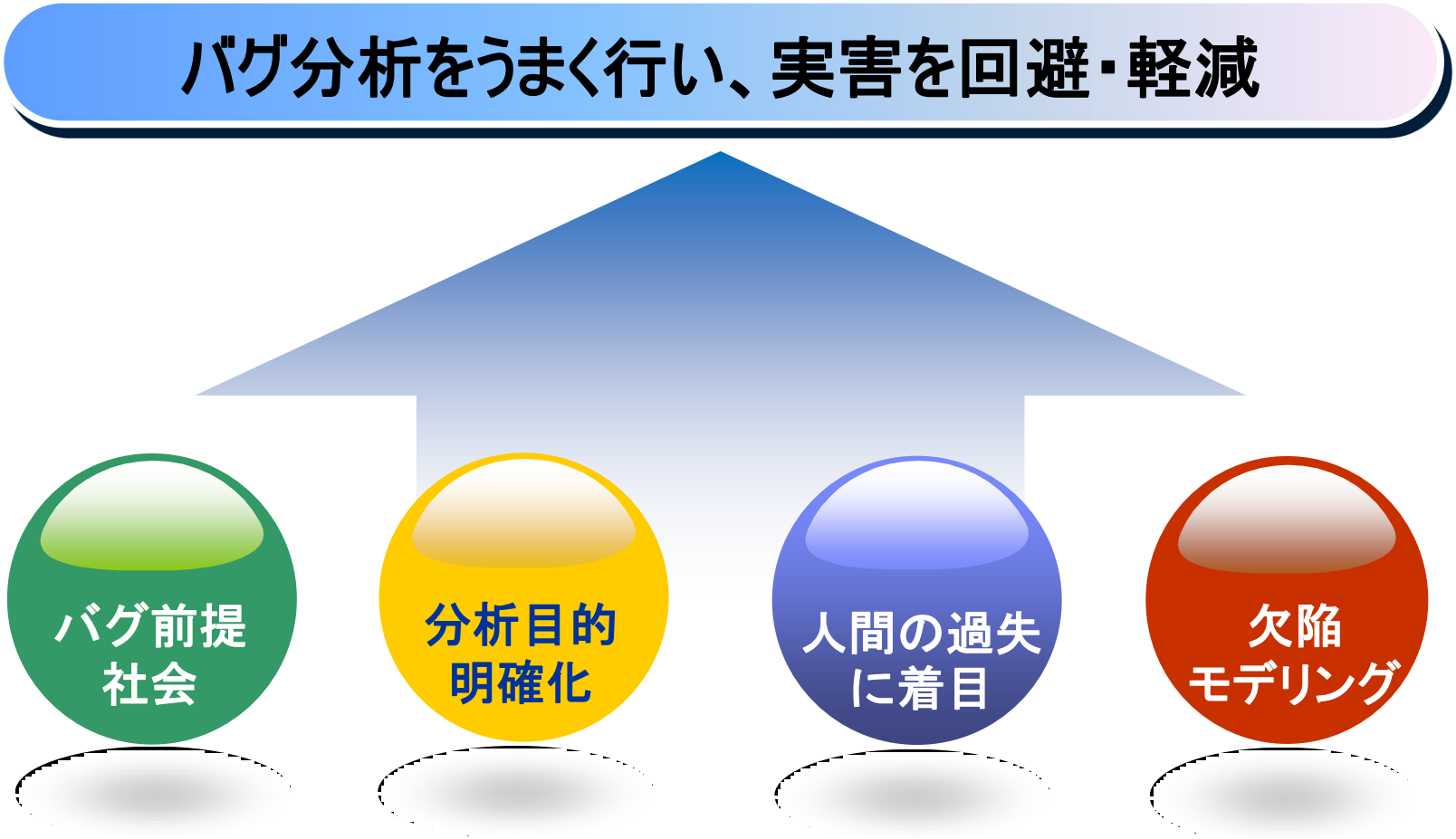
- 欠陥保存時の情報抽象度の高さに依存して、横展開幅（＝欠陥情報の適用範囲）が決定される
- 逆に広範囲な展開を狙わない場合は、抽象度を高めなくてもよい？




Copyright © 2013 Project Fabre All Rights Reserved. 

# 15. 本日のまとめ

- アンチパターン脱却⇒まずまず良いパターン実践⇒モデル化して議論
- 人間が陥る「罠」に注目し、バグ知識と合わせて横展開幅を広げる




Copyright © 2013 Project Fabre All Rights Reserved. 