ソフトウェアプロセスにおけるチェックに対する工程間の依存解析

2K－7

後藤 太樹　佐藤 明良　山之内 徹

NEC C&C研究所

1 はじめに

ソフトウェア開発では，仕様やプログラム中のデータ間の不整合によって発生し得る，バグや作業の後戻り等の問題を防ぐために，開発プロセス中にレビューやテスト等，チェックの工程を設けることが多い．しかし，このようなチェックにはそれ自体に工数がかかるため，全ての工程であらゆるチェックを行えばよいというわけではなく，より適用効果の高い，適切な工程を選んで実施する必要がある．

本稿では，このような開発プロセスにおける「チェック工程の配置箇所」に注目したソフトウェアプロセスの評価について，事例とともに述べる．

2 開発プロセスの評価

ソフトウェアプロセスの評価[1]については成熟度モデル[2]によるプロセス全般の評価等があるが，ここでは「開発プロセスの中でエラーチェックをどの工程で行うべきか」という特定の観点での評価を，そのチェックによってどのような作業の後戻りが発生し得るか，という「チェック工程依存度」をメトリクスとして行う．

但し，ここでの「チェック」とは，開発プロセス中に現れるプロダクト(仕様，プログラム等)に含まれるデータを x_i としたときに，$check(x_1, \cdots, x_i) = \{true, false\}$ で表されるような作業 $check$ のことを指す．

また，「チェックをどの工程で行うべきか」とは，この $check(x_1, \cdots, x_i)$ の入力となるデータ x_i を，開発プロセス中のどの段階のどのプロダクトから取ってきてチェックを行うべきか，ということに相当する．

2.1 評価方法

上記観点によるプロセスの評価は以下の手順で行う．

1. プロセスフロー図を記述する．
2. チェック項目を決定し，各工程，各プロダクトに関するチェック関連情報を1.の図に加える．
3. チェックを行う工程を仮定し，そのチェック工程の記述を2.の図に加える．
4. 3.までのプロセスフロー図に対してチェック工程依存解析を行い，チェック工程依存関係図を得る．
5. チェック工程依存度により比較評価する．

プロセスフロー図とは，後述する2.2節の図2右のような開発工程の流れを記述した図であり，図の四角が工程(作業)を，円が入出力プロダクト(仕様，プログラム等)を表している．また，工程-プロダクト間の矢印はプロダクトの流れる向きを表している．

Dependence Analysis among the Stages related to Check in Software Process
Taiki Gotoh, Akiyoshi Sato and Toru Yamanouchi
C&C Research Laboratories, NEC Corporation

チェック関連情報とは，後述のチェック工程依存解析を行うために必要な以下の情報である．

1. 各工程での作業内容に関する情報．
   - 動作(工程名, X, 操作名, Xが不正時の動作)
     工程「工程名」では，データXに対して操作「操作名」を行っていることを示す．
2. 各プロダクト中のデータの状態に関する情報．
   - 状態(プロダクト名, X, 可視属性, 修正属性)
     プロダクト「プロダクト名」の中での，データXの状態を示す．
3. 各チェック工程のチェック内容に関する情報．
   - チェック({X,Y,…}, チェック識別子)
     データX,Y,…に関してチェック「チェック識別子」を行う．
   - 影響(チェック識別子, X, 操作名)
     チェック「チェック識別子」は，データXに対して操作「操作名」を行った時に結果が変化し得る．

チェック工程依存解析とは，そのチェックを行った場合にどのような後戻り工程が発生し得るかを，チェック関連情報と工程-プロダクト間の関連をもとに導出する，図1の手順(『全体の検査』)による依存解析である．

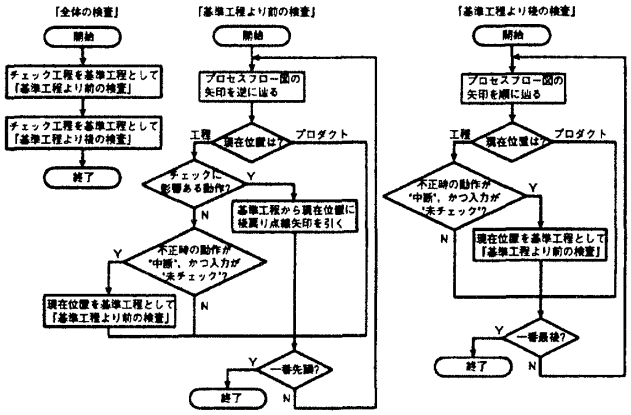

図1: チェック工程依存解析手順

チェック工程依存関係図とは，チェック工程依存解析によって得られる，発生し得る後戻り工程をプロセスフロー図に点線矢印で示した，後述する2.2節の図4のような図である．

チェック工程依存度とは，上記観点によるプロセスの評価を行うための以下のメトリクスである．

M-1: (チェック工程依存関係図中の)点線矢印の数(後戻りのパターン数)

M-2: 点線矢印が跨いでいる工程数の最大値 [M-2-1] と総和 [M-2-2] (後戻りの大きさ)

M-3: 点線矢印の起点の，先頭工程からの距離の最大値 [M-3-1] と総和 [M-3-2] (後戻りの発生箇所)

2.2 評価事例

ここでは評価事例として，図2のような開発プロセスを考える．これはGUI画面を持つアプリケーションにおいて，GUI部品の色，大きさ，位置等の属性 (プロパティ) を実行時にデータベースから読み込んで設定する「プロパティマネージャ」というコンポーネントに関連するコードの，部分的な開発プロセスである．

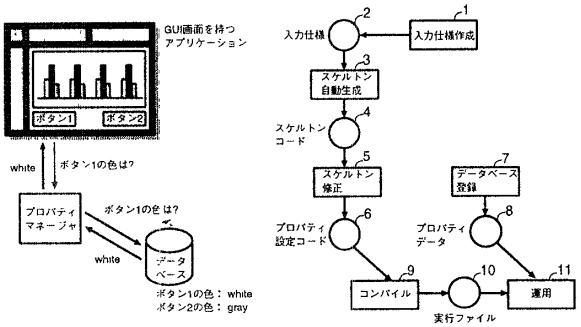

図 2: 概要とプロセスフロー図

この事例のプロセスでは，GUI部品のプロパティを設定する際に対応するデータが存在しないと正しく設定されず，しかもそれは実行画面上では確認しにくい，という問題がある．そのため以下のチェック項目を考える．

チェック項目: 「プロパティ設定コード」の中から参照しようとしているプロパティのデータが，実際にデータベースの中に存在するかどうか．

ここで，このチェック項目に関しては，

チェック1: プロダクト6と8に対してチェック
チェック2: プロダクト4と8に対してチェック
チェック3: プロダクト2と8に対してチェック

の段階でチェックを行うことが可能であるが，以下では，この中の「チェック1」と「チェック3」とを比較評価することとする．但し，この「チェック1」と「チェック3」とがチェック内容に関して等価である，ということは，ここでは前提として考える．

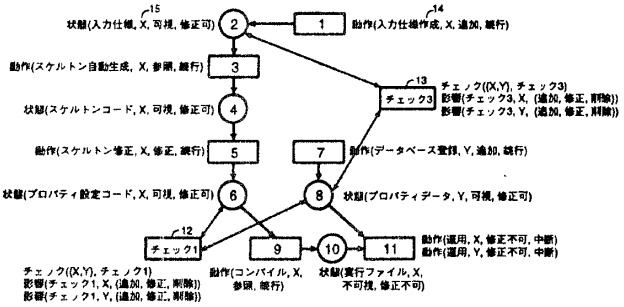

図 3: チェック関連情報を加えた図

まずは，このチェックを図3-12「チェック1」の段階で行う場合を考える．この図3は，図2右のプロセスフロー図に対して，このチェックに関するチェック工程の記述 (図3-12, 13) とチェック関連情報 (図3-14, 15等) を加えたものである．

ここでは例えば，工程3「スケルトン自動生成」では，「データXに対しては参照するだけである，また，Xに不正な値が来てもこの工程で中断することはない」，ということを意味する．また，プロダクト4「スケルトンコード」では，「データXに対して，このプロダクトからは値を参照することも修正することもできる」，ということを意味する．

この図3に対して，図1の手順で「チェック1」に関する依存解析を行うと，図4左のようなチェック工程依存関係図が得られる．この図4左の点線矢印が，発生し得る後戻り工程を示しており，例えば，図4-14の点線矢印は，工程12「チェック1」から工程1「入力仕様作成」への後戻りが発生し得ることを表している．

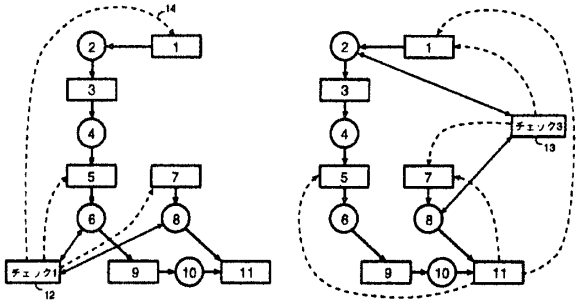

図 4: チェック工程依存関係図

次に，同じチェック項目について図3-13「チェック3」の段階でチェックを行う場合を同様に考えると，この場合のチェック工程依存関係図は図4右のようになる．

この図4左と図4右に対して「チェック工程依存度」を求めると以下のようになり，この結果が，どの段階でチェックを行うべきか，ここでは図3-12と図3-13のどちらで行うべきか，についての1つの判断材料となる．

	M-1	M-2-1	M-2-2	M-3-1	M-3-2
図4左	3	3	5	3	7
図4右	5	4	9	4	11

表 1: チェック工程依存度による比較評価

実際には，これらの情報に加えて，各工程の工数見積り等の情報も合わせて考えることにより，より適用効果の高い，適切な工程でのチェックが可能になり得る．

3 おわりに

本稿では，開発プロセス中のチェック工程に注目した「エラーチェックをどの工程で行うべきか」という観点でのソフトウェアプロセスの評価について述べた．

今後の課題としては，評価方法を詳細化すること，他の観点での評価方法について検討すること，実際のプロジェクトで事前評価を行うこと，等が挙げられる．

参考文献

[1] 藤野喜一:「ソフトウェアプロセス評価の動向」，情報処理, Vol.36, No.5, pp.399-408, 情報処理学会, 1995.

[2] W.S.Humphrey:「ソフトウェアプロセス成熟度の改善」，藤野喜一監訳，日本電気ソフトウェアプロセス研究会訳，日科技連, 1991.