2B-2

タグ付与による開発成果物のトレーサビリティに対する考察

阿部　玲子　　篠崎　衛　　山足　光義　　徳本　修一　　田村　直樹　　高橋　洋一

三菱電機株式会社　情報技術総合研究所

１．はじめに

トレーサビリティ技術は、ソフトウエア開発の際、各フェーズ間またはフェーズ内成果物の整合性を管理するために用いられる要求管理技術の一つである[1][2]。本技術は、各成果物内に任意のタグを記述しておくことによりタグ付け箇所のトレースを可能とするもので、利用する際にプロジェクト毎にタグ付けの仕様を決める必要がある。ところが、この仕様策定作業は一般的に難事である事が知られている。本稿では、仕様項目の一つであるタグ付け粒度の策定に対して注目し、利用目的に着目した「タグ付け粒度決定方式」を提案する。

２．トレーサビリティ技術

各成果物内に任意のタグを記述しておくことで、以下を目的としたタグ付け箇所のトレースを可能とするトレーサビリティ技術が存在する。

① 上位開発フェーズの要件が、下位フェーズのどこに展開されているかを示す。

② 各フェーズ毎に、仕様に対応する試験がどれであるかを示す

各成果物に記載したタグをマトリクス形式で一覧表にしたものを、トレーサビリティマトリクスと呼ぶ。トレースを行う場合、このトレーサビリティマトリクスを参照することで、上記①、②のトレースを行う。このため、成果物に付与するタグの数を増やすと、より細かい粒度でのトレースが可能になる。

しかし、粒度が細かくなるにつれ、タグ付け作業が複雑化しコストが増加する。よって、粒度をむやみに細かくするのではなく、適切な粒度で目的の効果を得る必要があり、目的やコストを考慮した上で、プロジェクトに適切なタグ付けの粒度を定めることが重要となる。

３．実プロジェクト適用時の課題

適用プロジェクトは昨年度から継続しているソフトウェア開発であり、成果物およびその仕様の変更は不可であった。このため、仕様策定期間および実作業負荷を極力少なくするという制約が存在し、トレース効果を最低限とする方針で粒度の決定が行われた。結果、開発における成果物とフェーズは図１のとおりとなった。

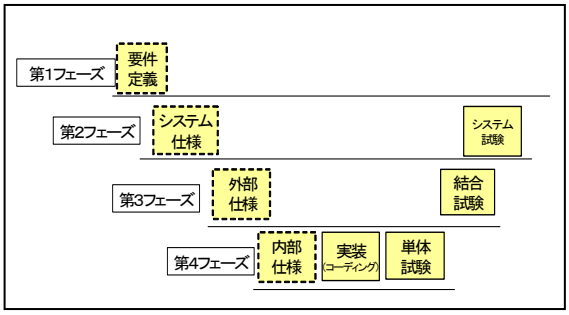

図１．開発フェーズに対する成果物の位置付け

本プロジェクトでは、トレース範囲はドキュメントのみとし、タグ付けを図の第４フェーズ「実装」を除く部分とした。粒度は、第４フェーズ「内部仕様」にて記載される仕様書の見出し（目次項目）を基準とすることとした。内部仕様書の見出し構成は、章（パッケージ）・項（クラス）・節（パブリックメソッド）となっており、本開発では「項」に対してタグ付けを行った。しかし、上記の仕様でトレーサビリティマトリクスを作成した結果、作成したトレーサビリティマトリックスを有効に利用することができなかった。これは、制約ありき、かつ技術者の感覚のみで　粒度を決定してしまったために生じた結果である。実際のプロジェクトへ適用する場合、トレーサビリティ技術に精通している見識者がプロジェクト内に存在するとは限らない。すなわち、トレーサビリティの利便性は理解しているが、実際にプロジェクト内においてどのように結果を利用するかが、トレーサビリティタグの仕様策定時点では明確にできない場合もある。

このように、実適用においては、トレーサビリティ技術を理想的な形式で適用することが難しく、明確な利用ビジョンも定められない場合がある。更に、適切な粒度策定が困難である場合に対応できる手段がないという課題がある。

４．タグ付粒度指標式

本課題を解決するために、誰にでも利用できるタグ付け指標として、タグ付け粒度指標式を考案した。これを社内、部門内などに適用し、成果物や開発スタイルの類似した組織内で指標

A study of traceability of development product by tag giving

ABE Reiko,SHINOZAKI Mamoru, TOKUMOTO Shuuichi, TAMURA Naoki, TAKAHASHI Youichi

Information Technology R&D Center, Mitsubishi Electric Corporation

として利用することを提案する。

ドキュメント数、推定メソッド数、可能な作業量などから推奨タグ付けレベルを導くための式を以下に示す。

$$Y = \frac{\sum_i (time_{1i} \times n_{1i})}{N \times time_2 \times n_2} + R \qquad (式１)$$

Y	: 推奨粒度
$time_{1i}$	: 成果物 i のタグ付け作業時間
n_{1i}	: 成果物 i のタグ付け作業者数
N	: 成果物の量（数）
$time_2$	: 保守作業時間
n_2	: 保守作業者数
R	: 要求数とソース量の比率

式１を用いて算出した値から、具体的なタグ付内容を得るためには、タグ付け粒度表を別途作成する。

５．評価

５．１．　タグ付け粒度指標式の適用と粒度表

前述のプロジェクトに対し、今回提案する方式を適用する。まず、プロジェクトタグ付け粒度表は以下のように作成した。

値	推奨トレース内容	最小タグ付け粒度
１	ドキュメント	ドキュメント
２	大項目要求	要求表大項目
３	中項目要求	要求表中項目
４	小項目要求	要求表小項目
≈	≈	≈
１３	ドキュメント・ソース	ドキュメント・モジュール
１４	大機能・ソース	大機能（章）・パッケージ
１５	中機能・ソース	中機能（項）・クラス
１６	小機能・ソース	小機能（節）・パブリックメソッド

表１　粒度表

式１にプロジェクトによる値を当てはめた結果、算出値は粒度４となった。

よって、表１の粒度表から、ドキュメントに対する小機能トレースができるよう、内部仕様書見出しの節単位が最小粒度となるようタグ付けを行うことが推奨される。

５．２　指標式利用前後の比較による式評価

タグ付け粒度表から、式利用前のタグ付け粒度を逆引き計算すると１１となった。ここで、保守作業コストCを以下と定義する。

$$C = time_2 \times n_2 \qquad (式２)$$

式２を用いて式１を組み替えると保守作業コストが算出できる。

$$C = \frac{\sum_i (time_{1i} \times n_{1i})}{N(Y - R)} \qquad (式３)$$

この結果を利用して、指標式利用前後で保守作業コストを比較すると、利用前のコストは式利用後のコストを上回る。式利用後の保守作業コストは、式２を用いたプロジェクトで対応可能な想定保守作業量を利用した値である。よって、利用前は想定コストオーバーであり、粒度レベル１１では、想定よりも保守作業の負荷が高いため、作成したマトリクスを利用しにくいことが分かる。つまり、実際に利用できるトレーサビリティマトリクスを得るためには、タグ付作業時の負荷を上げ、タグ付け粒度を細かくする必要があると言える。

一方、式利用後の粒度レベルは１６であり、式利用前の粒度レベル１１と比較するとタグ付け作業負荷は高まるが、細かい粒度でトレースができる。これにより、トレーサビリティマトリクスの活用シーンが増えるため、式の利用効果があることが確認できた。

６．まとめ

本稿では、技術者のレベルによらず、目的にあった効果を得るために必要なトレーサビリティ粒度を決定するための指標となる「タグ付け粒度指標式」を提案した。さらに本式を、弊社実プロジェクトの一つに対して適用し、効果があることを確認した。

ただし、本式の算出値は指標値であるため、算出値の粒度レベルで問題がないかどうかは別途確認が必要である。問題がなければ算出した粒度でタグ付けを行えばよいが、そうでない場合は、算出値に係わる値のうち、変更可能な値を変更し、算出結果を吟味しながら調節するという課題がある。

今後は、上記の課題を解決すると共に、類似する他プロジェクトへ本方式を適用し、現行の式を再評価し、継続的に式の改善を実施していく計画である。

７．参考文献

[1]Mary Beth Chrissis, JASPIC CMMI V1.1 翻訳研究会　他, “CMMI 標準教本”, 日経 BP 社, Aug,2005

[2]Automotive SIG, “Automotive SPICE プロセスアセスメントモデル(PAM)”,The SPICE User Group 2005, Jul,2006