要求プロセスにおける開発データの獲得方法の検討

糸賀 裕弥[†1]

アブストラクト 要求プロセスにおける開発データを獲得することで，要求プロセスの終了・要求の誤りと修正の定量的解析・効率的なレビューが可能である．しかし，自然言語で記述され，構造のない上流工程の成果物の開発データの獲得には困難がともなう．上流工程の開発データの獲得について検討したい．

Metrics Data Collection on Requirements Process

Hiroya ITOGA[†1]

Abstract We can review software requirements specification, detect, revise errors in it, and decide the time to finish requirements process if we collect the metrics of the process. There is difficulty, however, to collect metrics from requirements written in natural language without document structure. In this position paper and workshop we discuss how to collect the metrics of upper stream of software development process.

1. はじめに

ソフトウェア開発における要求仕様書の品質は，開発されるソフトウェア製品の品質に影響を与える．要求定義はソフトウェア開発における最も初期の手順であり，成果物である要求仕様書をもとに，その後のソフトウェア開発が進む．さらに要求仕様書はステイクホルダ（利害関係者）間，例えば顧客側と開発側の合意の根拠となる文書であり，納期やコストにも関わる．

これまでは，ソフトウェア要求仕様書における 8 つの品質特性，妥当性・非あいまい性・完全性・無矛盾性・重要度のランク付け・検証可能性・追跡可能性・変更可能性に基づき，ソフトウェア要求仕様書の品質を向上させることが重視されてきた．特に，前半 4 つの品質特性に関わる要求の誤りの発見と修正により，ソフトウェア要求仕様書の品質を向上させ，ソフトウェア製品の品質を維持するための努力である[1].

しかしながら，この努力には困難が伴う．妥当性については，変化し易くかつ明示されないステイクホルダの要望に基づいた，誤りの発見と修正の繰り返し，非あいまい性・完全性・無矛盾性については，ステイクホルダ間において意識されない未合意部分の発見と修正の繰り返しにより品質を維持する．すなわち，発見と修正の繰り返しプロセスが完了する条件，あるいは十分な品質になったとしてプロセスを終了（打ち切る）条件があいまいである．このため，要求プロセスが長期にわたり，ステイクホルダの要望がさらに変化し，要求プロセスの終わりが見えない，という状況が起こりうる．

このため要求プロセスの品質を高め改善するモデルや方法が必要である[2]．そこで，本研究では，

1） 要求プロセスをいつ，どのように終えるのか
2） 要求プロセスにおける誤りと修正の定量的解析
3） 要求仕様の効率的なレビュー

のため，要求プロセスに作業履歴の活用や統計解析手法を適用する方法を検討している．

これまで，パーソナルソフトウェアプロセス手法（Personal Software Process）[3]を要求プロセスに適用する方法を検討してきた[4].

2. 要求プロセスへの PSP の適用

要求プロセスは，1）要求獲得，2）要求分析，3）要求仕様化，4）要求の検証・妥当性確認・評価，の 4 つの要素プロセスからなる[5].

要求獲得においては，ステイクホルダから，文書，シナリオ（ユースケース），インタビューといった様々なかたちで表現された要望を得る．成果物は獲得要求である．要求分析においては，獲得された要求を整理・分類して，要求間の構造や依存関係，一貫性や整合性を明らかにする．成果物は分析要求である．要求仕様化においては，分析要求を文章や図表を用いて仕様化する．成果物は要求仕様書であり，前者 2 つと異なり，

†1 立命館大学 情報理工学部 情報システム学科
Department of Computer Science, College of Information Science and Engineering, Ritsumeikan University

ある程度の構造を持つ[6]. 要求の検証・妥当性確認・評価においては，要求仕様書をもとに，その品質を確認する. 成果物は要求仕様書の品質評価である.

パーソナルソフトウェアプロセスに基づき，1）要求プロセスの時間管理，2）要求プロセスの欠陥管理，3）要求プロセスのレビュー管理の手法を導入する. 要求プロセスの時間管理と欠陥管理は相互作用しつつ要求および要求仕様書の品質を向上させるため手法，要求プロセスのレビュー管理は効率的なレビューを実現するための手法である.

要求獲得プロセスにおける成果物（獲得要求）および，要求分析プロセスにおける成果物（分析要求）は，構造化されていない表現であることが多い. 前者においては例えば，要望リスト・利用者のふるまいを記述したシナリオ・性能に関する覚え書きなど，後者においてはユースケース図・シーケンス図・データフロー図といった図形式の表現も含まれる. 要求仕様化プロセスにおける成果物は要求仕様書であり，比較的構造が定まった表現である.

そこで，要求仕様書の 1 項目に該当する分析要求の一部，分析要求の一部に該当する獲得要求の一部，それぞれに対して，作業時間と作業量をもとに時間管理を行う. また，要求仕様書の 1 つの誤りの原因となった分析要求の一部，分析要求の誤りの原因となった獲得要求の一部に対して，要素プロセスごとにレビューを行い，欠陥が埋め込まれた要素プロセスと発見・修正された要素プロセスを記録することで，欠陥発生率と欠陥除去率の管理を行う.

3. PSP の適用における問題点

元来のパーソナルソフトウェアプロセス（Personal Software Process）においては，成果物はプログラム（ソースコード）であり，論理の単位がプログラムのステップ数（Lines of Codes, LOC）である. そのためプロセスにおける成果物の時間管理，成果物の量の管理が容易である. また欠陥についても，コードレビュー，コンパイラによるチェック，テストといった複数の手法によって管理されている.

要求プロセスにPSPを適用した場合，成果物が多岐にわたり，構造化されていないため，このような成果物の量の管理および欠陥の管理が困難である.

これまで提案した手法においては，要求獲得プロセスの成果物としては要望リストを，要求分析プロセスの成果物としては要望リストに対応したソフトウェアのふるまいを記述したシナリオ（ユースケース）を，要求仕様化プロセスの成果物としては構造化された要求仕様書を用いてきた.

しかし，あらかじめ決められている形式から外れた成果物の扱いや，決められた形式に合致させようとした場合の要求の詳細度などにおいては，課題が多い.

4. 議論と今後の課題

ソフトウェア開発プロセス，特に要求プロセスを含む上流工程においては，ソースコードではなく要求仕様書，設計書のような比較的構造を持ったものから，開発メモや議事録といった文書と呼びにくいものまで，多種多様なデータが用いられる. また，ソースコードと異なり，変更・ビルド・テスト・レポジトリへのコミットといった定型の処理も行われる保証が無いため，データの獲得と管理が困難だと考えられる.

最近は，ソースコードレポジトリ（例えば Github）において，自然言語によるドキュメントの変更管理を行っている例も聞こえてきており，版単位での管理は可能なようにも思われる.

要求項目ごと，要望リストごとといったより小さな単位，あるいは意味のある単位での変更管理を行う手法について検討を行いたい.

また，ソースコードにおける欠陥管理においてはバグ追跡が主である.

しかし，要求プロセスにおいては，欠陥のもととなった要望の理由までを欠陥の原因として把握する必要があり，その際の要求分析・要求獲得における問題点も明らかにする必要がある. このような作業履歴についてもデータの獲得・管理が必要となる.

参考文献

[1] 大西淳，郷健太郎，要求工学，ソフトウェアテクノロジーシリーズ，Vol. 9，共立出版，2002.

[2] S. Beecham, T. Hall, and A. Rainer, Defining a Requirements Process Improvement Model, Software Quality Journal, Vol. 13 (3), pp. 247-279, 2005.

[3] Watts S. Humphrey, Introduction to the Personal Software Process, Addison Wesley Longman, 1997.

[4] 糸賀裕弥，要求プロセスへのパーソナルソフトウェアプロセスの適用，信学技報知能ソフトウェア工学，Vol. 113, No. 414, 25-30, 2014.

[5] 一般社団法人情報サービス産業協会 REBOK 企画 WG 編，要求工学知識体系，近代科学社，2011.

[6] IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications, 1998.