ソフトウェア開発データあれこれ

門田暁人†

本稿では，ソフトウェア開発データに対する様々な疑問や課題を述べる．

Some Thoughts on Software Development Data

Akito Monden†

This paper presents questions and problems of software development data.

1. はじめに

筆者はこれまでに，産学官の共同研究を通して，ソフトウェア開発に関する様々なデータを分析してきた．本稿では，これまでに筆者が感じてきた疑問点や課題を述べる．

2. 生産性の定義

従来，生産性に影響する要因の分析が盛んに行われてきた[1]．また，企業においては，製品ごと，部署ごとに生産性の目標値を設けることも行われている．ただし，生産性の定義には問題があると考えられる．

生産性は，工数あたりの開発規模で与えられることが多い．しかし，この定義の下では，要件定義，設計，テストに労力をかけず，コーディングばかりをやっているプロジェクトの生産性が高くなってしまう．また，要件定義や一部のテストが請負外となっているプロジェクトでは，生産性が見かけ上高くなる恐れがある．さらに，差分開発ではコーディングをほとんど行わず，テスト工数だけが莫大となるケースがあり，見かけ上の生産性が著しく低いものとなる．

より良い生産性の定義についてはあまり議論されていないように思う．一つの考え方としては，プロジェクトの全工程をまとめて生産性の議論を行うのではなく，工程ごとに生産性の議論をすべきであると思う．ただし，テストの生産性については，定義が難しいように思う．

3. 規模の定義

3.1. 新規 vs 改造 vs 流用

ソフトウェア規模の測定においては，プロジェクトにおける「開発規模」を測定することが望まれる．そのために，特に派生開発や差分開発では，新規，改造，流用に分けて規模を計測することが必要である．ただし，その計測方法については，業界標準がないように思う．一つの方法は，ソースファイル単位で，全くの変更がなければ流用，1行でも変更していれば改造，それ以外は新規と判断することが考えられる．ただし，「ソースファイル単位」というのは適当でないかもしれない．

この問題は，バグ密度やテストケース密度による品質管理を行う場合にも影響する．テストが十分か否かを判断する一つの尺度としてテストケース密度（テストケース数÷規模）が用いられるが，分母における規模として何を与えるのが適当であるかについては難しい問題である．バグ密度（バグ数÷規模）の計測においても同様である．企業によっては，新規＋改造＋流用×0.1 を開発規模とする場合もあるが，その妥当性は不明である．

さらに言えば，特にテストにおける「規模」の定義は，どの部分をテストすべきかという問題にもかかわってくる．共通ライブラリを修正した場合，そのライブラリに依存するソースファイルについては，変更を加えていなくてもすべてテストすることが望ましい（特に，結合テスト以降のテストでは）．そのため，コーディングにおける開発規模とテストにおける規模は異なる定義を使うべきであるかもしれない．

3.2. FP vs SLOC

ソースコード行数は規模の尺度としてよく用いられているが，プログラミング終了後でないと計測できない，プログラミング言語に依存する，自動生成コードやテンプレートの行数を取り除いてカウントすることが容易でないという課題がある．一方，ファンクションポイントは，設計時に計測可能であり，言語にも依存しない．このことから，近年ではファンクションポイントを採用する企業が増えている．

†奈良先端化学技術大学院大学情報科学研究科
Graduate School of Information Science, Nara Institute of Science and Technology

ただし，ファンクションポイントを採用すればソースコード行数の計測は不要かと言うとそうではない．株式会社 NTT データにおいて実施された，同一仕様のソフトウェア開発を複数の企業に発注した事例[2]において，同一仕様，同一ファンクションポイントであってもソースコード行数には大きなばらつきがあり，そのために開発工数にも大きなばらつきがあることが示された．このことは，ファンクションポイントだけでは開発工数を正確に見積もることはできない，ということと，コーディングや詳細設計における実装上の工夫や問題点というのは少なからず工数に反映されるので，ソースコード行数が依然として重要であることを示唆している．ただし，ファンクションポイントとソースコード行数をどのように使い分けるかについては，まだまだ検討の余地があるように思う．

4. 工数の定義

特に大規模プロジェクトにおいて，工数データはあまり信頼できない．なぜならば，外注先における開発工数を正確に計測することが多くの場合難しいためである．外注先の工数は，契約時の見積もり工数がそのまま計上されている場合が少なくないようである．そのため，外注率がゼロであるプロジェクトのみを対象として，工数についての分析・議論をすべきであるという意見もある．ただし，外注元の立場としては，契約時の工数が把握できていれば問題ないともいえる．この点については整理が必要であるように思う．

JUAS（日本情報システムユーザ協会）のソフトウェアメトリックス調査報告書では，規模や工数に加えて発注金額としてのコストが強く意識され，金額あたりのバグ数，といった尺度も分析されている点は興味深い．

5. データ間の関係についての疑問

5.1. 規模と生産性の関係

生産性の要因分析においては，規模のファクターを無視することはできない．一般には，規模が大きいほど生産性が低くなると言われている．規模が大きいほど開発者が増え，コミュニケーションコストが増大するためである．規模＝ファンクションポイントと捉えた場合，IPA-SEC のデータ[3]ではこの関係が成立している．しかし，規模＝ソースコード行数と捉えた場合，なぜか規模が大きいほど生産性が高くなっている．この傾向は，他社のデータにおいても見られたため，何らかの共通の原因があると考えられる．

この状況証拠は，大規模プロジェクトほどコードクローンが多いことを示唆している．問題は，なぜ大規模プロジェクトほどコードクローンが多くなりやすいのかである．一つの解釈として，規模が大きいプロジェクトは，多数の企業が参加するため，各社にテンプレートとなるソースコードを配布する場合がある．そのため，会社間でコードクローンが生じている可能性がある．ソースコード行数の計測においては，コードクローンを考慮した計測が望ましいと考える[4].

5.2. 規模と出荷後バグ数の関係

ソフトウェア品質は，出荷後に発見されるバグ数（もしくはバグ密度）によって評価されることが多いため，その計測の実態を知ることは重要である．

IPA-SEC のデータでは，規模が小さいプロジェクトほど出荷後バグ（バグ密度）のばらつきが大きく，規模が大きなプロジェクトほど安定して出荷後バグが少ないように見える．

一つの解釈は，大規模プロジェクトほど，細かいバグは記録されていないのではないだろうか．ただし，なぜそのようなことが起こっているかについては不明である．もう一つの解釈は，規模が小さいプロジェクトは，より大きなプロジェクトの一部となっている場合があり，このようなプロジェクトでは，「出荷後バグ」というのは，社間結合テストや総合テストでの検出バグを含む可能性がある．いずれにしても，ソフトウェア品質＝出荷後バグと捉えるならば，より厳密な議論が必要であるように思う．

6. おわりに

本稿では，筆者が日ごろ感じている疑問点や課題を思いつくままに述べた．ワークショップではこれらの点について議論できれば幸いである．

参考文献

[1] Maxwell, K., and Forselius, P. “Benchmarking Software-Development Productivity,” IEEE Software, 17 (1), 2000, 80-88.

[2] 端山毅, “定量データ分析の実際”, 平成 23 年度第二回エンピリカルソフトウェア工学研究会, 2012.

[3] 独立行政法人情報処理推進機構（IPA）技術本部ソフトウェア・エンジニアリング・センター（SEC）, “ソフトウェア開発データ白書 2012-2013”, 独立行政法人情報処理推進機構 SECBOOKS, 2012.

[4] 門田 暁人, 内田 眞司, 松本 健一, "実装者に依存しないプログラム規模尺度の構築の試み," コンピュータソフトウェア, Vol.28, No.4, pp.377-382, November 2011.