自主研究

ソフトウェア開発における
フレームワーク使用と生産性

ソフトウェア開発におけるフレームワーク使用と生産性

松本　健一　奈良先端科学技術大学院大学　情報科学研究科
大岩佐和子　押野　智樹　一般財団法人　経済調査会　調査研究部　第二調査研究室

はじめに

ソフトウェア開発、特に、開発言語にJavaを用いた開発では、開発ツールの一種である「フレームワーク」の活用が広がりつつある。フレームワークとは、ソフトウェアの枠組みや汎用的な機能のひな型をあらかじめ用意した開発環境のことで、コーディング作業等の省力化により、ソフトウェア開発の生産性向上が期待される。実際、経済調査会が毎年実施している「ソフトウェア開発に関する調査」によると、フレームワークを使用するソフトウェア開発プロジェクトの割合は、使用状況を確認する設問を追加した平成19年度には45.0%であったが、直近の平成27年度では74.3%まで増加している。また、フレームワークの使用により、開発すべきソースコード量が減少し、生産性が大幅に向上したとの報告もある[1]。

本稿では、フレームワークの使用が、ソフトウェア開発における生産性に与える影響を分析した結果について報告する。特に、フレームワークとの親和性の高いJavaを主な開発言語とするプロジェクトについて、開発規模の大小や工程別にその影響を見ていく。

1 利用データ（分析対象データ）

分析に用いるのは、経済調査会が平成13～27年度に実施した「ソフトウェア開発に関する調査」で収集したプロジェクトデータ（ソフトウェア開発データリポジトリ）である。同調査は、ソフトウェア開発における生産性、工数、費用に及ぼす要因の特定などを通じて、ソフトウェア開発の実態を明らかにし、その成果を公表することを目的として、平成10年度からほぼ毎年実施している。同調査では、分析用データとして平成13～27年度までの15年度分延べ2,085プロジェクトのデータを蓄積しているが、本稿で用いるのは、次の条件を満足する167プロジェクトのデータである。

- 主な開発言語がJavaである（Javaで記述された機能量が全機能量の50%を超えている）。
- 経済調査会が共通フレームと対応付けし、定義した開発工程区分のうち、基本5工程（基本設計、詳細設計、ソフトウェア構築、結合テスト、総合テスト（ベンダ確認））※1がすべて実施されている。

※1 一般にソフトウェア開発では、設計を行い（設計工程）、設計に従ってプログラムを作成し（製造工程）、プログラムの動作をテストする（テスト工程）ことが実施される。経済調査会では、ソフトウェア開発の工程を共通フレームと対応付けて、システム・ソフトウェア要件定義、基本設計、詳細設計、ソフトウェア構築、結合テスト、総合テスト（ベンダ確認）、総合テスト（ユーザ確認）に区分した。各工程と共通フレームの対応関係については「月刊 積算資料」を参照されたい。なお、本稿ではシステム・ソフトウェア要件定義、基本設計をまとめて基本設計とし、基本設計から総合テスト（ベンダ確認）までの基本5工程を分析対象とした。

- ソフトウェアの規模を表す「実績FP規模」が記されており、値が10FP以上である。
- 開発全体の「実績工数」が記されており、値が1人月以上である。
- 開発全体での工数密度の対数値が、平均値±標準偏差×3の範囲である。（外れ値の除去が目的。工数密度の対数値は正規分布に従うと見なすことができる。この基準により除外されたプロジェクトは1件であった。）

分析に用いた主なプロジェクト特性値は、規模（実績FP規模）、工数、そして、生産性指標として（100FPあたりの）工数密度の計3個である。これら特性値の定義を**図表1**に、分析対象プロジェクトにおける基本統計量を**図表2**に、それぞれ示す。

なお、本稿では、先述の通り、生産性指標として、一般的な「規模／工数」ではなく、その逆数となる工数密度を用いている。ソフトウェア開発における生産性は、開発工程で生成・利用されたプロダクトの規模を開発工数で割った値を用いることが多い。しかし、本稿では工程別の生産性について論じようとしており、収集したプロジェクトデータには各工程で生成されたプロダクト規模データが含まれておらず、また把握も困難である。そのため、最終成果物の規模である「実績FP規模」を用いることとした。当然のことながら、実績FP規模は、工程に関係なくプロジェクト全体で一定である。この実績FP規模を分母にすることで、規模の異なるプロジェクト間で工数を比較できるだけではなく、工程間の工数が比較できることとなる。とすれば、「規模／工数」ではなく、「工数／規模」とする方が自然である。

また、本稿では、分析対象プロジェクト167件で使用されている様々なフレームワークを、**図表3**の通り4つに分類して分析を行う。この分類にもとづくフレームワークの使用状況は**図表4**のとおりである。

図表１　分析に用いたプロジェクト特性値

プロジェクト特性値	単位	定義
規模	FP	未調整ファンクションポイントの実績値（実績FP規模）
工数	人月	ソフトウェア開発の実績工数
工数密度【生産性指標】	人月／100FP	100FPあたりの工数を表す．生産性指標として工程毎に算出 （100FPあたりの）工数密度 ＝（当該工程における）工数／規模×100

図表２　プロジェクト特性値の基本統計量（外れ値を除いた分析対象167プロジェクト）

	平均値	標準偏差	最小値	第１四分位数	中央値	第３四分位数	最大値
規模（実績FP規模）	1,810	3,071	48	359.0	800.0	1,855	26,572
工数（人月）	213.7	649.7	2.4	23.5	53.0	164.3	7,471
工数密度（人月/100FP）	9.75	7.50	1.21	5.00	7.34	13.03	50.13

図表３　フレームワークの定義・特徴

フレームワーク名・種別	定義・特徴
Struts	JavaをWebアプリケーション開発のためのフレームワーク
自社開発フレームワーク	社内用に開発された独自のフレームワーク
部品型フレームワーク	事務処理システム向けに画面等のソフトウェア資産を機能部品として提供するフレームワーク
その他フレームワーク	上記以外のフレームワーク

図表４　フレームワークの使用状況

使用フレームワーク名等		プロジェクト数
(a)　フレームワーク使用		93
内訳	(a_1)　Struts	19
内訳	(a_2)　自社開発フレームワーク	11
内訳	(a_3)　部品型フレームワーク	22
内訳	(a_4)　その他フレームワーク	41
(b)　フレームワーク不使用		12
(c)　フレームワーク使用状況不明		62

2 フレームワーク使用が生産性に与える影響

ここでは、まず、フレームワークの使用・不使用による規模、工数、および、工数密度の違いについて見ていく。なお、工数と工数密度は、開発全体における値である（基本5工程以外の工程での工数も含めた値である）。**図表5**に、フレームワークが使用された93プロジェクトとフレームワークが使用されなかった12プロジェクトそれぞれにおける、規模、工数、および、工数密度の基本統計量を示す。

一見すると、フレームワークが使用されたプロジェクトの方が使用されなかったプロジェクトよりも規模が大きく、その一方で工数については両者に差がないことから、工数を規模で割って得られる工数密度は、フレームワークを使用したプロジェクトの方が小さくなる傾向にあるように見える。よって、フレームワークの使用により、工数密度が小さくなる、すなわち、生産性が向上する、と結論づけたいところであるが、統計的検定を行うと、有意水準5%で有意な差が認められるのは規模と工数密度の標準偏差のみで、いずれの平均値にも有意差は認められない。

なお、**図表5**では、有意差が認められた2つの標準偏差を網掛けで示す。

統計的検定の結果から言えることは、

「フレームワークが使用されたプロジェクトでは、フレームワークが使用されなかったプロジェクトに比べて大規模なソフトウェアも開発しているが、両者の工数に大きな差はなく、工数を規模で割って得られる工数密度のばらつきは、フレームワークが使用されたプロジェクトの方が小さい」

ということになる。これが、フレームワーク使用の効果であるならば、

「フレームワーク使用により、開発するソフトウェアの規模の違いが工数に与える影響は抑制され、工数密度のばらつきが小さくなる。」

となる。工数密度そのものが低下するわけではないが、小規模なソフトウェアの開発から大規模なソフトウェアの開発まで、そのばらつきが小さくなることは、プロジェクト管理の観点からは有益である。

3 規模と工数の関係に与える影響

1）フレームワーク使用の有無

2で示したように、フレームワークが使用されることで、規模が工数に与える影響は抑制される。このことは、規模と工数の関係に変化をもたらすことを意味する。そこで、ここでは、フレームワーク使用が規模と工数の関係にどのような影響を与えるのか、回帰分析により調べてみる。

図表6に、フレームワークが使用されたプロジェクト、および、使用されなかったプロジェクト、それぞれについて、規模（実績FP規模）と工数（実績工数）の関係を示す散布図（両対数）とそれらの間の回帰式（累乗近似式）とその決定係数R^2を示す。フレームワークが使用されたプロジェクトについては、使用されたフ

図表５　プロジェクト特性値の基本統計量

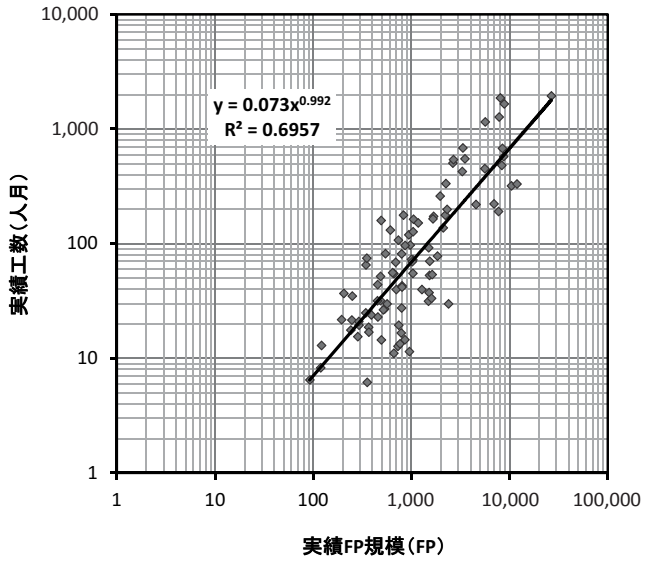
（a）フレームワークが使用された93プロジェクト

	平均値	標準偏差	最小値	第１四分位数	中央値	第３四分位数	最大値
規模（実績FP規模）	2,247	3,646	92	491.0	861.0	2,100	26,572
工数（人月）	204.4	373.8	6.2	26.6	65.4	178.0	1,954
工数密度（人月/100FP）	8.92	6.26	1.21	4.63	7.07	12.28	32.51

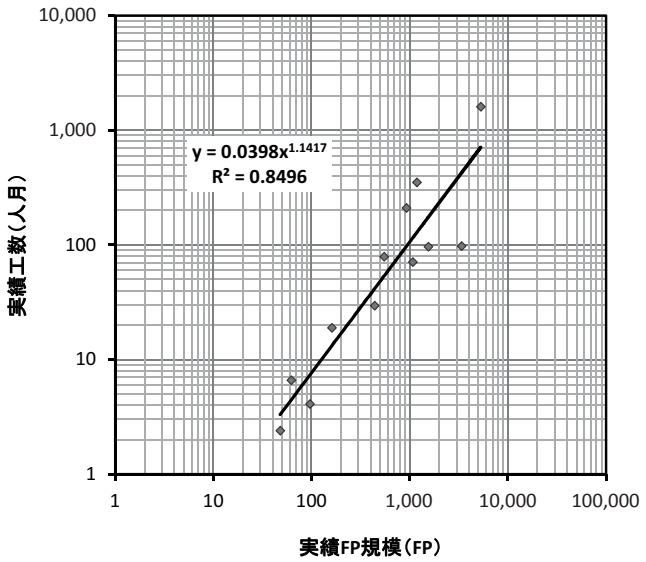
（b）フレームワークが使用されなかった12プロジェクト

	平均値	標準偏差	最小値	第１四分位数	中央値	第３四分位数	最大値
規模（実績FP規模）	1,227	1,582	48	144.8	737.0	1,270	5,290
工数（人月）	214.9	451.0	2.4	15.9	75.1	126.3	1,610
工数密度（人月/100FP）	12.65	9.80	2.92	5.95	8.70	16.54	30.43

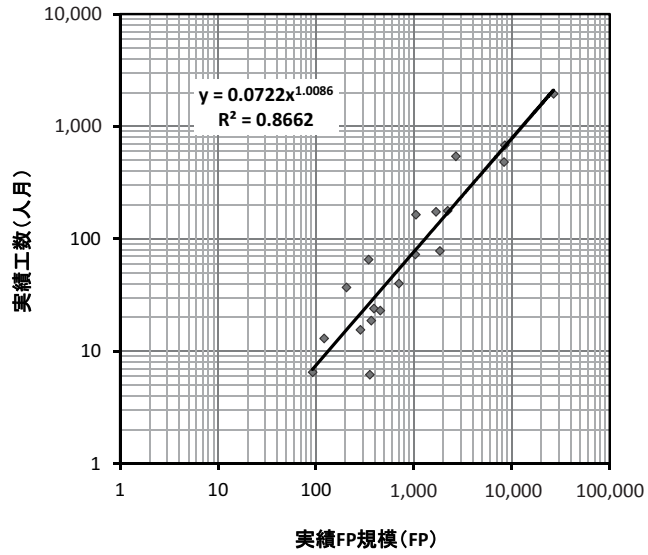
図表6 「規模」と「工数」の関係



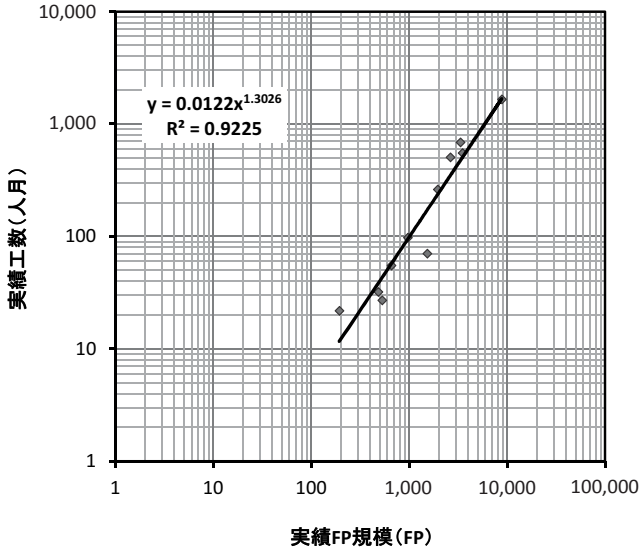
(a) フレームワークが使用された93プロジェクト



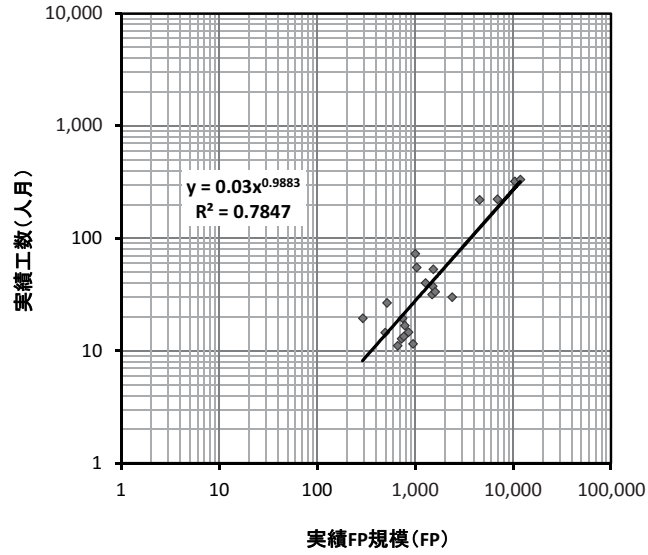
(b) フレームワークが使用されなかった12プロジェクト



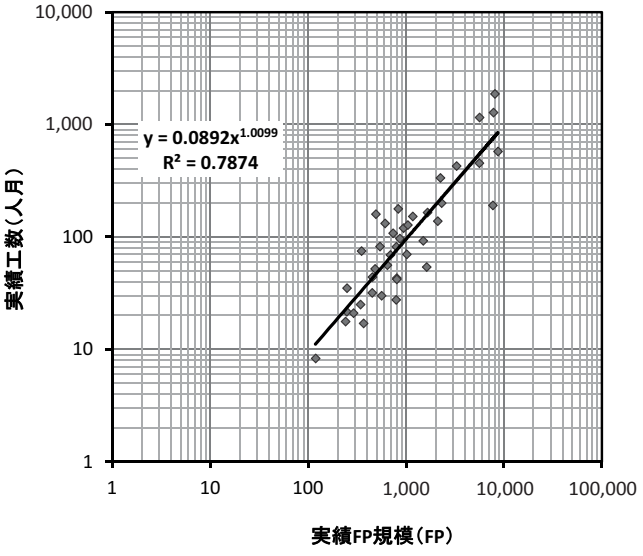
(a_1) Strutsが使用された19プロジェクト



(a_2) 自社開発フレームワークが使用された11プロジェクト



(a_3) 部品型フレームワークが使用された22プロジェクト



(a_4) その他のフレームワークが使用された41プロジェクト

レームワーク別にも示す。また、**図表7**には、**図表6**中で示した回帰式の係数a、b、および決定係数R^2の値を再掲すると共に、規模が100FP、1,000FP、および、10,000FPにおいて、それぞれの回帰式で求められる工数（人月）を示す。

図表6より、フレームワークが使用されたプロジェクトにおいても、使用されなかったプロジェクトにおいても、規模と工数の間には、ある程度以上に強い正の相関があることが分かる。

両対数の散布図上では、累乗近似式は直線となり、係数aがそのy切片（規模が1の時の工数の値）を、係数bがその傾きを、それぞれ表す。

y切片は、理論的には、規模に関係なく最低限必要となる工数に対応する。**図表7**より、いずれの場合も、小数点以下第2位から始まる値であり大差はない。敢えて言えば、Strutsが使用されたプロジェクトでは、y切片は、フレームワークが使用されなかったプロジェクトの2倍近くにもなる。一方で、自社開発フレームワークが使用されたプロジェクトでは、フレームワークが使用されなかったプロジェクトと比べて、y切片は3分の1以下となっている。

回帰式の傾き（係数b）についても、それほど大きな差は見られない。ただし、この傾きの違いは、規模が大きくなると、工数に大きな差をもたらす。**図表7**において、規模が100FPにおいて回帰式から得られる工数（人月）の値は、y切片に大きな差がないこともあり、大きな差は見られない。しかし、規模が1,000FPにおいては、フレームワークが使用されなかったプロジェクトでは106、フレームワークが使用されたプロジェクトではその約3分の2の69である。規模が10,000FPにおいては、フレームワークが使用されなかったプロジェクトでは1,468、フレームワークが使用されたプロジェクトではその約半分の678である。

2）使用フレームワーク別

使用されたフレームワーク別に回帰分析の結果を見ると、その傾向は次の通りである。

・Struts

工数抑制にある程度の効果を持つ。規模が1,000FPにおいては77人月で、フレームワークが使用されなかったプロジェクトの約4分の3、規模が10,000FPにおいては782人月で、フレームワークが使用されなかったプロジェクトの約半分となっている。

・自社開発フレームワーク

規模が大きくなるにつれ、より多くの工数を要する。規模が1,000FPにおいては99人月で、フレームワークが使用されなかったプロジェクトとほぼ同じであるが、規模が10,000FPにおいては1,980人月となり、フレームワークが使用されなかったプロジェクトよりもかえって多くの工数が必要となる。

・部品型フレームワーク

工数抑制に一番大きな効果を持つ。規模が1,000FPにおいても、10,000FPにおいても、フレームワークが使用されたプロジェクト全体と比較して、工数は40％ほどに抑えられている。

・その他フレームワーク

工数抑制の効果は限定的である。回帰式の傾き（係数b）の値が1.0099であり、これは、Strutsが使用されたプロジェクトの1.0086とほぼ同じであるが、切片（係数a）が0.0892であるため、規模が1,000においても、10,000FPにおいても、Strutsが使用されたプロジェクトよりも、その工数は25%ほど多くなる。

以上のように、フレームワークが使用されることで、ソフトウェアの開発規模が大きくなっても、工数の増加は半分程度に抑えられるが、それは、部品型フレームワーク、および、Strutsを使用したプロジェクトの影響によるところが大きいことが分かる。

図表7　規模を説明変数，工数を目的変数とする回帰分析の結果

使用フレームワーク名等		プロジェクト数	回帰式　工数 = a×規模^b					決定係数 R^2
			係数 a (y切片)	係数 b (傾き)	工数 (人月)			
					規模 = 100FP	規模 = 1,000FP	規模 = 10,000FP	
(a) フレームワーク使用		93	0.073	0.992	7	69	678	0.6957
内訳	(a_1) Struts	19	0.0722	1.0086	8	77	782	0.8662
内訳	(a_2) 自社開発フレームワーク	11	0.0122	1.3026	5	99	1,980	0.9225
内訳	(a_3) 部品型フレームワーク	22	0.03	0.9883	3	28	269	0.7847
内訳	(a_4) その他フレームワーク	41	0.0892	1.0099	9	96	977	0.7874
(b) フレームワーク不使用		12	0.0398	1.1417	8	106	1,468	0.8496

図表8　プロジェクト特性値の基本統計量
（分析対象167プロジェクトのうち，基本5工程の各工数が記されている82プロジェクト）

	平均値	標準偏差	最小値	第１四分位数	中央値	第３四分位数	最大値
規模 (実績FP規模)	2,351	3,966	62.0	497.3	893.5	2,285	26,572
工数 (人月)	282.2	872.9	4.1	28.2	70.7	208.3	7,471
工数密度 (人月/100FP)	10.08	8.12	1.21	5.10	7.27	13.04	50.13

4 工程別工数密度

最後に、ソフトウェアの開発規模が大きくなっても工数の増加を抑えることのできる部品型フレームワークとStrutsについて、工程別の工数密度 [2] [3] を見てみる。分析に用いるのは、ここまでの分析対象であった167プロジェクトのうち、基本5工程の各工数が記されている82プロジェクトのデータである。

図表8に、ここでの分析対象82プロジェクトにおける、規模、工数、および、工数密度の基本統計量を示す。そして、**図表9**に、分析対象82プロジェクト、そのうち、フレームワークが使用されなかった6プロジェクト、Strutsが使用された14プロジェクト、部品型フレームワークが使用された16プロジェクト、それぞれについて、基本5工程それぞれにおける工数密度の基本統計量と箱ひげ図を示す。

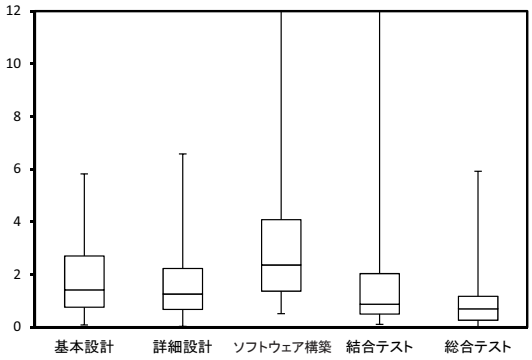
全82プロジェクトの値を基準にすると、フレームワークが使用されなかったプロジェクトでは、5つの基本工程いずれにおいても工数密度の平均値が増加しており、特に、詳細設計工程と結合テスト工程において顕著である。

部品型フレームワークが使用されたプロジェクトでは、基本5工程いずれにおいても工数密度の平均値は低下しており、基本設計工程で1.91から0.84と約2分の1に、詳細設計工程では1.69から0.42と約4分の1に、ソフトウェア構築工程で3.23から1.15、結合テスト工程で1.57から0.39、および、総合テスト工程で1.02から0.33、と約3分の1にまで低下している。特筆すべきは詳細設計と結合テストで、工数密度の低減の効果が他の工程よりも大きい。

部品型フレームワークでは、データベースや画面等、繰り返し利用可能なコードが切り出され、文字通り、機能部品として整理、蓄積（資産化）され、開発者に提供される。利用しやすく、また、信頼性が確認されている機能部品が提供されれば、ソフトウェア構築工程はもちろんのこと、基本設計工程や結合／総合テスト工程においても、工数密度の低下が期待される。本分析結果は、機能部品としてのコードの資産化が、基本5工程全体における工数密度の大幅な低下、すなわち、生産性の大幅な向上を実際にもたらすこと、可能であることを示している。

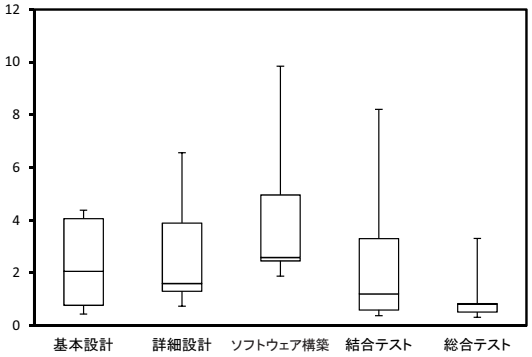
一方、変動係数を見ると、ソフトウェア構築工程で約2分の1に低下している一方で、総合テスト工程では逆に、1.11から1.48と30%ほど増加している。工数密度は低下するものの、基本5工程の最後の工程となる総合テスト工程においてそのばらつきが大きくなる傾向にあり、開発管理上のリスクとして備えるべきである。

図表9　基本5工程における工数密度



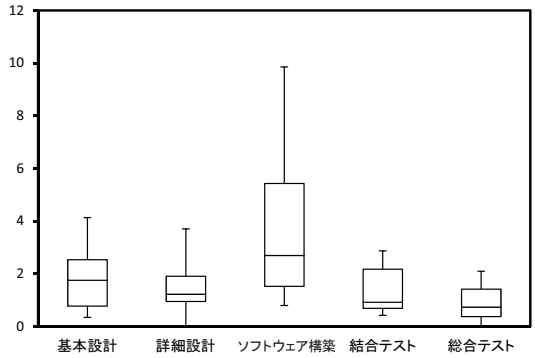
統計量＼工程	基本設計	詳細設計	PG設計製造	結合テスト	総合テスト(ベンダ確認)
平均値	1.91	1.69	3.23	1.57	1.02
標準偏差	1.48	1.48	2.96	2.13	1.14
変動係数	0.77	0.88	0.92	1.35	1.11
最大値	5.82	6.58	18.36	16.44	5.92
第３四分位数	2.70	2.23	4.08	2.03	1.17
中央値	1.42	1.26	2.36	0.87	0.70
第１四分位数	0.76	0.67	1.37	0.50	0.26
最小値	0.08	0.03	0.51	0.11	0.00

(i) 分析対象の82プロジェクト



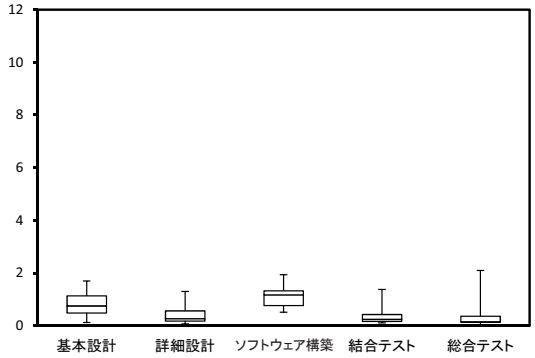
統計量＼工程	基本設計	詳細設計	PG設計製造	結合テスト	総合テスト(ベンダ確認)
平均値	2.33	2.73	4.18	2.56	1.08
標準偏差	1.86	2.34	3.10	3.05	1.11
変動係数	0.80	0.86	0.74	1.19	1.03
最大値	4.38	6.56	9.85	8.21	3.30
第３四分位数	4.06	3.89	4.96	3.30	0.83
中央値	2.06	1.60	2.59	1.21	0.81
第１四分位数	0.77	1.30	2.46	0.59	0.51
最小値	0.43	0.73	1.88	0.37	0.31

(ii) フレームワークが使用されなかった6プロジェクト



統計量＼工程	基本設計	詳細設計	PG設計製造	結合テスト	総合テスト(ベンダ確認)
平均値	1.85	1.51	3.58	1.33	0.91
標準偏差	1.17	0.96	2.81	0.89	0.71
変動係数	0.63	0.64	0.78	0.67	0.78
最大値	4.13	3.70	9.85	2.87	2.09
第３四分位数	2.54	1.91	5.43	2.17	1.42
中央値	1.76	1.23	2.70	0.92	0.74
第１四分位数	0.78	0.95	1.53	0.69	0.38
最小値	0.34	0.03	0.79	0.42	0.03

(iii) Strutsが使用された14プロジェクト



統計量＼工程	基本設計	詳細設計	PG設計製造	結合テスト	総合テスト(ベンダ確認)
平均値	0.84	0.42	1.15	0.39	0.33
標準偏差	0.45	0.35	0.46	0.35	0.49
変動係数	0.54	0.84	0.40	0.89	1.48
最大値	1.70	1.30	1.94	1.38	2.10
第３四分位数	1.13	0.56	1.33	0.43	0.36
中央値	0.76	0.27	1.17	0.25	0.15
第１四分位数	0.48	0.18	0.77	0.17	0.13
最小値	0.12	0.07	0.51	0.11	0.00

(iv) 部品型フレームワークが使用された17プロジェクト

Strutsが使用されたプロジェクトでは、5つの基本工程いずれにおいても工数密度の低下は見られず、逆に、ソフトウェア構築工程では若干増加が見られる。ただし、変動係数は、5つの基本工程いずれにおいても低下しており、工数密度のばらつきの低減には一定の効果はあるようである。Strutsでは、開発対象となるWebアプリケーションにおいて、その動作の流れが規定されている。開発者は、その流れに沿って必要なパーツを選び、あてはめていくことになるため、アプリケーションの実現方法や開発作業の進め方に、開発者間で大きな差が生じにくい。そのことが、工数密度のばらつき低減につながっている可能性がある。

以上のように、基本5工程別で見ても、部品型フレームワークによる工数密度の低下は顕著であり、生産性の向上に大きく寄与するフレームワークであると言える。

まとめ

本稿では、経済調査会が平成13年度から27年度に実施した「ソフトウェア開発に関する調査」で収集されたプロジェクトデータ（ソフトウェア開発データリポジトリ）を分析することで、その活用が広がりつつある開発ツールの一種「フレームワーク」の使用が生産性に与える影響を、開発規模の大小や工程別に明らかにした。得られた主な知見は次のとおりである。

(1) 工数を規模で割って得られる工数密度のばらつきは、フレームワークの使用により小さくなる。これは、規模が工数に与える影響をフレームワークが抑制する、と解釈することもできる。

(2) フレームワーク使用の有無で比較すると、規模が100FPであれば、フレームワーク使用の有無は、工数に大きな差を生じさせない。しかし、規模が1,000FPや10,000FPであれば、フレームワークの使用により、工数が約半分となる。

(3) フレームワーク別に見ると、部品型フレームワークが、工数抑制に一番大きな効果を持つ。規模が10,000FPであれば、フレームワークが使用されなかったプロジェクトに比べ、工数が約6分の1となる。Strutsも工数抑制にある程度の効果を持つ。規模が10,000FPであれば、フレームワークが使用されなかったプロジェクトに比べ、工数が約2分の1となる。自社開発フレームワークには、工数抑制の効果は必ずしも見られなかった。規模が10,000FPにおいては、フレームワークが使用されなかったプロジェクトよりもかえって工数が増加する。その他のフレームワークでは、工数抑制の効果は限定的であり、フレームワークが使用されなかったプロジェクトと工数に大きな差は見られなかった。

(4) 基本5工程別に見ると、フレームワークが使用されなかったプロジェクトでは、特に、詳細設計工程と結合テスト工程において工数密度が大きい。つまり、フレームワークの使用によって上記2工程の工数が減少している。

(5) 基本5工程別、かつ、フレームワーク別に見ると、部品型フレームワークの使用により、基本5工程全てにおいて工数密度は低下し、特に、詳細設計工程と結合テスト工程では、6分の1まで低下する。ただし、総合テスト工程ではそのばらつきが30%ほど増加する。Strutsの使用により、工数密度の低下はみられないが、そのばらつきは低減される。

以上のように、今回の分析では、フレームワークの使用は、工数密度の低下、すなわち、生産性の向上に一定の効果があるとの結果が得られた。特に、部品型フレームワークが生産性の向上に与える影響は、開発規模の大小や工程の別に関係なく大きいことが分かった。

フレームワークを使用するソフトウェア開発プロジェクトの割合は近年増加しており、その種類も増えつつある。フレームワークの使用が生産性の向上にどのようにつながるのか、その分析や解明の重要性は、今後もますます高くなると考えられる。ただし、今回分析対象とした部品型フレームワークとStrutsのよう

に開発支援のアプローチが異なれば、規模と工数の関係や生産性に与える影響も異なる。また、特定のアプリケーション開発に特化したフレームワークもあり、フレームワークというだけで区別や層別もせず分析することには限界がある。今回分析対象とした部品型フレームワークも万能ではなく、生産性の向上が見られるのは、類型的な事務処理システムの開発に使用した場合のみかもしれない。開発支援のアプローチや対象アプリケーション等で層別を行っても統計的有意が確認できるほど、更なるデータ収集に努めるとともに、フレームワーク開発者・使用者へのインタビュー等、定性的な評価や分析も行い、分析結果に対する解釈の合理性や実用性を高めていく必要がある。

参考文献

［1］中村伸裕、谷本收、楠本真二：“ソフトウェアプロダクトラインのエンタプライズ・システムへの適用と評価”、SEC journal、Vol.9、No.4、2014年。

［2］戸田航史、松本健一、大岩佐和子、押野智樹：ソフトウェア開発における工程別生産性に関する分析“、一財）経済調査会「経済調査研究レビュー」Vol.8 2011年3月

［3］松本健一、大岩佐和子、押野智樹：ソフトウェア開発における工程別生産性に関する分析～生産性変動要因に基づくリスク管理・予測に向けて“、一財）経済調査会「経済調査研究レビュー」Vol.17 2015年9月