ソフトウェア工学 125－4
（2000．1．14）

ソフトウェアプロジェクトにおけるリスク要因の推定の一方法[1)]

古山恒夫（東海大学[2)]）、石橋雄一（スタットラボ）、
岩尾俊二（構造計画研究所）、花本佳一（構造計画研究所）

概要：ソフトウェアプロジェクトの生産性や信頼性などの値が多数の蓄積データに比べて逸脱しているプロジェクトに対してその原因となるリスク要因を推定する方法を提案する。逸脱しているかどうかは経験的な絶対値ではなく統計的な基準により判断する。プロジェクトデータのような正の値で理論的には十分大きな値を取りうる分布に対しては、統計的な基準は対数変換後の回帰分析が簡便で適切である。リスク要因の推定に際しては米国での分析結果も利用する。提案した方法を実データに適用した結果、生産性の低いプロジェクトでは開発者の経験が低く、信頼性の低いプロジェクトでは要求分析の安定度と品質保証機構の整備度が低いことが判明した。

A Method for Identification of Software Project Risk Factors

Tsuneo Furuyama(Tokai University),
Yuichi Ishibashi(StatLab),
Shunji Iwao(Kouzo Keikaku Engineering Inc.) and
Yoshikazu Hanamoto(Kouzo Keikaku Engineering Inc.)

This paper proposes a method of identifying software risk factors of the project whose productivity or reliability are inferior to the historical data. The criteria for picking up the inferior projects are determined statistically. Among the various methods for determining the criteria, the regression analysis to logarithmic transformed data is most effective. The risk factors are first reduced mainly based on the relationships obtained by analyzing 7,000 projects in USA between each project attributes and productivity or reliability. The risk factors are identified based on the attributes showing lower values of each project.

1．はじめに

ソフトウェアプロジェクトは多かれ少なかれ常にリスクを抱えて運営されている。ソフトウェアプロジェクトの計画／見積もり段階あるいは運営時におけるリスク分析はプロジェクト管理上最も重要な事項のひとつであるが、その問題の難しさから、少なくとも国内においてはこれまでこれについて論じられることが少なかった。ソフトウェアプロジェクトにおけるリスク管理についての文献は世界的に見てもあまり多くない[1][2]。これもこ問題の難しさを表しているといえよう。失敗プロジェクトのデータが成功プロジェクトのデータに比べて関係者の名誉のために隠されてしまう傾向にあることもひとつの大きな理由であろう。

もともとソフトウェアプロジェクトに関するさまざまな情報を含んだ公開データはわが国では

1) 本研究は平成10年度通産省補正予算公募プロジェクト「ソフトウェアＥＣを支える定量化情報提供サービスセンター構築」の一環として行われたものである。
2) 〒410-0395 沼津市西野 317

非常に少なく皆無と言ってよいであろう。そのため、国内においては、平均的な生産性という最も基本的な統計量すらまったくわかっていない。したがってこのような状態でリスク問題を論じることは早計とも思われるが、この問題の重要性を思うと少ないデータから一歩でも2歩でも前進を試みることは無駄なことではない。

本論文では、まずリスク分析の方法（枠組み）について考察する。次に、リスク分析の対象とすべきプロジェクトの抽出方法（スクリーニング方法）、さらに抽出されたプロジェクトのリスク要因の推定方法を検討する。最後にソフトウェア情報定量化センター（JASMIC）で収集した国内企業15社約300プロジェクトから得られたデータに、それらの方法を実際に適用してその実用可能性を検証する。第2章でリスク分析の枠組みについて考察する。第3章で生産性と信頼性における異常値を抽出するための方法を検討する。第4章で、異常値を持つプロジェクトのリスク要因の候補を特定するために、米国の7,000プロジェクトを分析した結果とそのプロジェクトの持つ属性からリスク要因の候補を絞り込む方法について検討する。

2．リスク分析の方法

2．1　リスク分析の枠組み

リスク分析の枠組みを次のように捉える。

(1)　プロジェクトの評価項目（生産性など）の決定

(2)　それらの評価項目に（大きな）影響を与えると思われる要因の列挙

——実際にはそれら要因のうち計測可能な項目だけに絞込むことが必要である

(3) 蓄積データによる各リスク要因の評価項目への影響度の分析（モデルの構築）

——影響度の大きいもののみ、その評価項目に対するリスク要因とする

(4) 実際のプロジェクトに対して評価項目の値が悪いものを抽出

——計画段階のプロジェクトに対しては、計画値、あるいは構築されたモデルからの推定値が閾値を超えるものを抽出

——完了プロジェクトに対しては実績値が閾値を超えるものを抽出

(5) 抽出されたプロジェクトのリスク要因のうち評価項目に強い影響を与え、かつ対象プロジェクトでのその値が悪いものを抽出して提示

——計画段階のプロジェクトにとっては、評価項目を向上させるために改善すべき主な要因である

——完了プロジェクトにとっては、評価項目が低かった主な原因である

2．2　枠組みを具体化する上での問題点

前節で述べた方法は、精度の高いデータが大量にあり、かつ評価項目（目的変数）を十分記述するに足るだけの要因（説明変数）が列挙されている場合は、精度の高いモデルが構築できるため、十分に機能する。最も理想的な場合は用意された説明変数ですべてが記述されて、計画段階で評価項目の値を正確に求めることができる。完了プロジェクトの場合は評価項目の値が悪かった理由を正確に定量的に指摘することができる。

しかし、現実は次のような問題点を抱えていることが多い。

1) 蓄積したデータ数が不十分である

2) データのばらつきが大きい

3) 値の分布の広がりが少ない要因がある。「低値安定」すなわち多くのプロジェクトで値が悪いところに集中している場合は明らかにその要因を改善すれば評価項目の値が向上するはずであるがモデルには現れにくくなる。

JASMIC で収集したデータもこのような問題点を抱えているため、現時点では必ずしも十分なリスク要因（影響要因）の絞込みと信頼度の高い推定が行えない。

2．3　リスク分析方法の実装の方針

そこで今回のリスク分析では次のような方針で実装を試みることにした。

[方針1] すべてのプロジェクトにリスク要因が

潜んでいるが、特に問題となりそうな異常値に対してのみリスク分析（問題点の指摘）を行う。
[方針2] 米国で行われた7000プロジェクトの分析結果を活用する。

3．リスク分析対象プロジェクトの抽出方法

ここでは、各プロジェクトが蓄積データに比較して異常値を持つかどうかを検出する方法、すなわちデータのスクリーニング方法について検討する。

3．1　評価項目

リスク分析の対象とするプロジェクトを選定するための評価項目としてはユーザ、開発者ともに最も関心が高いと思われる次の2つを選ぶことにする。

（1）　生産性

——工数／規模と規模／工数の両方が生産性の指標として使われる

——一般には工数／規模は、規模の増加に伴い少しづつ値が増加していく（COCOMO モデル[3]による）

——生産性はガンマ分布（に近い分布）に従う

（2）信頼性

——欠陥数／規模が信頼性の指標として使われることが多い

——信頼性は指数分布（に近い分布）に従う

3．2　異常値の抽出方法

データが「異常値（あるいは、はずれ値）」であるかどうかの判断基準としては、一般に大きく次のふたつが考えられる。

1) データの取りうる値の範囲に関する既知情報の利用
2) 統計的傾向や統計的パラメータから逸脱範囲を定める方法

このうち既知情報の利用は、例えば現在のソフトウェア生産性は最高でもｘｘ（ｋｓ／人月）であるからこれ以上の値を持つデータは異常である、というようなものである。この方法は重要な方法であるが、組織やプロジェクト、開発するソフトウェア種別などに依存し、一般的な閾値とはしにくい。一方、後者の方法は前者に比べて精度が落ちるものの、機械的に処理できるという特長がある。ここでは、後者の方法について検討する。

3．3　統計的手法による異常値の抽出方法

3．3．1　1次元データの場合

1次元データ（例えば、開発規模など）に対して統計的な分布から異常値を抽出する方法として次のものが考えられる。

1) 生データの平均（Mean）±標準偏差×ｎの範囲を利用する方法
2) 生データの中央値（Median）と上下のヒンジを利用する方法[4]——順位情報の利用
3) 生データの対数変換後の値の平均±標準偏差×ｎの範囲を利用する方法

最初の方法は、分布が正規分布に近い場合、あるいは左右対称の分布（ただし一様分布のようなものは極端なので除く）に有効で説得力のある方法である。2番目の方法は、ガンマ分布（指数分布もその1種）のように分布が非対称である場合、あるいは平均から大きくはずれた値を多く含むデータ群に対してロバストな結果を与える点ですぐれた方法である。プロジェクトデータは経験的にこのような分布になることが多い。3番目の方法は、正の値のみをとり左右が非対称なガンマ分布などのような分布に対して用いると正規分布に近くなって扱いやすくなるという特徴をもっている。

3．3．2　2次元データの場合

これらの方法は容易に2次元データに拡張することができる。例えば、工数と規模などの2次元データの場合は平均の代わりに規模に対する工数の回帰直線を用い、標準偏差×ｎの代わりに回帰直線に平行で（±標準偏差×ｎ）だけ離れた2本の直線を用いて異常値を検出することができる。

生データを利用した生産性に関する回帰分析では、FP数がゼロのとき工数もゼロであるから、原点を通る直線となる。これは情報としては1次元情報に縮退する。ここで注意しないといけないことは工数／FP 数の分布と FP 数／工数の分布

では分布の形態が異なり、単純に平均と標準偏差×ｎσの閾値では異なった結果が出てくる。それを避けるためには対数log（工数／FP数）を求めるとよい。この変換によれば、log（工数／FP数）とlog（FP数／工数）は左右反転した分布となるだけであるから同じ結果を与えることになる。

対数変換後のデータに対しても同様に回帰直線およびそれと平行な直線を求めることにより、異常値を抽出するための閾値を設定することができる。例えば、生産性に関するデータについては、log(規模)（横軸）とlog（工数）（縦軸）の散布図に対して回帰直線を求め、それと平行な

$$\log(\text{工数}) = a \times \log(\text{規模}) + b \pm n\sigma \qquad (1)$$

なる２本の直線で囲まれる範囲から逸脱したデータ（異常値）に対して警告を出す。ｎの値は全体から何%のプロジェクトを抽出するかに依存する。ｎ＝１の場合は片方向で約 30%のプロジェクトが、ｎ＝1.5 の場合は片方向で約７%のプロジェクトが抽出される。

順位情報を利用する場合は、説明変数（この場合は工数）をある幅で切ってそれぞれの領域ごとに中央値と上下のヒンジを求めればよい[4]。幅の切り方は例えばSturgesの公式

$$L = [1 + \log_2 N] \qquad (2)$$

を用いるとよい[4]。

以上さまざまな統計的スクリーニング方法を比較した結果を表１に示す。

３．４　適用例――生産性データのスクリーニング方法

JASMIC で収集したデータの規模と工数の分布を図１に示す。図１のデータに対して、３．３で示した３つの方法でスクリーニングを行ってみる。ただし、ここでは目的変数としての工数に最も大きな影響を与える要因と考えられる、規模との２次元データとして分析する。

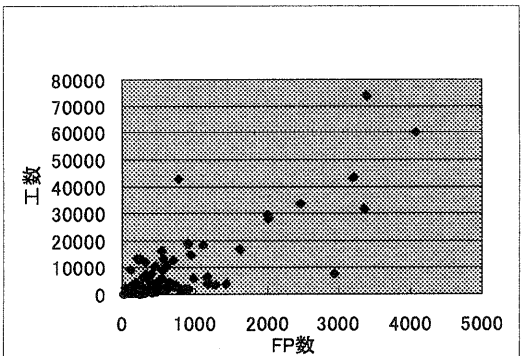

図１　規模（FP数）と工数の関係（FP数＜5000）

図２は、(1) 生データに対する回帰直線、(2) 対数変換後のデータに対する回帰直線、(3) 規模を９つの幅に分けたのち各領域に対して中央値を求めたもの（順位情報を利用したもの）をそれぞれ示したものである。ただし、順位情報を利用した方法では、それぞれの領域ごとに中央値を求めた後ハミングのならし[4]を行って全体を滑らかにしている。

図２によると対数変換法と順位情報利用法とが

表１　生産性データに対するスクリーニング方法の比較

分布の次元	尺度	閾値	特徴
1次元	log（工数／FP数）	平均±標準偏差×ｎσ	・生産性の尺度として逆数（FP数／工数）をとっても結果が変わらない
2次元	FP数と工数	工数＝a×FP数＋b±ｎσ	・ばらつきが大きいデータでは逆数を尺度とした場合の結果との差が大きくなる
		あるFP数の幅毎に工数の中央値±上下ヒンジ（順位情報の利用）	・ガンマ分布のような左右非対称な分布に対して実情に合った閾値と考えられる ・蓄積データの異常値にひきずられず結果がロバストである
	log(FP数)とlog(工数)	log（工数）＝a×log（FP数）＋b±ｎσ	・順位情報を利用する方法と同じような特徴をもつがそれより簡便に計算できる

同じような結果を与えることがわかる。これは、（規模を固定した場合）工数がガンマ分布のような左右非対称でゼロの値の方に傾いた分布のためこのような結果となったと考えられる。

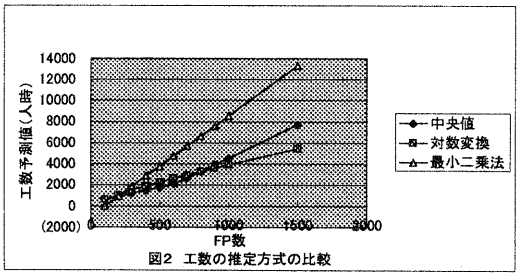

図2 工数の推定方式の比較

対数変換法と順位情報利用法を比較すると、対数変換法では係数が1より小さくなっている。すなわち規模が大きくなるにつれて少しづつであるが生産性が向上（規模あたりの工数が減少）していくことを示している。一方、順位情報利用法では規模の増大とともに生産性が減少していくことを示している。この例では順位情報利用法の方がCOCOMOでの結果やわれわれの常識と合致している。

4．リスク要因の抽出方法

ここではスクリーニングされたプロジェクトのリスク分析について検討する。

4．1　リスク分析の手順

スクリーニングの結果抽出されたプロジェクトに対して、具体的なリスク分析を以下の手順で行う。JASMICの現時点での収集データを分析した結果、システム特性と呼んでいる開発タイプ（新規、改造の別など）やシステム構成（クライアント／サーバシステムかスタンドアロンかなど）などは生産性に大きな影響を与えるため生産性推定モデルに取り込めることが判明したが、システム属性（開発者のレベルやツールの利用程度など）と呼んでいる開発者がコントロールできる要因のうちで統計的に有意でモデルに取り込めるものはそれほど多くない。システム特性ごとにシステム属性を再度分析するには今のところデータ数が不足している。そこで今回は2．3で述べた分析方針2に従って、米国の分析結果[2]を利用して、生産性に強い影響を与える要因群と信頼性に強い影響を与える要因群を調べ、それらの中から具体的なリスク要因の候補を選定することにした。

4．1．1　一般的なリスク要因の選定

異常値に相当するデータに対して、その原因となりそうなリスクの候補を表2に示すリスク分析表よりピックアップする。リスク分析表に示されたリスク要因の候補は、参考文献[2]に示されているソフトウェア開発における代表的なリスク、およびJASMICで収集した各プロジェクトの属性で構成する。例えば、品質データが異常値の場合は、リスク分析表の「低い品質」欄に○のついているリスク要因が影響を与えている可能性が高いので、その要因をピックアップする。

4．1．2　特異プロジェクトの属性からのリスク要因の抽出

リスク分析表からピックアップされた項目のうち、プロジェクト属性（アンケートによる回答）のレベルが悪い値を示しているものだけを選び出して重要なリスク要因として警告を出す。

4．2　リスク要因分析の実例

4．2．1　生産性の視点からのリスク分析例

JASMICで収集したデータのうち、FPによる規模と工数が計測されている新規プロジェクト118件について生産性の視点からリスク分析した例を示す。リスク分析対象とするプロジェクトは、3．2で述べた方法のうちの対数変換による方法を採用し、閾値の幅を決めるnは1.5とした。対数変換後のデータは近似的に正規分布をなす。したがって理論的には平均$\pm 1.5\sigma$を超えるデータ数は全体のそれぞれ6.68%となるはずである。実際に生産性が平均-1.5σ以下のプロジェクトは$118\times 0.0668=7.88$に最も近い8件であった。

これらのプロジェクトに対して4．1で述べた方法でリスク要因の分析をした結果を表3に示す。表3に示した以外のプロジェクト属性、例えばユーザの仕様書への関与などに対しても各プロジェ

表2 リスク要因分析表

原因(リスク)					結果(プロジェクトへの影響)	
リスク一覧(「ソフトウェア病理学」[2]より)		プロジェクト属性(JASMICのアンケート項目)			低い品質	低い生産性
人間要因	不適切な管理者教育	スタッフ	管理		○	○
	経験不足の管理者			プロジェクト管理者の経験	○	○
	管理者の不当行為				○	
	不適切な技術者教育				○	○
	未熟な技術者		開発者	業務分野の経験	○	○
				分析・設計経験		
				言語・ツール経験		
				開発プラットフォーム経験		
	技術者の不当行為				○	
	徐々に増大するユーザ要求(*)		ユーザ	ユーザのソフトウェア経験		
				ユーザの要求仕様関与	○	
				ユーザの設計レビュー関与		
				ユーザのシステム試験関与		
SE(ツールと技法)	不適切な管理ツールと手法(*)	技術	ツール利用	プロジェクト計画立案ツール	○	○
	不適切な管理ツールと手法(*)			プロジェクト管理ツールの利用	○	○
	不適切な構成管理			構成管理ツールの利用	○	○
	不適切なソフトウェア工学ツールと手法(*)			設計支援ツールの利用	○	○
	不適切な技術文書作成ツールと手法			ドキュメント作成支援ツールの利用		○
	不適切な品質保証ツールと手法			デバッグ／テストツールの利用		○
	不適切なソフトウェア工学ツールと手法(*)			CASEツールの利用	○	○
			開発プラットフォーム	プラットフォームの有効性		
				プラットフォームの安定性		
SE(再利用)	再利用性の低い構造		(再利用率)		○	○
	再利用性の低いデータ					○
	再利用性の低いプロジェクト計画					○
	再利用性の低い見積り					○
	再利用性の低い要求仕様			要求仕様	○	○
	再利用性の低い設計			設計文書	○	○
	再利用性に低い文書			ユーザ向け文書		○
	再利用性に低いプログラム			プログラム	○	○
	再利用性の低いテスト			試験仕様／データ／プログラム	○	
	再利用性の低いヒューマンインタフェース					○
SE(プロジェクト管理)	徐々に増大するユーザ要求(*)	プロセス	要求分析	要求仕様の明確さ		
				要求仕様の安定度	○	
			品質	品質保証の機構		
	不正確な品質評価				○	
	欠陥多発モジュール				○	○
	不適切な欠陥除去(*)		テスト	テスト体制	○	
	不適切な欠陥除去(*)			テスト期間	○	
	不適切な欠陥除去(*)			テスト環境の安定性	○	
SE(規範)	不正確な尺度			(FP数)	○	○
	不適切な測定方法				○	○
	不適切な規格				○	○
	不完全なソフトウェアライフサイクル				○	○
	不適切な方法論					○
	不正確な規模見積り	計画／運営	計画	見積り方法	○	○
SE(プロジェクト管理)	不正確なコスト見積り				○	
	不適切な計画作成			計画の妥当性	○	
	過酷なスケジュール				○	
	過大な文書作成					○
	生産性の誇大宣伝(に惑わされる)					○
	(専門分化の不足(*))		運営	役割と責任の明解さ		
	(曖昧な改善目標(*))			目標の共有		
企業政策	狭いオフィス環境	環境	オフィス	オフィス環境－スペース		○
				オフィス環境－騒音		
	専門分化の不足(*)	保守	スタッフ	保守スタッフの割り当て・経験		
			プロセス	保守のサイト数		
	老朽化システムの長期保守			保守における新版のリリース頻度	○	○
(SE(再利用))			プロダクト	現在のシステムの安定度		
				変更のし易さ(データ)		
				変更のし易さ(プログラム)		

(*)は重複して現れる項目

表3 生産性の低い8件のプロジェクトのリスク分析結果（×印がプロジェクト属性値の低い項目）

グループ	使用言語（注1）	スタッフ					技術								プロセス	計画／運営		環境	
		管理	開発者の経験度				ツールの利用度						プラットフォーム		品質	計画		オフィス	
		経験	業務分野	分析設計	言語ツール	プラットフォーム	計画立案	管理ツール	構成管理	設計支援	資料作成	デバッグ	有効性	安定性	保証機構	見積方法	妥当性	スペース	騒音
1	C																		
	VB	×				×	×	×	×	×		×							
	C						×											×	
	Cobol	×	×	×			×			×									
	不明	×		×								×				×			
	C	×					×	×	×	×	×								
	OO		×		×	×	×	×		×		×			×		×	×	×
	C		×				×	×	×						×				×

（注1）OO:オブジェクト指向言語

クトは悪い値を示しているが、生産性への影響要因は小さいと見てリスク要因としては指摘していない。

4．2．2 信頼性の視点からのリスク分析例

JASMICで収集したデータのうち、FPによる規模と工数が計測されている新規プロジェクト67件について信頼性の視点からリスク分析した例を示す。分析方法は生産性の場合と同様である。信頼性が平均-1.5σ以下のプロジェクトは$67\times 0.067=4.4$より1件少ない3件であった。

4．3 考察

（1）生産性に関するリスク分析

生産性のレベルによるグループごとの第三世代言語の使用比率を表4に示す。また同じグループごとに各属性値の平均を求めた結果を表5に示す。これによると、第1グループは開発者の経験を除いてほとんどの属性値で比較的よい値を示しており、主なリスク要因が実は開発言語と開発者の経験にあるのではないかと推測される。ただし、この結果から属性値の悪いところを改善する必要はないと判断するのは早計であろう。例えばツールをもっと有効に利用すれば今より生産性が改善されることは明らかであろう。第2グループでは第一グループとは逆に、第三世代言語の使用比率は低いが、プロジェクト属性値は一般に悪い。そのための生産性がやや低下していると推測される。

（2） 信頼性に関するリスク分析

信頼性のレベルによるグループごとの第三世代言語の使用比率を表6に示す。また同じグループごとに各属性値の平均を求めた結果を表7に示す。ここでも、第一グループでは開発言語に第三世代

表4 生産性のグループ毎の第三世代言語の使用比

グループ	データの範囲	プロジェクト数	開発言語 第三世代言語	第四世代言語	不明	第三世代言語比率
1	$-1.5\sigma \geqq$	8	5	2	1	71%
2	$-\sigma \geqq, > -1.5\sigma$	12	4	8	0	33%
3	$\sigma >, > -\sigma$	78	20	22	31	48%
4	$1.5\sigma >, \sigma \geqq$	16	1	1	14	6%
5	$1.5\sigma \geqq$	4	1	0	2	33%
合計		118	31	33	48	48%

表5 生産性のリスク要因対象候補のグループ毎の平均点

グループ	データの範囲	プロジェクト数	リスク要因の対象候補（ソフトウェア病理学[2]による） 開発者の経験	ツール利用の度合い	品質保証機構の整備度	準備された計画	オフィスの広さ
1	$-1.5\sigma \geqq$	8	2.75	3.54	2.63	2.69	2.63
2	$-\sigma \geqq, > -1.5\sigma$	12	3.10	4.03	3.42	2.92	4.08
3	$\sigma >, > -\sigma$	78	2.73	4.10	3.26	3.21	3.10
4	$1.5\sigma >, \sigma \geqq$	16	2.35	3.84	2.33	2.69	3.44
5	$1.5\sigma \geqq$	4	2.30	3.78	3.00	3.00	3.50
平均点			2.70	4.01	3.09	3.06	3.24

（注）1から5までの5段階評価で値が小さいほどよい評価である。

言語を使用している比率が高い。また属性値では品質保証機構の整備に問題があることがわかる。一方第５グループではすべての属性値が他のグループに比べてよい値を示している。

表6 信頼性のグループ毎の第三世代言語の使用比

グループ	データの範囲	プロジェクト数	開発言語 第三世代言語	第四世代言語	不明	第三世代言語比率
1	$-1.5\sigma \geqq$	3	2	0	1	68%
2	$-\sigma \geqq, > -1.5\sigma$	5	(*1) 1	3	1	20%
3	$\sigma >, > -\sigma$	51	12	20	18	24%
4	$1.5\sigma >, \sigma \geqq$	3	1	0	2	33%
5	$1.5\sigma \geqq$	5	0	1	4	0%
合計		67	15	24	26	23%

表7 信頼性のリスク要因対象候補のグループ毎の平均

グループ	データの範囲	プロジェクト数	リスク要因の対象候補（ソフトウェア病理学[2]による） 開発者の経験	ツール利用の度合い	要求分析の安定度	品質保証機構の整備度	テスト環境のよさ	準備された計画
1	$-1.5\sigma \geqq$	3	2.53	4.42	3.67	4	2.44	2.83
2	$-\sigma \geqq, > -1.5\sigma$	5	2.68	4.7	3.2	3.4	2.27	2.8
3	$\sigma >, > -\sigma$	51	2.72	4.31	3.12	3.18	2.87	3.08
4	$1.5\sigma >, \sigma \geqq$	3	2.73	4.86	3.67	2.5	2.67	3.17
5	$1.5\sigma \geqq$	5	2.52	3.9	2.6	2.6	2.33	2.6
平均点			2.7	4.33	3.13	3.17	2.76	3.01

（注）1から5までの5段階評価で値が小さいほどよい評価であ

5．おわりに

ソフトウェアプロジェクトにおけるリスク分析の方法を提案した。プロジェクトに対するリスク分析方法を確立するためには、本来はまず失敗データの収集と失敗の原因分析から始める必要がある。しかし、失敗データは成功データよりさらに収集が困難であるため、当面はここで検討したような方法でリスク分析を行うことから始めるのがよいと思われる。

データを統計的にスクリーニングする方法は既知情報を利用する方法に比べて画一的かつ形式的である。一見実情にそぐわないように思えるが、大量の蓄積データがある場合はかえって客観的なスクリーニングの判断基準を与えてくれるものと思われる。

ただし、この方法でも対象とするデータ群の分布や意味に依存する部分がある。実際には収集したデータをもとに試行錯誤して最適なパラメータを決めていく必要がある。

提案した方法を JASMIC で収集したデータに適用した結果、次のことが明らかになった。

1) ソフトウェアプロジェクトデータのようにガンマ分布を示すようなデータに対しては、対数変換後のデータに対して回帰分析を行い、それと平行な直線でスクリーニングを行うのが簡明である。
2) 生産性の観点からリスク要因を分析したところ、開発言語に第三世代言語を使っている比率が高いこと、開発者の経験が低いことが生産性の低いプロジェクトに多く見られることがわかった。またツールの利用度は他のグループと同様に低いことがわかった。米国での分析結果[2]で因果関係があるとされたオフィスの広さなどは JASMIC のデータでは因果関係は見られなかった。
3) 信頼性の観点からリスク要因を分析したところ、開発言語に第三世代言語を使っている比率が高いこと、要求分析の安定度と品質保証機構の整備度がそれぞれ低いことがわかった。ツールの利用度はここでも低く、改善の余地がある。生産性の場合と異なり、開発者の経験の度合いには因果関係が見られなかった。

[参考文献]

[1] Robert N. Charette: Software Engineering Risk Analysis and Management, McGraw-Hill book Co., 1989.

[2] Capers Jones 著、島崎恭一、富野　寿監訳：ソフトウェア病理学、共立出版、1998.

[3] B. W. Boehm: Software Engineering Economics, Prentice-Hall, 1981.

[4] 渡部　洋、鈴木規夫、山田文康、大塚雄作：探索的データ解析法、朝倉書店、1985.