情報処理学会研究報告
IPSJ SIG Technical Report

Vol.2010-SE-167 No.3
2010/3/18

ソフトウェア開発データに対する相関ルールマイニングを利用した不具合増加要因の調査

出張純也[†1]　尾形憲一[†1]　菊野亨[†1]
水野修[†2]　菊地奈穂美[†3]　平山雅之[†3]

ソフトウェア開発現場では，プロジェクト管理の重要性が益々高まってきている．本研究では，不具合が発生しているプロジェクトに共通して発生している事象の抽出を目指す．具体的には，まず，企業横断的なプロジェクトの属性データに対して相関ルールマイニングを適用し，不具合が発生しなかったプロジェクトに共通して発生している事象をルールとして抽出する．次に，抽出されたルールの前提部を分解し，メトリクスと値の組を抽出する．このメトリクスの値に関して，不具合が多いプロジェクトにおける値とルールで得られた値を比較し，得られたメトリクスが不具合に関連があるかどうかの調査を行った．

その結果，不具合が少ないプロジェクトの特徴として抽出されたルールから得られた 22 個のメトリクスのうち，6 個のメトリクスについて不具合と関連があることがわかった．また，得られたメトリクスの中には，経験によって導かれたリスク要因と一致するものもあった．

Finding the Cause of the Faults by Applying Association Rules to the Software Development Data

Junya Debari ,[†1] Kenichi Ogata ,[†1] Tohru Kikuno ,[†1]
Osamu Mizuno ,[†2] Nahomi Kikuchi [†3]
and Masayuki Hirayama[†3]

In the software development projects, the necessity of project management has been increasing. In this study, we search the common characteristics in the runaway software projects by data mining. Therefore, we first extracted the characteristics of the succeeded projects by association rule mining. Then, we researched the metrics in the antecedents of the extracted rules if there is any difference between succeeded projects and runaway projects. As a result, we extracted 22 metrics, and found that there is significant differences in the value of 6 metrics between succeeded projects and runaway projects.

1. はじめに

1.1 研究の背景

近年のソフトウェア開発の大規模化・複雑化に伴い，ソフトウェアを開発するプロジェクトを円滑に進めることが難しくなってきている．この結果として，プロジェクトの生産物であるシステムの品質不良によって稼働後に障害が発生するなど，失敗となるソフトウェアプロジェクトが数多く報告されている [9]．このため，ソフトウェアプロジェクトの開発現場では，プロジェクトが失敗する可能性を早期段階で察知することが重要である．しかしながら，開発プロジェクトの成功や失敗に繋がる要因は多岐にわたるため，プロジェクトを失敗に導く決定的なリスク要因を特定する事は難しい．こうしたリスク要因の特定や，特定した後のリスク回避方法の検討は経験に基づいて行われる事が多い．また，一方では，多くの開発プロジェクトから観察収集される種々の参考情報が有効に利用されていないという問題もある．そのため，現場で観察収集される規模・工数・工期・不具合数などの量的なデータや，プロジェクトの特性を示すカテゴリカルなデータなどを有効に活用して失敗を事前に回避する手法の確立が求められている．

1.2 研究の目的

プロジェクトの失敗回避のためには，プロジェクト早期のほうが回避策のバリエーションが多く有効な対策を実行しやすいため，出来る限り早い段階でプロジェクトの見通しや健全性を評価することが重要である．こうしたチェックを行うタイミングやチェックリストの内容は，プロジェクトマネージャの経験によって決定されることが多い．本研究では，こうしたチェックリスト作成の際に参考となる情報を企業横断データから抽出することを目指す．

具体的には，プロジェクト横断的データに対して相関ルールマイニングを適用し，プロジェクトの不具合に関するルールを抽出する．次に，抽出されたルールの前提部に含まれているメトリクスに関して，各メトリクスが不具合数の増加に関連があるかどうかの調査を行う．

†1 大阪大学 大学院情報科学研究科
Graduate School of Information Science and Technology, Osaka University
†2 京都工芸繊維大学 大学院工芸科学研究科
Graduate School of Science and Technology, Kyoto Institute of Technology
†3 情報処理推進機構 ソフトウェア・エンジニアリング・センター
Information-Technology Promotion Agency Software Engineering Center, Japan

Vol.2010-SE-167 No.3
2010/3/18

2. 関 連 研 究

ソフトウェア開発プロジェクトにおけるリスク分類は，Boehm [1] らによって古くから行われている．文献 [3] では，Keil らは，世界各地の経験豊かなプロジェクトマネージャを対象として調査を行い，リスク要因の特定を行っている．文献 [8] の 25 章では，ソフトウェア開発におけるリスクマネジメントについて述べられている．また，リスク対策のためのチェックリストの有効性についても述べられている．本研究では，定量的なデータ分析によって，チェックリストを作成する際に参考となる情報を抽出する事を目的としている．

相関ルールマイニングは様々な分野で有益な情報を抽出するために用いられている．ソフトウェア工学の分野における利用例として，例えば，Michail はアプリケーション内でライブラリが再利用されるパターンを相関ルールマイニングを用いて発見し，そのパターンをクラスライブラリの構築に反映させる試みを行っている [4]．また，Zimmermann らは，バージョン管理システムに記録されている変更履歴データに相関ルールマイニングを適用することで，同時に変更されやすいファイルやモジュールの組み合わせを特定する試みを行っている [6]．浜野らは開発の現場から得られたアンケートの回答に対して相関ルールマイニングを適用することで，ソフトウェアプロジェクトが混乱するリスク要因を特定する試みを行っている [10]．Morisaki らは相関ルールマイニングの拡張として，結論に連続値の範囲を利用できる手法を提案している [5]．この手法は従来の名義的尺度を用いたルールマイニングでは検出できなかったルールを抽出できるという点が特徴である．本研究では，相関ルールマイニングの結果として得られるルールを用いて，プロジェクトの初期に着目すべきメトリクスの抽出を目指している．

3. 分析の方針

3.1 データの特徴と分析の方針

本研究で分析に利用するソフトウェア開発プロジェクトのデータは，IPA/SEC が収集したものである [7]．このデータは国内の企業 20 社から収集されたものである．文献 [7] の付録 C に示されているように，欠損値 (未記入) が含まれているデータ項目がある．

本研究では，品質に関する要因として，発生不具合数 (現象数) についての分析を行う．発生不具合数 (現象数) は，文献 [7] の付録 A.4 の定義の通りに計算を行ったものである．発生不具合数 (現象数) の度数分布は表 1 の通りである．収集されたデータ [7] のうち，発生不具合数 (現象数) に関するデータが存在するプロジェクトが 816 件あり，その中で不具合数が 0 件であるようなプロジェクトが 306 件，不具合が存在するようなプロジェクトが 510 件である．図 1 は，発生不具合数 (現象数) の度数分布である．この 816 件のプロジェクトのうち，7 割のプロジェクトでは『発生不具合数 (現象数)』が 5 件以下となっている．

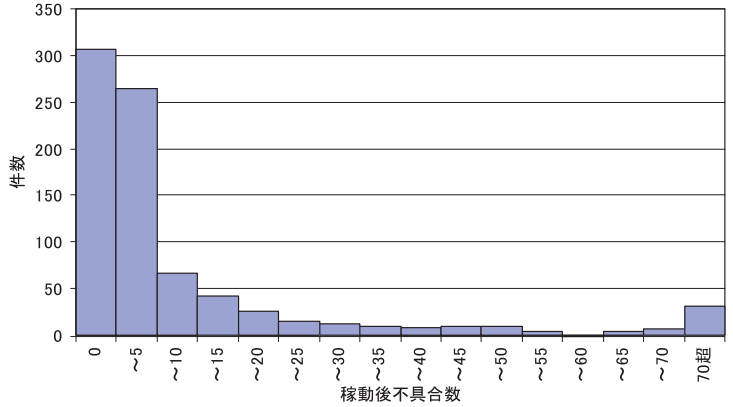

図 1 発生不具合数 (現象数) の分布 [7]

3.2 分析の方針

まず，分析のために，データ白書 [7] から説明変数として 75 個のメトリクスを選択した．この 75 個のメトリクスは，実績開発工数などの定量的なデータだけでなく，開発言語などの開発環境や開発作業の進め方，開発チームの要員スキルなどのカテゴリカルなデータも含んでいる．プロジェクトは，発生不具合数 (現象数) のデータを持つ 816 件である．

プレ実験として，このデータに対して「発生不具合数 (現象数) が中央値[⋆1]以上である」を結論部として相関ルールマイニングを適用した．その結果，得られた相関ルールは 816 件のプロジェクト中，1 割未満 (81 件未満) のプロジェクトでしか成立しないようなルールが大半であった．

そのため，不具合が多いプロジェクトでは，不具合が少ないプロジェクトと逆の事が起こっていると想定して分析を行う．まず，稼働後の不具合が 0 件であったプロジェクトで共通して発生している事象を相関ルールマイニングによって明らかにする．次に，得られた

⋆1 発生不具合数 (現象数) の中央値は 2 である．

Vol.2010-SE-167 No.3
2010/3/18

ルールの前提に現れているメトリクスについて，不具合数が多いプロジェクト（ここでは不具合数が多いプロジェクトの上位 5%とする）でのメトリクスの値を確認する．この不具合数が多いプロジェクトでの値が，不具合が少ないプロジェクトに関する相関ルールの値と異なっていれば，そのメトリクスは不具合と関係があると考える．

4. 分　　析

4.1 準　　備

本研究では，データマイニング手法として相関ルールマイニング [2] を用いる．ここで，相関とは，ある事象が発生すると別の事象が発生しやすいという共起性を意味している．例えば，$A \Rightarrow B$ という相関ルールは，A という事象が起こると B という事象も起こりやすいということを意味している．この相関ルールにおいて，A の部分を相関ルールの「前提部」，B の部分を相関ルールの「結論部」と呼ぶ．

相関ルールの重要性を測る指標として，支持度 (support) と信頼度 (confidence) が存在する．支持度は，全データ中でルールがどの程度出現しているかを示す割合である．一方，信頼度は，前提部が成立するという条件下で結論部が発生する確率である．$A \Rightarrow B$ という相関ルールが存在する場合，支持度は全データ中で A と B が同時に発生しているデータの割合を示す．信頼度は，A が発生している条件下で B が発生している割合を示す．

4.2 データの加工

分析に用いるために，説明変数となるメトリクスの選択を行った．その結果，75 個のメトリクスに関して分析を行うことに決定した．

4.2.1 データの二値化

ここでは，分析に利用するデータを以下の方針で二値化する．データの二値化によって，属性値ごとの件数が増えるため，相関ルールマイニングで特徴が抽出されやすくなる．

名義尺度の値をとるデータ

名義尺度の値をとるデータに関しては，二値化を行わない．

順序尺度の値をとるデータ

要員スキルなどの順序尺度の値をとるデータに関しては，データ白書 [7] 付録 A『データ定義』や付録 C『データごとの回答状況』を参考にして，2 群間のデータ数の差が少なくなるように二値化を行う．

例えば,『要員スキル_分析・設計経験』の回答状況は,『a:全員が十分な経験あり』が 162 件,『b:半数が十分な経験，残り半数はいくらか経験』が 315 件,『c:半数がいくらかの経験，残り半数は経験なし』が 104 件,『d:全員が経験なし』が 6 件である．ここで『a』と『b』をまとめると,『未経験者が 1 人もおらず，その中の半数は十分な経験を持っている』という意味になる．そのため,『a, b』を『スキルが高い』,『c, d』を『スキルが低い』と分割した．

連続値をとるデータ

工数や SLOC 値などの連続値をとるデータに関しては，相関ルールマイニングで取り扱うことができない．そのため，離散値に変換する必要がある．本研究では，中央値を閾値として機械的に二値化を行う．

例えば,『SLOC 実績値_SLOC』は，中央値が 62,000 であるため，62,000 未満のデータを『Low』として，62,000 以上のデータを『High』とする．

4.2.2 記入率が低いデータの削除

本研究で用いるデータ [7] には，未記入のデータが多く存在する．分析のために用意した 816 プロジェクト，75 メトリクスのデータセットの記入率は 49.4%である．

本研究では，記入率が低いデータを削除してから分析を行う．図 2 は，削除の手順を表した図である．まず，データの加工は Step 1，Step 2 の 2 段階に分けて行う．データ加工を行う前のデータセットが DataSet 0 である．図 2 において，m_i は i 番目のメトリクスを，PRJ_j は j 番目のプロジェクトを表している．

Step 1 では，メトリクスごとの記入率を調査する．メトリクス m_i の記入率 F_{m_i} は,『m_i のデータ件数÷目的変数の総数』で計算する．全てのメトリクスの記入率を計算した後に，F_{m_i} の統計値を計算し，メトリクスの削除を行う基準を決定する．Step 1 を実行した後のデータセットが DataSet 1 である．

Step 2 では，記入率が低いメトリクスが削除されたデータに関して，プロジェクトごとの記入率を計算する．j 番目のプロジェクト PRJ_j の記入率を F_{PRJ_j} とすると，F_{PRJ_j} は『プロジェクト PRJ_j のメトリクス記入件数÷メトリクスの総数』で計算する．全てのプロジェクト記入率を計算した後に，F_{PRJ_j} の統計値を計算し，プロジェクトの削除を行う基準を決定する．Step 2 を実行した後のデータセットが DataSet 2 である．

ここで，データ削除の例を図 2 を用いて説明する．まず，DataSet 0 のメトリクス毎の記入率を計算する (表 1)．メトリクス記入率の percentile 25[⋆1]を計算すると，0.3 となった．ここでは，例としてメトリクス記入率 0.3 未満のデータを削除している．削除されるメトリク

⋆1 percentile n とは，対象とする数値群を小さい順番に並び替えて n %目の数字を意味している．以後，percentile n を pn と書く．

Vol.2010-SE-167 No.3
2010/3/18

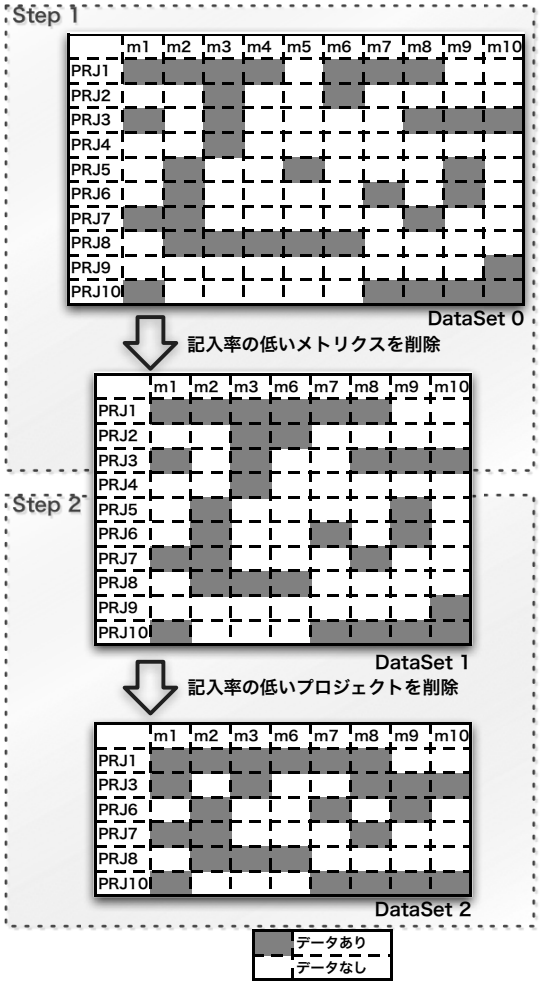

図 2 データ削除のイメージ

表 1 図 2 の DataSet 0 のメトリクス毎の記入率

メトリクス	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}
記入率	0.4	0.5	0.5	0.2	0.2	0.3	0.3	0.4	0.4	0.3

表 2 図 2 の DataSet 1 のプロジェクト毎の記入率

プロジェクト	PRJ_1	PRJ_2	PRJ_3	PRJ_4	PRJ_5	PRJ_6	PRJ_7	PRJ_8	PRJ_9	PRJ_{10}
記入率	0.75	0.25	0.625	0.125	0.25	0.375	0.375	0.375	0.125	0.625

表 3 メトリクス記入率の統計値とメトリクス数

統計値	記入率	メトリクス数
p10	19.7 %	67
p20	26.1 %	60
p25	28.2 %	56
p30	29.2 %	52
p40	31.5 %	45
p50	38.2 %	38

表 4 プロジェクト記入率の統計値

プロジェクト記入率	メトリクス記入率 p25 以上	メトリクス記入率 p50 以上
最小値	17.9	28.9
p25	42.9	59.9
p50	51.8	71.1
p75	75.0	89.5

スは m_4，m_5 である．この結果が DataSet 1 である．次に，DataSet 1 のプロジェクト毎の記入率を計算する (表 2)．プロジェクト記入率の p50 を計算すると，0.375 となった．ここでは例として，プロジェクト記入率 0.375 未満のプロジェクトを削除している．削除されるプロジェクトは PRJ_2，PRJ_4，PRJ_5，PRJ_9 である．この結果が Dataset 2 である．

Step 1：メトリクスの削除

表 3 は，加工前のデータ (図 2 における DataSet 0) のメトリクス毎の記入率の統計値と，その統計値より記入率の低いメトリクスを削除した後のメトリクス数をまとめたものである．左の列が統計値をあらわしている．右側の列は，記入率が該当する統計値以上であるメトリクスの数を表している．例えば，2 行目は，メトリクス記入率の p10 が 19.7% であり，記入率が 19.7%以上であるメトリクスが 67 件あったことを意味している．

Step 2：プロジェクトの削除

表 4 は，Step 1 で記入率 p25(28.2%) 未満のメトリクスを削除したデータのプロジェクト毎の記入率の統計値と，記入率 p50(38.2%) 未満のメトリクスを削除したデータのプロジェクト毎の記入率の統計値である．例えば，メトリクス記入率 p25 以上のデータで計算した場合のプロジェクト記入率 p50 は 51.8%である．

今回は，メトリクス記入率 p25(28.2%) 以上，プロジェクト記入率 p50(51.8%) 以上のデータを分析に利用することに決定した．その結果，本研究の対象データセットはプロジェクト

表 5 分析に利用したメトリクス

名義尺度の値をとるメトリクス	
開発プロジェクトの種別	開発プロジェクトの形態
受託開発の場合の作業場所	新規顧客
新規業種・業務	新規協力会社
新技術の利用	利用形態
システムの種別	業務パッケージ利用の有無
処理形態	アーキテクチャ
Web 技術の利用	DBMS の利用
開発ライフサイクルモデル	類似プロジェクト参照の有無
プロジェクト管理ツールの利用	構成管理ツールの利用
設計支援ツールの利用	ドキュメント作成ツールの利用
デバッグ・テストツールの利用	CASE ツールの利用
コードジェネレータの利用	開発方法論の利用
開発フレームワークの利用	
順序尺度の値をとるメトリクス	
開発プロジェクトチーム内での役割分担・責任所在の明確さ	達成目標と優先度の明確さ
作業スペース	計画の評価 (コスト)
計画の評価 (品質)	計画の評価 (工期)
実績の評価 (コスト)	実績の評価 (品質)
実績の評価 (工期)	要求仕様の明確さ
ユーザ担当者の要求仕様関与	要求レベル (信頼性)
要求レベル (性能・効率性)	PM スキル
要員スキル_業務分野経験	要員スキル_分析・設計経験
要員スキル_言語・ツール利用経験	要員スキル_開発プラットフォームの使用経験
連続値をとるメトリクス	
FP 実績値_調整前	SLOC 実績値_SLOC
平均要員数プロジェクト全体	ピーク要員数プロジェクト全体
検出バグ現象数結合テスト	実績開発工数
実績月数_プロジェクト全体	月あたりの FP
月あたりの SLOC	月あたりの工数
納期遅延	工数超過
ピーク時と平均時の要員数の比	

数は 425 件，メトリクス数は 56 個となった．表 5 がそのメトリクスの一覧であり，この節以降に述べる分析で利用する．名義尺度の値をとるメトリクスが 25 個，順序尺度の値を取るメトリクスが 17 個，連続値のとるメトリクスが 12 個となった．また，データの記入率は 74.3%となった．なお，表 6 はデータ加工後の発生不具合数 (現象数) の統計値である．

4.3 相関ルールマイニングのパラメータの決定

4.2 節の手順で整えたデータセットに対して，相関ルールマイニングでルールを抽出する．その際には，最低信頼度，最低支持度という 2 つの閾値を設定する必要がある．支持度が高いルールほど「多くのプロジェクトで共通して発生している事象である」と言えるため，支持度を決定する必要がある．そこで，信頼度を変化させて最大支持度を調査する．

表 6 データ加工後の発生不具合数 (現象数) の統計値

データ件数	最小値	p25	中央値	p75	最大値
425 件	0	0	1.0	6.0	362

表 7 最低信頼度と最大支持度

最低信頼度	最大支持度
0.7	0.28706
0.8	0.25882
0.9	0.17882
1.0	0.08235

表 7 は相関ルールマイニングを適用する場合の，最低信頼度を設定した場合の最大支持度，最大支持度のルールが成立するプロジェクトの件数，をまとめたものである．例えば，2 行目は，最低信頼度を 0.7 とした場合の最大支持度が 0.28706 である事を意味している．また，最大支持度のルールが成立するプロジェクトは，425 × 0.28706=122 件となる．最低信頼度 0.7 とは，前提部が成立するようなプロジェクトの 70%以上で結論部も成立するという意味である．

利用するデータには，不具合数が 0 であるようなプロジェクトが 205 件，不具合が存在するプロジェクトが 220 件存在する．このデータを用いた場合，支持度 0.1 のルールは 43 件以上のプロジェクトで成立するルールという意味になる．これは，不具合 0 であるプロジェクト 205 件のうち 2 割程度で成立するということになる．

そこで，本研究では，最低信頼度 0.9，最低支持度を 0.1 として相関ルールマイニングを行う．

4.4 相関ルールマイニングによるルールの抽出

4.2 節で加工したデータに対して，4.3 節で決定したパラメータを設定して相関ルールマイニングを行った結果，581 件の相関ルールを得た．以下のルール r_1 は，得られた相関ルールの中で最も支持度が高かったルールである．このルールの支持度は 0.17882 であり，信頼度は 0.95 であった．

r_1 実績の評価 (品質) = a,b (稼働後の不具合数が予定以下であった)
∧ 要員スキル_言語・ツール利用経験 = a,b (半数以上が十分な経験あり)
∧ SLOC 実績値_SLOC = 中央値未満
⇒ 発生不具合数 (現象数) = 0

4.5 ファクターの抽出

4.4 節で得られたルールは，ルール r_1 のように「メトリクス = 値」の組合せによって構成されている．このメトリクスと値の組合せを，本研究ではファクターと呼ぶこととする．例えば，ルール r_1 に含まれるファクターは,「実績の評価 (品質) = a,b」「要員スキル_言語・ツール利用経験 = a,b」「SLOC 実績値_SLOC = 中央値未満」「発生不具合数 (現象数) = 0」の 4 つである．

相関ルールの前提に含まれるファクターは，不具合が 0 件であるプロジェクトに共通して発生している事象であると考えられる．そこで，得られた 581 件の相関ルールの前提に含まれるファクターのみを抽出した．その結果，表 8 に示す 22 個のファクターが得られた．

5. 結果と考察

5.1 結果と解釈

4.5 節で得られた 22 個のファクターが不具合の発生と関連があるかどうかについて考えるために，不具合が多いプロジェクトでの各メトリクスの値を調査する．ここでは，不具合が多いプロジェクトとは，記入率が低いデータを削除する前の 816 件のプロジェクトのうち，発生不具合数 (現象数) が percentile 95 以上となっているプロジェクト (41 件) を指している．不具合が多いプロジェクトのメトリクスの値が，4.5 節で得られたファクターでの値と一致している割合を**一致率**とする．例えば，不具合が多い 41 件のプロジェクトのうち，38 件のプロジェクトが「実績開発工数が中央値以上」，1 件のプロジェクトが「実績開発工数が中央値より低い」と回答していた．また，2 件のプロジェクトについては空白回答であった．4.5 節で得られたファクターは「実績開発工数 = 中央値未満」であるため，一致率は $1 \div 39 = 2.6\%$ となる．一致率が低ければ低いほど，該当するメトリクスは不具合との関連が強いと考える事ができる．逆に，一致率が高い場合には，メトリクスと不具合に関連があるとは言えず，分析対象となったプロジェクトの背景となっていると考えることができる．一致率が 50%前後である場合は，強い関連があるとは言えないが，何らかの関連があると考えられる．

表 8 は，得られた不具合 0 ルールの前提に含まれているメトリクスの値と，不具合が多いプロジェクトでの値の一致率を計算し，一致率が低い順に並び替えたものである．本研究では，相関ルールマイニングによって得られたメトリクスを一致率によって三等分することとした．具体的には，一致率が 33.3% 以下のメトリクスを C_1，一致率 33.3%〜66.6%のメトリクスを C_2，一致率 66.7%以上のメトリクスを C_3 とした．

表 8 不具合 0 ルールと不具合が多いプロジェクトのメトリクスの値の一致率

	不具合 0 ルールに出現したメトリクス	一致率
C_1	実績開発工数 = 中央値未満	2.6 %
	SLOC 実績値_SLOC = 中央値未満	2.6 %
	実績月数_プロジェクト全体 = 中央値未満	4.9 %
	月あたりの SLOC$^{\star 1}$ = 中央値未満	7.3 %
	要員スキル_分析・設計経験 = a,b	33.3 %
C_2	新技術の利用 = b	54.2 %
	実績の評価 (品質) = a ,b	55.0 %
	要員スキル_開発プラットフォーム使用経験 = a,b	57.1 %
	要員スキル_業務分野経験 = a,b	58.8 %
	業務パッケージ_利用有無 = b:なし	63.9 %
	実績の評価 (コスト) = a ,b	65.7 %
C_3	要員スキル_言語・ツール利用経験 = a,b	68.8 %
	新規顧客 = b	70.6 %
	新規業種・業務 = b	71.4 %
	納期遅延$^{\star 2}$ = 遅延無し	72.0 %
	計画の評価 (品質) = a	73.9 %
	実績の評価 (工期) = a,b	77.1 %
	計画の評価 (工期) = a	78.3 %
	計画の評価 (コスト) = a,b	78.3 %
	システム種別 = a	100 %
	開発ライフサイクルモデル = a	100 %
	開発プロジェクト形態 = b	100 %

5.2 メトリクスの調査

ここでは，ファクターとして選ばれたメトリクスが実際に不具合と関連があるかどうかについて調査を行う．

C_1 に属するメトリクス

『月あたりの SLOC』『実績月数_プロジェクト全体』『SLOC 実測値_SLOC』『実績開発工数』『月あたりの SLOC=中央値未満』は，不具合 0 のルールでは『実績月数_プロジェクト全体=中央値未満』『SLOC 実測値_SLOC=中央値未満』『実績開発工数=中央値未満』のファクターが抽出されている．対して，不具合が多いプロジェクトでは 9 割以上のプロジェクトが『実績月数_プロジェクト全体=中央値以上』『SLOC 実測値_SLOC=中央値以上』『実績開

⋆1 月あたりの SLOC は，SLOC 実績値_SLOC を実績月数_プロジェクト全体で割ったものである．
⋆2 納期遅延は，文献 [7]275 ページのプロジェクト全体工期 (実績) をプロジェクト全体工期 (計画) で割ったものであり，1 より大きい場合に『遅延あり』，1 以下の場合に『遅延無し』としている．

発工数=中央値以上』となっている．このことから，規模の小さいプロジェクトでは稼働後の不具合が少ない傾向があり，規模が大きいプロジェクトでは稼働後の不具合が多い傾向がある．また，一ヶ月に開発するコード行数が少ないプロジェクトでは稼働後に不具合が出にくく，一ヶ月に開発するコード行数が多いプロジェクトでは稼働後の不具合が出ているプロジェクトが多い．

『要員スキル_分析・設計経験』は，不具合 0 ルールでは『a：全員が十分な経験』『b：半数が十分な経験，半数がいくらかの経験』のいずれか，となっている．データ白書 [7] では『a, b』と回答しているプロジェクトが 8 割あるが，一致率は 33%となっている．そのため，不具合が多いプロジェクトでは要員スキルが低い傾向があると言える．ただし，不具合が多いプロジェクト中のデータ数が 6 件しかなく，データ数が少ないためにこのような分布となった可能性がある．

C_2 に属するメトリクス

『新技術利用』の値は，不具合 0 のルールでは『b：利用なし』であり，一致率は一致率は 54%である．それに対して，データ白書 [7] によると，8 割のプロジェクトが『b：利用なし』と回答している．このことから，不具合の多いプロジェクトでは新技術を利用している傾向がある．

『要員スキル_業務分野経験』『要員スキル_開発プラットフォーム使用経験』は，不具合 0 ルールでは『a：全員が十分な経験』『b：半数が十分な経験，半数がいくらかの経験』が抽出されている．データ白書 [7] によると，8 割のプロジェクトが『a, b』と回答している．しかし，一致率は 6 割に満たない．このことから，これらのメトリクスに関しては，要員スキルが低いほうが不具合が多いプロジェクトが多い．

『業務パッケージ_利用有無』は，不具合 0 ルールでの値は『利用なし』となっている．データ白書 [7] によると，80%のプロジェクトが『利用なし』と回答している．一致率は 64%であるため，不具合の多いプロジェクトでは業務パッケージを利用している傾向がある．

『実績の評価 (コスト)』の一致率は 65%である．それに対して，データ白書 [7] によると，に『a』『b』と回答しているプロジェクトが 85%以上である．コスト超過するプロジェクトでは稼働後の不具合数が出ているプロジェクトは多い．

C_3 に属するメトリクス

『要員スキル_言語・ツール利用経験』は，不具合 0 ルールでは『a：全員が十分な経験』『b：半数が十分な経験，半数がいくらかの経験』のいずれか，となっている．データ白書 [7] によると,『a, b』と回答しているプロジェクトが 8 割ある．一致率は 7 割となっているため，不具合が多いプロジェクトの方では『c』『d』と回答している割合が若干高いが，不具合と関係があるとは言えない．

『新規業種・業務』及び**『新規顧客』**は，いずれも不具合 0 ルールでは『既存』となっている．データ白書 [7] では『既存』と回答しているプロジェクトが 8 割程度あり，不具合が多いプロジェクトでは 7 割程度のプロジェクトが『既存』と回答している． 不具合が多いプロジェクトでは『新規』と回答しているプロジェクトの割合が若干高いが，不具合と関連があるとは言えない．

『実績の評価 (工期)』は，整形前のデータでは 8 割のプロジェクトが『納期までに終了している』と回答している．**『納期遅延』**のメトリクスは，欠損値削除前のデータでは 692 件のプロジェクトが納期までにプロジェクトを終了しており，254 件のプロジェクトが納期を遅延していることから，73%のプロジェクトが納期までにプロジェクトを終了していることになっている．これらのメトリクスに関しては，不具合が多いプロジェクトでの分布と整形前の分布に差がほとんど無い事から，不具合数との関連は無い．

『計画の評価 (コスト)』『計画の評価 (品質)』『計画の評価 (工期)』の 3 つのメトリクスは，不具合 0 ルールとの一致率が 78%程度である．整形前のデータでは，８～９割のプロジェクトが『a：計画の根拠と実効可能性を検討済み』と回答しており，若干の分布の違いがあるが，不具合と関連があるとは言えない．

『開発プロジェクト形態』『システム種別』『開発ライフサイクルモデル』の 3 つのメトリクスは，不具合 0 ルールとの一致率が 100 %になっている．これは，分析に利用したデータに「アプリケーションソフトを作成する受託開発のプロジェクトで，開発ライフサイクルモデルとしてウォーターフォールモデルを用いている」という共通の背景があったということを意味している．

5.3 既存のルールとの比較

本研究では，定量的なデータに対して相関ルールマイニングを用いる事で，ソフトウェア稼働後の不具合発生と関係があると思われるメトリクスの抽出を行った．抽出されたメトリクスの中でも，C_1 や C_2 に属するメトリクスを重点的にチェックすることでプロジェクトの失敗を未然に防ぐ役に立つと考えられる．

一方，プロジェクトマネージャの経験をもとにチェック項目を導いた研究が存在する [3]. 研究 [3] では，表 9 の質問項目を導き出している．この質問項目は，No と回答している場合にプロジェクト失敗のリスク要因となる．C_1，C_2 に属するファクターは，一致しない場合に不具合が高くなる要因であるため，質問項目と比較が可能である．まず，(1)，(2)，(5)，

表 9 文献 [3] の質問項目

(1)	開発側および顧客側の上級管理職は，プロジェクトをサポートすることを正式に確約したか
(2)	エンドユーザはプロジェクトおよびシステムや製品の開発に熱心な態度で挑んでいるか
(3)	開発チームと顧客は要求事項を十分に理解しているか
(4)	顧客は要求事項の定義に十分に参加したか
(5)	エンドユーザの期待は現実的なものか
(6)	プロジェクトのスコープは安定しているか
(7)	開発チームには必要なスキルが揃っているか
(8)	プロジェクトの要求事項は安定しているか
(9)	プロジェクトチームには実装するテクノロジの経験があるか
(10)	プロジェクトチームの人数は作業に対して適切か
(11)	顧客，ユーザ全員がプロジェクトの重要性およびシステム，製品に対する要求事項に同意しているか

(6)，(8)，(10)，(11) に関しては，本研究で利用したメトリクスの中に含まれていないため，比較対象から除外する．(7) 及び (9) は『要員スキル_分析・設計経験』『要員スキル_言語・ツール利用経験』『要員スキル_言語・ツール利用経験』『要員スキル_業務分野経験』『要員スキル_開発プラットフォーム使用経験』に該当していると考えられる．(3)，(4) については，該当するメトリクスを抽出することはできなかった．[3] のリストでは触れられていないが本研究で抽出されたメトリクスとしては,『実績開発工数』『SLOC 実績値_SLOC』などの定量的なメトリクスが挙げられる．

このことから，本研究では，プロジェクトマネージャの経験によって導かれたリスク要因や，定量的なメトリクスを抽出できていると言える．この結果は，プロジェクトマネージャの経験によって導かれたリスク要因に対して，定量的な分析による裏付けができたことも示している．

6. ま と め

本研究では，IPA/SEC によって収集された企業横断的なソフトウェア開発プロジェクトデータ [7] に対して相関ルールマイニングを適用し，不具合が 0 件であるようなプロジェクトに共通して発生していると考えられる事象を抽出した．相関ルールマイニングの結果，22 個のメトリクスが抽出された．次に，不具合が多いプロジェクトでは不具合が 0 件であるプロジェクトとは逆の事象が発生していると想定し，実際に不具合が多かったプロジェクトでは不具合が 0 件であるプロジェクトと逆の事象が発生しているかどうかについて確認を行った．その結果，6 つのメトリクスについては，不具合が多かったプロジェクトと不具合が 0 件であったプロジェクトで逆の事象が発生していることが分かった．

また，従来研究と比較を行った結果，相関ルールマイニングによって抽出されたメトリクスの中には，プロジェクトマネージャの経験によって導き出されたリスク要因と一致しているものが含まれていることがわかった．この結果から，プロジェクトマネージャの経験的な知識に対して定量的な裏付けができたと考えている．

謝辞 この研究の一部は，経済産業省「平成 21 年度産業技術研究開発委託費（産学連携ソフトウェア工学実践事業)」，日本学術振興会科学技術研究費補助金基盤研究 (C)(課題番号：21500035)，及び日本学術振興会科学技術研究費補助金特別研究員奨励費 (課題番号：21・3963) の助成を受けている．

参 考 文 献

1) Boehm, B.W.: Software risk management: Principles and practice, *IEEE Software*, Vol.8, No.1, pp.32–41 (1991).
2) Han, J. and Kamber, M.: *Data Mining: Concepts and Techniques*, Morgan Kaufmann Publishers (2001).
3) Keil, M., Cule, P.E., Lyytinen, K. and Schmidt, R.C.: A framework for identifying software project risks, *Commun. ACM*, Vol.41, No.11, pp.76–83 (1998).
4) Michail, A.: Data Mining Library Reuse Patterns using Generalized Association Rules, *Proceedings of the 22nd international conference on Software engineering*, pp.167–176 (2000).
5) Morisaki, S., Monden, A., Tamada, H., Matsumura, T. and ichi Matsumoto, K.: Mining Quantitative Rules in a Software Project Data Set, 情報処理学会論文誌， Vol.48, No.8, pp. 2725–2734 (2007).
6) Zimmermann, T., Weißgerber, P., Diehl, S. and Zeller, A.: Mining Version Histories to Guide Software Change, *IEEE Trans.on Software Engineering*, Vol.31, No.6, pp.429–445 (2005).
7) (独) 情報処理推進機構ソフトウェア・エンジニアリング・センター（編）：ソフトウェア開発データ白書 2008，日経 BP 社 (2008).
8) ロジャー S. プレスマン：実践ソフトウェアエンジニアリング　ソフトウェアプロフェッショナルのための基本知識，日科技連 (2005).
9) 経済産業省，(独) 情報処理推進機構：2008 年版組込みソフトウェア産業実態調査報告書，http://sec.ipa.go.jp/reports/20080715.html (2008).
10) 浜野康裕，天嵜聡介，　水野修，　菊野亨：相関ルールマイニングによるソフトウェア開発プロジェクト中のリスク要 因の分析，コンピュータソフトウェア， Vol.24, No.2, pp.79–87 (2007).