ソフトウェアプロジェクト混乱に対するベイズ識別器による予測の試み

安部 誠也, 濱崎 考成, 水野 修, 菊野 亨

大阪大学 大学院情報科学研究科 情報システム工学専攻
E-mail: {s-abe, o-mizuno, kikuno}@ist.osaka-u.ac.jp

概要

ソフトウェアの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．特にプロジェクトが制御不能になる状態，いわゆる混乱状態になるかどうかをプロジェクトの早期に発見することが求められている．そこで本論文では，ベイズ識別器を用いたソフトウェア開発プロジェクトの混乱予測手法を利用した混乱予測ツールの開発について述べる．ベイズ識別器を用いた混乱予測手法では，開発プロジェクトの関係者に対してアンケートを実施して，プロジェクトが抱えるリスクの程度を抽出する．そしてベイズ識別器を用いてアンケートとプロジェクトの混乱の有無の関係を調べ予測を行う．本報告では，ある企業の協力を得て実際の開発現場において実施したアンケートを用いて適用実験を行った．その結果，全体の *79%*のプロジェクトにおいて正しく予測することができた．

1　まえがき

ソフトウェアの開発に求められる期間は年々短くなっているが，その反面，ソフトウェア製品は高い信頼性を求められるようになってきている．こうした状況下でソフトウェア開発プロジェクトが抱える問題点を早期に発見することの重要性が高まってきている．そのため，ソフトウェア開発プロジェクトにおけるリスクを分類して，リスクの回避を行う技術の開発が望まれている．

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 らによって古くから行われてきており [1]，リスクを分析してプロジェクトの最終状態に関する予測を行う研究も行われている [4, 6]．

我々は，ソフトウェア開発プロジェクトの早期に行うリスク調査アンケートによって，そのプロジェクトが最終的に混乱状態に陥るかどうかを判定する手法を提案してきた．従来の手法では，過去に収集したアンケートのデータに対して統計的分析を行い，ロジスティック曲線に基づくモデルを作成した．そして，新規に入力されるアンケートをこのモデルに当てはめることにより，プロジェクトの初期時点でも最終状態をある程度予測できることを示した [5]．また，アンケートのデータに対してクラスタ分析を行うことで，同様な予測を行う手法も提案してきた [7]．

これらの手法では，回答が得られなかった，あるいは回答者がその項目について分からないという回答を，他の回答と同様に一つの回答値として取り扱っていた．しかし実際の開発現場での使用を考えた際に全てのアンケート項目に対して回答が得られるという前提は不自然であり，未回答のデータを適切に取り扱える手法は開発現場への導入の容易さにもつながることから，このような評価が得られなかった回答を適切に取り扱うことが課題であった． さらに，これらの手法は回答を得てから予測モデルを作るまでの手順が煩雑であるため，進行中のプロジェクトに

対する迅速な適用が難しいという問題もあった．

そこで本研究では，未回答の項目のあるアンケートに対しても簡単に予測を行う手法としてベイズ識別器を用いた手法を提案する．ベイズ識別器はベイズの定理を利用してカテゴリカルデータをいくつかのクラスに高い精度で分類する手法としてよく知られている [2]．本研究では，アンケートの回答結果からプロジェクトの混乱の有無，及び最終状態に関するいくつかの要因をこのベイズ識別器を用いて予測する．

2　目的と準備

2.1　研究の背景

近年，ソフトウェアの規模が大きくなり，また開発期間が短くなるにつれ，プロジェクト管理の重要性が増している．

実際のソフトウェア開発現場では，稀にではあるがプロジェクトの進行状況すら誰にも把握できなくなってしまうようなことが発生する．このようなプロジェクトでは，多くの場合「混乱状態」と呼ばれる状態に陥ることを繰り返した後に最悪の状態にまで発展してしまう．混乱状態に陥ったプロジェクトの全てが最悪の事態を招くわけではないが，こうした混乱状態を事前に予測できれば最悪の事態を防ぐことが出来る．

これまでソフトウェア開発プロジェクトの混乱状態を予測するためにいくつかの研究が行われてきた．これらの研究を通じて「開発現場のプロジェクトマネージャは無意識にではあるが，プロジェクトの混乱につながるリスク要因をつかんでいる」ということが分かってきた．

そこで我々はプロジェクトマネージャへのアンケート調査を実施し，アンケート調査に基づいて，プロジェクトの混乱とリスク要因との間に統計的モデルを作成し，そのモデルを適用して新規プロジェクトの混乱状態を予測する手法を提案してきた [5, 7]．

2.2　プロジェクトの混乱状態

開発現場の状況が制御不可能になってしまうプロジェクトを混乱プロジェクトと呼ぶ．しかしこれでは漠然としているので，ここでは混乱プロジェクトを次のように定義する．

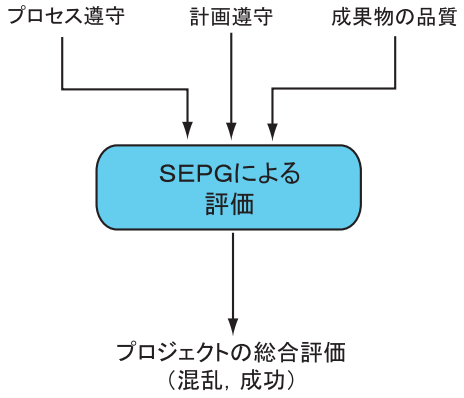

図 1. 混乱プロジェクトの評価

計画との相対誤差　混乱状態に陥ったプロジェクトは，工数・開発期間・納期などにおいて計画からのずれが生じる．そこでこれらの計画との相対誤差が基準値を超えているもの

開発プロセス　設計・レビュー・テストの各プロセスが組織の内部規約通り実施されていないプロジェクト

成果物の品質　プロジェクト成果物の品質が著しく悪いプロジェクト

これらの基準を基に，組織の SEPG(Software Engineering Process Group) により混乱状態に陥ったと認められたものを混乱プロジェクトと定義する (図 1 参照)．

3　ベイズ識別器によるプロジェクト混乱予測ツール

我々は，ソフトウェアの混乱予測のためにアンケートの収集から混乱の予測までを自動的に行うシステムの開発を目指している (図 2)．以降では，混乱予測の流れと，その中で使用するアンケート項目の作成，そしてベイズ識別器の理論について説明する．

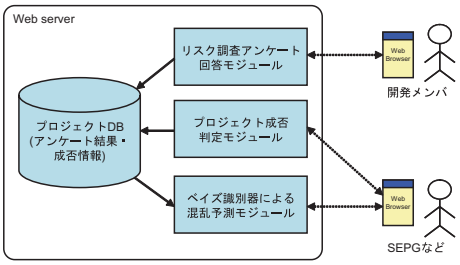

図 2. 混乱予測システム

3.1 ツールによる混乱予測の流れ

図 3 は本論文で提案する混乱予測手法の大まかな作業の流れを表している．

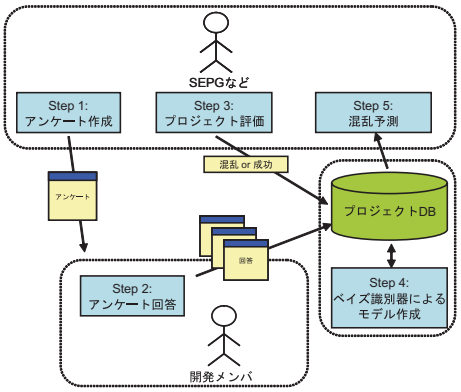

図 3. 混乱予測の流れ

まず，ステップ 1 でプロジェクトメンバーからプロジェクトの評価を聞き出すためのアンケートを作成する．本研究ではある企業の協力を得て企業の SEPG が中心となって作成した．作成に当たっては，私たちの研究グループが過去に他の企業と協力して作成したものを参考に，企業の内部規約・プロジェクト特性を考慮した．作成したアンケートは 9 つの観点から計 30 の質問項目を持つものになった．アンケートへの回答方法としては，質問項目ごとに回答選択肢をいくつか用意し，その何れかを選択するものとした．

続いてステップ 2 では作成したアンケートをプロジェクトメンバーに配布し，アンケートに回答してもらう．本研究では計 19 のプロジェクトを対象として延べ 116 人のプロジェクトメンバーにアンケートを実施した．

そのうち既に終了した 15 個のプロジェクトに携わったプロジェクトメンバーのべ 80 人から得られた回答を，以降の適用実験で使用する．また，残りの 4 つのプロジェクトは現在進行中であり，プロジェクトメンバーのべ 36 人からもアンケートの回答も既に得られているが，次に述べるステップ 3 での混乱判定が得られていないため，今回は利用しない．

提案手法ではプロジェクトの混乱予測を行うにあたり，過去のプロジェクトのデータを利用する．予測を行うモデル作成には混乱の有無が分かっているプロジェクトデータが必要であるため，この混乱の有無の判定をステップ 3 で行う．本研究では 2.2 節で述べたように

- 計画値との相対誤差
- 組織により定められた開発プロセス規約
- プロジェクト成果物の品質

を基に SEPG が総合的に判断した．

ステップ 4 では開発した混乱予測ツールを利用して，学習対象としたアンケート回答とプロジェクトの結果からベイズ識別器によるモデルを作成する．このとき，アンケートの回答結果は部分的に未回答の項目があるものも問題なく扱える．引き続き，ステップ 5 では新規プロジェクトについて混乱予測を行う．混乱予測は新規プロジェクトのアンケート結果をステップ 4 で作成したモデルに当てはめることで行う．ステップ 5 の作業も開発したツールを用いることで自動的に行える．

3.2 リスク調査アンケート

本節では，混乱予測を行うためのにプロジェクトメンバーに対して行ったリスク調査アンケートについて説明する．

このアンケートの目的は，プロジェクトを混乱状態へと陥らせるような要因の有無・程度をプロジェクトメンバーから聞き出すことである．

アンケートの作成に当たっては，我々が過去の研究において作成したアンケート [5] を基に[1]，企業の内部規約・プロジェクト特性などを考慮してＳＥＰＧが中心となって作成した．作成したアンケートは次の 9 つの主要な要因に整理される．

1 要求仕様の問題点

2 見積りの問題点

3 プロジェクト体制の問題点

4 工程計画の問題点

5 進捗管理の問題点

6 開発技術の問題点

7 開発メンバーの問題点

8 プロジェクト環境の問題点

9 その他の外的要因に関する問題点

これらの各要因について，より詳細な質問項目を設けてアンケートを作成した．例えば「(1) 要求仕様の問題点」では，

1.1 不適切な要求

1.2 開発側の要求理解不足

1.3 企画・開発側相互の要求に対する合意不足

1.4 企画からの要求変更の多発

という 4 つの質問項目を作成した．このように全要因について詳細な質問項目を設け，計 30 の質問項目が作成された．詳細を表 1 に示す．

アンケートでは，プロジェクトメンバーが各質問項目に対する評価を行う．評価は各項目ごとに回答選択肢を用意し，その何れかを選ぶことによって行う．例えば表 1 の「(1.1) 不適切な要求」の項目では，回答選択肢として「かなりある」「ある」「少しある」「ほとんどない」「ない」「分からない」の 6 つの選択肢を用意した．回答選択肢は基本的に 6 段階であるが，質問内容に合わせて 3〜5 段階に調整している．例えば「(3.4) コントロール不能な開発メンバー」では，回答選択肢として「複数いる」「一人いる」「いない」「分からない」の 4 段階に設定している．なお，全ての質問項目に対して「分からない」という回答選択肢は必ず設けている．

[1] 文献 [5] でのアンケートを作成する際には，これまで文献等で発表されているリスク要因などに基づき，できる限り広範な要因を含めることを目指した．

表 1. リスク調査アンケート

1．要求仕様	
1.1	不適切な要求
1.2	開発側の要求理解不足
1.3	企画・開発側相互の要求に対する合意不足
1.4	企画からの要求変更の多発
2．見積り	
2.1	見積り対象の項目の過不足
2.2	見積りの重要さの認識不足
2.3	見積りの根拠不十分
3．プロジェクト体制	
3.1	開発メンバーの知識・スキル不足
3.2	不明確な役割分担・責任範囲
3.3	開発作業負荷の集中
3.4	コントロール不能な開発メンバー
4．工程計画	
4.1	不十分／不適切な成果物定義
4.2	マイルストーンやレビューの過不足
4.3	計画に対する関係者全員のコミットメントなし
4.4	計画に対するマネージャのレビュー不足
4.5	過度に厳しい納期／工数／コスト
5．進捗管理	
5.1	要求管理，仕様変更管理不足
5.2	進捗状況把握不足
6．開発技術	
6.1	達成可能性が見えない機能要求／性能要求
6.2	固有技術の新規性，不慣れ度
6.3	新しい開発方法論・開発プロセスの採用
7．開発メンバー	
7.1	開発メンバーのモチベーションが低い
7.2	開発メンバーの健康不良，事故
7.3	プロジェクト途中での退職者
7.4	開発メンバーの余裕の欠如
8．環境	
8.1	開発環境・テスト環境の整備不十分
8.2	不適切，不十分な構成管理
9．その他の外的要因	
9.1	開発経費の圧縮
9.2	購入（選定）した製品・ツールの品質問題
9.3	世の中のソフト動作環境の変化

3.3 ベイズ識別器

ベイズ識別器 はベイズの定理を利用してカテゴリカルデータをいくつかのクラスに高い精度で分類する手法として良く知られている．ベイズ

識別器は各属性が互いに独立であるという仮定を置いているが，経験的に仮定が破られたデータセットであっても極めて精度の高い予測が可能であることが知られている [2]．この節では，ベイズの定理とベイズ識別器の概要について説明する．

3.3.1 ベイズの定理

ベイズの定理は，事前確率を事後確率に変換するものである [3].

確率変数 A，B において，事前確率 $P(A)$，$P(B)$，事後確率 $P(A \mid B)$ とするとき，次のような関係が成り立つことが知られている．

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}$$

3.3.2 ベイズ識別器

$Q_1, Q_2, \cdots, Q_n$ を属性集合とし，C を予測を行うクラスとする．ここでは，属性集合，クラス共に名義尺度として扱う．

属性集合が，$Q_1 = q_1, \cdots, Q_n = q_n$ と与えられたとき，クラス $C = confuse$ となる確率 $P(C = confuse \mid Q_1 = q_1 \wedge Q_2 = q_2 \wedge \cdots \wedge Q_n = q_n)$ は，ベイズの定理を用いて次のように表される．

$$\frac{P(Q_1{=}q_1 \wedge \cdots \wedge Q_n{=}q_n \,|\, C{=}confuse)}{P(Q_1{=}q_1 \wedge \cdots \wedge Q_n{=}q_n)} \times P(C = confuse)$$

$P(C = confuse)$ はトレーニングデータセットから簡単に求めることができる．また，$P(Q_1 = q_1 \wedge \cdots \wedge Q_n = q_n)$ はクラス C の値に関わらず一定の値であり事後確率の合計が 1 になるための正規化係数と考えることができる．そこで，$P(Q_1 = q_1 \wedge \cdots \wedge Q_n = q_n \mid C = confuse)$ を求めることに焦点を当てる．

ベイズの定理を再び適用すると，この部分は次のように変形できる．

$$\begin{aligned} &P(Q_1{=}q_1 \,|\, Q_2{=}q_2 \wedge \cdots \wedge Q_n{=}q_n, C{=}confuse) \\ &\quad \times \quad P(Q_2{=}q_2 \wedge \cdots \wedge Q_n{=}q_n \,|\, C{=}confuse) \end{aligned}$$

この式の第 2 項目も同様にして，

$$\begin{aligned} &P(Q_2{=}q_2 \,|\, Q_2{=}q_2 \wedge \cdots \wedge Q_n{=}q_n, C{=}confuse) \\ &\quad \times \quad P(Q_3{=}q_3 \wedge \cdots \wedge Q_n{=}q_n \,|\, C{=}confuse) \end{aligned}$$

ここで，各属性 $Q_i (1 \leq i \leq n)$ がお互いに独立であると仮定すると，

$$\begin{aligned} &P(Q_1{=}q_1 \,|\, Q_2{=}q_2 \wedge \cdots \wedge Q_n{=}q_n, C{=}confuse) \\ &\quad = \quad P(Q_1 = q_1 \mid C = confuse) \end{aligned}$$

が成り立つ．

よって，$P(Q_1 = q_1 \wedge \cdots \wedge Q_n = q_n \mid C = confuse)$ は次のようになる．

$$\prod_{i=1}^{n} P(Q_i = q_i \mid C = confuse)$$

以上より，ベイズ識別器 により，クラス $C = confuse$ となる確率は次の式で求められる．

$$\begin{aligned} &P(C{=}confuse \,|\, Q_1{=}q_1 \wedge \cdots \wedge Q_n{=}q_n) \\ &\quad = \quad \prod_{i=1}^{n} P(Q_i{=}q_i \,|\, C{=}confuse) \\ &\qquad \times P(C{=}confuse) \\ &\qquad / P(Q_1{=}q_1 \wedge \cdots \wedge Q_n{=}q_n) \end{aligned}$$

これを用いて，全てのクラスについて確率を計算し，最も大きいクラスに分類することで予測を行う．なお，各項は，属性・クラスの値が分かっている学習データセットから求められる．

3.4 混乱予測モジュールの実装

混乱予測モジュールは Web 上で動作するアプリケーションとして実装した．使用した言語は Perl と R で，開発規模はほぼ 1000 行であった．

このモジュールの動作について説明する．まず，プロジェクトデータベースから学習に必要なプロジェクトデータを取得する．そして，取得したデータをベイズ識別器によって分類し，確率モデルを作成する．次に，分類したいデータを再びプロジェクトデータベースから取得し，先に作成した確率モデルを用いて混乱と成功のどちらのクラスに属するかを判定し，その結果を出力する．

この開発したツールを用いることで，アンケート結果から混乱するかどうかの予測を非常に簡単に行える．なお，開発しているツールは当研究室の Web ページ (http://www-ise4.ist.osaka-u.ac.jp/) にて公開予定である．

4 適用実験

本章では，3.2 節で作成したアンケートを実際のソフトウェア開発プロジェクトに対して適用し，アンケートの回答結果から ベイズ識別器 による混乱予測を行った結果を示す．

4.1 対象プロジェクト

ある企業において実際に行われたソフトウェア開発プロジェクトを対象とした．

対象としたプロジェクトはある企業において 2001 年から 2002 年の間に実施された 15 個のプロジェクトで，アンケートは 1 プロジェクトにつき 1 人から 9 人の計 80 人のメンバーに対して行った．このメンバーには，プロジェクトの評価を行ったメンバーなど，一部分のみに携わったメンバーも含まれる．

これらのプロジェクトは SEPG の活動に用いるソフトウェアの開発プロジェクトで，スパイラル型の開発モデルを採用している．プロジェクトの規模は平均で約 7KLOC である．

4.2 プロジェクトの評価

本実験における対象プロジェクトは既に終了しているので，各プロジェクトの評価を SEPG が行った．プロジェクトの評価は，2.2 で示したように，プロジェクトの計画・プロセス，およびプロジェクト成果物の品質を基に SEPG が総合的に判断した．(表 2)

工数・期間については見積り値と実際の値との相対誤差を計り，その値により，成功（○），失敗（×）を決定した．納期は，計画通り守られたもの（○）と守られなかったもの（×）を判定した．設計，レビュー，テストのプロセスについては，企業の内部規約通り，混乱なく実施されたもの（○）と実施されなかったもの（×）を判定した．

そして，これらの判定値とプロジェクト成果物の品質から，SEPG が総合的にプロジェクトの混乱の有無（○：成功，×：混乱）を決定した．

4.3 アンケートの回答データの取り扱い

3.2 節で示したアンケートをプロジェクトメンバー 80 人に実施した．アンケートでは各質問項目に 3〜6 段階の回答選択肢を用意し，その何れかを選ぶことでアンケートを実施している．本適用実験では得られた回答値は名義尺度として取り扱っている．なお，すべての項目において「分からない」という回答選択肢を設けているが，この回答については他の回答選択肢とは異なり，一つの回答値として取り扱うのではなく値を持たない不明値として取り扱う．

4.4 混乱予測結果

得られたアンケート結果とプロジェクト評価を利用して，ベイズ識別器により混乱予測を行った．

提案手法の性能を評価するにあたり，ジャックナイフ法による検証を行った．ジャックナイフ法とは，データセットの中から 1 つのデータを取り出し，残り全てのデータを学習に用いて取り出したデータの予測を行い，その精度を評価する手法である [3]．このようにすることで，複数のモデルを作成すると共に全てのデータの予測を行うことができるので，モデルの性能評価の際によく利用されている．

本研究では，80 個のアンケート回答を使用して実験を行った．そして，得られた予測結果と SEPG によって評価された実際の結果を比較した．結果を表 3 に示す．

80 個の回答の中には未回答の項目が含まれるものもあったが，ベイズ識別器での分析は問題なく実行されている．表 3 に示すように，全回答 80 個中 63 個の回答について正しく予測することができ，予測精度は 79% となった．このことから提案手法によりプロジェクトの混乱予測が可能であることが確認できた．また，この結果にフィッシャーの正確確率検定を行ったところ，$p < 0.01$ となり統計的にもこの手法が有用であ

表 2. プロジェクト成否結果

Prj.	計画遵守			プロセス遵守			総合判定
	工数	期間	納期	設計	レビュー	テスト	
A	○	×	×	○	×	○	×
B	×	○	○	○	○	○	○
C	○	○	○	○	○	○	○
D	○	○	○	○	○	○	○
E	×	×	×	○	○	○	×
F	○	○	○	○	○	○	×
G	×	○	×	○	○	○	○
H	×	×	×	○	○	○	×
I	×	×	×	○	×	○	×
J	○	○	×	○	×	○	×
K	×	×	×	×	×	○	×
L	×	×	×	×	×	○	×
M	×	×	×	○	×	○	×
N	○	○	○	○	○	○	○
O	○	○	○	○	○	○	○

表 3. プロジェクトの混乱予測結果

実際	予測結果	
	混乱 (×)	成功 (○)
混乱 (×)	39	12
成功 (○)	5	24

予測精度：0.79

ることが分かった．なお，これとは別のプロジェクトデータに対して同様の実験を行ったところ，そちらでも予測精度は 80%となり非常に高い精度を示した．

ただ，表 3 に示す結果には問題点もある．予測で成功 (○) とされながら，実際には混乱 (×) した 12 プロジェクトの存在である．このようなプロジェクトは全体的な精度に関わらず好ましくないものであるため，今後削減していかなければならない．

5 まとめと今後の課題

本研究では，ソフトウェア開発プロジェクトの混乱予測を行う手法として，プロジェクトメンバーへのアンケートとベイズ識別器を用いる手法を提案し，そのためのツールの実装を行った．また実際のプロジェクトデータを用いて適用実験をおこなった．その結果，非常に高い精度でプロジェクトの混乱予測が可能であることがわかった．

しかし，本手法はあくまで簡易的な混乱と成功の目安を提供するだけにとどまっており，具体的に何を改善すべきかを提示することはできない．今後，まず，混乱すると予測されたプロジェクトに施した改善案によって混乱が回避されたプロジェクトについて，その改善案を収集する．そして，リスク要因と改善案の関係を調べることによって，混乱すると予測された新たなプロジェクトに改善案を提供できる手法へと発展させていく必要がある．また，プロジェクトの早期と終了時にアンケートを実施することによって，リスクマネジメントプロセスの存在がプロジェクトの結果にどのように影響を与えるかを考慮できるフレームワークへの拡張も必須である．

また，現在進行中のプロジェクトに対しても同様のアンケートを行っており，そのプロジェクトの成否についても今後，プロジェクトの終了時に評価する予定である．

謝辞

本研究を行うにあたって，アンケートの作成・収集に協力していただき，また貴重なデータを提供して頂いた株式会社リコーの松瀬健司様，坂田英信様にこの場を借りて感謝致します．

参考文献

[1] B. W. Boehm. Industrial software metrics top 10 list. *IEEE Software*, 4(5):84–85, 1987.

[2] P. Domingos and M. J. Pazzani. On the optimality of the simple bayesian classifier under zero-one loss. *Machine Learning*, 29(2-3):103–130, 1997.

[3] R. O. Dura, P. E. Hart, Daviid G., 尾上守夫監訳. パターン識別. 新技術コミュニケーションズ, 2001.

[4] J. Jiang and G. Klein. Software development risks to project effectiveness. *Journal of Systems and Software*, 52:3–10, 2000.

[5] O. Mizuno, T. Kikuno, Y. Takagi, and K. Sakamoto. Characterization of risky projects based on project managers' evaluation. In *Proc. of 22nd International Conference on Software Engineering*, pp. 387–395, 2000.

[6] C. Wohlin and A. A. Andrews. Prioritizing and assessing software project success factors and project characteristics using subjective data. *Empirical Software Engineering*, 8:285–303, 2003.

[7] 濱崎考成, 水野修, 菊野亨, 高木徳生. リスク管理のためのアンケート回答のクラスタ分析と混乱プロジェクト発見への応用. ソフトウェアシンポジウム 2002, pp. 159–166, 2002.