アンケート調査に基づく開発中のプロジェクトの混乱予測とその予測作業支援システムの開発

足立 卓也†，水野 修†，菊野 亨†，高木 徳生‡，坂本 啓司‡
†大阪大学 大学院基礎工学研究科 情報数理系専攻
E-mail: adati@ics.es.osaka-u.ac.jp
‡オムロン株式会社

Abstract

本報告では，開発中のプロジェクトの混乱予測をロジスティック回帰分析に基づいて行う試みについて述べる．ソフトウェアの開発現場では，開発初期の段階でプロジェクトが混乱するかどうかを予測できることが望ましい．多くの場合，プロジェクトマネージャは経験からそうした混乱要因がある程度分かっている．我々はプロジェクトマネージャに対するアンケートを行い，その集計結果に基づいてプロジェクトの混乱を予測する手法の開発を進めてきた．ここではこの手法を実際に開発現場に導入することを目指して，開発初期の 2 つの時点でアンケートを行うことを試みる．まず，そのためのアンケート調査表の作成を行った．引き続いて，実際のプロジェクトデータを利用した適用実験を行い，得られた予測結果について評価した．さらに，この手法の開発現場への導入を容易にするための支援システムの試作についても述べる．

1 まえがき

ソフトウェアの開発期間が次第に短くなり，その反面，高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予測することの重要性が高まってきている．

実際の開発現場では，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「失敗プロジェクト」と見なされるプロジェクトが発生する．このようなプロジェクトでは，多くの場合，「混乱状態」と呼ばれる状況に陥ることを繰り返した後，最悪の事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全てが最悪の事態になるわけではないが，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もできる．

我々は，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を行ってきた．これらの研究を通じて分かってきたこととして，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を予測できるリスク要因をつかんでいることが多い．そこで，プロジェクトマネージャへのアンケート調査に基づいて，プロジェクトのリスク要因とプロジェクト混乱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の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 [10].

この手法の開発に当たってプロジェクトマネージャへの調査表を作成した．この調査表は開発終了時点で記入することを前提としたため，一部には開発中に記入することは困難と思われる項目も含まれていた．しかし，現実にはプロジェクトの混乱予測は開発の早い段階で行えることが強く望まれる．

本研究では，設計開始前の時点と設計完了時点の 2 つのそれぞれの時点で記入できる調査表を作成し，それらを用いて算出された予測結果を比較する．また，それぞれの予測精度の違いについて考察する．更に，この手法を実際の開発現場で運用する際にその支援を行うシステムを実装し，その評価を行う．

2 本研究の目的

(1) 開発の初期段階で記入できるアンケート調査表の作成とその評価.

文献 [10] で開発しているアンケート調査表は，リスク管理に関する専門書や論文 [3, 5, 9] と協力企業における内部規約を調査して作成していた．しかし，その調査表には開発が終了してからでないと判断できない項目が含まれていた．しかも，混乱予測は早い段階で適用できなければ実効性に乏しい．実際，協力企業からも設計終了時点までででの適用が求められた．

そこで，設計作業の開始時点と終了時点の 2 つで記入可能な 2 種類の調査表を作成する．次に，それら

の調査表に基づいて適用実験を行い，混乱予測結果の違いについて分析する．

(2) 提案する予測作業を支援するシステムの開発．

提案する混乱予測手法は，調査表の配付・回収やデータの集計，ロジスティックモデルの作成などに多くの手間がかかっていた．そこで，これらの作業を支援するシステムの開発を目指す．

3 プロジェクトの混乱予測 [10]

ここでは，文献 [10] で提案しているプロジェクトの混乱予測手法の概要を説明する．

3.1 混乱プロジェクト

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 [13]．しかし，これだけでは定義がかなり漠然としているので，混乱プロジェクトを次のように定式化する．

基本的には開発コストと開発期間の 2 つについて開発計画作成時の推定値と実績値のずれを見て，いずれかが基準値を超えていれば，それは混乱プロジェクトであると判断する．しかし，これでも必ずしも十分ではないので，プロジェクト管理者と開発者にインタビューを行って同意が得られたものだけを最終的に混乱プロジェクトと定めることにする (図 1)．

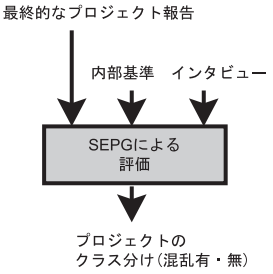

図 1. 混乱プロジェクトの判定

3.2 予測モデルの作成

これまでの検討で，我々はプロジェクト管理者が混乱プロジェクトの兆候を明らかに感じていることを分かっていた．しかし，それが混乱プロジェクトであると断言するだけの理論的根拠が無かったので，確信するには至っていなかった．そこでプロジェクト管理者の直感の中味を引き出すことによって混乱プロジェクトを特性づけるヒントが得られるのではないかと考えた．

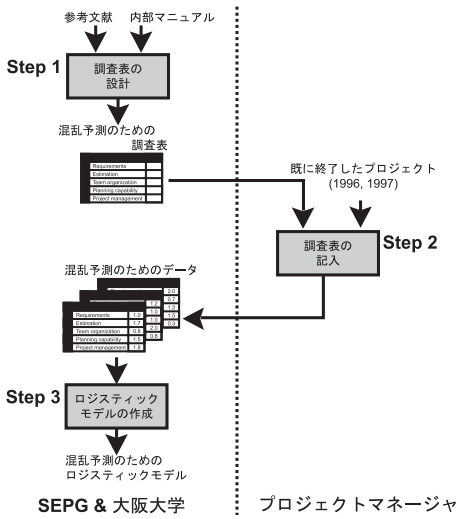

図 2. 混乱予測モデル作成の概要

図 2 に文献 [10] で行った予測モデルの作成手順の概要を示す．Step 1 では，まず幾つかのリスク要因についてプロジェクト管理者の評価を引き出すための調査表を作成する．Step 2 では，Step 1 で作成された調査表を，既に終了しているプロジェクトの担当マネージャに配布して，各項目に記入後，回収する．Step 3 では SEPG によるプロジェクトの評価 (既に終了しているプロジェクトについて混乱したかどうかの判定) を行い，その評価と調査表のリスク要因からロジスティック回帰分析を行う．その結果，混乱予測のためのロジスティックモデルが作成される．

3.3 調査表の作成 (Step 1)

調査表の設計に当たっては，リスク管理に関する専門書や論文 [3, 5, 9] と協力企業における内部規約を調査した．その結果，混乱プロジェクトを引き起こすリスク要因を次の 5 つの主要な要因に整理した．

(1) 要求仕様の定義と理解に関する問題点

(2) 実現すべきプロダクトの規模や機能の見積りに関する問題点

(3) 開発チームの編成と人材 (能力) に関する問題点

(4) 開発計画の作成方法とその内容に関する問題点

(5) 技術的な事項や外的事項に対するプロジェクト管理に関する問題点

5つの主要な要因はそれぞれより詳細なレベルの項目に展開されている．なお，これらの調査項目は開発終了時点で全て記入可能であることを目安にして作成されている．(なお，文献 [10] の調査表の項目は全て設計終了時点で記入可能であることが分かった．今回使用した調査表についてはその詳細を 4 節で説明する．)

3.4 調査表への記入 (Step 2)

この調査表への回答を混乱予測モデルを作成するための基礎データとするために，“ある”，“恐らくある”，“ない” という 3 種類の回答に評価値 2，1，0 を対応づける．また “分からない” という回答については “恐らくある” と同じだと解釈して，1 を割り当てた．続いて，5 つの各リスク要因ごとに評価値の総和を取り，それぞれの項目数で割った．結果として，5 つのリスク要因ごとに 0〜2 の範囲の値をとる測定データを得た．

3.5 ロジスティック回帰分析 (Step 3)

ロジスティック回帰モデルは，目的変数 Y が 2 値の事象であるとき，最も一般的な回帰分析手法である．本研究では，目的変数は「プロジェクトが混乱する/しない」という 2 値の事象である．プロジェクトマネージャに混乱プロジェクトに対する理論的根拠を与えるという目的からも，統計的手法として確立されているロジスティックモデルを採用している．ロジスティックモデルは次式

$$E(Y|x_1,\cdots,x_n) = \frac{e^{b_0+b_1x_1+\cdots+b_nx_n}}{1+e^{b_0+b_1x_1+\cdots+b_nx_n}}$$

に基づいている [1, 12]．E はそのプロジェクトが混乱プロジェクトである確率を表す．$x_1,\cdots,x_n$ はこのモデルにおける説明変数である．調査表の 5 つの主要な混乱要因がこれらの説明変数の候補になる．係数 $b_0, b_1, \cdots, b_n$ の値は調査表への回答 (評価) を用いて決定される．なお，求まるモデル式の妥当性の議論は付録にその概要のみを示す．

4 開発中のプロジェクトへの適用

2 節でも述べたように現実の開発現場を考えた場合，開発のできるだけ初期の時点で混乱予測ができることが望まれる．そこで，ここでは図 3 に示す開発初期の 2 つの時点を考える．すなわち，設計開始前と設計完了時点の 2 つである．設計開始前に配付する調査表を Q1，設計終了時に配付する調査表を Q2 と表す．

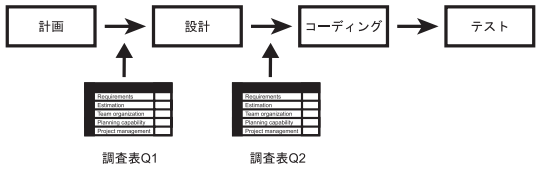

図 3. Q1，Q2 の配付時期

4.1 対象プロジェクト

あるソフトウェア開発企業の協力を得て，1996 年から 1998 年までに開発された 40 プロジェクトのデータを利用する．今回の実験では 1996 年と 1997 年に実施された 32 プロジェクトをロジスティックモデル作成に利用する．次に，1998 年に実施された 8 プロジェクトが新規に実施されたものと仮定して，これらについて実際に混乱が予測できるか否かを調べる．

4.2 調査表の設計

ここではプロジェクトマネージャに対して配付する調査表について，次の 2 つを作成した (表 1，表 2)．

調査表 **Q1:** 設計開始前で記入可能な調査表

調査表 Q1(表 1) は計画作成段階だけで判断できる項目で構成する．そのため 3.3 節で説明した 5 つの要因のうち，要求仕様，見積り，工程計画の 3 つだけを含む．

表 1. 調査表 Q1

1. 要求仕様	
1.1	要求仕様が不明確なままでの要求
1.2	要求仕様に対する顧客側，実現側相互の合意不足
2. 見積り	
2.1	非技術的圧力に妥協した
2.2	見積りの根拠不足
3. 工程計画	
3.1	作業に対する責任分担不明確
3.2	作業成果物定義不充分
3.3	マイルストーンやレビュー時期の設定不足
3.4	進捗管理方法不明

調査表 **Q2:** 設計完了時点で記入可能な調査表

調査表 Q2(表 2) は 3.3 節で説明した 5 つの要因をすべて含んだものとなる．Q1 と同じ要因であっても，設計完了時になって初めて回答できる項目もあるため Q2 は Q1 より大きくなっている．

今回の実験では，開発現場のマネージャに対しては調査表 Q2 だけを配付し Q2 に対する回答を回収した．調査表

表 2. 調査表 Q2

1. 要求仕様	
1.1	要求仕様が不明確なままでの要求
1.2	要求仕様に対する顧客側，実現側相互の合意不足
1.3	要求仕様の引き出し力不足
1.4	要求仕様の理解力不足
2. 見積り	
2.1	非技術的圧力に妥協した
2.2	見積りの根拠不足
2.3	見積り項目不足
2.4	見積りの重要さの認識不足
2.5	技術的課題に対する楽観的すぎる見積り
3. プロジェクト体制	
3.1	要員のスキル不足
3.2	プロジェクト体制整備の必要性認識不足
4. 工程計画	
4.1	作業に対する責任分担不明確
4.2	作業成果物定義不充分
4.3	マイルストーンやレビュー時期の設定不足
4.4	進捗管理方法不明
4.5	計画に対する関係者全員のコミットメントなし
4.6	計画に対する上級マネージャのレビュー不足
5. 進捗管理	
5.1	技術的側面でのリスク管理不足
5.2	モラール不足
5.3	工数不足
5.4	要件，仕様変更管理不足
5.5	進捗状況報告不足
5.6	進捗管理データ不足

Q1は実際には配付せず，Q2について回収したデータからQ1に対するデータを得ている．

4.3 適用実験

2つの時点でアンケート調査が実施されたと仮定して，それぞれ96年と97年の32プロジェクトのデータ(PJ01〜PJ32)からロジスティックモデルを作成する．そして，それぞれの場合に98年の8件の新規プロジェクト(PJ33〜PJ40)が混乱する確率を予測する．

なお，この実験では説明変数として先に述べた調査表の大項目を使用する．各変数の値は大項目中の小項目毎の点数を合計し，小項目数で割ったものである．作成されたモデルは，試作したシステムを用いてステップワイズ法により最適な変数の組を選んでいる．この時に採用されなかった(つまり，削除された)説明変数はモデルに適していないと統計的に判断されたものである．

(1) 設計開始前での予測

調査表Q1を使用してモデルを構築した場合に採用された変数，係数を表3に示す．作成されたモデルのp値は0.00006で$p \ll 0.05$が成立しており，統計的に有意であることが示された．また，デビアンスの値は23で自由度が15なので，モデルの適合度も悪くないと判断できる．

このモデルを使用した8件のプロジェクトに対する設計開始前での混乱予測の結果を表4に示す．詳しくは4.4節で考察するが，PJ33，PJ34の混乱確率は極めて高く，PJ36，PJ37，PJ39のそれは極めて低い．表4で*印を付けたプロジェクトは実際に混乱したことを表す[1]．

表 3. Q1によるモデルでの説明変数

変数名	係数
(定数項)	-4.1736
1. 要求仕様	1.4740
3. 工程計画	3.0223

表 4. Q1による予測の結果

プロジェクト名	混乱確率
PJ33 *	96.59%
PJ34 *	92.90%
PJ35 *	38.90%
PJ36	3.17%
PJ37	3.17%
PJ38	23.35%
PJ39	3.17%
PJ40	23.69%

(2) 設計完了時での予測

調査表Q2を使用してモデルを構築した場合に採用された変数，係数を表5に示す．作成されたモデルのp値は0.000004で$p \ll 0.05$が成立しており，統計的に有意であることが示された．また，デビアンスの値は18で自由度が26なので，モデルの適合度も悪くないと判断できる．

このモデルを使用した設計終了時点での混乱予測の結果を表6に示す．表6からもPJ33，PJ34の2つのプロジェクトの混乱確率が極めて高いことが分かる．一方，PJ36，PJ37，PJ38，PJ39，PJ40の値はいずれも極めて低くなっている．

表 5. Q2によるモデルでの説明変数

変数名	係数
(定数項)	-7.0303
1. 要求仕様	1.7471
2. 見積り	2.3972
4. 工程計画	4.6466

4.4 実験結果の考察

表4と表6の結果より，混乱プロジェクトの予測について次のことが観測された．

[1] 先にも述べたように，PJ33〜PJ40は実際には全て開発が終了しているので，混乱したかどうかの判定もすでにSEPGによって行われている．

表 6. Q2による予測の結果

プロジェクト名	混乱確率
PJ33 *	99.57%
PJ34 *	99.39%
PJ35 *	21.51%
PJ36	0.19%
PJ37	1.07%
PJ38	3.71%
PJ39	0.19%
PJ40	7.33%

(1) Q1を適用した設計開始前の時点での混乱予測においては，混乱確率の高い方から上位3つのプロジェクトは実際に混乱した3つのプロジェクトと一致した．

(2) Q2を適用した設計終了時点での混乱予測においても，混乱確率の高い方から上位3つのプロジェクトは実際に混乱した3つのプロジェクトと一致した．

(3) 表4と表6の結果の比較より，予測を行うタイミングを後ろにずらすことによって，予測の精度が上がった．

ここで，Q1とQ2の予測精度の違いについてより詳細に考察してみる．表4に示される混乱確率によって8個のプロジェクトは次の3つのグループに分けられる．

グループ **a:** PJ33, PJ34

グループ **b:** PJ35, PJ38, PJ40

グループ **c:** PJ36, PJ37, PJ39

グループaのプロジェクトについては，ほぼ間違いなく混乱すると予測できる．同様にグループcのプロジェクトについては混乱しないと予測できる．しかし，グループbのプロジェクトについてはこの時点で混乱を判断するのは難しい．

次に，表6の結果を見るとグループaとcについては表4に比べて予測精度が上がっている．グループbの各プロジェクトについては混乱確率が小さくなった．このことより，PJ38とPJ40はこの時点で混乱しないと予測できる．しかし，プロジェクトPJ35については今回の実験からは予測することは難しい．実際に運用する際に混乱確率の基準値を低く(例えば20%以上のプロジェクトを混乱とみなす)設定することで，混乱プロジェクトの早期発見は可能になる．

この結果より，設計開始前の時点での予測でもほぼ正確な予測結果を得られることが確認された．また，設計完了時点までの情報を加えることによって，確率の予測が補強されることも確認された．

5 予測作業支援システム

これまでに説明してきた混乱プロジェクトの予測を支援するシステムを開発した．本節では，このシステムの概要と評価について説明する．

5.1 支援システムの目的

混乱の予測を行う際に次の作業について手間がかかるため，企業の開発現場への導入を難しくさせる恐れがある．

- 調査表の配布と回収
- 統計パッケージを用いたロジスティックモデルの作成

そのため，本手法の開発現場での運用を支援するシステムを開発する必要が生じた．本システムの目的は，提案する手法そのものをシステムに実装し，プロジェクトの混乱予測をより簡単に行えるようにすることである．具体的には次の(1)–(4)の実現を目指す．

(1) プロジェクトマネージャへの調査表の配付と回収．

(2) SEPGによる過去のデータに基づくロジスティックモデルの作成．

(3) 新規に入力されたプロジェクトに対する混乱予測．

(4) 上述の(1)–(3)の操作をWebブラウザを通じて実行できるようにすること．

5.2 システムの構成

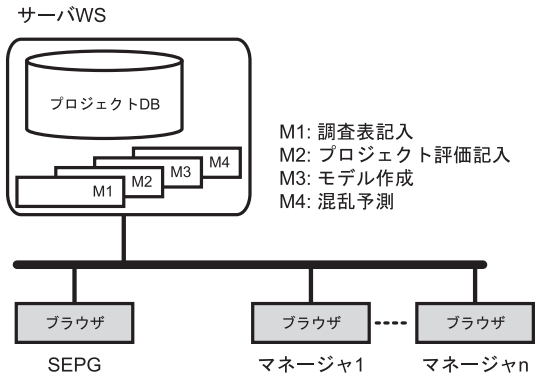

図 4. システム構成図

システムの構成を図4に示す．1つのプロジェクトデータベースを中心に4つのモジュールM1〜M4から構成されている．これらのモジュールの開発と実行はLinuxワー

クステーション (CPU: Alpha21164 500MHz, Memory: 64MB) の上で行う．

なお，データベースおよび，実行のためのプログラムは全てワークステーション上に置く．システムを利用する側 (プロジェクトマネージャや SEPG) は Web ブラウザ以外に特別なシステムを必要としない．

各構成要素についてその概要を説明する．

(1) プロジェクトデータベース：この中には (a) 調査表の項目，(b) 過去に実行されたプロジェクトに対する調査表への回答，(c) そのプロジェクトの終了時点での SEPG の評価結果 (混乱したか，しなかったか)，が蓄積されている．また，(d) 新規のプロジェクトに関する調査表への回答，も蓄積される．

管理を容易にするために専用の Web サーバ上にプロジェクトデータベースを構築する．

(2) 調査表記入 (M1) モジュール：Web ブラウザを通してプロジェクトマネージャに調査表の項目を表示し，記入を促す．記入された調査表への回答は，プロジェクトデータベースに書き込まれる．このモジュールは調査表の項目の追加，削除，変更に対しても必要最低限の労力で対応できるようになっている．

モジュール M1 の実装には Perl を使用した．

(3) プロジェクト評価記入 (M2) モジュール：SEPG が過去のプロジェクトを評価した結果を入力する．

モジュール M2 の実装には Perl を使用した．

(4) ロジスティックモデル作成 (M3) モジュール：プロジェクトデータベースに蓄積された過去のデータを利用して混乱プロジェクトを予測するモデルを作成する．モデルの作成に関しては指定した変数を全て使用する方法 (強制投入法) と，最適な変数の組を見つけてそれを使用する方法 (ステップワイズ法) のいずれか一方を選ぶ．このモジュールも (2) と同様，Web ブラウザを通して全ての操作を行うことができる．

モジュール M3 の実装には Perl と Octave[4] を使用した．

(5) プロジェクトの混乱予測 (M4) モジュール：新規に実行中のプロジェクトのデータを (4) で作成したモデルに適用して，そのプロジェクトが混乱する確率を計算する．これら一連の作業も Web ブラウザを通して行うことができる．

モジュール M4 の実装には Perl と Octave を使用した．

なお，システムの規模は全てのモジュールで 2000 行程度である．

5.1節で述べたシステムの開発目的のうち，(1) の調査表の配付と回収を調査表記入 (M1) モジュールで，(2) のロジスティックモデルの作成をプロジェクト評価記入 (M2) モジュールとロジスティックモデル作成 (M3) モジュールでそれぞれ実現している．(3) の混乱予測は混乱予測 (M4) モジュールで行う．

5.3 操作例

プロジェクトマネージャと SEPG による典型的な利用例を次に示す．まず，プロジェクトマネージャは次の操作を行う．

操作 **A:** 新規プロジェクトについての調査表への記入 (図 5)．

調査表記入モジュール M1 はプロジェクトデータベースから調査表を読み出し，その内容をブラウザに表示する．プロジェクトマネージャはブラウザを介して調査表の回答を記入する．回答はモジュール M1 によってプロジェクトデータベースに保存される．

図 5. 調査表記入

SEPG は次に示す 2 つの操作を行う．なお，プロジェクトマネージャは SEPG 用に準備された機能を操作できない．

操作 **B:** 蓄積されているデータを用いたロジスティックモデルの作成 (図 6)．

SEPG は既に終了したプロジェクトについての評価 (混乱したか，していないか) をブラウザを通して入力する．評価結果はプロジェクトデータベースに保存される．

一方，評価結果が入力されたプロジェクトだけを使用して，新規プロジェクトの混乱予測をするモデルを作成する．SEPG はブラウザを通して，モデルに使用する変数を選択し，入力する．モデル作成モジュール M3 は，プロジェクトデータベースから読み出した過去のプロジェクトのアンケート調査データを使ってロジスティックモデルを作成する．先にも述べたように，モデル作成には，強制投入法とステップワイズ法の 2 つを使える．求まったモデルはプロジェクトデータベースに保存される．

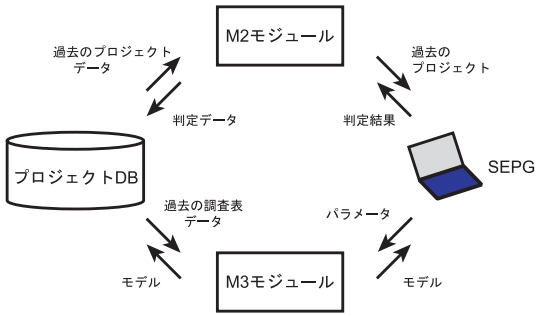

図 6. モデル作成

操作 **C:** アンケート結果に基づいた新規プロジェクトの混乱予測 (図 7).

SEPG はブラウザを通して混乱予測モジュール M4 を起動させる．操作 B において作成されたモデルと新規プロジェクトデータを読み出し，新規プロジェクトに対して混乱予測をする．その結果をブラウザに表示する．

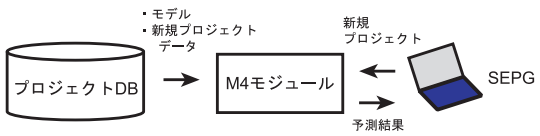

図 7. 混乱予測

実際には幾つかのプロジェクトが並行して行われている．1つのプロジェクトに注目してシステム操作の流れを示すと次のようになる．設計終了までのある時点でプロジェクトマネージャはアンケート調査表の記入を行い (操作 A)，そのアンケート結果に対して SEPG がロジスティックモデルに基づき混乱予測を行う (操作 C)．そしてプロジェクトが終了した後，SEPG はそのプロジェクトの評価を入力する．引き続いて，その時点までに得られているプロジェクトのデータから新たなロジスティックモデルを作成する (操作 B).

5.4 評価

本システムは未だ実際の開発現場に導入するには至っていないため，操作面での評価をすることは難しい．ここでは，次のような視点からの部分的な評価を試みることにする．

動作速度**:** システムは主に Perl と Octave を用いて実装されている．これらは共にインタプリタ型言語であるため，その動作速度は決して速くはない．

本システムで最も時間を要する処理はステップワイズ法によるロジスティック回帰モデル構築である．実験的に 40 プロジェクトと 23 説明変数に対して最適なモデルを作成したところ，17 秒で計算を完了した．この程度であれば十分に実用的であると考える．

拡張性**:** 調査表のフォーマットは容易に変更可能な仕様になっているので，今後もし項目が増減したとしてもその対応に問題はない．

6 まとめ

本報告では，開発の初期時点でプロジェクトマネージャに対するアンケート調査を行ってプロジェクトが混乱するかどうかを予測することを試みた．具体的には，アンケート調査の時期の違いが混乱の予測精度に与える影響を調べた．実験の結果，設計の開始前でも高い精度でプロジェクトの混乱予測が可能であることを確認した．また，提案した手法を支援するシステムの開発について述べた．試作したシステムではアンケートの配布・回収，ロジスティックモデルの構築，プロジェクトの混乱予測を Web ブラウザを通じて行えるようになっている．

今後の課題としては，試作システムを実際の開発現場に導入した評価を行うことがある．今回は既に終了したプロジェクトに対して適用実験を行っているので，アンケートに対する回答は比較的率直なものであった．実際に運用中のプロジェクトに適用した場合，回答者の受けとめ方が今回とは異なるため，正確なデータが得られないことも考えられる．そのため，より正確なアンケート手法の検討も課題の 1 つである．さらに，混乱すると予測した場合にそれを回避する方法 (具体的には注目すべきリスク要素の抽出やそのリスク要素への対処法) についても研究を進めていく必要がある．

付録**:** ロジスティックモデルの統計的評価

3.5節に示されるロジスティックモデルを統計的に評価する基準として，p 値とデビアンスを用いる．

p 値は仮説「$H_0 : b_0 = b_1 = \cdots = 0$」が成りたつ確率を示し，モデルの有意性を判定する基準となる．通常，$p < 0.05$ であれば，仮説 H_0 は棄却されるとする．

一方，デビアンスはモデルの適合度を示す値であり，デビアンスがモデルの自由度に近い値をとると適合度が良いとされる．ただし，デビアンスはかなり極端に値が自由度から外れない限り，適合度が悪いと判断する根拠にはならない．また，一般的には，デビアンスの値よりも p 値によるモデルの有意性を問題にする場合が多い．

参考文献

[1] V. R. Basili, L. C. Briand and W. L. Melo, "A validation of object-oriented metrics as quality indicators," IEEE Trans. on Software Eng., vol. 22, no.10, pp.751–761, 1996.

[2] L. C. Briand, V. R. Basili and C. Hetmanski, "Developing interpretable models with optimized set reduction for identifying high risk software components," IEEE Trans. Software Eng., vol.19, no. 11, pp.1028–1044, 1993.

[3] E. H. Conrow and P. S. Shishido, "Implementing risk management on software intensive projects," IEEE Software, Vol.14, No.3, pp.83–89, 1997.

[4] J.W. Eaton, et al., Octave Home Page, `http://www.che.wisc.edu/octave/`.

[5] R. Fairley and P. Rook, "Risk management for software development," In *Software Engineering*, IEEE CS Press, pp.387–400, 1997.

[6] N. E. Fenton and S. L. Pfleeger, *Software Metrics : A Rigorous & Practical Approach*, PWS Publishing, 1997.

[7] W. S. Humphrey, *Managing the Software Process*, Addison Wesley, Reading, MA, 1989.

[8] W. S. Humphrey, *A Discipline for Software Engineering*, Addison Wesley, Reading, MA, 1995.

[9] D. W. Karolak, *Software Engineering Risk Management*, IEEE CS Press, CA, 1996.

[10] O. Mizuno, T. Kikuno, Y. Takagi and K. Sakamoto, "Characterization of Risky Projects based on Project Managers' Evaluation," Submitted to ICSE2000.

[11] J.D. Musa, A Iannino and K. Okumoto, *Software Reliability: measurement, Prediction, Application*, McGraw-Hill, 1987.

[12] 丹後, 山岡, 高木, ロジスティック回帰分析 –SASを利用した統計解析の実際–, 朝倉書店, 1996.

[13] E. Yourdon, *Death March : The Complete Software Developer's Guide to Surviving 'Mission Impossible' Projects*, Prentice Hall Computer Books, 1997.