

PSP コースデータを対象とした論理エラー欠陥の影響分析と予測モデル

野 中 誠[†] 東 基 衛[†]

ソフトウェア開発の初期段階に、欠陥傾向モジュールを識別して重点的にレビューを行うことは、レビュー効率の改善に有効である。本稿では、論理エラー欠陥に関して、PSP (Personal Software Process) 技法を拡張して測定された欠陥記録データを用いることにより、欠陥傾向モジュールおよび欠陥修正時間の予測が行えることを示す。また、予測モデルの構築にあたって、欠陥修正時間に影響を与えている要因を明らかにするために、41 人の PSP コースデータを分析した。その結果、欠陥型の詳細な分類は欠陥修正時間に影響を与えていないが、課題の差異は影響を与えていることが確認できた。この結果に基づいて、欠陥傾向モジュール予測モデルの構築実験を行った。現時点ではデータ数が少ないために、十分に検証された予測モデルは得られなかったが、予測モデル構築の可能性を示すことができた。

Effect Analysis and Prediction Models for Logical Error Defects using PSP Course Data

MAKOTO NONAKA[†] and MOTOEI AZUMA[†]

Identifying defect-prone modules at earlier stages of the software lifecycle is important for improving review and inspection efficiency. In this paper, a method for generating precise models that identify defect-prone modules and predict defect-fix-time by using the Personal Software Process (PSP) is described. Defects that were injected at the design phase by human logical errors are treated. Prior to create such models, the factors that affected defect-fix-time were analyzed, in terms of the differences of defect types and program exercises in the PSP training course. The result showed that the difference among defect types was not statistically significant; however, there was statistically significant difference among program exercises. An additional experiment for generating a model for identifying defect-prone modules was conducted with the data that had been recorded by using the extended PSP defect recording method. The generated model is yet to be validated, such models can be generated from PSP data.

1. はじめに

ソフトウェア開発において、テスト工程での欠陥摘出には多大なコストがかかる。レビューやインスペクションは有効な欠陥摘出技法であるが、すべての中間成果物を完全にレビューするには多大なコストを要する。したがって、修正コストの高いモジュールを重点的にレビューし、レビュー効率を高める必要がある。そのためには、開発の初期の段階において、欠陥が作り込まれた可能性の高いモジュールまたはクラス(以下、欠陥傾向モジュールと呼ぶ)の識別および欠陥修正コストの予測が必要である。

本研究では、設計仕様書などを対象に測定可能

なメトリクスをもとに、欠陥傾向モジュールを識別する技法を、欠陥傾向モジュール予測と呼ぶ。欠陥傾向モジュール予測に関する研究は、これまでも数多く行われている(例えば[1]~[3])。しかし Fenton が指摘したように[4]、欠陥の属性を分類したうえでの分析は十分に行われていない。これに対して、Humphrey が提唱した PSP (Personal Software Process) 技法[5]を用いることで、欠陥の型、作り込み工程、除去工程および修正時間といった欠陥の属性が開発者によって記録される。これらの欠陥データを用いることで、精密な欠陥傾向モジュールの予測モデル構築が期待できる。

本研究では、論理的なエラー によって作り込

[†]早稲田大学理工学部経営システム工学科
Dept. of Industrial & Management Systems Engineering,
School of Science and Engineering, Waseda University.
E-mail: {nonaka, azuma}@azuma.mgmt.waseda.ac.jp

ISO/IEC 14598-1 の定義に従い、本稿では、メトリクスを「定義された測定方法および測定尺度」としている。

IEEE STD 729-1983 に従い、本稿では、エラーを「ソフトウェアの中に欠陥を作り込む人間の動作」、欠陥を「ソフトウェアにおいてエラーの表面化すること」としている。

まれた欠陥(以下,論理エラー欠陥と呼ぶ)のうち,設計工程で作り込まれ,テスト工程で除去された欠陥を対象としている。これは,論理エラー欠陥はソフトウェア開発工数を増大させる主なリスク要因になると考えられるためである。構文エラー欠陥やコーディング時に作り込まれる欠陥は,コンパイラやCASE(Computer-Aided Software Engineering)ツールなどの適用によってある程度は除去できる。しかし論理エラー欠陥は,仕様書の解釈ミスや問題領域の知識不足に起因するため,構文エラー欠陥とは異なるアプローチが求められる。

本稿では,PSP技法を拡張して測定された欠陥記録データを用いることで,欠陥傾向モジュールの識別および欠陥修正時間の予測が行えることを示す。また,予測モデルの構築に先立って,PSPコースデータを対象として欠陥修正時間に影響を与えている要因について分析した結果と,欠陥傾向モジュールの予測モデル構築の試行結果について述べる。

本稿の構成は以下の通りである。2章では,欠陥傾向モジュール予測に関する研究,PSPの欠陥記録方法およびPSP技法の効果を評価した研究を紹介する。3章では欠陥傾向モジュールおよび欠陥修正時間の予測に利用可能な統計モデルと,その説明変数の候補となる設計メトリクスについて述べる。4章では,レビュー工程の有無,欠陥型の差異およびプログラム課題の差異といった要因が,欠陥修正時間に与える影響を分析した結果を述べる。この結果に基づき,5章では欠陥傾向モジュールの予測モデルの構築実験を行った結果を述べる。最後に6章でまとめを述べる。

2. 関連研究

この節では,欠陥傾向モジュール予測に関する従来研究を紹介する。次に,PSPの欠陥記録方法の概要を述べ,PSPコースデータの分析およびPSP技法の効果に関する従来研究を紹介する。

2.1 欠陥傾向モジュールの予測

ソースコードを測定対象として,SLOC(Source Lines of Code)といった規模メトリクスや,McCabeのサイクロマチック数[6]といった複雑度メトリクスを用いて欠陥傾向モジュールの予測を行う研究が数多く行われている。しかし,

開発プロセスの早い段階で欠陥傾向モジュールを予測しレビュー効率を高めるためには,要求仕様または設計仕様を測定対象とする必要がある。

一方,設計メトリクスを用いた欠陥傾向モジュール予測の研究も行われている。Ohlssonらは設計仕様書から自動的に計測可能な設計メトリクスを用いて,欠陥傾向モジュールの予測が行えることを示している[2]。Basiliらは,ChidamberとKemererが提案したオブジェクト指向設計メトリクス(CKメトリクス)[7]が欠陥傾向クラスの予測に有効であることを示している[1]。神谷らも同様に,アプリケーションフレームワークに基づくソフトウェア開発における欠陥傾向クラスの予測に,CKメトリクスを用いている[3]。しかしFentonが指摘したように[4],欠陥の属性を分類したうえでの欠陥傾向モジュール予測に関する研究は十分に行われていない。

欠陥の属性には,欠陥の型,作り込み工程,除去工程および欠陥修正時間などが考えられる。欠陥傾向モジュールを予測する際には,これらの欠陥属性を測定することが重要である。

2.2 PSPの欠陥記録方法

欠陥データ記録方法のひとつに,PSPの欠陥記録方法が挙げられる。PSPとは,個人のプロセスを改善するための定義されたプロセスおよび帳票[5]であり,PSP0からPSP3までの4つのプロセスレベルに大別される。PSP0では,工程別時間および欠陥データを測定してベースラインデータを得る。PSP1では,PROBE(PROxy-Based Estimation)という規模および工数の見積り手法が導入される。PSP2では,欠陥除去のために設計レビューおよびコードレビューが導入され,プロセス品質メトリクスを用いてその効果を評価する。PSP3では,循環型の開発プロセスが導入される。また,これらのプロセスを習得するための10個のプログラム課題(1Aから10A)からなる演習コースが用意されている。

PSPでは,欠陥データを記録する際に,欠陥型,作り込み工程,除去工程および修正時間といった詳細なデータを記録する。欠陥型は,表1に示した欠陥型分類に基づいて記録する。本研究では,このうちの型番号40から80までを論理エラー欠陥と呼ぶ。欠陥の修正時間は,分単位で開発者自身が記録する。

表 1 PSP で用いる欠陥型[1]

型番号	欠陥型 (論理エラー欠陥)	型番号	欠陥型
40	アサインメント	10	文書
50	インタフェース	20	構文
60	チェックング	30	ビルド、
70	データ		パッケージ
80	機能	90	システム
		100	環境

2.3 PSP データの分析

Hayes らは 298 人のソフトウェア技術者の PSP コースデータの分析を行っている[8]。この報告では、プロセスレベルが上がるにつれて、生産性を損ねることなく、規模および工数見積り誤差が減少し、欠陥密度も低減したと述べられている。国内でも大学教育に PSP を適用した事例およびデータを分析した結果が報告されている（例えば[9]）。また、PSP 技法の効果を分析した研究として、Prechelt らは PSP コースを受講した学生と他の教育コースを受講した学生とを比較し、PSP コースを受講した学生の方が品質の良いソフトウェアを開発したと報告している[10]。実際のソフトウェア開発プロジェクトに PSP 技法を適用し、見積り精度および品質が向上したという報告も数多く存在する（例えば[11]）。

これらの研究の多くは、PSP のプロセス改善効果に着目したものである。これは PSP の主要な目的であるが、PSP によって測定された精密なプロセスデータを用いることで、プロセス改善とは異なる観点からの分析が可能である。

3. 欠陥傾向モジュールの予測モデル

欠陥傾向モジュールを予測するには、説明変数と予測モデルについて検討しなければならない。以下に、それぞれについて述べる。

3.1 説明変数

設計工程で論理エラー欠陥が作り込まれる主な原因として、要求仕様の解釈ミスや問題領域の知識不足といった個人の能力に起因する原因が挙げられる。これらの要因は無視できないため、要求仕様の複雑度および開発者の知識レベルを測定し、それらの差分から論理エラー欠陥の発生を予測することが望ましい。しかし、実際のソフ

トウェア開発プロジェクトでこれらを定量的に測定することは困難である。過去の欠陥傾向モジュール予測に関する従来研究では、説明変数に設計メトリクスを用いたものがほとんどである([1]～[3])。本稿でもこれにならない、モジュールまたはクラスに適用可能な設計メトリクスを説明変数とする。

説明変数の候補として、設計段階で測定可能なメトリクス、PSP で適用可能なメトリクスおよびモジュール等の分類が挙げられる。これらの中から、予測モデルの説明変数として適切なものを統計解析により選択する。

(1) 規模メトリクス

モジュールの規模を測るメトリクスとしては、擬似コード行数が考えられる。オブジェクト指向設計の場合は、クラスの規模を測るメトリクスとして、メソッド数や属性数などが考えられる。

PSP では、規模を見積る際に PROBE と呼ばれる見積り手法を用いる。これは、実績データと比較してモジュール相対規模を決定し、モジュール見積り規模を算出する方法である[5]。モジュール相対規模は、「非常に大きい」から「非常に小さい」までの 5 段階で与えられる。モジュール見積り規模は、ソースコード行数で表される。ここで、モジュール見積り規模はモジュール相対規模に基づいて求められるため、両者の相関は必然的に強くなる。これらを説明変数として用いる場合は、どちらか一方のみを用いなければならない。

(2) 複雑さメトリクス

規模の大きいモジュールは一般に複雑になるため、規模と複雑さを直交した概念として厳密に区別することは困難である。例えば、CK メトリクス[7]はオブジェクト指向ソフトウェアの設計の複雑さを表しているが、規模を表したメトリクスであるとも言える。

構造化設計の場合は、擬似コードに McCabe のサイクロマチック数を適用し、モジュール複雑度を測定できる。このほかに、モジュールの fan-in および fan-out 数が挙げられる。fan-in 数とは、あるモジュールを呼び出す上位モジュールの数である。一方 fan-out 数とは、あるモジュールが呼び出す下位モジュールの数である。オブジェクト指向設計の場合、オブジェクトの状態によって振る舞いが異なるため、クラスの状態数が複雑度を表すと考えられる。

(3) モジュールまたはクラスの分類

PROBE 手法では、モジュールまたはクラスの分類として、論理、入出力、計算、テキスト、データ、セットアップを用いている。オブジェクト指向ソフトウェアの場合は、MVC (Model, View and Controller) などの分類が考えられる。

3.2 欠陥傾向モジュールの予測モデル

欠陥傾向モジュールの予測は、各モジュールに欠陥が作り込まれている確率を予測し、その確率がある閾値を超えたものを欠陥傾向モジュールとして識別する。この確率を予測する方法の一つに、多重ロジスティック回帰分析がある。多重ロジスティック回帰モデルは、Basili らの欠陥傾向クラス予測でも適用されている[1]。

予測モデルの構造は式(1)で表される。

$$P_i = \frac{1}{1 + e^{-(C_0 + C_1 X_{i1} + \dots + C_n X_{in})}} \quad (1)$$

ここで、 P_i はモジュールに欠陥が作り込まれている確率、 X_m は予測モデルの説明変数、 C_n は偏回帰係数である。添え字 i は第 i 番目のモジュールを、 n は第 n 番目の説明変数を表す。説明変数には 3.1 節で述べたメトリクス(1)(2)および分類(3)が用いられる。

3.3 欠陥修正時間の予測モデル

欠陥修正時間の予測は、説明変数が質的データか量的データかによって適用できるモデルが異なる。3.1 節で示した説明変数は、(3)のモジュールまたはクラス分類を除いて、すべて量的データである。したがって、モジュール分類についてデータを層別したうえで、重回帰分析が適用できる。神谷らは、欠陥傾向クラスについて、欠陥修正時間の予測に重回帰モデルを適用している[3]。

予測モデルの構造は式(2)で表される。

$$y_j = a_0 + a_1 x_{j1} + \dots + a_n x_{jn} \quad (2)$$

ここで、 y_j は欠陥修正時間の予測値、 X_{jn} は予測モデルの説明変数、 a_n はデータから得られる偏回帰係数である。添え字の j は第 j 番目の欠陥を、 n は第 n 番目の説明変数を表す。説明変数は、3.1 節で示した(1)および(2)が適用できる。

ここで示した予測モデルを、PSP コースデータを用いて構築するには、欠陥型、プロセスレベル、および課題の差異といった PSP 固有の変動要因

の影響を考慮しなければならない。影響がある場合は、その要因についてデータを層別する必要がある。影響分析の結果は4節で述べる。

3.4 PSP の欠陥記録方法の拡張

欠陥傾向モジュールの予測モデルを構築するには、どの欠陥がどのモジュールに作り込まれたのかが記録されていなければならない。標準の PSP 技法では欠陥が作り込まれたモジュール名が記録されない。したがって、PSP の欠陥記録帳票を拡張して、欠陥が作り込まれたモジュール名を記録しておく必要がある。

4. 欠陥修正時間への影響分析

この節では、PSP コースで得られた欠陥データを用いて、レビュー工程の有無、欠陥型の差異およびプログラム課題の差異といった要因が欠陥修正時間に与える影響を分析した結果を示す。著者らはすでに同様の分析結果を報告しているが[12]、本稿はより多くの技術者の PSP コースデータを用いて分析したものである。

4.1 論理エラー欠陥が工数に与える影響

分析対象とした PSP コースデータは、企業のインハウスコースおよび PSP ネットワークのオンラインゼミ[13]で投稿されたものを対象とした。これらのデータのうち、以下の条件を満たしたデータを分析対象として抽出した。

- (1) 実務経験を持ったソフトウェア技術者のデータである。
- (2) 1A から 10A までのプログラム課題のうち、少なくとも 6A まで作成済である。
- (3) 開発言語に C 言語を用いている。
- (4) 表 1 の欠陥型のうち、欠陥型番号 40 から 80 までの 5 種類を論理エラー欠陥と判断し、これらを分析対象とした。その他の欠陥型は分析対象から除外した。
- (5) 作り込みが設計工程で、除去がテスト工程である欠陥を対象とした。

これらの条件により抽出された欠陥データの主な統計量を表 2 に、ヒストグラムを図 1 に示す。最初に、外れ値の有無について検討しておく必要がある。欠陥修正時間の最大値は 130 分であり、100 分を超える修正時間の欠陥もいくつか観測された。ひとつの論理エラー欠陥の修正に 100

分を超えることは、経験のあるソフトウェア技術者でも発生しうる事象と考えられるので、これらは外れ値ではないと判断する。

表 2 分析対象データの主な統計量

ソフトウェア技術者数（人）	41
プログラム数（個）	157
平均開発時間（分／プログラム）	409.6
欠陥数（個）	273
欠陥修正時間 平均値（分）	19.2
中央値（分）	10
最頻値（分）	5
最小値（分）	1
最大値（分）	130

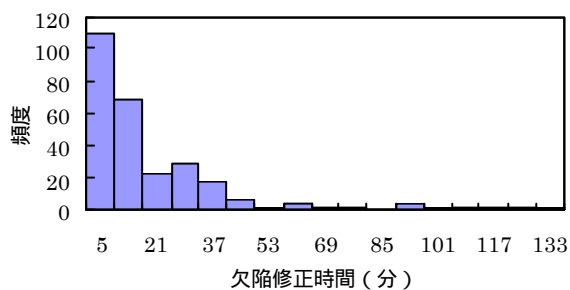


図 1 欠陥修正時間のヒストグラム（実測値）

論理エラー欠陥の修正時間がプログラム課題の開発時間数に占める比率と、各階級に含まれたプログラム数、およびその累積構成比率を表 3 に示す。なお、ひとつのプログラムに複数の論理エラー欠陥が含まれていた場合は、その合計値を用いて修正時間比率を計算している。

表 3 開発時間に対する論理エラー欠陥の修正時間比

開発時間に占める 論理エラー欠陥の 修正時間比率	プログラム数	累積構成比率 （％）
30%以上	2	1.3
20%～30%	10	7.6
10%～20%	38	31.8
10%未満	107	100.0
合計	157	

表 3 より、論理エラー欠陥が作り込まれたプログラムについて、論理エラー欠陥の除去に開発時間の 10%以上を費やしたプログラムが構成比で 30%以上存在していたことがわかる。このように論理エラー欠陥の除去は開発工数の大きな割合を占めることがあり、工数増加のリスク要因となっていると考えられる。

以降では、欠陥修正時間に影響を与えている要因を統計的手法により分析する。その際に、図 1 のように分布に偏りのあるデータでは統計的手法の適用が不適切となる場合が多い。そこで、欠陥修正時間の対数をとった値のヒストグラムを示したものが図 2 である。図 2 は若干右に歪んだヒストグラムではあるものの、統計的手法を用いた分析に適した分布であると判断できる。以降では修正時間の対数値について分析を行う。

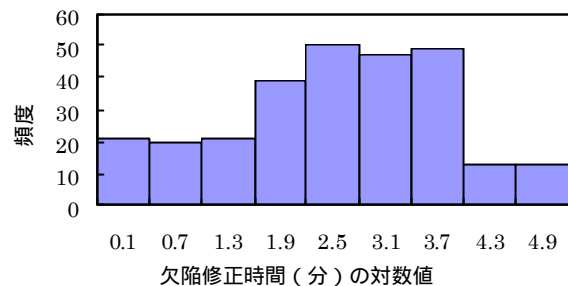


図 2 欠陥修正時間のヒストグラム（対数値）

4.2 レビュー工程の有無と欠陥修正時間

最初に、レビュー工程の有無による論理エラー欠陥の修正時間への影響を分析した。PSP2 ではレビュー工程が新たに導入される。欠陥傾向モジュールを予測するにあたって、レビューを実施せずにテスト工程で検出された欠陥データと、レビュー工程を経てもなおテスト工程まで残存していた欠陥データとではその性質に違いがあると思われる。したがって、レビュー工程の影響を分析し、予測モデル構築に用いることのできるデータを特定しておく必要がある。

図 3 に、レビュー無し（PSP0、PSP1）とレビュー有り（PSP2）の場合の欠陥修正時間の対数値の箱ひげ図を示す。ここで、グラフ中の箱はデータの四分位範囲（第 1 四分位から第 3 四分位までの範囲）を表し、線は最大値および最小値を表す。レビュー無しの平均修正時間は 2.22（9.2 分）、レビュー有りの平均修正時間は 2.73（15.3 分）であり、レビューを経て検出された論理エラー欠陥の方が、平均修正時間が長いと思われる。両者の平均の差の有無についてウェルチの検定を行った結果、有意水準 1% で有意差が認められた。すなわちレビュー工程を経て検出された論理エラー欠陥と、レビュー無しに検出された欠陥とでは、修正時間に差があるといえる。

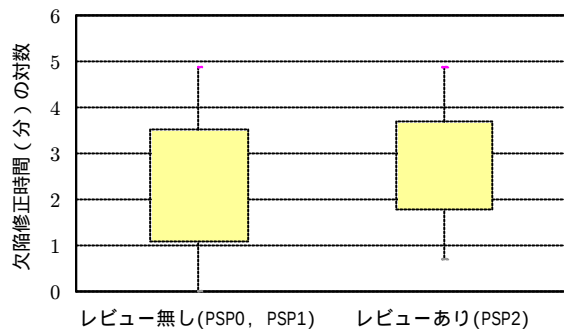


図 3 レビューの有無の違いによる修正時間の分布

この結果から、欠陥傾向モジュール予測モデルの構築には両データを同時に用いるのは不適切といえる。レビュー効率を高めるために欠陥傾向モジュール予測モデルを利用することを想定しているため、レビュー無しのデータを用いて予測モデルを構築する。

4.3 欠陥型と欠陥修正時間

次に、表 1 の欠陥型の違いによる欠陥修正時間の差の有無を、分散分析により分析した。本稿では欠陥データを論理エラー欠陥に絞り込んでいるが、さらに詳細な欠陥型に絞り込む必要の有無を確認するため、この分析を行った。図 4 に欠陥型別の修正時間に関する箱ひげ図を示す。なお、図 4 の横軸の欠陥型番号は表 1 の通りである。

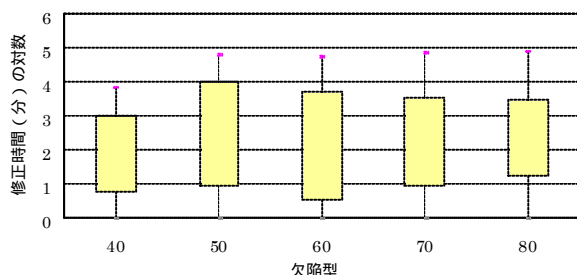


図 4 欠陥型と欠陥修正時間との関係

表 4 に欠陥型別の修正時間の平均と分散を、表 5 に分散分析の結果を示す。欠陥型 40 のアサインメントに関する欠陥の平均修正時間が、他の欠陥型よりも短いように思われる。しかし、欠陥修正時間の平均値の差の有無を確認するために分散分析を行った結果、有意水準 5% で統計的な有意差はみられなかった。

ここで、実際には有意差があるにもかかわらず、その差異を検出できていない可能性がある（第 1 種の過誤）。検定の検出力はサンプル数が少ないと弱まるが、この分析では、各欠陥型について十

分なサンプル数が得られていると思われる。したがって第 1 種の過誤の可能性は低いと思われる。

以上より、今回のデータに関しては、欠陥型の違いは欠陥修正時間に影響するとはいえない。

表 4 欠陥型別の欠陥修正時間の平均と分散

欠陥型	40	50	60	70	80
標本数	32	15	15	23	187
平均	1.88 (6.6)	2.48 (11.9)	2.12 (8.3)	2.24 (9.4)	2.38 (10.8)
分散	1.31 (3.7)	2.31 (10.1)	2.53 (12.6)	1.72 (5.6)	1.32 (3.8)

上段：対数値 下段：分

表 5 分散分析表

要因	平方和	自由度	分散	分散比
欠陥型	7.9	4	1.96	1.34
誤差	392.5	267	1.47	
合計	400.4	271		

4.4 課題と欠陥修正時間

最後に、課題の違いによる欠陥修正時間の差の有無を、分散分析により分析した。図 5 に課題別の修正時間に関する箱ひげ図を示す。表 6 に課題別の修正時間の平均と分散を、表 7 に分散分析の結果を示す。課題 2A および 6A の平均欠陥修正時間が他の課題よりも長いように思われる。分散分析を行った結果、有意水準 1% で有意差が確認できた。すなわち少なくともひとつの課題は、他の課題と欠陥修正時間の対数の平均値が異なる。

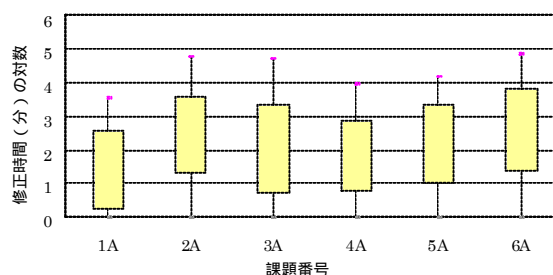


図 5 課題と欠陥修正時間との関係

分散分析はサンプル数が大きくなると検出力が強まるため、実際には有意差が無いにもかかわらず、僅かな差異を有意と判定することがある（第 2 種の過誤）。今回のデータは、各課題ともサンプル数が非常に大きいとはいえず、分散比も十分に大きな値である。したがって、第 2 種の過誤の可能性は低いと思われる。以上より、課題の違いは修正時間に影響を与えているといえる。

表 6 課題別の欠陥修正時間の平均と分散

課題	1A	2A	3A	4A	5A	6A
標本	23	50	47	20	22	55
平均	1.44 (4.2)	2.46 (11.7)	2.07 (7.9)	1.83 (6.2)	2.20 (9.0)	2.60 (13.5)
分散	1.39 (4.0)	1.31 (3.7)	1.77 (5.8)	1.11 (3.0)	1.32 (3.7)	1.51 (4.5)
上段：対数値 下段：分						

表 7 分散分析表

要因	平方和	自由度	分散	分散比
課題	29.0	5	5.80	4.00**
誤差	306.2	211	1.45	
合計	335.2	216		

4.5 考 察

以上の分析結果から、設計工程で作り込まれた論理エラー欠陥について、以下の考察が得られた。

- (1) 3.3節で述べた欠陥修正時間の予測モデルを作成するにあたって、欠陥型 40 から 80 といった詳細な欠陥型について層別する必要はないと思われる。
- (2) 課題の違いが欠陥修正時間に影響を与えていたのは、課題の規模、複雑度および問題領域の違いと思われる。ソフトウェア機能規模を要求仕様から定量的に測定する手法にはファンクションポイント法などが挙げられる。また、機能規模測定の方法が FSM (Functional Size Measurement) [14]に定義されるなど、機能規模の測定は洗練されつつある。一方、要求の複雑度を要求仕様から定量的に測定することは一般に困難である。そこで、課題の複雑度が設計仕様に反映されていると考え、設計メトリクスに基づいて欠陥傾向モジュールの予測を行う。

5. 予測モデルの構築実験

前節の分析結果をうけて、欠陥傾向モジュール予測モデルを構築する試行を行った。現時点で得られたデータ数が少ないため十分に検証されたモデルは得られていないが、予測モデル構築の指向結果を以下に述べる。

5.1 実験の概要

4 節とは別の PSP コースを対象にして、欠陥傾向モジュール予測モデルを構築するために欠陥

データを収集した。3.4節で述べたように、欠陥傾向モジュールの予測を行うには PSP の欠陥記録方法を拡張する必要がある。そこで、通常の PSP 技法に加えて、被験者には欠陥を作り込んだモジュール名を記録してもらった。また、設計仕様としてモジュール構造図の作成を要求した。被験者数は 8 名、いずれも実務経験が 2 年以上のソフトウェア技術者である。開発に用いた言語は C 言語であり、構造化設計技法を用いた。

4.2節の結果に基づいて、レビュー工程無しの論理エラー欠陥データを分析対象とした。また、3.1節で述べた説明変数の候補のうち、PROBE 手法の適用により測定されるものを選択した。PSP コースにおいて PROBE 手法が適用されるのは 4A 以降であり、また、レビュー工程が 7A から導入されるため、4A から 6A までの欠陥データを抽出した。こうして抽出したデータの主な統計量を表 8 に示す。

表 8 分析データの概要

被験者数 (人)	8
総モジュール数 (個)	44
論理エラー欠陥を含むモジュール数 (個)	5
論理エラー欠陥数 (個)	7
欠陥修正時間 平均値 (分)	17.4
中央値 (分)	12
最小値 (分)	5
最大値 (分)	60

5.2 欠陥傾向モジュールの予測モデル

得られた論理エラー欠陥数は 7 件、論理エラー欠陥を含んだモジュールは 5 個のみであり、予測モデルを構築するのに十分なデータ数とはいえない。しかし、ここでは試行として、多重ロジスティック回帰分析により欠陥傾向モジュールの予測モデルを作成した。表 9 に多重ロジスティック回帰分析の結果を示す。これは、3.1節で示した説明変数のうち、モデル式の相関比がもっとも良くなるものを選んで結果である。

表 9 多重ロジスティック回帰分析の結果

説明変数	偏回帰係数	標準偏回帰係数	P 値
X_{11}	0.42	0.38	0.34
X_{22}	-0.98	-0.74	0.36
定数	-0.22		

標準偏回帰係数の値が大きいほど欠陥修正時間に影響を与えている要因である。また、P 値は 0.05 より小さければ重要な要因であると判断する。しかし、どちらの説明変数も統計的に重要な要因とは判断されなかった。今回の試行では少ないデータしか得られなかったため、現時点では十分に意味のあるモデルは得られなかった。しかし、多くの精密な欠陥データを集めることによって、精密な予測モデルが得られると思われる。

5.3 欠陥修正時間の予測モデル

欠陥修正時間の予測モデルは、重回帰分析に十分なデータ数が得られなかったため、モデルが構築できなかった。しかし、欠陥データとモジュール構造図を詳細に調べてみると、同じ課題について同じ設計を行った 2 人の被験者に関して、一方は論理エラー欠陥を作らなかったが、他方は 1 つの欠陥の修正に長時間かかっている事例が発見できた。このことから、欠陥傾向モジュール予測と欠陥修正時間予測は独立で行うのではなく、両者のモデルを複合させて用いた方が良い。

6. ま と め

本稿では、設計工程で作り込まれた論理エラー欠陥を対象として、PSP 技法で測定された欠陥データを用いた欠陥傾向モジュールおよび欠陥修正時間の予測モデルについて述べた。予測モデルの作成に先立って、欠陥修正時間に影響を与えている要因を調べた。その結果、課題の差異は欠陥修正時間に影響を与えていたが、詳細な欠陥型の差異は影響を与えていないことが確認できた。この結果に基づいて、欠陥傾向モジュール予測モデルの構築実験を行った。データ数が少ないため十分な検証は行えなかったが、予測モデル構築の可能性を示すことができた。

今後の課題として、多くのデータを収集し、予測モデルの正確性、完全性および予測精度について評価を行う必要がある。また、実プロジェクトのデータを用いて予測モデルを構築し、その有効性を検証したいと考えている。

謝辞 (株) NEC 情報システムズ佐谷鉄夫氏をはじめ PSP データ提供にご協力いただいた方々、および PSP データ分析の機会を与えていただいた日本電気(株)砂塚利彦氏に謝辞を示す。本研究の一部は早稲田大学特定課題研究助成費

No. 2001A-120 の援助による。

参 考 文 献

- [1] Basili, V.R., Briand, L.C. and Melo, W.L.: A Validation of Object-Oriented Design Metrics as Quality Indicators, *IEEE Trans. Software Eng.*, Vol.22, No.10, pp.751-761 (1996).
- [2] Ohlsson, N. and Alberg, H.: Predicting Error-Prone Modules in Telephone Switches, *IEEE Trans. Software Eng.*, Vol.22, No.12, pp.886-894 (1996).
- [3] 神谷年洋, 楠本真二, 井上克郎, 毛利幸雄: 複雑度メトリクスを用いたエラー予測の一手法 —アプリケーションフレームワークを用いた開発への適用, 情報処理学会論文誌, Vol.42, No.6, pp.1601-1609 (2001).
- [4] Fenton, N.E. and Neil, M.: A Critique of Software Defect Prediction Model, *IEEE Trans. Software Eng.*, Vol.25, No.5, pp. 675-689 (1999).
- [5] Humphrey, W.S.: *A Discipline for Software Engineering*, Addison-Wesley, MA (1995). (ソフトウェア品質経営研究会: パーソナルソフトウェアプロセス技法, 共立出版 (1999)).
- [6] McCabe, T.J.: A Complexity Measure, *IEEE Trans. Software Eng.*, Vol.2, No.4, pp.308-320 (1976).
- [7] Chidamber, S.R. and Kemerer, C.F.: A Metrics Suite for Object-Oriented Design, *IEEE Trans. Software Eng.*, Vol.20, No.6, pp. 476-493 (1994).
- [8] Hayes, W. and Over, J.W.: The Personal Software Process (PSP): An Empirical Study of the Impact of PSP on Individual Engineers, *CMU/SEI-97-TR-001* (1997).
- [9] 海谷治彦, 小島彰, 海尻賢二: 大学におけるプロセス改善教育のあり方について —PSP 法実践の経験をもとに—, ソフトウェアシンポジウム 2001 論文集, pp.137-142, ソフトウェア技術者協会 (2001).
- [10] Prechelt, L. and Unger, B.: An Experiment Measuring the Effects of Personal Software Process (PSP) Training, *IEEE Trans. Software Eng.*, Vol.27, No.5, pp.465-472 (2001).
- [11] Kamater, J. and Hayes, W.: An Experience Report on the Personal Software Process, *IEEE Software*, Vol.17, No.6, pp.85-89 (2000).
- [12] 野中誠, 東基衛: PSP コースデータにおける論理エラー欠陥の分析, 情報処理学会第 63 回全国大会講演論文集 (分冊 1), pp.225-226 (2001).
- [13] PSP ネットワークホームページ (2001) <http://www.azuma.mgmt.waseda.ac.jp/psp/>.
- [14] ISO/IEC 14143-1: 1998, Information Technology —Software Measurement, —Functional Size Measurement, Part1: Definition of Concepts (1998).