NAIST-IS-DD0961015

博士論文

ソフトウェア開発における定量データを活用した開発工数予測とテスト工数低減

田村 晃一

2010年2月4日

奈良先端科学技術大学院大学
情報科学研究科 情報システム学専攻

本論文は奈良先端科学技術大学院大学情報科学研究科に
博士 (工学) 授与の要件として提出した博士論文である。

田村 晃一

審査委員：

松本 健一 教授	（主指導教員）
飯田 元 教授	（副指導教員）
門田 暁人 准教授	（副指導教員）
森崎 修司 助教	（副指導教員）

ソフトウェア開発における定量データを活用した開発工数予測とテスト工数低減*

田村 晃一

内容梗概

ソフトウェア開発の大規模化と短納期化への要求にこたえるには，適切な計画を立案すること及び開発の効率化による工数低減が重要となる．計画立案はプロジェクト初期に予測した開発工数に基づいて行われるため，精度の高い開発工数予測が求められる．開発工数の低減には，大規模プロジェクトにおいて最も多くの工数がかかるテスト工程の工数低減が鍵となる．その手段として，テストよりも早期に実施可能であり，早期の欠陥検出に伴ってテスト工数低減が図れるレビューが注目されている．本論文では，開発工数の予測精度向上及びテスト工数低減を目的とし，(1) 工数予測モデル構築に用いる過去プロジェクトデータに含まれる欠損の処理法の評価，(2) 修正確認テスト工数の低減を効率的に行うコードレビュー手法の提案を行った．本論文の具体的な成果は次のとおりである．

(1) 工数予測モデル構築に適した欠損値処理法の評価

重回帰モデルをはじめとした開発工数予測モデル構築手法の多くは，欠損を含まないデータの使用が前提となっている．しかし，一般に過去プロジェクトデータに欠損が含まれることは避けられない．そこで従来，欠損を補完または除去することで欠損を含まないデータを作成する手法（欠損値処理法）が提案されている．しかし，現実に欠損を含むプロジェクトデータに対してどの手法を用いることで高精度の工数予測モデルが構築できるのかは必ずしも明らかでない．本論文では，複数の企業で収集された 706 件（欠損率 47%）のプロジェクトデータに対

* 奈良先端科学技術大学院大学 情報科学研究科 情報システム学専攻 博士論文, NAIST-IS-DD0961015, 2010 年 2 月 4 日.

し，欠損値補完法（平均値挿入法，k-nn 法，CF 応用法），欠損値除去法（無欠損データ作成法），重回帰モデルに特化した手法（ペアワイズ除去法）を適用し，重回帰モデルの構築を行う．構築したモデルを用いて欠損のない 143 件のプロジェクトに対して工数予測を行った結果，類似性に基づく補完法（k-nn 法，CF 応用法）を用いることで高精度のモデルが構築されることがわかった．さらに，欠損値処理法を用いた工数予測モデルにおけるプロジェクト件数と精度の関係を明らかにするために，同一のプロジェクトデータからプロジェクト件数の異なるプロジェクトデータを複数抽出し，同様の実験を行った．その結果，プロジェクト件数が少ない場合（220 件以下）の場合には，無欠損データ作成法を用いることで高精度のモデルが構築されることがわかった．

(2) 修正確認テスト工数の低減を目的としたコードレビュー手法の提案

ソフトウェア開発におけるテスト工程での欠陥修正には，欠陥が修正されていることを確認するテスト及び修正により新たな欠陥の混入がないことを確認するテストの両方（合わせて修正確認テストと呼ぶ）が必要となるケースが多い．本論文では，欠陥の修正に伴って必要となる修正確認テスト工数の低減を効率的に行うためのコードレビュー手法を提案する．提案手法では，レビューアがテスト工数を推定できる情報を用いることにより，多くの修正確認テスト工数が必要となる箇所を特定し，その特定箇所から優先的にレビューする．商用開発の実務経験者 6 名を含む 18 名の被験者の間で，提案手法と Test Case Based Reading（TCBR），Ad-Hoc Reading（AHR）を比較したところ，TCBR と比較して平均 2.1 倍，AHR と比較して平均 1.9 倍の修正確認テスト件数の削減が確認できた．

キーワード

開発工数予測，欠損値処理法，メトリクス，ステップワイズ重回帰分析，ソフトウェアレビュー，ソフトウェアインスペクション，回帰テスト

Full Use of Software Engineering Data for Development Effort Estimation and Test Effort Reduction*

Koichi Tamura

Abstract

In a software development project, software effort estimation is an important issue for management of schedule and resources. In addition, it is necessary to reduce the development effort under the circumstances of increasing demands for growing in product size and for shortening of project duration. The goal of this dissertation is the improvement of prediction performance and the reduction of development effort, by (1) identifying the effective missing data technique for historical project data sets to build effort estimation models and (2) a source code review technique to efficiently reduce the fix assurance test effort. Achievements of this dissertation are described below:

(1) Identifying the effective missing data technique for effort estimation modeling

Most of the model construction methods require a complete data set that has no missing values for modeling. One of the practical problems in building effort estimation models arises from the fact that many historical project data sets contain substantial amounts of missing data. Thus far, several missing data techniques have been proposed. However, it is unclear which technique is appropriate to missing data in historical data sets used to build effort estimation models. In this dissertation, using project data of 706 cases (containing 47% missing values)

* Doctoral Dissertation, Department of Information Systems, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-DD0961015, February 4, 2010.

collected from several companies, we applied five missing data techniques (mean imputation, k-nn method, applied CF method, row-column deletion, and pairwise deletion) to build regression models. Then, using project data of 143 cases (with no missing values), we evaluated the estimation performance of models after applying each missing data technique. The result showed that the similarity-based imputation methods (k-nn method and applied CF method) showed better performance than other methods. In addition, using multiple data sets with different project cases each extracted from the above data set, we conducted a similar experiment. The result showed that the row-column deletion showed better performance than other methods for data sets not exceeding 220 cases.

(2) A code review technique to reduce fix assurance test effort

In testing phases of software development projects, most of the detected defects require tests to confirm that the correction is done as expected and additional tests to confirm that the correction does not cause new defects. In this dissertation, we propose a code reading technique in order to reduce effort of such testing. The proposed method preferentially detects defects that potentially require larger size of testing by use of information that helps reviewer estimate the size of testing. In an evaluation experiment, the proposed technique reduced the size of testing 2.1 times compared with test case based reading and 1.9 times compared with ad-hoc reading among 18 subjects including 6 commercial software developers.

Keywords:

Software effort estimation, Missing data technique, Metrics, Stepwise regression analysis, Software review, Software inspection, Regression test

関連発表論文

学術論文誌

1. 田村 晃一, 柿元 健, 戸田 航史, 角田 雅照, 門田 暁人, 松本 健一, 大杉 直樹. 工数予測における類似性に基づく欠損値補完法の実験的評価. コンピュータソフトウェア, Vol.26, No.3, pp.44-55, August 2009.（第 2 章に関連する）

2. 田村 晃一, 門田 暁人, 松本 健一. 欠損値処理法を用いた工数予測におけるデータ件数と予測精度の関係. コンピュータソフトウェア（採録決定）.（第 3 章に関連する）

3. 田村 晃一, 亀井 靖高, 上野 秀剛, 森崎 修司, 松本 健一. 修正確認テスト規模の低減を目的としたコードレビュー手法. 情報処理学会論文誌, Vol.50, No.12, pp.3074-3083, December 2009.（第 4 章に関連する）

国際会議

1. Koichi Tamura, Takeshi Kakimoto, Koji Toda, Masateru Tsunoda, Akito Monden, Ken-ichi Matsumoto. Empirical evaluation of missing data techniques for effort estimation. In *Proc. 2nd International Workshop on Software Productivity Analysis and Cost Estimation*, pp.4-9, December 2008.（第 2 章に関連する）

査読付き国内研究集会発表

1. 田村 晃一, 柿元 健, 戸田 航史, 角田 雅照, 門田 暁人, 松本 健一, 大杉 直樹. 工数予測における類似性に基づく欠損値補完の効果. ソフトウェア工学の基礎 XIV, 日本ソフトウェア科学会 FOSE2007, pp.85-93, November 2007.（第 2 章に関連する）

国内研究集会発表

1. 田村 晃一, 柿元 健, 戸田 航史, 角田 雅照, 門田 暁人, 松本 健一. プロジェクト間の類似性に基づくソフトウェアメトリクスの欠損値の補完. ソフトウェア信頼性研究会 第 4 回ワークショップ, pp.17-23, June 2007.（第 2 章に関連する）

2. 田村 晃一, 亀井 靖高, 上野 秀剛, 森崎 修司, 松村 知子, 松本 健一. 見逃し欠陥の回帰テスト件数を考慮したコードレビュー手法. 電子情報通信学会技術研究報告, ソフトウェアサイエンス研究会, Vol.108, No.173, pp.61-66, July 2008.（第 4 章に関連する）

その他の発表論文

学術論文誌

1. 田村 晃一, 池田 隼, 國島 丈生, 横田 一正. パノラマ画像の 4 次元配置による仮想空間の拡張. 日本データベース学会 Letters, Vol.6, No.1, pp.101-104, June 2007.

国内研究集会発表

1. 高井 雄治, 田村 晃一, 森崎 修司, 松本 健一. Web アプリケーションを対象とした故障モード影響解析の試行. 情報処理学会研究報告, ソフトウェア工学研究会, Vol.2009-SE-166, No.10, October 2009.

目 次

第 1 章　はじめに **1**

1. 研究の背景と目的 1
2. 論文構成 4

第 2 章　工数予測における欠損値処理法の実験的評価 **6**

1. はじめに 6
2. ステップワイズ重回帰分析による工数予測 8
3. 欠損値処理法 9
   - 3.1 平均値挿入法 10
   - 3.2 ペアワイズ除去法 10
   - 3.3 類似性に基づく補完法（k-nn 法） 10
   - 3.4 類似性に基づく補完法（CF 応用法） 11
   - 3.5 無欠損データ作成法 13
4. 評価実験 14
   - 4.1 概要 14
   - 4.2 実験用データセット 14
   - 4.3 評価基準 17
   - 4.4 手順 18
5. 結果と考察 18
   - 5.1 実験結果 18
   - 5.2 類似性に基づく補完法と平均値挿入法・ペアワイズ除去法との比較 19
   - 5.3 類似性に基づく補完法と無欠損データ作成法との比較 . . . 20
   - 5.4 類似性に基づく補完法間の比較 21

5.5 ステップワイズ重回帰分析により選択された説明変数の比較 22
6. 関連研究 27
7. まとめ 28

第 3 章 欠損値処理法を用いた工数予測におけるプロジェクト件数と予測精度の関係 30
1. はじめに 30
2. 評価実験 31
2.1 概要 31
2.2 実験用データセット 32
2.3 評価基準 32
2.4 手順 33
3. 結果と考察 33
4. 関連研究 38
5. まとめ 38

第 4 章 修正確認テスト規模の低減を目的としたコードレビュー手法 40
1. はじめに 40
2. 関連研究 41
3. 提案手法 42
3.1 概要 42
3.2 手順 43
3.3 実施例 45
4. 実験 48
4.1 概要 48
4.2 レビュー対象 48
4.3 手順 51
4.4 評価基準 52
4.5 結果 53
5. 考察 56

6. まとめ 58

第 5 章 おわりに **59**

謝辞 **61**

参考文献 **64**

図 目 次

1.1 プロジェクトのメンバ形態の一例（ウォーターフォールモデル）. 2

2.1 プロジェクト間の類似度計算 12
2.2 MAE の箱ひげ図 24
2.3 MRE の箱ひげ図 25
2.4 MER の箱ひげ図 26

3.1 各プロジェクト件数における欠損値処理法を用いた工数予測モデルの精度（SD）. 34
3.2 各プロジェクト件数における欠損値処理法を用いた工数予測モデルの精度（MdMRE） 35

4.1 ソースコードの依存関係と修正確認テストの例 45
4.2 イベント受付システムのデータの流れ 49
4.3 削減件数の累積（平均値）. 54
4.4 優先欠陥検出数の累積（平均値） 55

表 目 次

2.1 実験に用いたデータセットに含まれるメトリクス 16
2.2 欠損値処理法適用後の無欠損プロジェクト件数 16
2.3 ステップワイズ重回帰分析適用時の調整済み R^2 と選択された変数 23
2.4 ステップワイズ重回帰分析に各欠損値処理を適用した際の予測精度 23

3.1 類似性に基づく補完法または無欠損データ作成法適用時の予測精度（SD） 36
3.2 類似性に基づく補完法または無欠損データ作成法適用時の予測精度（MdMRE） 37

4.1 図 4.1 のテスト項目，テスト件数，対象ソースコードユニット . . 46
4.2 実験に使用したテスト計画書の一部 50
4.3 被験者に提示した欠陥型（文献 [10]） 51
4.4 レビューでの検出により削減できる修正確認テスト件数 51
4.5 実験における各レビュー手法の修正確認テスト削減能力と欠陥検出能力 53

第1章　はじめに

1．研究の背景と目的

ソフトウェア開発プロジェクトを成功に導くために，定量データを活用したプロジェクト支援技術の研究が数多く行われている．定量データを活用したプロジェクト支援技術には，例えば，開発工数や信頼性の予測手法，開発プロセスの管理手法などが存在する．

図 1.1 にウォーターフォールモデルを用いたソフトウェア開発プロジェクトのメンバ形態の一例を示す．ソフトウェア開発プロジェクトでは，図 1.1 に示すように，各工程において各成果物の作成に直接関わる部門（以降，ライン部門と呼ぶ），及びそのメンバを技術的に支援する部門（以降，スタッフ部門と呼ぶ）が別々に存在する場合が多い．スタッフ部門は，ソフトウェア開発におけるプロジェクト管理や品質管理などに関する知識，技術，ツールなどを開発部門が利用可能なように体系化する活動を行う．例えば，開発工数や信頼性の予測モデルを構築する活動などである．ライン部門は，これら体系化された技術やツールによる支援を活用しながら，各工程において成果物を作成する．加えて，成果物作成支援のための設計・プログラミング技法，レビュー・テスト手法なども活用する．

まず，スタッフ部門による技術的支援においては，プロジェクトの初期に実施する開発工数の予測が特に重要である．これは，開発工数の予測に基づいて開発計画の立案と開発要員の確保を行うため，不適切な開発工数予測はコスト超過や納期の遅れなどを引き起こし，プロジェクト失敗の主な原因となるからである．

従来，開発工数の予測を行うために，過去のプロジェクトで計測・収集されたメトリクス値を用いる手法が数多く提案されている [3][37][40][46]．ここで，メトリクスとはソフトウェア開発において計測されるデータであり，例えば，開発規

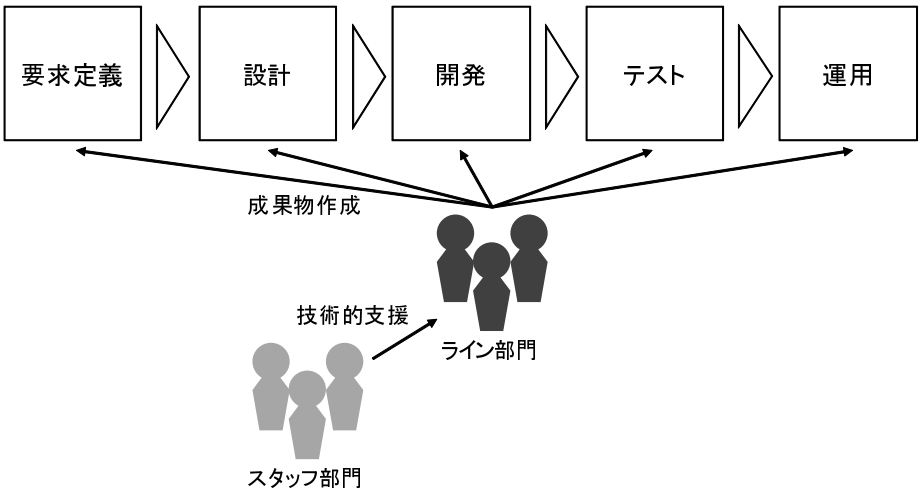

図 1.1 プロジェクトのメンバ形態の一例（ウォーターフォールモデル）

模，開発期間，欠陥数などがある．これまでに提案されている重回帰分析やニューラルネットなどの多変数モデルを用いた開発工数の予測は，予測時点までに計測されているメトリクス値を入力として行われ，数学的に予測結果を得ることができる．

次に，ライン部門による成果物作成活動においては，開発計画どおりに開発を進めるために，効率的な開発を実施することが求められる．ソフトウェアの規模が大きい場合には，総開発工数に占める工程別の工数比率において，テスト工程の比率が最も高くなることが報告されており [14]，テスト工程を効率化することは開発全体を効率化する上で重要である．

従来，テスト工程の効率化にはレビューが有効であることが報告されている [7][21][27][45]．これは，レビューはテストと比較して早い段階での実施が可能であり，欠陥の早期検出に伴う早期修正によりテスト工程にかかる工数の低減につながるためである．例えば，設計工程で作り込まれた欠陥がテストまで見逃された場合，設計書の修正だけでなく，欠陥を含んだ設計書を基に作成されたソースコードに対する修正，及びソースコードが正しく修正されたことを確認するための再テストも必要となる．一方，この設計書に含まれる欠陥が設計書レビューで

検出され，修正された場合，テストで検出する場合と比較してソースコードの修正及び再テストにかかる工数が削減できる．なお，レビューには計画の状態や進捗の妥当性，または標準規格どおりに開発されていることを評価するために実施されるものも存在するが [1]，ここでは成果物や中間成果物を人手により読み合わせることにより欠陥を検出するために実施されるレビューのみを対象としている．

本論文では，開発工数の予測精度を向上させること及びテスト工数を低減させることを目的として，次の問題点にそれぞれ取り組む．

まず，工数予測モデルの構築に用いるソフトウェア開発プロジェクトの実績データ（以降，データセットと呼ぶ）に含まれる欠損値の適切な処理法が必ずしも明らかでないという問題（**問題 1**）である．一般に，収集されたデータセットに欠損値が含まれることは避けられない [19][47]．欠損値が含まれる理由としては，収集メトリクスが異なる複数の組織のデータセットをマージしたこと，時期によって収集メトリクスが異なる場合があること，時間的制約や不注意による記録漏れなどが挙げられる．しかしながら，従来提案されているモデル構築手法の多くは，欠損値を含まないデータセットの使用が前提となっている．そのため，欠損値処理法を用いて欠損値を含むデータセットから欠損値を含まないデータセットを作成する必要がある．この欠損値処理法には，大きく分けて，欠損値を含むメトリクスやプロジェクトを除去する手法（欠損値除去法），及び欠損値に何らかの値（平均値など）を補完する手法（欠損値補完法）の 2 つが存在する．欠損値除去法がその単純さから最も広く利用されているものの，工数予測モデルを構築・利用する者にとっては，多少ノイズが入ってでも情報量を減らさない（欠損値補完法）方が良いのか，情報量を減らしてでもノイズを避ける（欠損値除去法）方が良いのかを知ることは重要である．

次に，修正工数（欠陥の修正をするための工数）及び確認工数（欠陥の修正が期待どおり行われたかを確認する工数，欠陥の修正に伴い新たに別の欠陥が混入されていないかを確認する工数）の低減に寄与しない欠陥ばかりをレビューで検出しても，テスト工数の低減につながらないという問題（**問題 2**）である．レビュー実施によりテスト工数の低減が図れる理由の 1 つは，欠陥を早期に検出することによって，その欠陥がテストまで見逃された場合に必要となる修正工数と

確認工数を削減できることである．しかしながら，例えば成果物に含まれる誤字脱字など，早期検出により低減される修正工数と確認工数が小さい欠陥ばかりがレビューで検出されると，レビュー実施に伴うテスト工数低減の効果が小さくなるおそれがある．従来提案されているレビュー手法は，欠陥の広範囲（網羅的）な検出を主目的としており，検出される欠陥数が多くなることが期待される一方で，欠陥が見逃された場合の修正工数と確認工数の大小については必ずしも考慮されていない．

まず，**問題 1** を解決するために，どの欠損値処理法を用いることで高い精度の工数予測モデルを構築できるか実験的に評価する．本論文では，複数のソフトウェア開発企業で収集されたデータセットに対して，欠損値補完法（平均値挿入法，k-nn 法，CF 応用法），欠損値除去法（無欠損データ作成法），及び重回帰モデルに特化した手法（ペアワイズ除去法）を適用する．そして，欠損値処理法適用後のデータセットを用いて，工数を予測するための重回帰モデルをそれぞれ構築し，その精度を評価する．さらに，予測時に使用可能なデータセットに含まれるプロジェクト件数は組織や開発の種類によって異なる場合があるため，欠損値処理法を適用するデータセットのプロジェクト件数を変化させ，同様の評価を行う．

次に，**問題 2** を解決するために，文献 [48] の調査結果をふまえ，特に欠陥がテスト工程まで見逃された場合に必要となる修正確認テスト規模が大きくなる欠陥（確認工数が大きくなる欠陥）を優先的に検出するコードレビュー手法を提案する．提案手法では，レビューアがテスト工数を想定できる情報（テストケース数など）を活用することで，ソフトウェアの各構成要素（ソースコードにおける関数やメソッドなど）に仮に欠陥が含まれていた場合に必要となる修正確認テスト規模を予測する．続いて，この予測結果から多くの修正確認テスト規模が必要な部分を特定し，その特定した部分から優先的に欠陥の検出を試みる．

2. 論文構成

本論文の主要部分は大きく 3 つの章から構成される．

第 2 章では，**問題 1** を解決するために，欠損値を含むソフトウェア開発企業

の実績データに対して，どの欠損値処理法を用いることで精度の高い工数予測モデルを構築できるのかを実験的に評価する．実験では，International Software Benchmarking Standards Group (ISBSG)[11] が収集した，複数の企業で収集されたデータセット（以降，ISBSG データセットと呼ぶ）から欠損を含む 706 件のプロジェクト (データ欠損率 47%) を選び，5 つの欠損値処理法 (平均値挿入法，k-nn 法，CF 応用法，無欠損データ作成法，ペアワイズ除去法）を適用し，重回帰モデルの構築をそれぞれ行う．そして，ISBSG データセットから選んだ欠損値を含まない別の 143 件のプロジェクトを用いて各モデルの予測精度を評価することで，各欠損値処理法の性能を評価する．

第 3 章では，**問題 1** を解決するために，欠損値処理法を用いた工数予測モデルにおけるプロジェクト件数と予測精度の関係を明らかにする．実験では，第 2 章で用いたデータセットを基にプロジェクト件数の異なるデータセットを複数作成し，各プロジェクト件数での欠損値処理法の性能を第 2 章と同様に評価する．

第 4 章では，**問題 2** を解決するために，修正確認テスト規模を増大させる欠陥を優先的に検出することを目的としたコードレビュー手法を提案する．提案手法により優先的にレビューすべき部分の特定及び修正確認テスト規模の低減につながる欠陥を検出できるかどうかを評価するために，レビューの進め方の戦略である Test Case Based Reading 及び Ad-hoc Reading との比較実験を行う．実験では，商用開発の実務経験者 6 名を含む 18 名の被験者を，提案手法を実施するレビューア，Test Case Based Reading を実施するレビューア，Ad-hoc Reading を実施するレビューア（それぞれ学生 4 名，実務経験者 2 名）に分割し，各手法を実施したレビューアによって検出された修正確認テスト規模の低減につながる欠陥の個数，及び欠陥の検出に伴い低減された修正確認テスト規模を比較する．

そして，本論文の最後の第 5 章で全体のまとめを述べる．

第2章　工数予測における欠損値処理法の実験的評価

1. はじめに

ソフトウェア開発プロジェクトの初期段階において開発工数を予測すること，及び各工程の終了段階で開発工数の再予測を行うことは，プロジェクト完遂に必要な資源の確保や，スケジュール管理を行う上で重要である．そのために，過去のソフトウェア開発プロジェクトで計測された実績データを予測の根拠に用いる定量的予測手法が数多く提案され，用いられてきた [3][37][40][46]．なかでも，プロジェクトのメトリクスを説明変数として用い，目的変数である開発工数との関係を一次式で表現する重回帰モデルを用いた予測は，ツールが普及しており適用が容易なことから，広く実施されてきた [4][44]．

重回帰モデルの構築にあたっては，欠損値を含まないデータセットが必要となるが，一般に，多数の部署・組織から収集されたデータセットには欠損値が含まれる [19][47]．欠損値が生じる原因としては，収集メトリクスが異なる複数の組織のデータセットをマージしたことや，時間的制約や不注意による記録漏れなどが挙げられる．

従来，欠損値を含むデータセットから重回帰モデルを構築するために，欠損値処理法が広く用いられてきた．欠損値処理法には，例えば，欠損値を含むメトリクスやプロジェクトを除去し，欠損値を含まないデータセットを作成すること（欠損値除去法）がある [32][41]．ただし，この場合には，データセットのサイズが小さくなり，予測の根拠となる情報量を減らしてしまうため，モデルを構築したとしても十分な予測精度が得られない可能性がある．他に，欠損値を何らかの値で補完することにより，データセットのサイズを保つこと（欠損値補完法）がある

[4][15][19][26][32][40]．この場合の問題は，補完する値によってはデータセットにノイズが混じることとなり，必ずしも妥当なモデルが得られないことである．さらに，重回帰モデルに特化した欠損値処理法として，ペアワイズ除去法も提案されている [26]．しかし，いずれの手法が工数を予測するための重回帰モデルに適しているのかは，必ずしも明らかでない．

本章では，ISBSG データセット [11] から欠損値を含む 706 件のプロジェクト（データ欠損率 47%）を選び，欠損値補完法（平均値挿入法，k-nn 法，CF 応用法），欠損値除去法（無欠損データ作成法），及び重回帰分析に特化した手法（ペアワイズ除去法）を適用し，重回帰モデルの構築をそれぞれ行う．そして，ISBSG データセットから選んだ欠損値を含まない別の 143 件のプロジェクトを用いて各モデルの予測精度を評価することで，各欠損値処理法の性能を実験的に比較する．ここで，CF 応用法とは，協調フィルタリングを用いた工数予測手法 [46] を応用した手法であり，欠損値補完への適用は本論文が初めてである．CF 応用法と k-nn 法はいずれもプロジェクト間の類似性に基づく補完法といえるが，類似度及び補完値を求めるアルゴリズムが異なっている．各々のアルゴリズムの詳細は 2 章 3 節で述べる．

従来，欠損値処理法の比較は，全く行われていないわけではないが，いずれも不十分である．Jonsson ら [15] や Strike ら [41] は，欠損値を含まないデータセットを人為的（ランダム）に欠損させ，欠損値補完法の比較を行っている．しかし，現実のデータ欠損はランダムというよりはむしろバースト的であるため，現実のデータセットを用いても同等の効果が得られるかは明らかでない．バースト的な欠損となる理由はいくつか存在する．企業ごとに計測対象のメトリクスが異なる場合，それらのデータセットを結合した際にバースト的な欠損が発生する．同一の企業であっても，時期によって計測対象のメトリクスが異なっている場合があり，バースト的な欠損が発生する原因となる．

また，Cartwright ら [5] や Myrtveit ら [32] は，欠損値を含む現実のデータセットに対する欠損値処理を行ってから重回帰モデルを構築し，データセットに対するモデルの適合度を評価している．しかし，適合度のみの評価では，実際の予測を行うにあたって有用であるかは不明である．特に，モデル構築用のデータセッ

トに対して予測モデルが過剰に適合（オーバーフィッティング [9]）した場合，モデル構築用データセット以外のデータセットに対して予測を行った際に高い予測精度が得られない可能性がある．そのため，適合度の評価に加えて，予測性能の評価を行うことが求められる．

さらに，いずれの従来研究においても，欠損値補完法と無欠損データ作成法との比較は行われていない．工数予測モデルを構築・利用する者にとって，多少ノイズが入ってでも情報量を減らさない（欠損値補完法）方が良いのか，情報量を減らしてでもノイズを避ける（欠損値処理法）方が良いのかを知ることは重要である．

以降，2 章 2 節では工数予測モデル構築手法として用いたステップワイズ重回帰分析について述べ，2 章 3 節では従来の欠損値処理法，及び本論文で新たに欠損値補完法として適用する CF 応用法について述べる．2 章 4 節では，各手法の精度を評価するための実験について説明し，2 章 5 節で評価実験の結果と考察について述べる．2 章 6 節で関連研究について述べ，最後に 2 章 7 節で本章の結論について述べる．

2. ステップワイズ重回帰分析による工数予測

重回帰分析は多変量解析の一手法であり，ソフトウェア開発に要する工数を予測するために広く用いられている [29]．本章では，工数予測手法として重回帰分析の一手法であるステップワイズ重回帰分析を用いた [6]．

重回帰分析では，予測対象の変数 Y（目的変数）と目的変数に影響を与える n 個の変数 $X_k(k = 1\text{〜}n)$（説明変数）との関係を表した一次式（重回帰モデル）を作成する．ここで，偏回帰係数を a_k，定数項を C とすると重回帰分析のモデル式は式 (2.1) で定義される．

$$Y = a_1X_1 + a_2X_2 + \cdots + a_nX_n + C \tag{2.1}$$

式 (2.1) の a_k と C は，残差平方和が最小となるように決定される．作成された重回帰式に，予測対象のケース（プロジェクト）の説明変数の値を与えることで，

目的変数の値を予測することが可能となる．

重回帰分析では，予測精度を向上させるために，多数の説明変数候補の中から，予測精度の向上に寄与すると考えられる変数を選択して重回帰式を作成する方法がとられる．ステップワイズ重回帰分析は，ステップワイズ変数選択法により採用する変数を決定し，重回帰分析を行う手法である．ステップワイズ変数選択法は次の手順で行われる [51].

手順 1． 変数を全く含まないモデルを初期モデルとして作成する．

手順 2． 作成されたモデルに対して，各説明変数の係数が 0 でないかの検定を行い，指定した有意水準（本章の評価実験では，偏 F 値の有意水準を $p_{in} = 0.05$, $p_{out} = 0.1$ とした）で棄却されない場合に変数を採択する．ただし，多重共線性を回避するために，採択する変数の分散拡大要因（VIF）が一定値以上の場合，またはその変数を採択することによって，他の変数の VIF が一定値以上となる場合，その変数は採択しない．本章の評価実験では，VIF は 10 とした．

手順 3． 検定により適切な変数が選択されたと判断されるまで手順 2 を繰り返す．

3. 欠損値処理法

欠損値を含むデータセットにステップワイズ重回帰分析を適用するための前処理として，データセットに含まれる欠損値を何らかの値で補完する，もしくは欠損値を含むプロジェクト及びメトリクスを除外するといった欠損値処理が行われる．本章では，欠損値補完法として従来よく使われている平均値挿入法，k-nn 法に加えて，今回新たに欠損値補完に適用する協調フィルタリングを応用した補完法（CF 応用法）を用いて欠損値補完を行う．また，データセットから欠損値を多く含むメトリクスやプロジェクトを除去し，欠損値を含まないデータセットを作成する手法である無欠損データ作成法，及び重回帰分析に特化した手法であるペアワイズ除去法も適用し，相互に比較を行う．各欠損値処理法の詳細を以下に示す．

3.1 平均値挿入法

データセットの当該メトリクスの平均値を補完値とし，欠損値に挿入する．欠損値補完法としては最も単純かつ一般的な方法である [5][32][41].

3.2 ペアワイズ除去法

重回帰分析に特化した手法で，重回帰分析の過程においてメトリクス間の共分散を求める際に，メトリクスのいずれかが欠損しているプロジェクトを除外して共分散を求める手法である [19].

3.3 類似性に基づく補完法（k-nn 法）

類似性に基づく補完法（k-nn 法）は欠損値に対して，類似したプロジェクトのメトリクス値を用いて欠損値を補完する [5]．k-nn 法は 3 つの手順（正規化，類似度計算，補完値計算）から構成される．各手順の詳細とアルゴリズムについては以下で述べる．

手順 1．メトリクス値の正規化 各メトリクスは値域に大きなばらつきがあるため，値域をそろえるための正規化を行い，値域を [0,1] にする．ここで，p_i は i 番目のプロジェクト，m_j は j 番目のメトリクスと定義すると，プロジェクト p_i のメトリクス m_j の値 v_{ij} を正規化した値 v'_{ij} は式 (2.2) で定義される．

$$v'_{ij} = \frac{v_{ij} - \min(P_j)}{\max(P_j) - \min(P_j)} \tag{2.2}$$

ここで，P_j はメトリクス m_j が計測されているプロジェクトの集合，$\max(P_j)$ と $\min(P_j)$ はそれぞれ $\{v_{x,j}|p_x \in P_j\}$ の最大値，最小値を表す．

手順 2．プロジェクト間の類似度計算 メトリクス値を補完するプロジェクトと類似したプロジェクトを見つけるため，プロジェクト間のユークリッド距離を計算し，それを類似度とする．メトリクス値を補完するプロジェクト p_a と

他の各プロジェクト p_i とのユークリッド距離による類似度 $Euclidean(p_a, p_i)$ は式 (2.3) で定義される．

$$Euclidean(p_a, p_i) = \sqrt{\sum_{j \in M_a \cap M_i} \left(v'_{aj} - v'_{ij}\right)^2} \tag{2.3}$$

ここで，M_a と M_i はそれぞれプロジェクト p_a と p_i に関して記録されている（欠損値を含まない）メトリクスの集合を表している．

手順 3．類似度に基づく補完値の算出 補完値の算出には，補完対象のメトリクスが欠損していない，プロジェクト p_a と類似度の高い上位 k 個のプロジェクト（*k-nearestProjects*）を用いる．プロジェクト p_a のメトリクス m_b の値 v_{ab} を補完対象とすると，*k-nearestProjects* のメトリクス m_b の平均値を補完値とする．k は実験的に別途求める必要がある．

3.4 類似性に基づく補完法（CF 応用法）

類似性に基づく補完法（CF 応用法）は協調フィルタリングを用いた工数予測手法 [46] を応用した手法を用いて欠損値を補完する．CF 応用法は k-nn 法と同様に，補完対象のプロジェクトと類似しているプロジェクトのメトリクス値を用いて欠損値を補完するが，類似度計算及び補完値計算のアルゴリズムが異なる．CF 応用法は 3 つの手順（正規化，類似度計算，補完値計算）から構成される．各手順の詳細とアルゴリズムについては以下で述べる．

手順 1．メトリクス値の正規化 2 章 3.3 項の手順 1 と同様の方法で正規化を行う．

手順 2．プロジェクト間の類似度計算 メトリクス値を補完するプロジェクトと類似した他のプロジェクトを見つけるため，プロジェクト間の類似度を算出する．メトリクス値を補完するプロジェクト p_a と他の各プロジェクト p_i との類似度 $sim(p_a, p_i)$ は式 (2.4) で定義される．

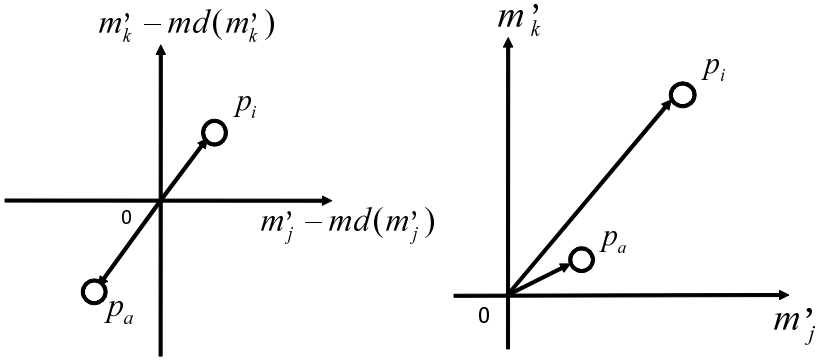

図 2.1 プロジェクト間の類似度計算

$$sim(p_a, p_i) = \frac{\sum_{j \in M_a \cap M_i} \left(v'_{aj} - md(m'_j)\right)\left(v'_{ij} - md(m'_j)\right)}{\sqrt{\sum_{j \in M_a \cap M_i} \left(v'_{aj} - md(m'_j)\right)^2} \sqrt{\sum_{j \in M_a \cap M_i} \left(v'_{ij} - md(m'_j)\right)^2}} \quad (2.4)$$

ここで，m'_j はメトリクス m_j を正規化したものであり，$md(m'_j)$ は j 番目のメトリクスの正規化した値の中央値を表す.

v'_{ij} から $md(m'_j)$ を減算することで，中央値よりも大きなメトリクス値は正の値をとり，小さい値は負の値をとるようになる．類似度の計算例を図 2.1 に示す．図 2.1 の左側の図は正規化したメトリクス値から中央値を減算して類似度を計算した場合，右側の図は正規化したメトリクス値をそのまま使って類似度を計算した場合を示している．この計算により，類似度 $sim(p_a, p_i)$ の値域が [-1,1] をとるようになり，大きくメトリクス値が離れたプロジェクト間の類似度が小さくなる.

手順 3. 類似度に基づく補完値の算出 補完対象となる欠損値について，その補完値の算出に類似したプロジェクトの対応するメトリクスの実測値を用いる．手順 2 の類似度計算では,（ベクトルのなす角を用い，ベクトルの大きさを用いないため）規模が異なるが傾向が似ているプロジェクト同士は類似度

が高いとみなしている．そこで，補完値の算出において，類似度 $sim(p_a, p_i)$ を重みとして，プロジェクト p_a と類似したプロジェクトのメトリクス値 v_{ib} に，プロジェクトの規模を補正する $amp(p_a, p_i)$ を乗じた値で加重平均を行う．プロジェクト p_a のメトリクス m_b の補完値 $\hat{v}_{ab}$ は式 (2.5) で定義される．

$$\hat{v}_{ab} = \frac{\sum_{i \in k-nearestProjects} (v_{ib} \times amp(p_a, p_i) \times sim(p_a, p_i))}{\sum_{i \in k-nearestProjects} sim(p_a, p_i)} \tag{2.5}$$

ここで，*k-nearestProjects* は，メトリクス m_b が欠損しておらず，かつ，プロジェクト p_a と類似度の高い上位 k 個のプロジェクトの集合を表す．k の値は実験的に別途求める必要がある．

また，$amp(p_a, p_i)$ は，プロジェクト p_a のファンクションポイント（FP）を f_a と定義すると，式 (2.6) で定義される．

$$amp(p_a, p_i) = \frac{f_a}{f_i} \tag{2.6}$$

$amp(p_a, p_i)$ は，プロジェクト p_a の規模が p_i の規模のおよそ何倍になっているかを，正規化された FP により求めている．この $amp(p_a, p_i)$ により，多様な規模のプロジェクトを含むデータセットを用いた場合にも補完が可能となる．

3.5 無欠損データ作成法

無欠損データ作成法では，欠損値を含まないデータセットを作成するために，欠損値を含むプロジェクトやメトリクスを除去する．従来研究では欠損値を 1 つでも含むプロジェクトを全て除去する手法（リストワイズ除去法）がよく用いられている [32][41]．しかし，欠損値が多くのプロジェクトにおいて存在していると，リストワイズ除去法を適用するとモデル構築に使用できるプロジェクトがほとんど残らない場合がある．この場合，モデルの構築ができない，または，わずかに残ったプロジェクトにオーバーフィッティングしてしまうおそれがある．そこ

で，無欠損データ作成法では，モデル構築に使用できるプロジェクト件数を確保するために，あらかじめ高い割合で欠損値を含んでいるメトリクスを除去してから，欠損値を 1 つでも含むプロジェクトを除去することで欠損値を含まないデータセットを作成する．ただし，多くのプロジェクトを残そうとすれば，多くのメトリクスの除去が必要となるため，除去するメトリクスの数には注意が必要である．

4. 評価実験

4.1 概要

本章では，各欠損値処理法の有効性をソフトウェア開発企業で収集された実績データを用いて実験的に評価する．実験では，2 章 3 節で述べた各欠損値処理法を適用したデータセットを用いて工数予測モデルを構築し，その予測モデルの精度を比較することにより評価を行う．なお，本章ではもともと欠損値を含むデータセットを用いるために，欠損値の真の値は不明である．そのため，補完値そのものの正確さの評価（真の値との比較）をすることはできない．そこで，欠損値補完を行ったデータセットを用いて工数予測モデルを構築し，その工数予測モデルの予測精度を（欠損値を含まないデータセットを用いて）評価することで，間接的に欠損値補完法の精度を評価する．

4.2 実験用データセット

実験で利用したデータセットは，ISBSG[11] が収集した 20ヶ国のソフトウェア開発プロジェクトの実績データから抽出した．ISBSG データセットは，一般公開されている実績データであり，多くの欠損値を含んでいる（欠損率 58%）．さらに，ISBSG データセットは企業横断データセットとして工数予測手法の評価に広く使用されている [17][30]．これらの特徴により，本章の目的である欠損値処理法の性能を検証するのに適したデータセットであると考えられる．

本実験では，設計終了時を予測時期と想定し，総開発工数の予測を行った．ISBSG データセットから，FP 計測手法が IFPUG，開発形態が新規開発，データの

質（Data Quality Rating）がAもしくはB（A〜Dの全4段階中上位2段階），及び総開発工数（目的変数）が欠損していないプロジェクトを抽出したデータセット（プロジェクト件数849件（欠損率39%））を実験に用いた．ISBSGデータセットは複数の企業から収集したデータセットであるため，データの質に大きなばらつきが存在している．そこで，データの質が欠損値補完及び工数予測の精度に大きな影響を与えることを防ぐため，文献[13][28]にならってデータセット抽出を行った．

実験に使用したデータセットに含まれるメトリクスの詳細を表2.1に示す．データセットに含まれるメトリクスのうち総開発工数を目的変数とし，4種類のメトリクスを説明変数として用いている．本章で欠損値補完に適用する平均値挿入法，k-nn法，CF応用法は比尺度変数の補完が前提となっており，比尺度変数の補完に焦点を当てた評価を行うために，開発プラットフォーム，開発言語，業種などのカテゴリ変数については採用しなかった．開発期間は実測の値が記録されているため，説明変数として選択するか否かは，その因果関係から議論が分かれる．しかし，(1) 開発期間の計画値がISBSGデータセットに含まれていないこと，(2) 一般に，(失敗プロジェクトを除いて）開発期間の計画値と実測値に極端に大きなずれはないと考えられること，(3) 本実験の目的は欠損値処理法の評価であり，そのためには，生産性や工数に影響を与え，欠損値の比較的少ない開発期間を含めた方が望ましいことから，開発期間を計画値と見なし，説明変数に含めた．一方で，説明変数として用いるメトリクスを増やしすぎると，予測用のデータセットとして使用できるプロジェクトの数が大きく減少してしまう．以上の理由により，本実験では表2.1の4種類のメトリクスを説明変数として選択した．

このデータセットを，欠損値を含むプロジェクトのみのデータセット（プロジェクト件数706件（欠損率47%））と欠損値を含まないプロジェクトのみのデータセット（プロジェクト件数143件）の2つのデータセットに分割した．前者を予測モデルを構築するデータセット（以降，フィットデータと呼ぶ）とし，後者を構築されたモデルを用いて実際に予測を行うデータセット（以降，テストデータと呼ぶ）とした．

無欠損データ作成法を適用するにあたっては，欠損率の高いメトリクスを削除

表 2.1 実験に用いたデータセットに含まれるメトリクス

変数の種類	名称	欠損率 (%)
説明変数	FP	0.0
	開発期間（単位:月）	8.8
	システム化計画工数（単位:人時）	77.0
	要件定義・設計工数（単位:人時）	71.2
目的変数	総開発工数（単位:人時）	0.0

表 2.2 欠損値処理法適用後の無欠損プロジェクト件数

欠損値処理法	無欠損プロジェクト件数 (件)
平均値挿入法	706
ペアワイズ除去法	706
類似性に基づく補完法 (k-nn 法)	706
類似性に基づく補完法 (CF 応用法)	706
無欠損データ作成法 (P 削除)	72
無欠損データ作成法 (S 削除)	28
無欠損データ作成法 (P, S 削除)	631

表中の P はシステム化計画工数，S は要件定義・設計工数を表す

した．本実験では，システム化計画工数を削除，要件定義・設計工数を削除，及びシステム化計画工数と要件定義・設計工数を削除する 3 つの方法を用いた．各欠損値処理法を適用した結果，モデル構築に使用できるプロジェクト件数は表 2.2 のようになった．表 2.2 中の P はシステム化計画工数，S は要件定義・設計工数を表している．また，k-nn 法における *k-nearestProjects*，及び CF 応用法における式 (2.5) の *k-nearestProjects* は，それぞれ *k-nearestProjects* の値を 1〜20 まで変化させて欠損値補完を行ったフィットデータを用いて総開発工数を目的変数としたステップワイズ重回帰モデルを構築し，そのモデルの残差平方平均が最小となった 3，及び 8 を用いた．

4.3 評価基準

予測精度の評価基準として，絶対誤差（MAE），実測値に対する誤差（MRE），予測値に対する誤差（MER），及び Pred(25) の 4 種類の評価基準を用いた．M 件のプロジェクトについて，その実測値と予測値をそれぞれ E_i，$\hat{E}_i(i = 1 \sim M)$ と定義すると，それぞれの評価基準は次の式 (2.7)〜(2.10) により計算される．

絶対誤差（MAE)

$$MAE = \left|\hat{E}_i - E_i\right| \tag{2.7}$$

実測値に対する誤差（MRE）

$$MRE = \frac{\left|\hat{E}_i - E_i\right|}{E_i} \tag{2.8}$$

予測値に対する誤差（MER）

$$MER = \frac{\left|\hat{E}_i - E_i\right|}{\hat{E}_i} \tag{2.9}$$

Pred(25)

$$Pred(25) = \frac{\sum_{i=1}^{M} isAccurate\left(\frac{\left|\hat{E}_i - E_i\right|}{E_i}\right)}{M} \tag{2.10}$$

$$isAccurate\left(\frac{\left|\hat{E}_i - E_i\right|}{E_i}\right) = \begin{cases} 1 & \left(\frac{\left|\hat{E}_i - E_i\right|}{E_i}\right) \leq 0.25 \\ 0 & \left(\frac{\left|\hat{E}_i - E_i\right|}{E_i}\right) > 0.25 \end{cases}$$

これらの基準の内，MRE は実測値を分母としているため，過大予測したモデルの誤差を求めるには有用な評価基準である一方で，過小予測したモデルの誤差が小さくなってしまうという問題点がある．これに対し，MER は予測値を分母としているため，MRE とは逆に，過大予測したモデルの誤差が小さくなってしまうという問題点がある．そこで，MRE と MER 両方の評価基準を用いることで，どの欠損値処理を行ったモデルが過大予測や過小予測でもない最も予測精度の高いモデルなのか，正しく判断できる [8][18]．Pred(25) は予測値の MRE が 25%以下となったプロジェクトの割合を示す．Pred(25) は値が大きいほど予測精度が高いことを表し，その他の評価基準は値が小さいほど予測精度が高いことを表す．

4.4 手順

評価実験は次の手順で行った．

手順 1. フィットデータに対し，平均値挿入法，ペアワイズ除去法，k-nn 法，CF 応用法，及び無欠損データ作成法を適用して欠損値処理を行う．

手順 2. 各欠損値処理法を適用した後のフィットデータに対して，ステップワイズ重回帰分析を行い総開発工数を目的変数としたモデルを構築する．

手順 3. 構築したモデルを用いてテストデータの総開発工数を予測し，各評価基準の値を算出する（テストデータの総開発工数は未知数とみなす）．

5. 結果と考察

5.1 実験結果

各欠損値処理法をフィットデータに対して適用し，ステップワイズ重回帰分析で総開発工数の予測を行った．この時の自由度調整済み決定係数（調整済み R^2）と，表 2.1 に示した説明変数のうちステップワイズ変数選択法により選択された変数を表 2.3 に示す．調整済み R^2 はフィットデータに対するモデルの適合度を表す尺度であり，0〜1 の範囲の値を取る．調整済み R^2 が 1 に近いほど適合度が高いことを示す．表 2.2 に示したとおり各モデルで構築に用いるプロジェクト件数が異なるため，厳密な適合度の比較は行えないが，参考として記述している．また，予測した時の MAE，MRE，MER それぞれの中央値（MdMAE，MdMRE，MdMER），及び Pred(25) を表 2.4 に示す．T は開発期間，P はシステム化計画工数，S は要件定義・設計工数を表している．

MAE，MRE，MER における各手法の箱ひげ図を図 2.2〜図 2.4 に示す．グラフの上方は省略している．グラフの縦軸は予測誤差を示し，箱の下端は第 1 四分位，上端は第 3 四分位，箱中の線分は中央値，ひげの下端は第 1 四分位から IQR（Inter-Quartile Range，箱の幅）の 1.5 倍以内に含まれる最小誤差のプロジェク

ト，ひげの上端は第 3 四分位から IQR の 1.5 倍以内に含まれる最大誤差のプロジェクト，◦ は外れ値の可能性があるプロジェクトを表す．

さらに，各欠損値処理法間の評価基準の差が統計的に有意かどうかを検定した．本章の実験では同じフィットデータに対して各手法を適用しているが，MAE, MRE，MER の値は正規分布していないため，ノンパラメトリックな検定である Wilcoxon の符号付順位和検定を行った [51]．また，Pred(25) に対しては，二群の比率の差の検定を行った．それぞれの検定において有意水準は 0.05 以下とした．それぞれの検定結果は以降の項で述べる．

5.2 類似性に基づく補完法と平均値挿入法・ペアワイズ除去法との比較

表 2.4 より，全ての評価基準において類似度に基づいた補完法（k-nn 法，CF 応用法）の方が平均値挿入法，ペアワイズ除去法よりも高い精度が得られたことがわかる．また，図 2.2〜図 2.4 の箱ひげ図より，類似性に基づく補完法の方が平均値挿入法，ペアワイズ除去法よりも箱の位置が下部にあり，より高い精度であることがわかる．また，IQR が小さい，すなわち誤差のばらつきが小さくなっていることもわかる．

平均値挿入法とペアワイズ除去法を比較すると，ペアワイズ除去法の方が平均値挿入法よりも高い精度が得られたことがわかる（表 2.4）．図 2.2 において，平均値挿入法はペアワイズ除去法と比べて IQR が小さく，誤差のばらつきが小さくなっているが，箱の位置が上部に位置しており精度はペアワイズ除去法よりも低いといえる．一方で，平均値挿入法及びペアワイズ除去法は，MdMRE が MdMER と比較して小さく，過小予測する傾向があるといえる．

類似性に基づく補完法と平均値挿入法，ペアワイズ除去法の評価基準の差について 2 章 5.1 項で述べた検定を行った結果，全ての評価基準において有意差がみられた．

以上の結果より，類似性に基づく補完法を用いることで平均値挿入法，ペアワイズ除去法を用いるよりも高い精度の工数予測モデルを構築できるといえる．ま

た，ペアワイズ除去法の方が平均値挿入法よりも高い精度の工数予測モデルを構築できるといえる．

5.3 類似性に基づく補完法と無欠損データ作成法との比較

表 2.4 より，全ての評価基準において類似度に基づいた補完法（k-nn 法，CF 応用法）の方が 3 つの無欠損データ作成法よりも高い精度が得られたことがわかる．図 2.2〜図 2.4 の箱ひげ図より，類似性に基づく補完法（k-nn 法）は全ての評価基準において 3 つの無欠損データ作成法よりも IQR が小さいことがわかる．また，類似性に基づく補完法（CF 応用法）は，MRE において 3 つの無欠損データ作成法よりも IQR が小さく，MAE 及び MER において無欠損データ作成法（P 削除）よりも IQR が大きいものの箱の位置は下部にあり，より高い精度であることがわかる．

類似性に基づく補完法と無欠損データ作成法の評価基準の差について 2 章 5.1 項で述べた検定を行った結果，MER の CF 応用法と無欠損データ作成法（P 削除）間を除いて有意差がみられた．

無欠損データ作成法の中では，MdMRE と Pred(25) に関しては P 削除と S 削除間に明確な差が認められないが，MdMAE，MdMER に関しては P 削除の方が小さな誤差となっていることがわかる．また，図 2.2〜図 2.4 の箱ひげ図においても，無欠損データ作成法の中では P 削除の IQR が概ね最も小さくなっている．以上より，無欠損データ作成法の中では，システム化計画工数を削除した方法（P 削除）が最も高い精度であるといえる．

P 削除では 72 件のプロジェクトが残ったのに対し，S 削除では 28 件となっており，予測モデル構築に使用できるプロジェクトが少なくなりすぎたことが精度を低下させた一因であると考えられる．また，P, S 削除では 631 件のプロジェクトが残ったが，今度は説明変数が少なくなりすぎた（FP と開発期間の 2 つのみとなった）ことが精度を低下させた一因であると考えられる．

以上の結果より，無欠損データ作成法における，プロジェクト件数及びメトリクス数を減らしすぎることの弊害が確認できた．ただし，P 削除，S 削除，P, S 削除のいずれにおいても，平均値挿入法よりも高い精度が得られたことから，欠

損値補完が常に優れているとはいえない．あくまでも適切に欠損値を補完できる場合に限り，欠損値を含むプロジェクトデータを捨てずに利用することが望ましく，その有力な方法が類似性に基づく補完法であることがわかった．

この結果は，プロジェクトデータを収集・利用する者にとって，欠損値を含むデータであっても収集・利用する価値があることを示している．

5.4 類似性に基づく補完法間の比較

類似性に基づく補完法である CF 応用法と k-nn 法を比較すると，表 2.4 より予測精度に大きな差はないが，MdMAE 及び MdMER においては k-nn 法の方が高い精度が得られ，MdMRE 及び Pred(25) においては CF 応用法の方が高い精度が得られたことがわかる．図 2.2〜図 2.4 の箱ひげ図より，全評価基準において k-nn 法の方が IQR が小さい，即ち誤差のばらつきが小さくなっている．

CF 応用法と k-nn 法の評価基準の差について 2 章 5.1 項で述べた検定を行った結果，MAE と MER において有意差がみられた．

以上の結果より，k-nn 法と CF 応用法は大きな差はないものの，どちらかといえば k-nn 法の方が高い精度が得られたといえる．また，MER において CF 応用法は k-nn 法よりも精度が低いことから，CF 応用法は k-nn 法と比べて過小予測する傾向があるといえる．一方，MRE において k-nn 法は CF 応用法よりも精度が低いことから，k-nn 法は CF 応用法と比べて過大予測する傾向があるといえる．以上より，欠損値の補完を行って工数予測を行う場合には，k-nn 法と CF 応用法の両方を実施し，予測値に大きな差がないことを確認することが望ましいといえる．

さらに，予測に用いるメトリクスが異なるため厳密な評価は行えないが，ISBSG データセットを用いた従来研究と予測精度を比較する．本章の実験結果において精度の高かった CF 応用法の MdMRE が 0.274，Pred(25) が 46%であり，文献 [28] において精度の高かったステップワイズ重回帰分析による工数予測では MdMRE が 0.617，Pred(25) が 21%，文献 [13] において精度の高かった Robust Regression による工数予測では MdMRE が 0.683，Pred(25) が 0%，文献 [12] において精度の高かった Ordinary least-squares regression による工数予測は MdMRE が 0.38，

Pred(25) が 21%であった．以上より，類似性に基づく補完法（CF 応用法）を用いることで，従来研究と比較しても高い予測精度が得られたといえる．

5.5 ステップワイズ重回帰分析により選択された説明変数の比較

表 2.3 のように，各手法適用後にステップワイズ重回帰分析を行った時の選択された変数がばらついている理由は，現段階では不明である．しかし，高い精度で工数予測できたモデルが選択した変数は工数予測に有用なメトリクスであると仮定すると，要件定義・設計工数は工数予測に有用なメトリクスであることがわかる．また，FP は欠損値を含まないメトリクスであるにも関わらず，必ずしも選ばれないことがわかる．しかし，平均値挿入法及び無欠損データ作成法（P, S 削除）では FP が最も高い標準化係数となっている．特に，平均値挿入法においては，他の欠損値補完法と比較して FP 以外のメトリクスの欠損値に不適切な値が補完されたため，もともと欠損値を含んでいない FP が工数予測において最も有用なメトリクスとして選択されたと考えられる．さらに，開発期間も必ずしも選ばれないことがわかる．以上より，後工程で計測されたメトリクスほど工数予測を行うための変数としては有用である可能性がある．

表 2.3 ステップワイズ重回帰分析適用時の調整済み R^2 と選択された変数

	調整済み R^2	選択された変数（標準化係数）
平均値挿入法	0.562	FP(0.57),S(0.23),T(0.18),P(0.10)
ペアワイズ除去法	0.957	S(0.74),FP(0.46)
類似性に基づく補完法 (k-nn 法)	0.923	S(0.53),P(0.39),FP(0.12),T(0.04)
類似性に基づく補完法 (CF 応用法)	0.925	S(0.61),P(0.41)
無欠損データ作成法 (P 削除)	0.822	S(0.74),FP(0.32)
無欠損データ作成法 (S 削除)	0.964	P(0.65),T(0.38)
無欠損データ作成法 (P, S 削除)	0.548	FP(0.63),T(0.20)

表中の T は開発期間，P はシステム化計画工数，S は要件定義・設計工数を表す

表 2.4 ステップワイズ重回帰分析に各欠損値処理を適用した際の予測精度

	MdMAE	MdMRE	MdMER	Pred(25)
平均値挿入法	2648	0.818	1.112	20%
ペアワイズ除去法	1036	0.461	0.609	28%
類似性に基づく補完法 (k-nn 法)	760	0.304	0.268	43%
類似性に基づく補完法 (CF 応用法)	829	0.274	0.295	46%
無欠損データ作成法 (P 削除)	1050	0.458	0.416	28%
無欠損データ作成法 (S 削除)	1463	0.479	0.526	27%
無欠損データ作成法 (P, S 削除)	1875	0.555	0.563	18%

表中の P はシステム化計画工数，S は要件定義・設計工数を表す

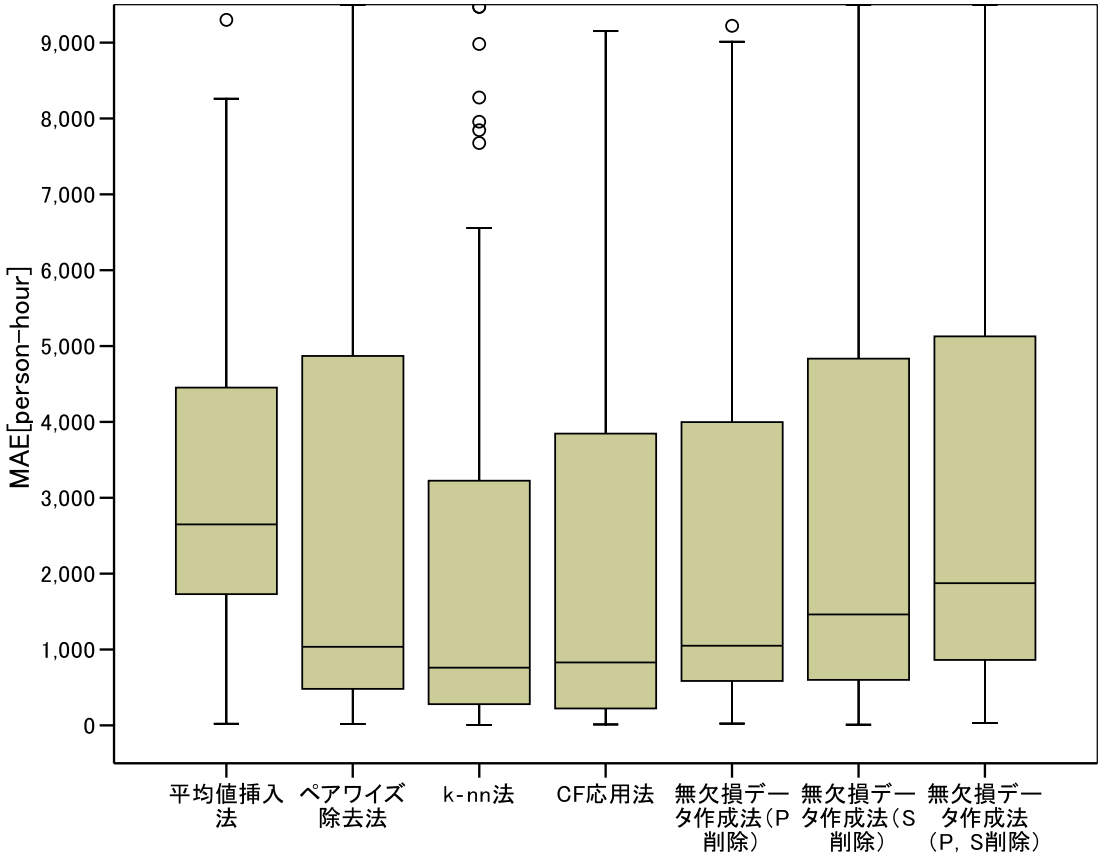

図 2.2 MAE の箱ひげ図

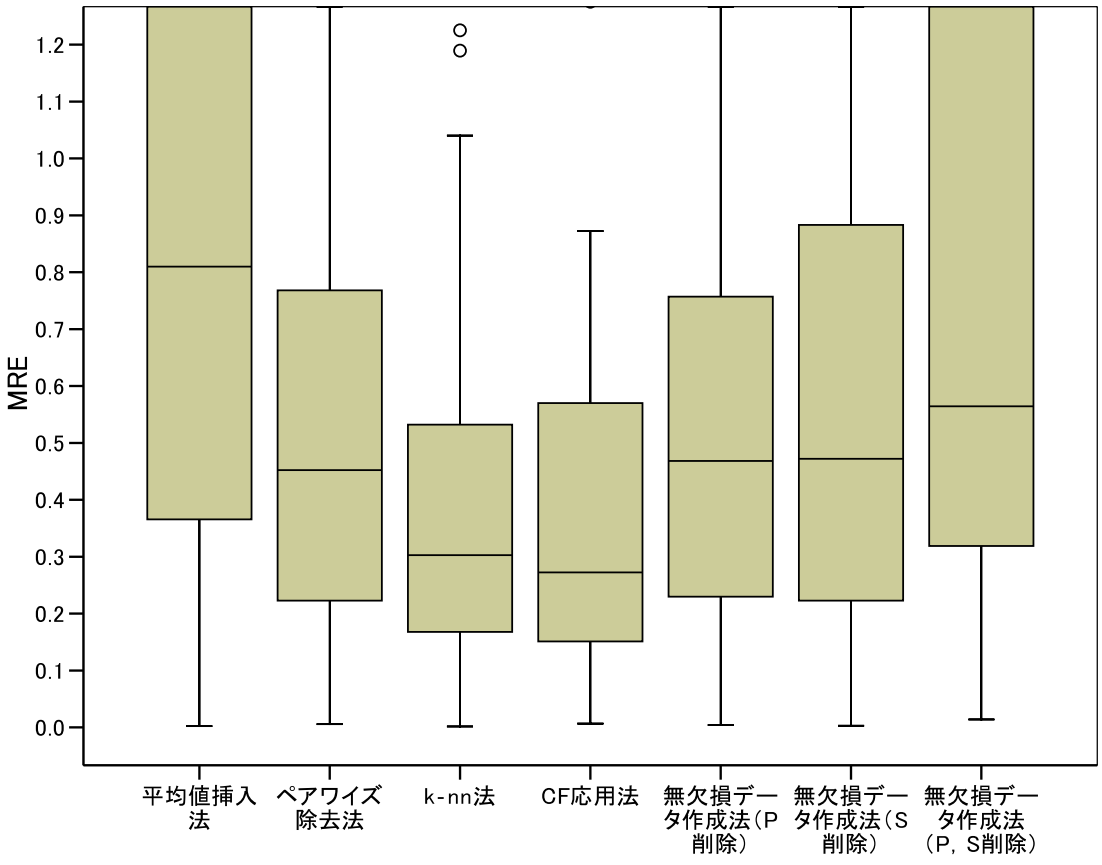

図 2.3 MRE の箱ひげ図

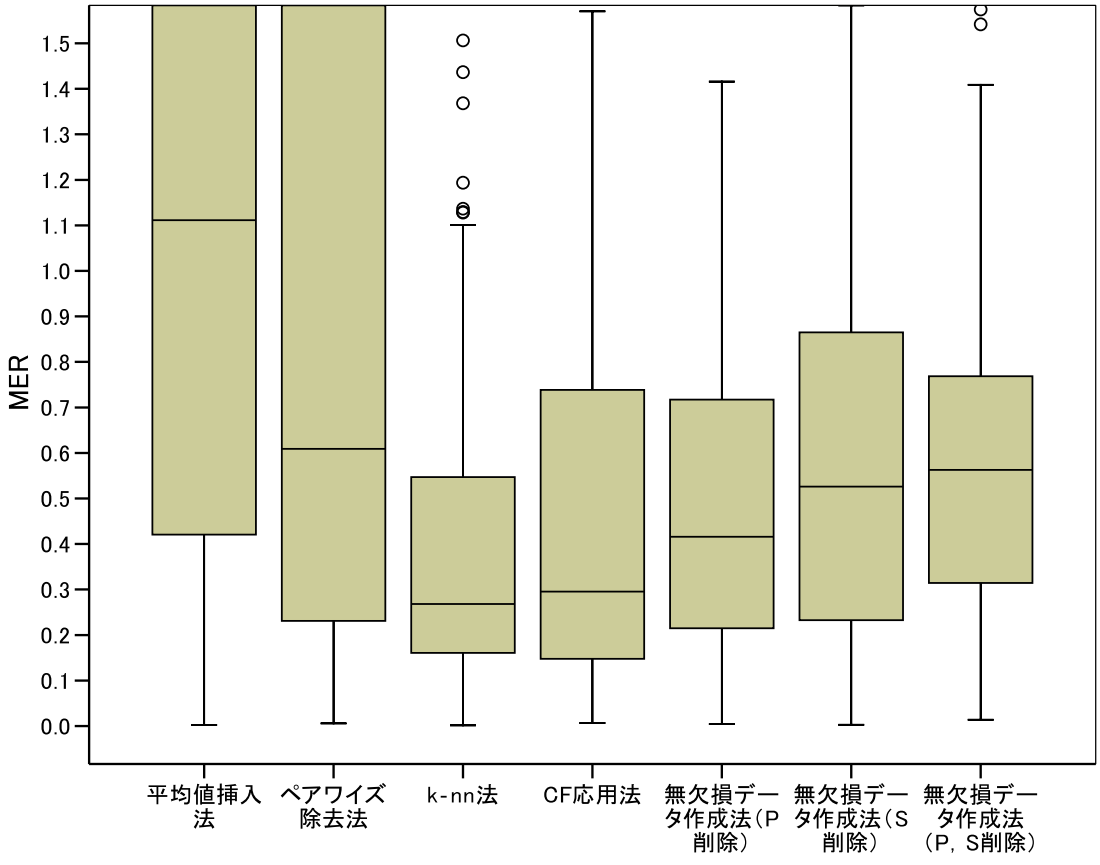

図 2.4 MER の箱ひげ図

6. 関連研究

Jonsson ら [15] は，欠損値を正確に補完することに焦点を当てており，欠損値を含まないデータセットに対して人為的（ランダム）に欠損値を設けてから補完を行い，真の値との比較を行っている．この結果から，k-nn 法適用時に，欠損を含むプロジェクトも類似プロジェクトの候補として補完値の算出を行う方が，欠損を含まないプロジェクトのみを類似プロジェクトの候補とするよりも補完精度が高くなることが示されている．

Sentas ら [36] 及び Strike ら [41] は，欠損値を含まないデータセットに対して人為的（ランダム）に欠損値を設けてから欠損値処理を行い，工数予測を行っている．文献 [36] では，カテゴリ変数に焦点を当てた欠損値処理法の評価を行っており，多項ロジスティック回帰分析を用いることで，リストワイズ除去法，平均値挿入法，EM アルゴリズム，及び Regression Imputation よりも高い精度で工数予測が行えることが示されている．文献 [41] では，リストワイズ除去法，平均値挿入法，及び 8 種類のホットデック法の比較実験を行っている．この結果から，標準化に z-score を用いた k-nn 法により欠損値補完を行ったデータセットを用いることで，高い精度で工数予測が行えることが示されている．しかし，現実のデータセットの欠損値の発生は，ランダムというよりはむしろバースト的であり，欠損値の分布に大きな偏りがある．

また，Cartwright ら [5] 及び Myrtveit ら [32] は，欠損値を含むデータセットに対して欠損値処理を行い，重回帰モデルを構築している．文献 [5] では，k-nn 法と平均値挿入法の比較実験を行っており，k-nn 法を用いて欠損値補完を行うことで，データセットに対して当てはまりの良い（適合度の高い）モデルを作成できることが示されている．文献 [32] では，リストワイズ除去法，平均値挿入法，k-nn 法の一手法である Similar Response Pattern Imputation，完全情報最尤推定法（Full Information Maximum Likelihood）の比較実験を行っている．この結果から，データセット中に存在する欠損が完全にランダムでない場合には完全情報最尤推定法が，欠損が完全にランダムな場合にはリストワイズ除去法が重回帰モデル構築に最も適していることが示されている．しかし，モデル構築に使用したデータセットに対して当てはまりの良いモデルを作成することに焦点が当てられ

ており，実際に作成したモデルが他のデータセットに対してどの程度の精度で予測を行えるのか評価を行っていない．

本章では，意図的に欠損させたのではなく実際に欠損値を多く含んだデータセットに対して欠損値処理を行ったことと，モデル構築後に（欠損値のないデータセットを用いて）工数予測を行い，精度の評価を行った点が従来と異なる．さらに，本章では欠損値補完法と無欠損データ作成法との比較を行った点が従来と異なる．

7. まとめ

本章では，欠損値を含む過去のソフトウェア開発プロジェクトの実績データである ISBSG データセットを用いて開発工数予測モデルを構築する際の，類似性に基づく補完法の効果を実験的に明らかにした．得られた主な結果は次のとおりである．

- 類似性に基づく補完法（CF 応用法，k-nn 法）によって欠損値を補完することで，比較対象とした欠損値処理法である平均値挿入法，ペアワイズ除去法，無欠損データ作成法よりも高い精度の工数予測モデルが構築できた．
- 無欠損データ作成法よりも，類似性に基づく補完法が優れていたことから，プロジェクトデータを用いて工数予測を実施する者にとって，欠損値を含むデータであっても収集・利用する価値があることが示された．
- 平均値挿入法よりも，無欠損データ作成法が優れていたことから，データセットの欠損値が適切に補完された場合に限り，欠損値を含むデータが工数予測モデルを構築するにあたって有用であるといえる．
- CF 応用法はやや過小予測する傾向があり，k-nn 法はやや過大予測する傾向があることから，欠損値の補完を行って工数予測を行う者は，k-nn 法と CF 応用法の両方を実施し，予測値に大きな差がないことを確認することが望ましい．

欠損値をより適切な値で補完できるようにさらなる手法の改善を行うこと，及び欠損値を前提とした工数予測手法（Optimized Set Reduction 法など）との精度比較を行うことが今後の課題である．

第3章　欠損値処理法を用いた工数予測におけるプロジェクト件数と予測精度の関係

1. はじめに

過去のソフトウェア開発プロジェクトの実績データに欠損が含まれることは，一般に避けられない [32][41]．しかしながら，重回帰モデルをはじめとする工数予測モデル構築手法の多くは欠損していないデータセットの使用を前提としているため，データセットに含まれる欠損値の存在は大きな問題となっている．そこで従来，欠損値を含むデータセットから重回帰モデルを構築するために欠損値処理法の適用が提案されている [5][32][41].

第 2 章では，プロジェクト件数 706 件のデータセットに対して類似性に基づく補完法を用いて欠損値処理することで，高い予測精度の重回帰モデルを構築できることを示した．類似性に基づく補完法は補完対象プロジェクトに類似したプロジェクトのメトリクス値を用いて欠損値を補完する手法である．そのため，データセットに含まれるプロジェクトの件数が多いほど，補完対象プロジェクトと類似したプロジェクトが存在する可能性が高くなり，適切な欠損値補完が行えると予測される．しかし，プロジェクト件数の少ないデータセットを用いた場合には類似プロジェクトの数も少なくなるため，適切な欠損値補完ができないおそれがあるが，この点については従来明らかにされていない.

予測モデル構築に用いることのできるデータセットのプロジェクト件数は，組織や開発の種類によって異なる場合がある．欠損値を含むデータセットを用いて工数予測モデルを構築する者にとって，予測時に使用可能なデータセットに対し

て，どの欠損値処理法を用いれば良いのかを知ることは重要である．

本章では，プロジェクト件数が少ない場合においても類似性に基づく補完法により精度の高い工数予測モデルを構築できるのか，そうでない場合にはどの欠損値処理法がより適しているのかを明らかにすることを目的とする．実験では，ISBSG データセットから抽出した第 2 章と同様のデータセット（プロジェクト件数 706 件（欠損率 47%））からプロジェクト件数の異なる複数のデータセットを作成し，2 章と同様に欠損値補完法（平均値挿入法，類似性に基づく補完法（k-nn 法及び CF 応用法)），欠損値除去法（無欠損データ作成法），重回帰分析に特化した手法（ペアワイズ除去法）をそれぞれ適用する（各欠損値処理法については 2 章 3 節参照のこと）．そして，欠損値処理後のデータセットから構築した重回帰モデルを用いて，2 章と同様の欠損値を含まない 143 件のプロジェクトの工数を予測した際の精度を用いて，各欠損値処理法の評価を行う．

以降，3 章 2 節では各欠損値処理法を用いた工数予測モデルの精度を評価するための実験について説明し，3 章 3 節では評価実験の結果と考察について述べる．3 章 4 節で関連研究について述べ，最後に 3 章 5 節で本章のまとめについて述べる．

2. 評価実験

2.1 概要

実験では，ISBSG データセットから抽出したプロジェクト件数の異なる 8 種類のデータセットに対して 5 つの欠損値処理法（平均値挿入法，k-nn 法，CF 応用法，無欠損データ作成法，ペアワイズ除去法）を適用する．そして，欠損値処理後のデータセットから構築した工数予測モデルの精度により比較を行うことで，各欠損値処理法におけるプロジェクト件数と予測精度の関係を明らかにする．本章では，工数予測モデルの構築手法として第 2 章と同様にステップワイズ重回帰分析を用いた（ステップワイズ重回帰分析については 2 章 2 節参照のこと）．

2.2 実験用データセット

実験では，ISBSG データセットから 2 章 4.2 項と同様の方法で抽出した 846 件のデータセットを用いた．このデータセットを，欠損値を含むデータセット（プロジェクト件数 706 件（欠損率 47%））と欠損値を含まないデータセット（プロジェクト件数 143 件）の 2 つに分割した．前者（以降，第 2 章と異なり基礎データと呼ぶ）から，プロジェクト件数の異なる複数のデータセットを作成した．これらプロジェクト件数の異なるデータセットに対して欠損値処理法を適用し，工数予測モデルを構築する．また，後者（以降，第 2 章と同様にテストデータと呼ぶ）を構築されたモデルを用いて実際に予測を行うデータセットとした．

無欠損データ作成法を適用するにあたっては，第 2 章と同様にシステム化計画工数を削除，要件定義・設計工数を削除，及びシステム化計画工数と要件定義・設計工数を削除する 3 つの方法（以降，第 2 章と同様にそれぞれ P 削除，S 削除，P, S 削除と呼ぶ）を用いた．また，類似性に基づく補完法における類似プロジェクト（*k-nearestProjects*）件数は，第 2 章と同様に，k-nn 法は 3，CF 応用法は 8 を用いた．

2.3 評価基準

予測精度の評価基準として，文献 [8] で工数予測モデルの評価基準として有用であると報告されている絶対誤差の標準偏差（SD），及び実測値に対する誤差（MRE）の中央値（MdMRE）を用いた．SD 及び MdMRE は値が小さいほど予測精度が高いことを表す．M 件のプロジェクトがあり，実測値と予測値をそれぞれ E_i，$\hat{E}_i (i = 1 \sim M)$ と定義すると，SD は次式により計算される（MRE については 2 章 4.3 項参照のこと）．

絶対誤差の標準偏差（SD）

$$SD = \sqrt{\frac{\Sigma\left(\hat{E}_i - E_i\right)^2}{M-1}}$$

2.4 手順

評価実験は次の手順で行った.

手順 1. 基礎データのプロジェクト件数の (1/8) 倍を最少のプロジェクト件数とする．このプロジェクト件数を 1, 2, ..., 7 倍にしたデータセット，すなわち，プロジェクト件数 88，176，265，353，441，530，618 件のデータセットをそれぞれ基礎データからランダムに 10 回ずつ抽出する（全 70 データセット）.

手順 2. 抽出した各データセット及び基礎データに対し，平均値挿入法，ペアワイズ除去法，k-nn 法，CF 応用法，及び無欠損データ作成法を適用して欠損値処理を行う.

手順 3. 各欠損値処理後のデータセットに対して，ステップワイズ重回帰分析を行い総開発工数を目的変数としたモデルを構築する.

手順 4. 構築したモデルを用いてテストデータの総開発工数を予測し，各評価基準の値を算出する（テストデータの総開発工数は未知数とみなす）.

3. 結果と考察

それぞれのプロジェクト件数において，各欠損値処理後のデータセットから構築したステップワイズ重回帰モデルの SD 及び MdMRE の値をそれぞれ図 3.1 及び図 3.2 に示す．各グラフの横軸はプロジェクト件数を，縦軸は評価基準の値を示している．なお，図 3.1 及び図 3.2 には，706 件の場合を除き，各プロジェクト件数において 10 個のデータセットから構築した 10 個のモデルが概ねどの程度の予測精度であるかを示すために，SD 及び MdMRE の平均値を示した．また，各プロジェクト件数において，ステップワイズ重回帰分析におけるステップワイズ変数選択法 [6] を適用した結果，説明変数が 1 つも採択されなかった，すなわちモデルの構築に失敗することが 1 回以上あった欠損値処理法については各評価基準の値は示していない.

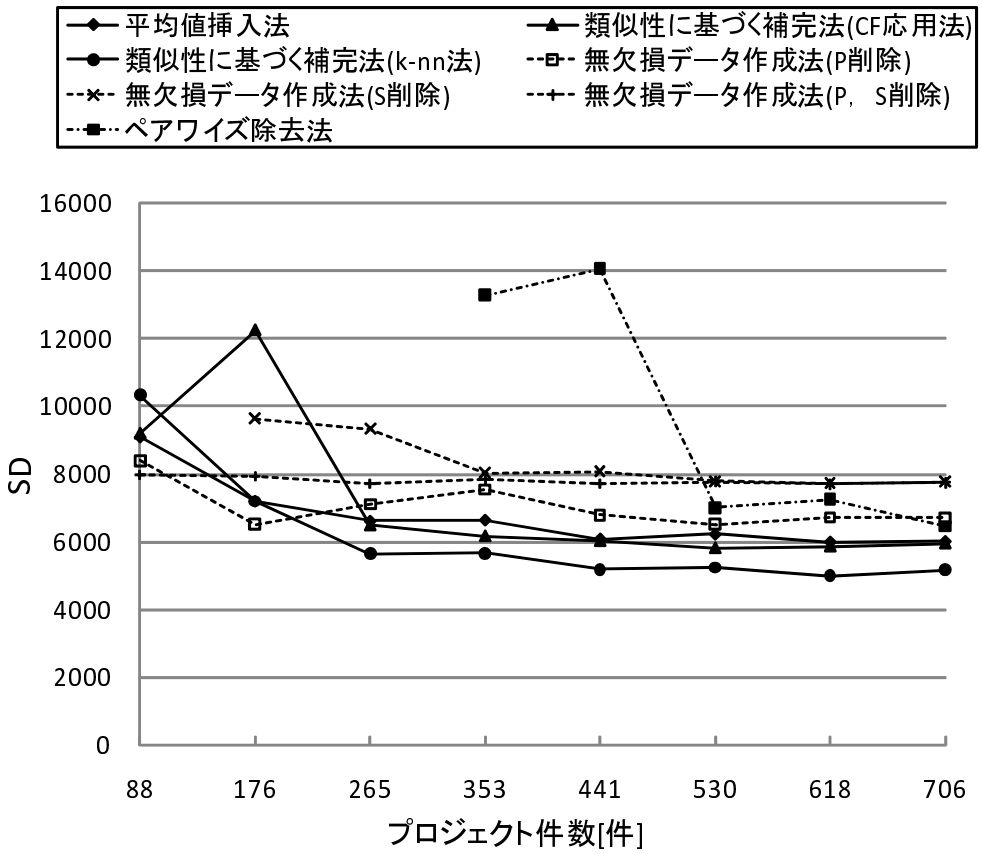

図 3.1 各プロジェクト件数における欠損値処理法を用いた工数予測モデルの精度（SD）

図 3.1 より，プロジェクト件数が 88 件の場合には無欠損データ作成法（P, S 削除），176 件の場合には無欠損データ作成法（P 削除）が最も高い予測精度であった．265 件以上の場合には類似性に基づく補完法が最も高い精度であった．k-nn 法及び CF 応用法間では，88 件の場合には CF 応用法，176 件以上の場合には k-nn 法の方が高い精度であった．ペアワイズ除去法は，441 件以下の場合には最も低い精度であったが，530 件以上の場合には大きく精度が向上した．平均値挿入法は，265 件以上の場合には無欠損データ作成法（P 削除）よりも高い精度となり，88 件の場合には類似性に基づく補完法よりも高い精度であった．

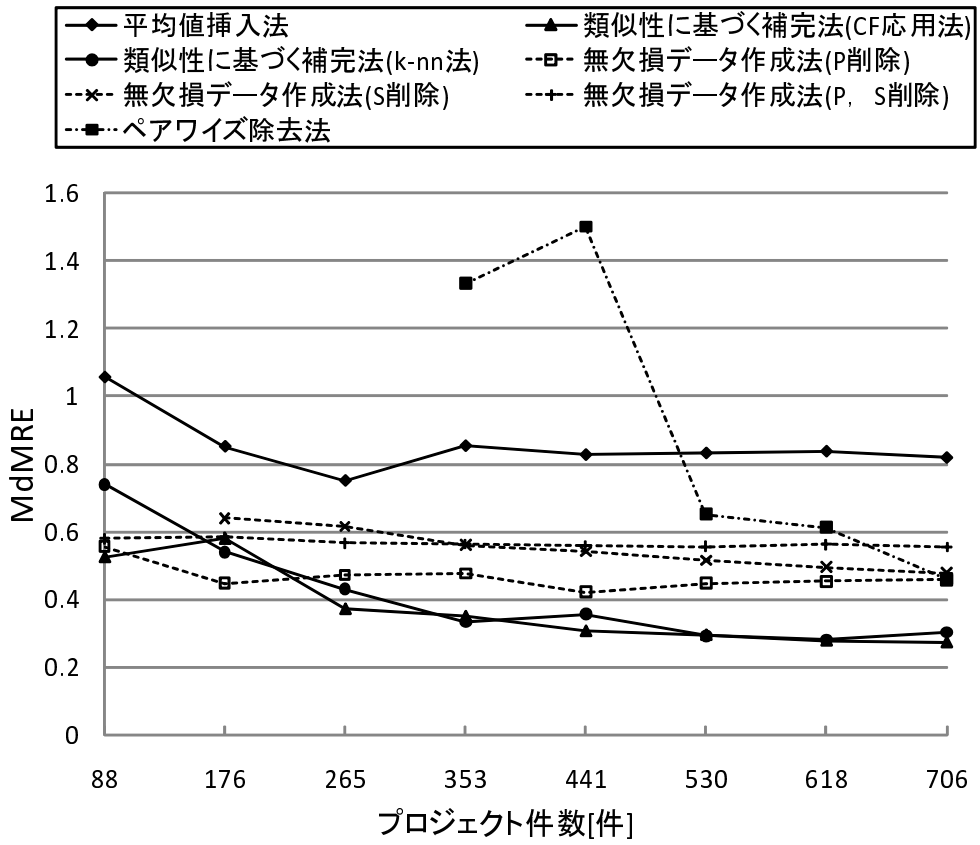

図 3.2 各プロジェクト件数における欠損値処理法を用いた工数予測モデルの精度（MdMRE）

図 3.2 より，プロジェクト件数が 88 件の場合には類似性に基づく補完法（CF 応用法），176 件の場合には無欠損データ作成法（P 削除）が最も高い予測精度であった．265 件以上の場合には類似性に基づく補完法が最も高い精度であった．k-nn 法及び CF 応用法間では，88 件の場合には CF 応用法の方が高い精度であったが，176 件以上の場合には精度に大きな差はないといえる．441 件以下の場合にはペアワイズ除去法は最も低い精度であった．平均値挿入法は，プロジェクト件数にかかわらず類似性に基づく補完法及び無欠損データ作成法よりも低い精度であった．

表 3.1 類似性に基づく補完法または無欠損データ作成法適用時の予測精度（SD）

統計量	欠損値処理法	SD			
		220 件	225 件	230 件	235 件
75 パーセンタイル	類似性に基づく補完法 (k-nn 法)	8454	7549	7530	7423
	無欠損データ作成法 (P 削除)	7604	7605	7866	7907
平均値	類似性に基づく補完法 (k-nn 法)	10294	7114	7120	7052
	無欠損データ作成法 (P 削除)	7265	7192	7255	7256
中央値	類似性に基づく補完法 (k-nn 法)	7433	5992	5931	5902
	無欠損データ作成法 (P 削除)	7414	7409	7472	7457
25 パーセンタイル	類似性に基づく補完法 (k-nn 法)	5608	5042	5049	4999
	無欠損データ作成法 (P 削除)	6683	6683	6692	6679

ここで，プロジェクト件数が何件以下であれば無欠損データ作成法が類似性に基づく補完法よりも高い精度となるのかを特定するために追加実験を行った．この追加実験では，まず，プロジェクト件数を 180〜260 件の間で 5 件ずつ変化させたデータセットをそれぞれ基礎データからランダムに 10 回ずつ抽出する．次に，抽出した各データセットに対し，類似性に基づく補完法（k-nn 法）または無欠損データ作成法（P 削除）をそれぞれ適用する．以降は，3 章 2.4 項の手順 3 及び手順 4 と同様の手順で実験を行う．

表 3.1 及び表 3.2 に，類似性に基づく補完法（k-nn 法）と無欠損データ作成法（P 削除）の精度が同程度となったプロジェクト件数 220〜235 件の時の結果を示す．この実験結果から，SD 及び MdMRE において，プロジェクト件数が 220 件以下の場合には無欠損データ作成法（P 削除）が，235 件以上の場合には類似性に基づく補完法（k-nn 法）がもう一方よりも高い精度となることがわかった．また，表 3.1 及び表 3.2 の 75 パーセンタイル及び 25 パーセンタイルの値から，無欠損データ作成法の方が類似性に基づく補完法よりも精度のばらつきが小さいことがわかった．類似性に基づく補完法において，プロジェクト件数が 220 件以下の場合には他手法よりも精度が低くなったのは，補完対象プロジェクトによっては，類似プロジェクトが十分に存在していなかったことが原因として考えられる．

表 3.2　類似性に基づく補完法または無欠損データ作成法適用時の予測精度（MdMRE）

統計量	欠損値処理法	MdMRE			
		220 件	225 件	230 件	235 件
75 パーセンタイル	類似性に基づく補完法 (k-nn 法)	0.655	0.572	0.566	0.531
	無欠損データ作成法 (P 削除)	0.576	0.570	0.566	0.566
平均値	類似性に基づく補完法 (k-nn 法)	0.575	0.498	0.500	0.486
	無欠損データ作成法 (P 削除)	0.509	0.493	0.499	0.500
中央値	類似性に基づく補完法 (k-nn 法)	0.536	0.461	0.474	0.470
	無欠損データ作成法 (P 削除)	0.541	0.519	0.529	0.528
25 パーセンタイル	類似性に基づく補完法 (k-nn 法)	0.396	0.377	0.375	0.373
	無欠損データ作成法 (P 削除)	0.413	0.413	0.433	0.427

以上の実験結果から，プロジェクト件数が異なれば，高い精度の工数予測モデルを構築できる欠損値処理法も異なることがわかった．また，類似性に基づく補完法（k-nn 法）において，プロジェクト件数が 88 件の場合と 706 件の場合との精度差が，SD では 5167，MdMRE では 0.44 となった一方で，無欠損データ作成法（P 削除）において，プロジェクト件数が 88 件の場合と 706 件の場合との精度差が，SD では 1687，MdMRE では 0.10 となった．このことから，類似性に基づく補完法は，無欠損データ作成法と比較して，プロジェクト件数の減少に伴う精度の低下が著しいといえる．

使用するデータセットまたはメトリクスが変わると，欠損値処理の結果や構築されるモデルも異なるため，以上の実験結果を一般化するには注意が必要である．しかしながら，同程度の欠損率のデータセットを用いて工数予測モデルを構築する者にとって，本実験結果は 1 つの目安となるといえる．例えば，使用可能なデータセットのプロジェクト件数が少ない場合には，無欠損データ作成法の適用も検討する価値があるといえる．一方で，無欠損データ作成法（S 削除）において，プロジェクト件数が 88 件の時には欠損値を含まないプロジェクトの件数が少なくなりすぎたため，モデル構築ができない場合があった（図 3.1 及び図 3.2）．この

ような場合には，SD においては平均値挿入法よりもやや精度が低くなる場合もあったが，MdMRE においては平均値挿入法よりも大幅に精度の高い類似性に基づく補完法（CF 応用法または k-nn 法）を用いることが望ましいといえる．

4. 関連研究

Jonsson ら [15]，Song ら [39]，及び Strike ら [41] は，欠損値を含まないデータセットを人為的（ランダム) に欠損させ，欠損値処理法間の比較を行っている．しかし，現実のデータ欠損はランダムというよりはむしろバースト的であるため，現実のデータセットを用いても同等の効果が得られるかは明らかでない（バースト的な欠損となる理由については 2 章 1 節参照のこと）．

また，Cartwright ら [5] や Myrtveit ら [32] は，欠損値を含む現実のデータセットに対する欠損値処理を行ってから重回帰モデルを構築し，データセットに対するモデルの適合度を評価している．しかし，適合度のみの評価では，実際の予測を行うにあたって有用であるかは不明である．さらに，上記のいずれの文献においても欠損値処理法を用いた工数予測モデルのプロジェクト件数と予測精度の関係については調査されていない．

5. まとめ

本章では，ISBSG データセットから抽出した欠損値を含むプロジェクト件数の異なる複数のデータセットを用いて，いずれの欠損値処理法を適用すれば高い工数予測モデルを構築できるのかについて実験的に評価した．得られた主な結果及び知見は，次のとおりである．

- プロジェクト件数が異なれば，高い精度の工数予測モデルを構築できる欠損値処理法も異なる（プロジェクト件数が少ない場合（220 件以下）にはシステム化計画工数を削除した無欠損データ作成法が，プロジェクト件数が多い場合（235 件以上）には類似性に基づく補完法が，比較対象としたその他の手法よりも高い精度の工数予測モデルを構築できた）．

- 類似性に基づく補完法は，無欠損データ作成法と比較して，プロジェクト件数の減少に伴う精度の低下が著しい．

実験結果より，欠損値処理を効果的に行うために次の方針が考えられる．

- プロジェクト件数が少ない場合には無欠損データ作成法の適用を，多い場合には類似性に基づく補完法の適用を検討する．
- プロジェクト件数が非常に少なく，無欠損データ作成法を適用するとモデルを構築できない場合には，類似性に基づく補完法を用いる．

他のプロジェクトデータを用いて同様の評価実験を行い，上記の結果に一般性を持たせることが今後の課題である．

第4章　修正確認テスト規模の低減を目的としたコードレビュー手法

1. はじめに

ソフトウェア開発における成果物もしくは中間成果物に対して作成者またはその他の開発者がその内容を確認することを目的として，レビューが実施されている．特に，欠陥を検出及び除去することを目的として実施されるレビューは，テストと比較して早い段階での実施が可能であるため開発効率の向上につながることが報告されており [7][21][27][45]，これまでにレビュー手法やその効果についての様々な研究が行われてきた [20][23].

レビューによる開発効率向上の理由の 1 つは，欠陥を早期に検出することによる修正工数（成果物の欠陥を修正する工数）及び確認工数（欠陥修正が期待どおり行われたかを確認する工数，欠陥修正に伴い新たに別の欠陥が混入されていないかを確認する工数）などの工数を低減できることである．これは，テスト実施前にレビューを実施した場合の修正工数と確認工数の合計が，レビューを実施しなかった場合よりも小さくなることを前提としている．そのため，例えばエラーメッセージの誤字脱字といった，早期検出により低減される修正工数と確認工数が小さい欠陥ばかりがレビューで検出されると，レビューによる効率化の効果が小さくなるおそれがある．

しかしながら，従来提案されているレビュー手法は欠陥の広範囲な検出に寄与することを主目的としており，検出される欠陥数が多くなることが期待される一方で，必ずしも欠陥が見逃された場合の修正工数と確認工数の大小について考慮

されていない．そのため，レビューによる修正工数と確認工数の低減の効果が小さくなるおそれがある．例えば，Checklist-Based Reading[7] では，見つけるべき欠陥を過去の事例や経験に基づいて事前にチェックリスト化し，それを確認しながらレビューを行うが，チェックリストの作成時やレビュー時に考慮すべき点として，欠陥の修正工数と確認工数への明示的な言及はない．

本章では，文献 [48] の調査結果をふまえ，特にレビュー時に見逃した場合に必要となる修正確認テスト規模に着目し，修正確認テスト規模が大きくなると推定される欠陥（確認工数が大きくなる欠陥）を優先的に検出するコードレビュー手法を提案する．実験では，ソースコード，外部仕様書，内部設計書に加えて，テスト計画書を用いることによりレビュー時に修正確認テスト規模をレビューアが実際に見積り，優先的にレビューすべき部分の特定及び欠陥の検出ができるか否か確認し，提案手法により潜在的に軽減された修正確認テストの規模を調べる．比較対象の手法として，テストに関するドキュメントとあらかじめ用意した欠陥の種類（以降，欠陥型と呼ぶ）の一覧を用いてレビューする Test Case Based Reading（TCBR），及び特定のレビュー手法を用いることなくレビューアの知識や経験に基づいてレビューを実施する Ad-Hoc Reading（AHR）を用いる．

以降，4 章 2 節では関連研究について述べる．4 章 3 節では修正確認テストの低減を目的としたコードレビュー手法を提案する．4 章 4 節では実験概要，レビュー対象，実験手順，評価基準，実験結果について説明し，4 章 5 節で考察する．4 章 6 節でまとめと今後の課題について述べる．

2. 関連研究

レビューにおける欠陥検出を効率的，効果的にするために様々な手法が提案されている．代表的なレビュー手法として Checklist-Based Reading（CBR）[7][31]，Scenario-Based Reading（SBR）[24] がある．SBR には，Perspective-Based Reading（PBR）[22][38]，Defect-Based Reading（DBR）[33][34]，Usage-Based Reading（UBR）[43] などの手法が含まれる．

CBR は過去の経験などに基づいて，見つけるべき欠陥をチェックリスト化し，

それを用いてレビューを行う手法である．PBR は対象システムの利用者，設計者，テスト担当者などの立場別に着目すべき観点を明示し，多面的に欠陥を検出する手法である．それぞれのレビューアが異なる観点を持つことで，対象システムに対して網羅的なレビューができる．DBR は定義された欠陥の種類ごとにレビューを行うことで効率的に欠陥を検出する手法である．異なる種類の欠陥を複数のメンバが個別にレビューすることで重複を減らし，PBR と同様に網羅的に欠陥を検出することができる．CBR，PBR，及び DBR はより多くの欠陥を検出することに焦点を当てているが，欠陥の優先度及び修正確認テスト規模に関して明示的には考慮していない．

UBR はユースケースを用いることで，システムの利用者に大きな影響を与える欠陥を優先的に検出する手法である．提案手法では UBR と同様に欠陥に優先度を割り当てレビューを行うが，UBR ではユースケースに記述されていない動作についてはレビュー対象とならない場合がある．また，UBR はユースケースの優先度を考慮しているが，修正確認テスト規模とユースケースの優先度は必ずしも一致していない．

テストに関するドキュメントを用いたレビュー手法として TCBR が提案されている [49]．TCBR は，開発作業前にあらかじめブラックボックステスト用のテストケースを作成しておき，各テストケースを参考に，設計書またはソースコードの実行フローを追跡し，欠陥型（例えば，文献 [10] の欠陥型）に基づいて欠陥の検出を行う手法である．しかし，TCBR は優先的にレビューすべき部分や優先的に検出すべき欠陥，及び必要となる修正確認テスト規模に関して明示的には考慮していない．

3. 提案手法

3.1 概要

提案手法は，修正確認テスト規模の低減を目的とし，テスト工程で検出された場合に大きな修正確認テスト規模を必要とする欠陥をレビューで優先的に検出する．修正確認テストは，(1) 欠陥の修正後に当該機能の動作を確認するためのテ

スト（確認テスト），及び (2) 欠陥の修正に伴って新たに欠陥が混入されていないか（デグレード）を確認するためのテスト（回帰テスト）からなる．提案手法では，特にソースコードユニット（関数やメソッドなど）間の呼び出し依存関係に着目し，欠陥が含まれていた場合に大きな修正確認テスト規模を必要とするソースコードユニットを特定する．

提案手法では，修正確認テスト規模を修正確認に必要なテストケース数（修正確認テスト件数）によって計測する．修正確認テスト件数は，確認テスト件数，回帰テスト件数の和である．これら確認テスト件数及び回帰テスト件数はテスト計画書から得る．テスト計画書は，テスト観点の設定とテストの所要時間を見積ることを目的としたドキュメントであり，提案手法ではテスト計画書がコードレビューに先立って作成されていることを前提とする．テスト計画書 I には次のテスト項目 I_k が含まれる．

$$I_k = < Description_k, c_k >$$

ここで，$Description_k$ は自然言語で記されたテスト内容，c_k は予定確認テスト件数を表す．

テスト計画書のテスト項目は，テスト実施フェーズ（単体テスト，結合テスト，システムテスト），テスト視点（論理テスト，機能テスト），テスト方法（ブラックボックス，ホワイトボックス）を問わない．ただし，提案手法は，修正確認テスト件数の削減によるテスト工数の低減を目的としているため，テスト実施フェーズにおいては結合テスト及びシステムテスト，テスト視点においては機能テストにおいて特に効果が得られると期待される．

3.2 手順

以下に示す手順で対象システムのソースコードのレビューを行う．

手順 1. ソースコードユニットへの分割

レビュー対象のソースコード U をソースファイル，クラス，関数，メソッドなどの構文上の区切りで分割し，それぞれをソースコードユニット u_i と

する．分割の区切りは，ソースコードの規模，レビュー時間の制約によって決定する．

手順 2. テスト項目 I_k のテスト対象となるソースコードユニットの列挙

$Description_k$ をもとに，テスト項目 I_k のテスト対象となるソースコードユニットの集合 U_k を求め，I_k に追加し I'_k とする．

$$I'_k = < Description_k, c_k, U_k >$$

手順 3. ソースコードユニットごとの修正確認テスト件数の算出

ソースコードユニット $u_i(i = 1, ..., |U|)$ の修正確認テスト件数 n_{u_i} を次の手順で求める．

for u_i in U **do**
  for I'_k in I' **do**
    if $u_i \in U_k$ **then**
      $n_{u_i} = n_{u_i} + c_k$
    end if
  end for
end for

手順 4. レビューの実施

与えられたレビュー工数の範囲で n_{u_i} が大きいソースコードユニット u_i から順番に，欠陥が含まれないかレビューする．

なお，具体的なテスト計画立案が完了していないなど，何らかの事情で，修正確認テスト件数 c_k が得られない場合でも，手順 2 において c_k どうしの大小関係を相対的に求めることができれば，本手順を実施できる．具体的には，テスト内容 $Description_k$ から c_k の確認テスト件数の大小関係を決定し，文献 [16] に示されている Analytic Hierarchy Process（AHP）[35] を利用する．AHP の代わりに Cumulative Voting（CV）[25] を利用することもできる．ただし，AHP や CV はテスト項目数 $|I|$ が大きくなると適用が難しくなることが予想されるため，I_k を

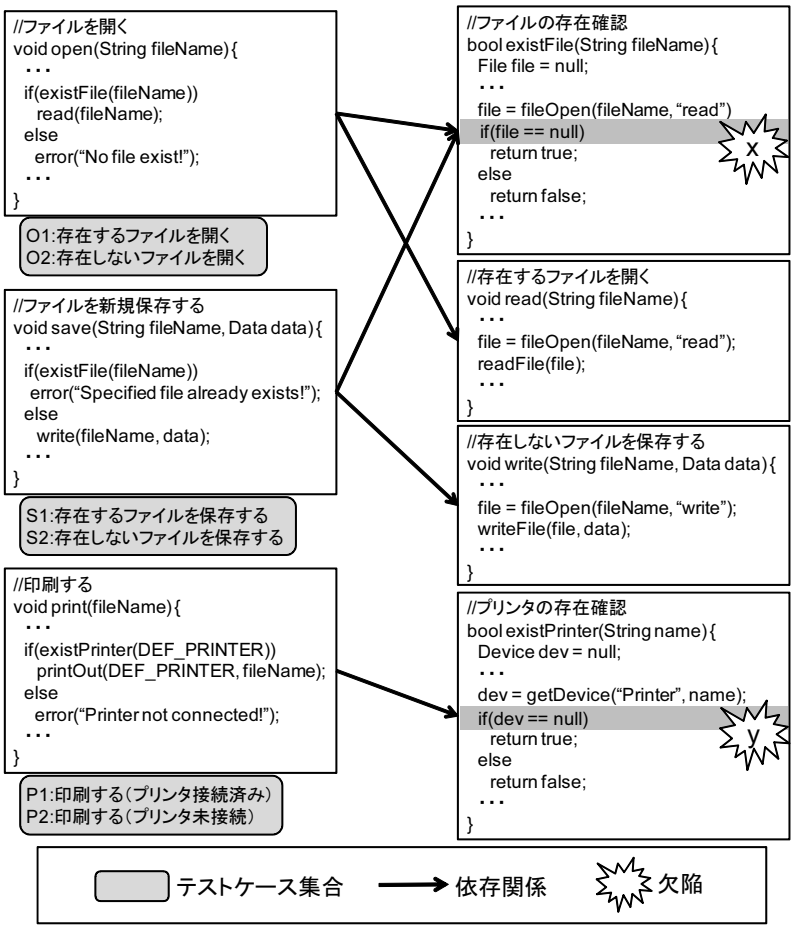

図 4.1 ソースコードの依存関係と修正確認テストの例

抽象化することによって Hierarchical Cumulative Voting[2] を適用する，または複数人で手分けすることによって大規模 AHP[50] を利用することなどが考えられる．

3.3 実施例

図 4.1 に示すソースコード及び図中のテスト $O1$，$O2$，$S1$，$S2$，$P1$，$P2$ を対象に提案手法の実施例を示す．本実施例では，特に手続き型プログラミング言語

表 4.1 図 4.1 のテスト項目，テスト件数，対象ソースコードユニット

	テスト内容	予定テスト件数 (件)	テスト対象となるソースコードユニット
I'_1	O：ファイルを開くことができるかを確認する	2	open，existFile，read
I'_2	S：ファイルを保存することができるかを確認する	2	save，existFile，write
I'_3	P：印刷することができるかを確認する	2	print，existPrinter

で記述されたソースコードを対象とし，ソースコードを関数単位で読み進めていく（ソースコードユニットを関数とする）．この時，テスト工程まで欠陥が見逃された場合に修正確認テスト件数が最も大きくなる関数から順番に読み進めるものとする．なお，本実施例では，レビューアはシステムの大まかな動作とテスト件数は把握できているが，ソースコードは事前には理解できていないことを想定している．

図中の x，y は欠陥を表している．以降では，x，y の修正確認テスト件数は x のほうが大きいことを示し，その後，提案手法で x のほうが y よりも先に検出されることを手順に照らしながら示す．関数 existFile に含まれる欠陥 x を関数 open のテストで検出した場合，確認テスト件数が 2（O1 と O2），回帰テスト件数が 2（S1 と S2）であり，修正確認テスト件数は 4 である．一方，関数 print のテスト（P1 または P2）で関数 existPrinter に含まれる欠陥 y を検出した場合，確認テスト件数は 2（P1 と P2），関数 existPrinter は他の部分との依存関係を持たないため回帰テスト件数は 0 であり，修正確認テスト件数は 2 となる．以下は 4 章 3.2 項で述べた手順に沿って，図 4.1 に示すソースコードとテストケースを対象として提案手法を実施したものである．

手順 1. ソースコードユニットへの分割

対象ソースコードをソースコードユニット（open，save，print，existFile，read，write，existPrinter）に分割する．

手順 2. テスト項目 I_k のテスト対象となるソースコードユニットの列挙

テスト計画書 I から表 4.1 に示される I' を求める．具体的には以下のとおりである．

I'_1: ファイルを開くことができるかを確認する際に実行されるソースコードユニットを図 4.1 のソースコードから目視で確認する．open 関数，及び，open 関数から呼び出される existFile 関数，read 関数をテスト対象となるソースコードユニットの集合 U_1 とする．

I'_2: ファイルを保存することができるかを確認する際に実行される save 関数，及び，save 関数から呼び出される existFile 関数，write 関数をテスト対象となるソースコードユニットの集合 U_2 とする．

I'_3: 印刷することができるかを確認する際に実行される print 関数，及び print 関数から呼び出される existPrinter 関数をテスト対象となるソースコードユニットの集合 U_3 とする．

手順 3. ソースコードユニットごとの修正確認テスト件数の算出

I' から，ソースコードユニット（関数）existFile の修正確認テスト件数 4 を得る．また，それ以外のソースコードユニットの修正確認テスト件数 2 を得る．

手順 4. レビューの実施

最も修正確認テスト件数が大きい existFile に欠陥がないかレビューし，次に修正確認テスト件数が大きい open，save，print，read，write，existPrinter を与えられたレビュー工数の範囲でレビューする．

本実施例は，関数の呼び出し依存関係を用いた見積りの例を示したが，観点は必ずしも呼び出し関係だけに依存するものではなく，他にも，データ構造の依存関係や計算機リソースの共有（性能，メモリ，デッドロックをはじめとするタイミング問題）に起因する修正は修正範囲が広く，修正確認テスト件数を増大させる傾向があるため，修正確認テスト件数の見積りを行う対象となる．

4. 実験

4.1 概要

提案手法が修正確認テスト件数の低減につながること，及び多くの修正確認テスト件数が必要となる欠陥を数多く検出できることを確認するために，提案手法，TCBR，及び AHR の比較実験を行う．全てのレビューアには，レビュー対象である C 言語で記述されたソースコード，外部仕様書，及び内部設計書を与える．これらに加えて，提案手法を実施するレビューアにはテスト計画書，TCBR を実施するレビューアにはテスト計画書及び欠陥型の一覧を与える．

本実験におけるレビューアは，C 言語によるプログラミング経験のある学生 12 名と，商用のソフトウェア開発を実務で行った経験のある実務経験者 6 名である．提案手法の効果がレビューアの開発経験の長短に強く依存するかどうかを調べることを目的として，実務経験者，学生を被験者に含めている．この 18 名を提案手法を実施するレビューア 6 名，TCBR を実施するレビューア 6 名，AHR を実施するレビューア 6 名（それぞれ学生 4 名，実務経験者 2 名）に分割し，比較実験を行った．レビューアの C 言語及びその他のプログラミング言語の経験年数が均等になるようグループの分割を行った．

4.2 レビュー対象

レビュー対象は次のようなイベント受付システム（Web サーバで実行される CGI）を実現するためのソースコードである．システムには「システム管理者」「イベント管理者（主催者）」「イベント参加者」の 3 者が関わる．システムは，複数のイベントの参加登録を同時に受け付けることができる．イベントは URL によって区別され，システム管理者がイベントを登録した時点で URL が発行される．イベント管理者は自身が管理するイベントの URL を参加者に伝え，イベント参加者は住所，氏名といった情報をシステムに送信する．

図 4.2 はイベント受付システムの情報のやりとりを示したものである．個々のイベントを主催するイベント管理者がシステム管理者にイベント作成の依頼をし，

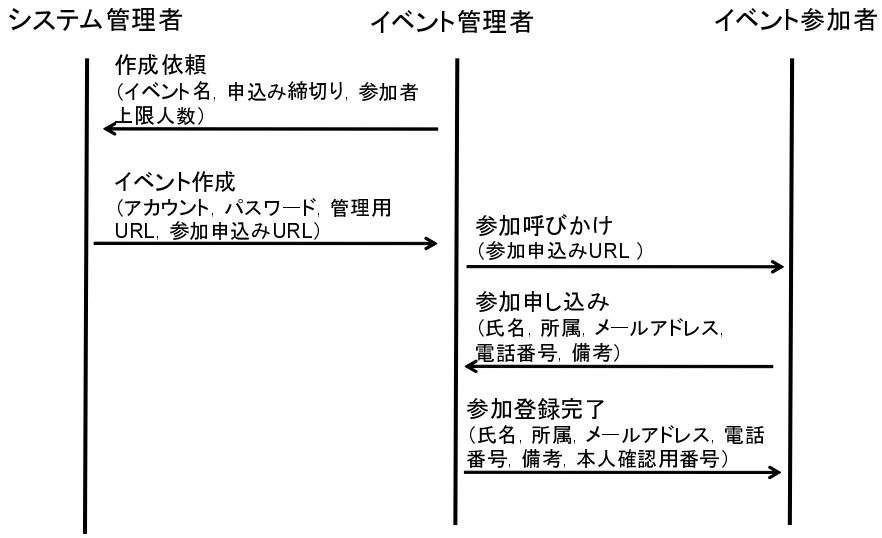

図 4.2 イベント受付システムのデータの流れ

作成画面からイベントを作成する．イベントを作成すると作成されたイベントの参加申込み用 Web ページの URL が生成され，システム管理者からイベント管理者に伝えられる．イベント管理者はこの参加申込み用の URL をイベント参加者に通知し，イベント参加者は参加申込画面を通じて，参加に必要な情報をイベント管理者に送信する．イベント管理者は，イベント管理者画面を通じて，登録者リストを見ることができる．

対象としたソースコードは C 言語で書かれており，Linux 上で Apache Web サーバの CGI として動作する．規模は約 1,500LOC であり，7 個のファイルからなる．本システムは本実験用に，類似システムの業務での開発経験を持つ実務経験者により開発されたものであり，類似システムの開発経験を持つ別の実務経験者により，現実的な開発や設計がされていること，ドキュメント（外部仕様書，内部設計書，テスト計画書）に不備がないことを確認した．

全てのレビューアには外部仕様書，内部設計書，ソースコードが渡される．外部仕様書，内部設計書は自然言語で書かれている．加えて，提案手法でレビューを行うレビューアにはテスト計画書，TCBR でレビューを行うレビューアにはテ

表 4.2 実験に使用したテスト計画書の一部

機能大分類	画面名称	機能小分類	予定テスト件数 (件)
参加登録機能	登録情報入力画面	イベント名ブラウザタイトル表示機能	4
		期限切れ確認機能	6
		参加人数超過確認機能	4
		ボタン押下によるページ遷移確認	1
	登録情報確認画面	「氏名」条件判断とエラーメッセージ	7
		「所属」条件判断とエラーメッセージ	7
		「メールアドレス」条件判断とエラーメッセージ	8
…	…	…	…

スト計画書及び欠陥型の一覧が渡される．テスト計画書は表 4.2 に示すような表形式のものであり，機能大分類，画面の名称，機能小分類，予定テスト件数が記述されており，3 個の機能大分類と 43 個の機能小分類から成っている．欠陥型は文献 [49] と同様に文献 [10] の PSP 欠陥型標準を用いた．表 4.3 に被験者に提示した欠陥型の一覧を示す．

ソースコード中には開発中に混入された欠陥がそのまま残されており，計 34 個の欠陥が確認されている．レビューで検出した場合に削減できる欠陥 1 件あたりの修正確認テスト件数の最小値，中央値，平均値，及び最大値を表 4.4 に示す．表中の値は，欠陥が存在するソースコードユニットの修正確認テスト件数から求めた．具体的には，ソースコードユニットを関数とし，4 章 3.2 項で示した手順と表 4.2 のテスト項目からソースコードユニットごとの修正確認テスト件数を算出した．

修正確認テスト件数は原則として確認テストと回帰テストの合計件数とした．ただし，確認テストと明らかに重複する回帰テストのテストケースが存在している場合には，重複するテストケースの一方を省略可能と見なした．例えば，文字列の長さをチェックするライブラリ関数の欠陥を修正した場合，文字列の長さを

表 4.3 被験者に提示した欠陥型（文献 [10]）

型番号	型名	型番号	型名
10	文章	60	チェック
20	構文	70	データ
30	ビルド，パッケージ	80	機能
40	割り当て	90	システム
50	インタフェース	100	環境

表 4.4 レビューでの検出により削減できる修正確認テスト件数

	最小値	中央値	平均値	最大値
修正確認テスト件数	1	7	10.6	71

チェックするテストの中の「参加者氏名」「参加者住所」といった類似した（半角，全角文字で入力され最大長が設定された入力項目）複数のテストケースを全て実施する必要はないため，必要のないテストケースを省略する．具体的には次式のとおりである．

$$\begin{aligned} &\text{修正確認テスト件数} \\ &\quad = \text{確認テスト件数} + \text{回帰テスト件数} - \text{省略可能テスト件数} \end{aligned}$$

ここで，省略可能テスト件数は，ホワイトボックステストと考えた際に，確認テスト及び回帰テストの中で他のテストケースと類似または重複しているテストケースの件数を表す．

4.3 手順

評価実験は以下の手順で行った．

手順 1. レビューアにレビュー対象としているシステムの説明を行う．説明では，ソースコード以外のドキュメントには欠陥は含まれないとし，ソースコードに含まれる欠陥の数は伝えていない．また，レビューア全員に，評価実

験におけるレビューを「テストで検出する場合よりも，レビューで検出することにより，修正確認工数がより小さくなるような欠陥を目視で検出すること」であると説明した．

手順 2. 提案手法及び TCBR を実施するレビューアには各レビュー手法についての説明を行い，AHR を用いるレビューアには特定のレビュー手法に関する説明は行わない．

手順 3. 提案手法を実施するレビューアにはテスト計画書を与え，ソースコードユニットを関数とすることを指示した．TCBR を実施するレビューアにはテスト計画書と欠陥型の一覧を与える．

手順 4. レビューアは手順 2 で説明したレビュー手法を用いてレビュー作業を行う．欠陥を検出した際にはその内容と検出時刻を記録する．また，制限時間を過ぎた場合は，レビュー作業が途中であっても終了する．

実験に要する時間は，手順 1 が 30 分，手順 2 及び手順 3 が 10 分（AHR のレビューアはなし），手順 4 が 60 分の計 100 分（AHR のレビューアは計 90 分）であり，手順 4 では全ての欠陥を検出できなくても終了するものとする．

4.4 評価基準

テスト計画書に記述されている予定テスト件数に基づいて，削減できた修正確認テスト件数（削減件数）を評価する．また，削減される修正確認テスト件数が上位 50%（上位 17 個）に含まれる欠陥を優先的に検出すべき欠陥と考え，それらを検出した個数（優先欠陥検出数）も評価する．

これらに加えて，文献 [24] 及び [42] で定義されている次式の評価基準も用いる．

$$\text{検出数} = \frac{\text{検出欠陥数}}{\text{レビュー時間（時）}}$$

$$\text{検出率} = \frac{\text{検出欠陥数}}{\text{全欠陥数}}$$

表 4.5 実験における各レビュー手法の修正確認テスト削減能力と欠陥検出能力

		平均値		
		提案手法	TCBR	AHR
全体	削減件数	45.6	21.7	24.2
	優先欠陥検出数	2.5	1.2	1.2
	検出数	4.5	2.2	1.7
	検出率	13%	6%	5%
学生	削減件数	30.0	11.3	17.8
	優先欠陥検出数	2.3	1.0	1.0
	検出数	3.8	1.8	1.0
	検出率	11%	5%	3%
実務経験者	削減件数	76.5	42.5	37.0
	優先欠陥検出数	3.0	1.5	1.5
	検出数	6.0	3.0	3.0
	検出率	18%	9%	9%

4.5 結果

実験結果を表 4.5 に示す．レビューア全体では，削減件数の平均値は提案手法が，TCBR の約 2.1 倍，AHR の約 1.9 倍の値であった．優先欠陥検出数の平均値は提案手法が，TCBR 及び AHR の約 2.1 倍の値であった．検出数及び検出率の平均値は提案手法が，TCBR の約 2.0 倍，AHR の約 2.6 倍の値であった．レビューア全体において，提案手法は TCBR 及び AHR よりも修正確認テスト件数を削減できたといえる．また，TCBR と AHR で削減できる修正確認テスト件数に大きな差はないが，TCBR の方が AHR よりも数多くの欠陥を検出できたといえる．

学生のレビューアでは，削減件数の平均値は提案手法が，TCBR の約 2.7 倍，AHR の約 1.7 倍の値であった．優先欠陥検出数の平均値は提案手法が，TCBR 及び AHR の約 2.3 倍の値であった．検出数及び検出率の平均値は提案手法が，TCBR の約 2.1 倍，AHR の約 3.8 倍の値であった．学生のレビューアにおいても，提案手法は TCBR 及び AHR よりも修正確認テスト件数を削減できたといえる．ま

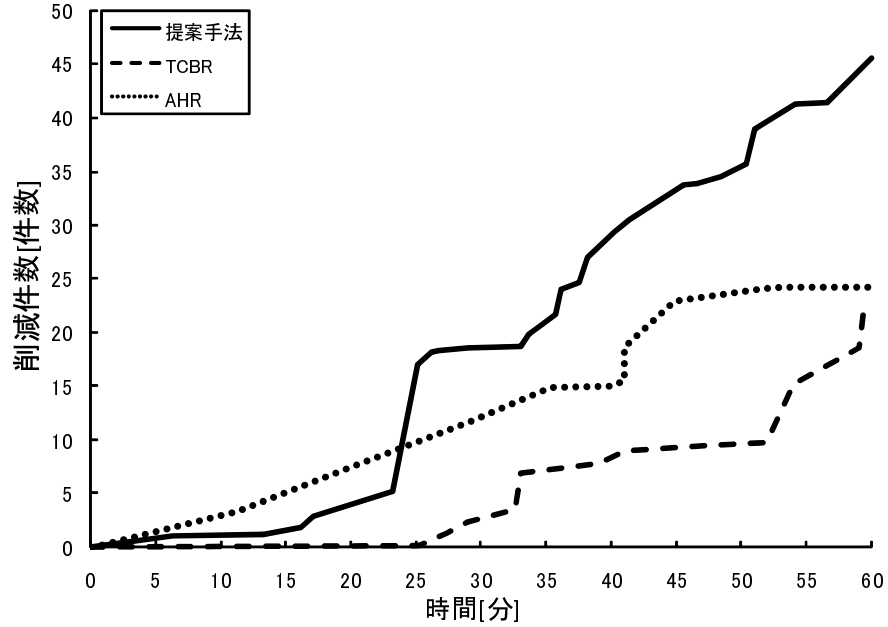

図 4.3 削減件数の累積（平均値）

た，AHR の方が TCBR よりも修正確認テスト件数を削減できたが，TCBR の方が AHR よりも数多くの欠陥を検出できたといえる．

実務経験者のレビューアでは，削減件数の平均値は提案手法が，TCBR の約 1.8 倍，AHR の約 2.1 倍の値であった．優先欠陥検出数の平均値は提案手法が，TCBR 及び AHR の約 2.0 倍の値であった．検出数及び検出率の平均値は提案手法が，TCBR 及び AHR の約 2.0 倍の値であった．実務経験者のレビューアにおいても，他のレビューアと同様に，提案手法は TCBR 及び AHR よりも修正確認テスト件数を削減できたといえる．また，TCBR の方が AHR よりも修正確認テスト件数を削減できたが，TCBR と AHR で検出できる欠陥数に差は見られなかった．

図 4.3 は，削減件数の累積（平均値）の推移を示している．横軸はレビュー開始からの経過時間を示し，縦軸は削減された修正確認テスト件数を示している．実線は提案手法，破線は TCBR，点線は AHR のテスト件数を表している．なお，全てのレビューアは制限時間である 60 分間レビューを行った．開始から約 23 分

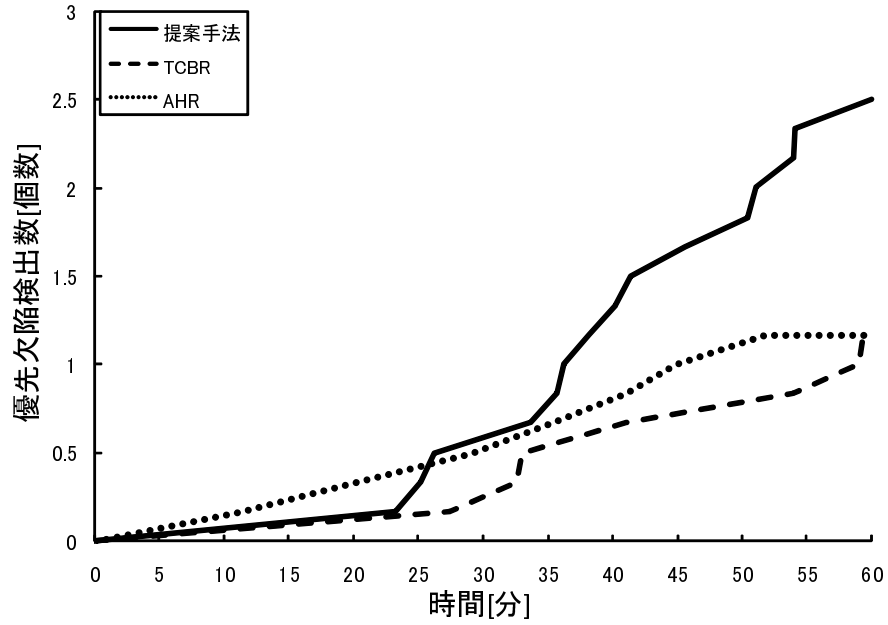

図 4.4 優先欠陥検出数の累積（平均値）

後までは AHR の累積削減件数の平均値が最も大きかったが，それ以降は提案手法の累積削減件数の平均値が最も大きくなった．TCBR の累積削減件数の平均値は時間にかかわらず最も低かった．また，開始から 23 分以降と 25 分以降は，それぞれ提案手法と TCBR において，累積削減件数の増加の割合も大きくなった．

図 4.4 は，優先欠陥検出数の累積（平均値）の推移を示している．横軸はレビュー開始からの経過時間を示し，縦軸は累積優先検出欠陥数を示している．実線は提案手法，破線は TCBR，点線は AHR の累積欠陥数を表している．開始から約 26 分後までは AHR の累積数の平均が最も大きかったが，それ以降は提案手法の累積数の平均が最も大きくなった．開始から 59 分まで，TCBR の累積削減件数の平均値は最も低かった．また，開始から 23 分以降と 26 分以降は，それぞれ提案手法と TCBR において，累積優先欠陥検出数の増加の割合も大きくなった．

5. 考察

TCBR 及び AHR との比較実験により提案手法の有効性を示す結果が得られた．修正確認テストの削減件数，修正確認テスト件数上位 50%に含まれる欠陥が検出された数（優先検出欠陥数）ともに，提案手法は TCBR 及び AHR よりも大きな値を示した．提案手法によりレビューアは修正確認テスト件数が大きくなるような欠陥を優先して検出でき，それに伴い多くの修正確認テスト件数を削減できたと考えられる．また，学生，実務経験者によらず，提案手法の効果を確認できた．

実際にレビューアが提案手法により修正確認テスト件数を考慮し優先的にレビューすべき部分を特定できたかどうかをレビューアへのインタビューにより確認したところ，提案手法が期待どおりの効果をもたらしていることを裏付ける次のような回答が得られた.「提案手法を用いることで優先的にレビューすべき部分の特定ができ，レビューが円滑に行えたように感じた」,「レビュー中に，削減できる修正確認テスト件数が小さい（削減に寄与しない）と予想される欠陥を検出した場合において，その欠陥と類似した別の欠陥を見つけることに固執せず，修正確認テスト件数が大きくなると予想される欠陥を優先的に検出することに注力することができた」．これらの回答は提案手法により修正確認テストが大きくなるような欠陥の検出を優先できることを示していると考えられる．提案手法を実施したレビューアは 4 章 3.2 項の手順 2 の作業を，テスト内容に対応する入出力画面を類推し，テストを実施した際に最初に実行されるソースコードユニットを探すことからはじめていた．手順 3 では，最初に実行されるソースコードユニットから呼び出されている関数を類推し，データ構造などその他にも，依存関係のある部分を探していた．一方，AHR を実施したレビューアのレビュー方針についてもインタビューしたところ,「他からよく呼び出されているライブラリ周りからレビューした」「設計書に書かれているシステムの利用手順に従ってレビューした」といった回答が得られた．

AHR と比較して提案手法は修正確認テストが大きくなる部分を特定するための時間を要する可能性を示す結果も得られた．図 4.3 が示しているように，レビューを開始してから 23 分後までは AHR の方が修正確認テスト件数の平均が大きな値となっている．レビューアへのインタビューにおいても「レビュー開始直後はテ

スト計画書及び内部設計書を読み，各機能の呼び出し依存関係を理解し，優先的にレビューすべき部分を特定する作業を行っていた」という回答が得られた．

本評価実験では，4 章 3.2 項の提案手法の手順 2 及び手順 3 で必要となるソースコードユニットごとに修正確認テスト件数を見積る時間をレビュー時間に含めている．コードレビューに先立ってこの見積り作業を実施している場合には，修正確認テスト件数を見積る作業を省略することができるため，より効率的に修正確認テスト規模の低減につながる欠陥を検出できる可能性がある．また，本評価実験のようにレビュー対象に関する知識の少ないレビューアが提案手法を実施する場合，優先的に読み進めるべき箇所を特定する作業に多くの時間を費やしてしまい，AHR などと比較して十分にレビューできない可能性がある．この可能性が高いと予想される場合には，レビュー対象のドキュメントやソースコードの量，レビューアの対象ソフトウェアに対する理解度を考慮し，事前に対象ソフトウェアを理解する時間を確保するなどの対策を実施することが望ましい．

本評価実験のレビュー対象は手続き型プログラミング言語（C 言語）で記述されており，ソフトウェアの機能とプログラムの構成要素との対応が比較的明確なものが多い傾向にあった．そのため，提案手法で実施する修正確認テスト規模の見積りは比較的容易に実施可能であったと考えられる．一方，オブジェクト指向プログラミングやアスペクト指向プログラミングによってソースコードが記述されている場合においても，テスト内容（$Description_k$）にテスト対象を特定するための情報（例えば，ソースコードファイル名，機能名など）が記述されていれば，これらの情報及び内部設計書などを用いることで提案手法は適用可能である．しかしながら，ソフトウェアの機能とプログラムの構成要素（クラス，メソッドなど）が直交概念となっているなどの理由で，4 章 3.2 項の手順 2 のテスト対象となるソースコードユニット群の特定に相対的に大きな時間を要する場合がある．このような場合においても，商用開発であれば，テスト完了基準として命令網羅などが指定されることが通常であるので，あるテストケースでテスト対象となるソースコードユニットの特定に，極端に大きな工数はかからないと考える．ただし，プログラミング手法がもたらす影響は，今後検討すべき重要な課題の 1 つである．

6. まとめ

本章では，修正確認テスト規模が大きくなると推定される欠陥を優先的に検出し，潜在的な修正確認テスト規模を低減させることを目的としたレビュー手法を提案した．提案手法を評価するために，Test Case Based Reading（TCBR）及び Ad-Hoc Reading（AHR）との比較実験を行い，以下の結果が得られた．

- 提案手法の方が修正確認テスト件数を TCBR よりも約 2.1 倍，AHR よりも約 1.9 倍多く削減できた．
- 多くの修正確認テスト件数が必要となる欠陥について，TCBR 及び AHR よりも提案手法の方が約 2.1 倍多く検出できた．
- 提案手法には優先的にレビューすべき部分を特定する作業が必要なため，AHR を実施した場合と比較してレビュー開始からしばらくの間は欠陥検出数，削減される修正確認テスト件数が少なくなる可能性がある．

なお，本章では修正確認テスト規模（修正確認テスト工数）と修正確認に必要なテストケース件数（修正確認テスト件数）との間に強い相関があるとし，修正確認テスト規模の低減を修正確認テスト件数の低減により計測した．対象ソフトウェアや対象プロジェクトに応じて，修正確認テスト規模を計測するための他の評価基準を検討することが今後の課題である．

本章ではソースコードをレビュー対象としたが，提案手法はテスト規模を考慮できる情報が存在する開発であれば，設計書レビューにも適用可能であると考えられる．

提案手法の効果と制限を明確にするために，異なる規模のソースコードを用いて同様の実験を行うこと，及び，提案手法実施にあたり必要となる準備のコストを考慮した評価を行うことが今後の課題である．

第5章　おわりに

本論文では，工数予測モデルの精度向上及びテスト工数の低減を目的とし，(1) 工数予測モデル構築用のプロジェクトデータに含まれる欠損値の処理法の評価，及び (2) 修正確認テスト規模の低減を効率的に行うコードレビュー手法の提案を行った．

まず，欠損を含むプロジェクトデータに対して，どの欠損値処理を用いることで高い精度の工数予測モデルが構築できるか実験的に評価した．実験では，20ヶ国のソフトウェア開発プロジェクトの実績データから抽出した706件のデータセットに対し，5つの欠損値処理法（平均値挿入法，k-nn法，CF応用法，無欠損データ作成法，ペアワイズ除去法）を適用し，重回帰モデルの構築を行った．各手法の予測精度を評価するために欠損のない143件のプロジェクトの工数予測を行った結果，706件のデータセットを用いた場合には，類似性に基づく補完法（k-nn法及びCF応用法）を用いることで高い精度のモデルが構築されることがわかった．この結果から，欠損値を含むプロジェクトデータであっても，数多く収集することで利用する価値があることがわかったため，開発現場における実績データの収集と利用を促進できると考える．

また，欠損値処理法を用いた工数予測モデルにおけるプロジェクト件数と予測精度の関係を明らかにするために，上記と同じデータセットからプロジェクト件数の異なるデータセットを複数抽出し，同様の実験を行った．実験の結果，プロジェクト件数が多く，補完対象プロジェクトと類似したプロジェクトが存在する場合には，類似性に基づく補完法により高い精度の工数予測モデルを構築できることがわかった．一方で，プロジェクト件数が少ない場合には，無欠損データ作成法により高い精度の工数予測モデルを構築できることがわかった．この結果は，予測時に使用可能なデータセットに対して適用する欠損値処理法を選定する際の

1 つの基準となると考える．さらに，プロジェクト件数が異なれば，高い精度の工数予測モデルを構築できる欠損値処理法も異なること，及び，プロジェクト件数の減少に伴う欠損値処理法を用いた工数予測モデルの精度低下に着目すると，類似性に基づく補完法は無欠損データ作成法と比較して精度の低下が著しいという知見も得た．

次に，修正確認テスト規模の予測に基づいて優先的にレビューすべき箇所を特定することで，修正確認テスト規模の大きくなる欠陥を効率的に検出するコードレビュー手法を提案した．実験では，提案手法により修正確認テスト規模が大きい欠陥を優先的に検出できることを確認するために，商用開発の実務経験者 6 名を含む 18 名の被験者の間で，提案手法，Test Case Based Reading（TCBR），及び Ad-Hoc Reading（AHR）の比較を行った．実験の結果，TCBR と比較して平均 2.1 倍，AHR と比較して平均 1.9 倍の修正確認テスト規模の削減が確認できた．この結果から，提案手法を用いることで修正確認テスト規模の削減によるテスト工数の低減が期待される．

これら論文の成果から，ソフトウェア開発プロジェクトにおいて，過去の実績に沿った開発計画の立案が容易になるとともに，開発工数の低減が見込まれる．これによって，納期の遅延や工数超過のリスクを低減できることから，ソフトウェア開発の大規模化と短納期化への要求にこたえることを支援できると考える．

謝辞

本研究を進めるにあたり多くの方々に，御指導，御協力，御支援頂きました．ここに謝意を添えて御名前を記させて頂きます．誠にありがとうございました．

奈良先端科学技術大学院大学 情報科学研究科 松本 健一 教授には，論文執筆やプレゼンテーションの作法といった研究に対する直接的な指導に限らず，研究に取り組む上での心構えから，研究者としての在り方など，ありとあらゆる面で熱心な御指導を頂きました．私が本研究を成し遂げることができたのは，先生の御理解と暖かい御助言に励まされたおかげです．先生の御尽力に敬意を表し，心より感謝致します．

奈良先端科学技術大学院大学 情報科学研究科 飯田 元 教授には，本研究に対する御意見，御指導のみならず，ソフトウェアプロセス改善技術及びソフトウェア設計技術に関して熱心な御指導を賜りました．頂いた御指導により，本研究をより有意義なものとすることができました．心より厚く御礼申し上げます．

奈良先端科学技術大学院大学 情報科学研究科 門田 暁人 准教授には，本研究に対して的確なアドバイスを下さり，研究の進め方，実験の設定についても数々の御指導を頂きました．論文執筆や国内外研究会の発表に関しては，論文執筆の作法からプレゼンテーションの作成に渡り，時には昼夜，平日休日を問わず，大変親身な御指導を頂きました．私の研究は，先生の御指導がなければ成しえませんでした．心より厚く感謝致します．

奈良先端科学技術大学院大学 情報科学研究科 森崎 修司 助教には，本研究を進めるにあたり，研究の観点，実務の観点の両面から数限りない御指導を賜りました．論文執筆や国内研究会の発表に関しては，論文執筆の作法からプレゼンテーションの作成に渡り大変親身に御指導を頂きました．また，研究活動以外における様々な相談事についても数多くの御助言を頂きました．心より厚く御礼申し上げます．

奈良先端科学技術大学院大学 情報科学研究科 大平 雅雄 助教には，研究を進めるにあたり，有益な御助言から研究者としての心構えまで，幅広い御指導を頂きました．また，学生生活において，しばしば私を鼓舞させるような励ましの言葉も賜りました．心より深く感謝致します．

奈良先端科学技術大学院大学 情報科学研究科 角田 雅照 特任助教には，研究を進める中で直面した疑問点や問題点に関して，適切かつ丁寧な御指導を賜りました．また，研究に対する姿勢，研究の難しさ，楽しさなど精神的な面についても御助言を頂きました．先生の御協力がなければ，本研究は完遂することはできませんでした．心より厚く御礼申し上げます．

奈良先端科学技術大学院大学 情報科学研究科 松村 和子 特任助教には，ソフトウェアレビューの研究についての御指導を頂きました．心より深く感謝致します．

奈良先端科学技術大学院大学 情報科学研究科 川口 真司 助教，名倉 正剛 特任助教（現 株式会社日立製作所）には，研究活動や学生生活に関して親身な御指導を賜りました．心より厚く御礼申し上げます．

大阪大学大学院 情報科学研究科 柿元 健 特任助教には，大学院入学当初から，実験の設定，論文執筆の作法等，様々な御指導を賜りました．論文執筆にあたり，先生の高い品質を求める姿勢から多くのことを学ばせて頂きました．心より厚く御礼申し上げます．

奈良工業高等専門学校 情報工学科 上野 秀剛 助教には，本研究を進めるにあたり，実験方法，論文執筆の作法，プレゼンテーションの作法等，数多くの御指導を頂きました．また，研究以外においても，日々の学生生活において数多くの御助言を賜りました．心より深く感謝致します．

株式会社野村総合研究所 佐瀬 直樹 氏，片山 正樹 氏，石塚 裕 氏，森信 飛鳥 氏には，博士後期課程で研究を進めるにあたり，御理解と御支援を賜りました．心より厚く御礼申し上げます．

奈良先端科学技術大学院大学 ソフトウェア工学講座 戸田 航史 氏には，本研究を進めるにあたり，実験方法，論文執筆の作法，プレゼンテーションの作法等，様々な有益な御指導を頂きました．また，日々の学生生活においても充実した時間を共有させて頂きました．心より深く感謝致します．

奈良先端科学技術大学院大学 ソフトウェア工学講座 亀井 靖高 氏には，本研究を進めるにあたり，実験方法，論文執筆の作法，プレゼンテーションの作法等，数多くの御指導を頂きました．学生生活においても，大学院入学以来から大変親しく接していただき，私の学生生活はより充実したものとなりました．また，氏

の研究に対する姿勢から数限りない励みと刺激を賜りました．心より厚く御礼申し上げます．

奈良先端科学技術大学院大学 ソフトウェア工学講座 伊原 彰紀 氏には，研究発表に関する御助言，日々の研究生活において数多くの御支援を賜りました．また，大学院入学以来，充実した時間を共有させて頂きました．心より深く感謝致します．

奈良先端科学技術大学院大学 情報科学研究科 ソフトウェア工学講座ならびにソフトウェア設計学講座の皆様には，日頃より多大な御協力と御助言を頂きました．研究発表の準備や研究の過程で賜った御助力，励ましの言葉により，私は研究を完遂することができました．皆様のおかげで，誠に有益な研究生活を過ごすことができました．特に，同期生の木浦 幹雄 氏（現 キヤノン株式会社），左藤 裕紀 氏（現 株式会社日立製作所），松田 侑子 氏（現 シャープ株式会社），石田 響子 氏（現 株式会社日立製作所），高田 純 氏（現 マイクロソフト株式会社），山科 隆伸 氏（現 日本ユニシス株式会社）には，大学院入学以来，多くの有意義な時間を共有させて頂きました．心より厚く御礼申し上げます．

本研究を進めるにあたり，評価実験に御協力頂いた被験者の皆様に深く感謝致します．

最後に，大学院において研究するにあたり，私を励まし，支え，温かく見守ってくれた私の家族に心より感謝致します．

参考文献

[1] *IEEE Standard for Software Reviews and Audits.* IEEE Std. 1028-2008, IEEE Computer Society, 2008.

[2] P. Berander and P. Jönsson. Hierarchical cumulative voting (HCV) - prioritization of requirements in hierarchies. *International Journal of Software Engineering and Knowledge Engineering*, Vol. 16, No. 6, pp. 819–849, 2006.

[3] B.W. Boehm. *Software Engineering Economics.* Prentice Hall, New Jersey, 1981.

[4] L. Briand, T. Langley, and I. Wieczorrek. A replicated assessment and comparison of common software cost modeling techniques. In *Proc. 22nd IEEE International Conference on Software Engineering*, pp. 377–386, 2000.

[5] M. Cartwright, M.J. Shepperd, and Q. Song. Dealing with missing software project data. In *Proc. 9th IEEE International Software Metrics Symposium*, pp. 154–165, 2003.

[6] N.R. Draper and H. Smith. *Applied Regression Analysis.* John Wiley and Sons, New York, 3rd edition, 1998.

[7] M.E. Fagan. Design and code inspection to reduce errors in program development. *IBM Systems Journal*, Vol. 15, No. 3, pp. 182–211, 1976.

[8] T. Foss, E. Stensrud, B.A. Kitchenham, and I. Myrtveit. A simulation study of the model evaluation criterion mmre. *IEEE Transactions on Software Engineering*, Vol. 29, No. 11, pp. 985–995, 2003.

[9] D.M. Hawkins. The problem of overfitting. *Journal of Chemical Information and Modeling*, Vol. 44, No. 1, pp. 1–12, 2004.

[10] W.S. Humphrey. *A Discipline for Software Engineering.* Addison-Wesley, Boston, 1994.

[11] International Software Benchmarking Standards Group. ISBSG Estimating, Benchmarking and Research Suite Release 9. http://www.isbsg.org/.

[12] R. Jeffery, M. Ruhe, and I. Wieczorek. A comparative study of two software development cost modeling techniques using multi-organizational and company-specific data. *Information and Software Technology*, Vol. 42, No. 14,15, pp. 1009–1016, 2000.

[13] R. Jeffery, M. Ruhe, and I. Wieczorek. Using public domain metrics to estimate software development effort. In *Proc. 7th IEEE International Software Metrics Symposium*, pp. 16–27, 2001.

[14] C. Jones. *Estimating Software Costs: Bringing Realism to Estimating*. McGraw-Hill, New York, 2nd edition, 2007.

[15] P. Jonsson and C. Wohlin. An evaluation of k-nearest neighbour imputation using likert data. In *Proc. 10th IEEE International Software Metrics Symposium*, pp. 108–118, 2004.

[16] J. Karlsson and K. Ryan. A cost-value approach for prioritizing requirements. *IEEE Software*, Vol. 14, No. 5, pp. 67–74, 1997.

[17] B.A. Kitchenham, E. Mendes, and G. Travassos. Cross versus within-company cost estimation studies: A systematic review. *IEEE Transactions on Software Engineering*, Vol. 33, No. 5, pp. 316–329, 2007.

[18] B.A. Kitchenham, L. Pickard, S.G. MacDonell, and M.J. Shepperd. What accuracy statistics really measure. *IEE Proceedings Software*, Vol. 148, No. 3, pp. 81–85, 2001.

[19] J. Kromrey and C. Hines. Nonrandomly missing data in multiple regression: an empirical comparison of common missing-data treatments. *Educational and Psychological Measurement*, Vol. 54, No. 3, pp. 573–593, 1994.

[20] S. Kusumoto, K. Matsumoto, T. Kikuno, and K. Torii. A new metric for cost effectiveness of software reviews. *IEICE Transactions on Information and Systems*, Vol. E75-D, No. 5, pp. 674–680, 1992.

[21] O. Laitenberger. Studying the effects of code inspection and structural testing on software quality. In *Proc. 9th International Symposium on Software Reliability Engineering*, pp. 237–246, 1998.

[22] O. Laitenberger and J.M. Debaud. Perspective-based reading of code documents at robert bosch gmbh. *Information and Software Technology*, Vol. 39, No. 11, pp. 781–791, 1997.

[23] O. Laitenberger, J.M. Debaud, and P. Runeson. An encompassing life cycle centric survey of software inspection. *The Journal of Systems and Software*, Vol. 50, No. 1, pp. 687–704, 2000.

[24] F. Lanubile, T. Mallardo, F. Calefato, C. Denger, and M. Ciolkowski. Assessing the impact of active guidance for defect detection: A replicated experiment. In *Proc. 10th International Symposium on Software Metrics*, pp. 269–279, 2004.

[25] D. Leffingwell and D. Widrig. *Managing Software Requirements: A Use Case Approach.* McGraw-Hill, New York, 2nd edition, 2003.

[26] R.J.A. Little and D.B. Rubin. *Statistical Analysis with Missing Data.* John Wiley and Sons, New York, 2nd edition, 2002.

[27] W. Melo, F. Shull, and G.H. Travassos. Software review guideline. Technical report, COPPE/UFRJ Systems Engineering and Computer Science Program Technical Report ES556/01, 2001.

[28] E. Mendes, C. Lokan, R. Harrison, and C. Triggs. A replicated comparison of cross-company and within-company effort estimation models using

the ISBSG database. In *Proc. 11th IEEE International Software Metrics Symposium*, p. 36, 2005.

[29] E. Mendes, I. Watson, C. Triggs, N. Mosley, and S. Counsell. A comparative study of cost estimation models for web hypermedia applications. *Empirical Software Engineering*, Vol. 8, No. 2, pp. 163–196, 2003.

[30] J. Moses and M. Farrow. Assessing variation in development effort consistency using a data source with missing data. *Software Quality Journal*, Vol. 13, No. 1, pp. 71–89, 2005.

[31] G.J. Myers. *The Art of Software Testing.* John Wiley and Sons, New York, 1st edition, 1979.

[32] I. Myrtveit, E. Stensrud, and U.H. Olsson. Analyzing data sets with missing data: An empirical evaluation of imputation methods and likelihood-based methods. *IEEE Transactions on Software Engineering*, Vol. 27, No. 11, pp. 999–1013, 2001.

[33] A.A. Porter and L.G. Votta. An experiment to assess different defect detection methods for software requirements inspections. In *Proc. 16th International Conference on Software Engineering*, pp. 103–112, 1994.

[34] A.A. Porter, L.G. Votta, and V.R. Basili. Comparing detection methods for software requirements inspection: A replicated experiment. *IEEE Transactions on Software Engineering*, Vol. 21, No. 6, pp. 563–575, 1995.

[35] T.L. Saaty. *The Analytic Hierarchy Process: Planning, Priority Setting, Resource Allocation.* McGraw-Hill, New York, 1980.

[36] P. Sentas and L. Angelis. Categorical missing data imputation for software cost estimation by multinomial logistic regression. *The Journal of Systems and Software*, Vol. 79, No. 3, pp. 404–414, 2006.

[37] M.J. Shepperd and C. Schofield. Estimating software project effort using analogies. *IEEE Transactions on Software Engineering*, Vol. 23, No. 12, pp. 736–743, 1997.

[38] F. Shull, I. Rus, and V.R. Basili. How perspective-based reading can improve requirements inspections. *IEEE Computer*, Vol. 33, No. 7, pp. 73–79, 2000.

[39] Q. Song, M.J. Shepperd, and M. Cartwright. A short note on safest default missingness mechanism assumptions. *Empirical Software Engineering*, Vol. 10, No. 2, pp. 235–243, 2005.

[40] K. Srinivasan and D. Fisher. Machine learning approaches to estimating software development effort. *IEEE Transactions on Software Engineering*, Vol. 21, No. 2, pp. 126–137, 1995.

[41] K. Strike, K. El Eman, and N. Madhavji. Software cost estimation with incomplete data. *IEEE Transactions on Software Engineering*, Vol. 27, No. 10, pp. 890–908, 2001.

[42] T. Thelin, C. Andersson, P. Runeson, and N. Dzamashvili-Fogelstrom. A replicated experiment of usage-based and checklist-based reading. In *Proc. 10th International Symposium on Software Metrics*, pp. 687–704, 2004.

[43] T. Thelin, P. Runeson, and C. Wholin. An experimental comparison of usage-based and checklist-based reading. *IEEE Transactions on Software Engineering*, Vol. 29, No. 8, pp. 687–704, 2003.

[44] C. Walston and C. Felix. A method of programming measurement and estimation. *IBM Systems Journal*, Vol. 16, No. 1, pp. 54–73, 1977.

[45] K.E. Wiegers. *Peer Reviews in Software - A Practical Guide.* Addison-Wesley, Boston, 2002.

[46] 角田雅照, 大杉直樹, 門田暁人, 松本健一, 佐藤慎一. 協調フィルタリングを用いたソフトウェア開発工数予測方法. 情報処理学会論文誌, Vol. 46, No. 5, pp. 1156–1164, 2005.

[47] 柿元健, 角田雅照, 大杉直樹, 門田暁人, 松本健一. 協調フィルタリングに基づく工数見積り手法のデータの欠損に対するロバスト性の評価. 電子情報通信学会論文誌, Vol. J89-D, No. 12, pp. 2602–2611, 2006.

[48] 松村知子, 門田暁人, 森崎修司, 松本健一. マルチベンダ情報システム開発における障害修正工数の要因分析. 情報処理学会論文誌, Vol. 48, No. 5, pp. 1926–1935, 2007.

[49] 野中誠. 設計・ソースコードを対象とした個人レビュー手法の比較実験. 情報処理学会研究報告, 2004-SE-146, Vol. 2004, No. 118, pp. 25–31, 2004.

[50] 八巻直一, 関谷和之. 複数の評価者を想定した大規模 AHP の提案と人事評価への適用. 日本オペレーションズ・リサーチ学会論文誌, Vol. 42, No. 4, pp. 405–421, 1999.

[51] 田中豊, 垂水共之 (編). Windows 版 統計解析ハンドブック ノンパラメトリック法. 共立出版, 東京, 1999.