社団法人 電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.
SS96-6 (1996-07)

オブジェクト指向分析・設計ドキュメント計測システムの作成

松村 崇史†, 島 和之‡, 松本 健一‡, 鳥居 宏次‡

†日立公共システムエンジニアリング
‡奈良先端科学技術大学院大学 情報科学研究科
〒 630-01 奈良県 生駒市 高山町 8916 番地の 5
e-mail: shima@is.aist-nara.ac.jp

あらまし　本稿では，オブジェクト指向分析・設計ドキュメントを測定するためのソフトウェアメトリクスの特徴とそれらのソフトウェアメトリクスを用いてオブジェクト指向分析・設計ドキュメントを測定するシステムについて述べる．このシステムは，分析・設計ドキュメントの作成を支援する CASE ツールの出力に対してオブジェクト指向ソフトウェアメトリクスを適用し，計測データを収集する．開発者は，収集したデータを元に分析・設計ドキュメントの質を向上させることができる．また，ドキュメントをモデル化して表現することにより，将来新たなメトリクスをシステムに追加することを容易にした．

キーワード　オブジェクト指向，分析，設計，ドキュメント，メトリクス，ツール

A System for Measuring Documents of Object-Oriented Analysis and Design

Takafu Matsumura†, Kazuyuki Shima‡, Ken'ichi Matsumoto‡, and Koji Torii‡

†HITACHI Government and Public System Engineering
‡Graduate School of Information Science
Nara Institute of Science and Technology
8916-5 Takayama-cho, Ikoma-shi, Nara 630-01
e-mail: shima@is.aist-nara.ac.jp

Abstract　In this paper, we characterize many types of software metrics proposed for the measurement of object-oriented analysis and design documents. Also, a system is proposed which measures documents resulting from object-oriented analysis and design by using these metrics. The measurement is done by using the output of CASE tools which support the documentation, and by using object-oriented software metrics. By using the proposed system, the quality of documents can be improved since they can be measured by the system. Since more metrics are likely to be proposed in the future, the documents are modeled and transformed into an inner representation before being measured so that new metrics can be incorporated.

key words　object-oriented, analysis, design, document, metrics, tool

1 はじめに

オブジェクト指向分析・設計は，1980年代後半から大きな注目を集めてきた．それは，オブジェクト指向が，実世界におけるイメージによって自然にモデル化でき，上流工程から下流工程における成果物に一貫性があり，部品化再利用に適しており，保守の局所化が図れるなどの特徴を有しているからである．現在，オブジェクト指向分析・設計には，Shlaer&Mellor方法，Coad&Yourdon手法，OMT，Booch法，RDD，HOODなどの手法が提案されている[10]．近年，これらのオブジェクト分析・設計法を用いて開発を行った事例が多く報告されるようになった．

オブジェクト指向分析・設計を用いることによって，従来の構造化設計等の方法論に比べ再利用，拡張性，保守性に優れたソフトウェアの設計がしやすいと言われている．これは，継承やデータ抽象などにより実現されるが，これらを使っても必ずしも良いソフトウェアができるとは限らない．よって，オブジェクト指向ソフトウェアを評価するためのソフトウェアメトリクスが必要になる．

従来のソフトウェアメトリクスはオブジェクト指向の特徴を考慮していないため，オブジェクト指向を用いて分析・設計されたソフトウェアシステムにそのまま用いるには問題がある[6]．オブジェクト指向ソフトウェアメトリクスとして，クラス，操作，継承関係，メッセージパッシングなどに着目したものが，いくつか提案されている[1, 12, 6]が，オブジェクト指向分析・設計法自体が十分に確立していない現状では，これらのメトリクスが変更される，もしくは新たなメトリクスが追加されることが予想できる．

ソフトウェア高品質の鍵は，要求の明確化，品質の作り込み，確認である[15]．特に，品質の作り込みのためには，開発中にドキュメントを定期的に計測し，その結果をドキュメントに反映させなければならない．しかし，ドキュメントの規模が大きくなれば計測に多くの労力が必要となるので，ドキュメントを計測するツールを導入して，計測の効率化を図ることが必要になる．現在，C++言語のソースコードを計測するツールがいくつか提案されている[12, 14]が，分析・設計ドキュメントを十分に計測できるツールは確立していない．理由としては，オブジェクト指向分析・設計法およびそのCASEツールが十分でないこと，そして，オブジェクト指向メトリクスが十分でないことが考えられる．

本研究では，オブジェクト指向分析・設計を支援するCASEツールの出力を計測するシステムを作成した．作成においては，将来の拡張性（メトリクスの追加，対応するCASEツールの追加）を考慮している．作成した計測システムは，Booch法[4]のCASEツールのドキュメントを対象とする．作成したシステムを小規模な分析・設計工程に適用した結果，ドキュメントの静的な性質の一部を確認することができた．また，本システムの計測結果を進捗管理に利用できることも分かった．

本論文の以降の構成は次の通りである．まず,2節では,オブジェクト指向分析・設計法やオブジェクト指向ソフトウェアメトリクスに関連する他の研究について紹介し,本研究の位置付けを明らかにする．3節では,オブジェクト指向ソフトウェアのための既存のメトリクスツールについて述べる．4節では,本研究において作成したシステムの概略，構成，使用例について述べる．5節では，まとめと残された課題とについて述べる．

2 ソフトウェアメトリクス

DeMarcoが“You can't control what you can't measure.”[5]というように，ソフトウェア開発プロセスまたはソフトウェアプロダクトの品質を制御するためには，それらを定量的に測定しなければならない．ソフトウェアメトリクスは，ソフトウェア開発プロセスまたはソフトウェアプロダクトを測定するための方法であり，以下のような目的で用いられる．

- ソフトウェア開発技術の進歩を評価する．
- ソフトウェア開発工数の予測性を改善する．
- ソフトウェア品質を効果的に制御する．
- 潜在する問題を早期に発見する．
- 開発時間，保守工数，故障率などの見積りモデルを改良する．

従来の手続き指向パラダイムにおけるソフトウェアメトリクスとしては，Halsteadのソフトウェア科学メトリクス(software science metric)[8]やMcCabeのサイクロマティック複雑さメトリクス(cyclomatic complexity metric)[13]などの他，多数のメトリクスが提案されている．表2は，測定対象，構造，計測基準の3つの基準によるソフトウェ

表 1: ソフトウェアメトリクスの分類

測定対象	例
プロダクト	ユーザマニュアルの語長，ソースコードの行数，データベースの関連の数
プロセス	設計期間，コーディングの工数，保守コスト，テスト工数
ハイブリッド	ファンクションポイント当たりの工数，入出力装置当たりの平均故障率

構造	例
基本	要求仕様の語数，コードの行数，設計完了までの時間
複合	コード 1000 行当たりの人月，誤り 1 つ当たりの平均訂正時間，テスト効率

計測基準	例
客観的	コメント以外のソースコードの行数，年平均生産バージョン数，入力画面の数
主観的	プログラマの経験，平均学習時間，アプリケーションの使い易さ

アメトリクスの分類を示している [2]．測定対象による分類では，開発成果物の定量的属性を示すプロダクトメトリクス，開発プロセスの定量的属性を示すプロセスメトリクス，プロダクトメトリクスとプロセスメトリクスを合成したものを示すハイブリッドメトリクスがある．構造による分類では，単一の属性で示される基本メトリクスと，幾つかの基本メトリクスから計算される複合メトリクスとがある．計測基準による分類では，正確に定義され，収集者や時間に関わらず繰り返し計測しても等しい計測値を得られる客観的メトリクスと，収集者の判断に依存し，一貫性がなく，繰り返し計測すると計測値が異なるような主観的メトリクスとがある．

しかし，これらのメトリクスは近年急速に広まったオブジェクト指向パラダイムにおける，クラス，メソッド，メッセージ，継承，ポリモルフィズム，オーバーロード，カプセル化などの新しい概念は扱っていない [3]．このため，これらの概念を取り入れたオブジェクト指向ソフトウェアメトリクスが提案されるようになった．Abreu は，オブジェクト指向ソフトウェアメトリクスを以下の 6 種類のカテゴリ (設計，サイズ，複雑さ，再利用，生産性，品質) と 3 段階の粒度 (メソッド，クラス，システム) の 2 つの独立した軸によって分類した．

設計メトリクス： 設計メトリクスの主な目的の 1 つとして，コーディング前に設計の問題点を見つけることがある．設計メトリクスは，ソフトウェアプロダクトの信頼性，テスト容易性，保守性などの様々な品質特性に関係する．

サイズメトリクス： サイズメトリクスは，ソフトウェアの作成，理解，保守に必要な工数を反映するので，資源の見積りにおいて重要である．また，サイズメトリクスは合成メトリクスの正規化でも重要な役割を果たす．つまり，異なるプロジェクト間で総工数，最適なテストケース数，故障数などを比較する場合に用いられる．

複雑さメトリクス： 多くの場合，時間や人員の不足のため，徹底的なコード・インスペクションや全モジュールの単体テストは不可能である．システムの大多数 (8 0 %以上) のフォールトが少数 (2 0 %) のモジュールに含まれる傾向がある [9]．この傾向はオブジェクト指向システムについても実験データによって確認されている [16]．複雑さメトリクスは，これらのフォールトを多く含むモジュールを見つけるために用いることができる．

再利用メトリクス： オブジェクト指向パラダイムは再利用に適しているが，ソフトウェアを効果的に再利用するためには，社内での奨励，管理，教育と共に，再利用を支援する機構を創ることが重要である．ここで再利用の支援とは，再利用項目の収集，分配，品質評価，洗練などである．これらの再利用のための投資に対する効果 (利益) を評価するために，幾つかの企業では再利用メトリクスの収集と分析を始めている [7]．

生産性メトリクス： 開発コストの見積りには開発チームの生産性メトリクスが必要である．開発チームは企業によって異なるので，各企業で過去のプロジェクトから生産性メトリクスを収集しなければならない．ただし，伝統的な生産性メトリクスはオブジェクト指向開発には適さない．例えば，オブジェクト指向開発では工数当たりの新規の LOC は低くなるだろう．これは，再利用がオブジェクト指向開発の一つの目的であるからである．

品質メトリクス： ソフトウェアの品質としては，保守性や可搬性などのユーザからは見えない品質特性も考慮すべきではあるが，信頼性，有用性，機能性といったユーザから見える品質特性は，ソフトウェア製品の成否に関わるので特に重要である．例えば，信頼性メトリクスはソフトウェア製品の出荷前に，その製品が十分な信頼性を達成しているかどうかを調べるために用いられる．

3 メトリクスツール

既に幾つかのオブジェクト指向ソフトウェアメトリクスが提案されているが，それらの有効性を実験的に確認する必要がある．そのためには，それらのメトリクスを用いて多様かつ多数のソフトウェア開発を測定しなければならない．ソフトウェアメトリクスを用いてソフトウェアプロセスやプロダクトを測定したり，測定結果を分析するためのツールとしてメトリクスツールがある．メトリクスツールを用いることによって低コストで正確な測定が可能となる．

メトリクスツールを用いることによってプロダクトメトリクスとハイブリッドメトリクスのデータ収集を自動化することができる．また，主観的メトリクスはツールによって自動的に測定することはできないが，主観的メトリクスの測定結果を用いて複合メトリクスを求める場合はメトリクスツールの利用が有効となる．オブジェクト指向開発のためのメトリクスツールとしては以下のものがある．

OOMetric [12]

- C++とSmalltalkのソースコードを計測する．
- 計測に適応するメトリクスを選択できる
- 計測値の最低値 (Lower threshold) と最高値 (Upper threshold) をユーザが設定できる．
- 計測値をいくつかのフォーマットでインポートもしくはエクスポートできる．
- 計測値を検索することができる．

OOMetTool [14]

- Borland C++のプロジェクトファイルとソースコードを計測する．

  プロジェクトファイルとは，コンパイルやリンクに関係する情報を納めたファイルである．
- 計測値をスプレッドシートが読めるフォーマットでエクスポートできる．

Classic-Ada Metric Analyzer [11]

- Classic-Ada の設計とソースコードを分析し，メトリクスを収集する．
- UNIX の LEX と YACC を用いて実装されている．

文献 [12] では，OOMetric の内部情報を示していない．従って，ツール内におけるデータの管理方法などは不明である．OOMetTool は，データの管理を Paradox [1]を用いて行なうが，データベースのスキーマは明らかにされていない．OOMetric および OOMetTool はソースコードを計測し，設計は計測できない．Classic-Ada Metric Analyzer は，Classic-Ada の設計とソースコードを中間言語に変換してから計測するが，中間言語の仕様が不明であるので，Classic-Ada 以外の設計やソースコードには使えない．

4 設計ドキュメント計測システム

オブジェクト指向ソフトウェアについては，ソフトウェアメトリクスの研究が十分ではなく，メトリクスツールの数も少ない．そこで，本研究ではオブジェクト指向分析・設計ドキュメントを計測するためのシステムを作成した．

表2は，作成した計測システムで扱うオブジェクト指向ソフトウェアメトリクスを示している．DIT と NOC は，Chidamber と Kemerer のメトリクスの中の2つである．PPM はメソッド当たりの平均パラメータ数，NAC は抽象クラスの数，GUS はメソッド当たりの大域参照の平均数，NIV はインスタンスから参照可能な非公開および保護メンバ変数の数，NIM はインスタンスで定義された公開，保護，非公開の全てのメソッドの数である．

4.1 システムの特徴

本システムの特徴は以下の通りである．

[1] Borland Paradox は米国 Borland 社の登録商標です．

表 2: 作成したシステムで扱うメトリクス

	Class	System
Design	DIT, PPM, NAC	GUS
Size	NIV, NIM	
Complexity	NOC	

- オブジェクト指向設計ドキュメントを計測する.

  工数の見積りや品質の作り込みのためには, 開発の上流工程における計測が必要である. Classic-Ada Metric Analyzer は, Classic-Ada の設計を計測することはできるが, オブジェクト指向の設計法として代表的な OMT 法や Booch 法における設計ドキュメントは計測できない.

- 計測結果を標準的なフォーマットで出力できる.

  ソフトウェアメトリクスを用いて計測した結果は, 各組織の事例データベースに保存されたり, 統計処理されたりする. よって, 計測結果はそれらのデータベースシステムや統計処理システムが読み込むことができるフォーマットであることが望ましい.

- 新たなメトリクスを容易に追加できる.

  現在のオブジェクト指向ソフトウェアメトリクスは研究段階であり, 今後も多くのメトリクスが提案されると予想される. 局所的な変更によって新たに提案されたメトリクスを追加できることが望ましい.

- 異なる CASE ツールにも対応できる.

  ソフトウェア開発管理のためにメトリクスデータの収集は重要ではあるが, データ収集のために開発環境が制限されることは問題である. 今後, オブジェクト指向開発に関する研究が進むにつれ, より優れたオブジェクト指向 CASE ツールが開発されることが予想される. それらの CASE ツールへの対応を効率的に行えることが望ましい.

4.2 システムの構成

図 1は作成したシステムの内部 (灰色部) と外部との関係を示している. システム外部には, Rational Rose [2] とオブジェクト指向分析・設計ドキュメントがある.

1. Rational Rose ··· Booch 法に基づくオブジェクト指向分析・設計ドキュメントを作成する CASE ツール.

2. オブジェクト指向分析・設計ドキュメント ··· 本システムでは Rational Rose からの出力に相当する. このドキュメントは, クラス図, シナリオ図, モジュール図, プロセス図を含んでいる. 本システムはオブジェクト指向ソフトウェアメトリクスに着目したため, 現時点ではクラス図とシナリオ図のみを用いて, モジュール図とプロセス図は用いていない.

システム内部には, 一時データファイル, 一時データ抽出ツール, ドキュメントデータベース, ドキュメントデータベース作成ツール, メトリクスデータベース, メトリクスデータベース作成ツールがある. 本システムでは, 複数のオブジェクト指向分析・設計 CASE ツールの出力に対応することを目的として, CASE ツールの出力を一時データファイルに変換する方式をとっている. 本研究では, CASE ツール Rational Rose の出力を一時データファイルに変換するためのツールを作成した. 他の CASE ツールに対応するためには, その CASE ツールの出力を一時データファイルに変換する一時データ生成ツールを作成する必要がある. 一時データファイルの内容は関係データベースに登録される. このデータベースをドキュメントデータベースと呼ぶ. ドキュメントデータベースには, クラスカテゴリ, クラスカテゴリ階層, クラス, 関連, 継承などの関係を示すテーブルが含まれる. メトリクスデータベースには, ドキュメントデータベースを元にドキュメントを計測した結果が, 測定の粒度によってクラス毎とクラスカテゴリ毎とに分けて登録される.

4.3 使用例

この節では, 小規模なオブジェクト指向開発実験から得られた分析・設計ドキュメントの履歴に対して本システムを適用し, 進捗管理に用いる方法について述べる. 進捗管理では以下の作業を行う.

1. 正確な状況把握と問題点の早期発見

[2] Rational Rose は Rational Software 社の登録商標です.

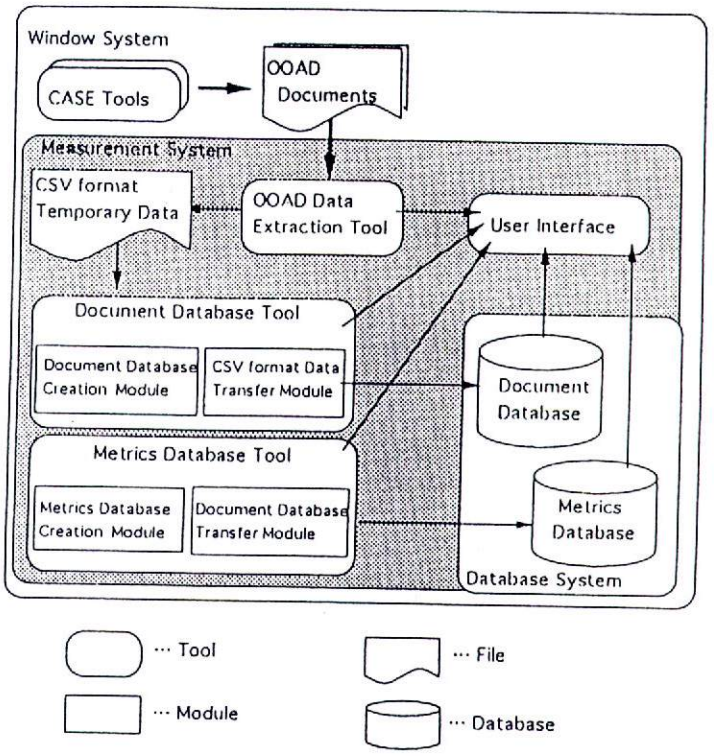

図 1: 計測システムの構成

2. 問題点の分析，他の部分も含めた状況判断

3. 適切な対策

作業 1 と作業 2 の判断基準としてソフトウェアメトリクスを用いることが可能である．例えば，作業 1 については，サイズメトリクスを進捗のマイルストンとして用いた状況把握や，複雑さメトリクスを用いた問題点の早期発見が挙げられる．また，作業 2 については，設計メトリクスを用いた問題点の分析が挙げられる．

測定したオブジェクト指向開発実験では，二人の開発者が CASE ツールを用いて一つの電卓システムの分析と設計を行い，一日に一回以上の頻度で分析・設計ドキュメントの変更履歴を保存した．

図 2は Booch におけるクラス図を示している．図中の雲型の破線はクラスを示し，クラス間の直線は関係を示し，端に黒丸のある関係は保有関係を示している．計測システムを用いて，保存された分析・設計ドキュメントを測定した結果を図 3に示す．この図では，横軸は保存された分析・設計ドキュメントのバージョンを示し，縦軸がメトリクスを用いて測定したデータを示している．この図より，NIV と NIM が時間の経過に従って徐々に増加していることが分かる．

よって，これらのメトリクスを進捗管理におけるマイルストンとして用いる方法が考えられる．これらのメトリクスは Abreu の分類では，サイズメトリクスに分類される．サイズメトリクスは，ソフトウェアの作成，理解，保守に必要な工数を反映する．開発途中のソフトウェアのサイズは，開発のやり直し（手戻り）などが発生した場合を除くと，時間の経過に対して単調増加すると考えられる．よって，サイズメトリクスは進捗のマイルストンとして適していると考えられる．

一方，サイズメトリクス以外のカテゴリのメトリクスは，進捗のマイルストンに適しているとは言えない．設計メトリクスの主な目的の一つとして，コーディング前に設計の問題点を見つけることが挙げられる．設計の問題点は設計の過程で混入していくが，問題点が多くなれば，通常はそれらを解決する作業が行われる．よって，設計メトリクスのデータは，時間の経過に対して単調に増加あるいは減少するとは考えにくい．複雑さメトリクスはサイズメトリクスと同様に，ソフトウェアの作成，理解，保守に必要な工数を反映する．開発の過程でソフトウェアは複雑になっていくが，その一方で複雑さを減らすための変更も行われる．よって，複雑さメトリクスのデータは，時間の経過に対して単調に増加

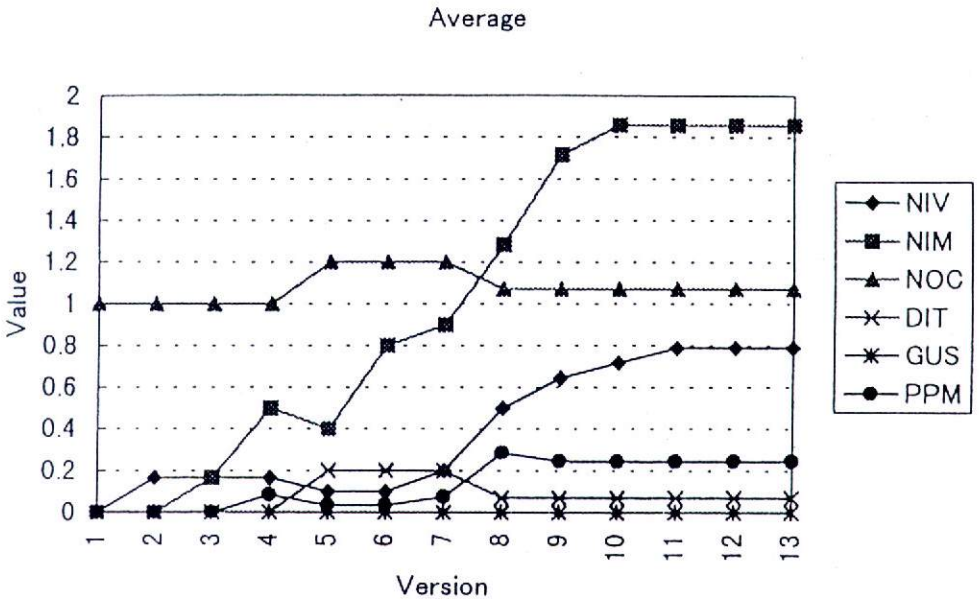

図 3: Result of Measurement

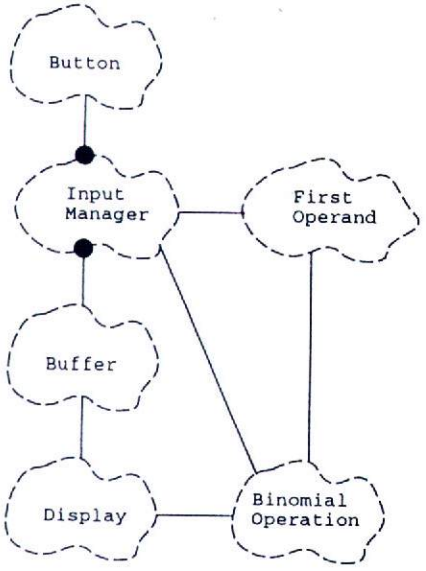

図 2: 電卓システムのクラス図

あるいは減少するとは考えにくい．再利用メトリクスや生産性メトリクスのデータは，プロジェクトによって大きく変わると予想されるが，開発中の時間の経過に対して単調増加あるいは単調減少することは考えにくい．品質メトリクスは信頼性，有用性，機能性などの品質特性を示す．ソフトウェアの品質は，テスト工程では向上していく傾向にあると予想されるが，設計工程では常に向上するとは限らない．

5 おわりに

本稿では，ソフトウェアメトリクスを用いてオブジェクト指向分析・設計ドキュメントを計測するシステムについて述べた．作成した計測システムは，メトリクスの 6 つのカテゴリの中で，設計，サイズ，複雑さのメトリクスについてドキュメントを計測することができる．また，本システムを小規模なオブジェクト指向分析・設計工程に適用した結果，工程の途中においてサイズメトリクスを測定することで，本システムを進捗管理に応用できることを示した．

今後は，複数の CASE ツールへの対応や計測できるメトリクスの追加などを行い，実際のプロジェクトにおけるデータ収集に役立つシステムの構築を目指したい．

参考文献

[1] M. M. Ammann, R.D. Cameron: “Measuring program structure with inter-module metrics”, Proc. Eighteenth Annual International Computer Software and Applications Conference (COMPSAC’94), pp. 139–144 (1994).

[2] F. B. Abreu: “Metrics in the management of information systems development projects (in Portuguese)”, in Proc. of the 6th Jornadas de

Qualidade no Software, APQ, Lisbon, Portugal (1992).

[3] F. B. Abreu and R. Carapuca: "Candidate metrics for object-oriented software within a taxonomy framework", J. Systems Software, 26, pp. 87–96 (1994).

[4] G. Booch: "Object-Oriented Analysis and Design with Applications", Benjamin Cummings Publishing Co. (1991).

[5] T. DeMarco: "Controlling Software Projects ", Prentice-Hall, Englewood Cliffs, New Jersey (1982).

[6] 藤崎 智宏, 荒野 高志: "オブジェクト間の動的な依存関係の定量化", 電子情報通信学会論文誌, Vol. J77-D-I, No. 10, pp. 720–728 (1994).

[7] M. L. Griss: "The economics of software reuse", in Proc. of OOPSLA'91, pp. 264–270 (1991).

[8] M. H. Halstead: "Elements of software science", Elsevier, North-Holland, New York (1977).

[9] W. S. Humphrey: "Managing the software process", Addison-Wesley (1989).

[10] 本位田 真一, 山城 明宏: "オブジェクト指向分析・設計", 情報処理, Vol. 35, No. 5, pp. 392–401 (1992).

[11] Wei Li and Sallie Henry: "Object-oriented metrics that predict maintainability", Journal of Systems and Software 23 (2), pp. 111–122 (1993).

[12] Lorenz M., Kidd J.: "Object-Oriented Software Metrics", Prentice-Hall, Englewood Cliffs, New Jersey (1994).

[13] T. McCabe: "A complexity measure", IEEE Trans. Software Eng., 2, 308-320 (1976).

[14] Stiglic B., Hericko M., Rozman I.: "How to evaluate object-oriented software development?," SIGPLAN Notices., Vol. 30, No. 5, pp. 3–10(1995)

[15] 山田 淳, 建部 周二: "高品質ソフトウェア環境とメトリクス," 電子情報通信学会論文誌, Vol. J73, No. 5, pp. 481–485 (1990).

[16] J. F. Walsh: "Preliminary defect data from the iterative development of a large C++ program (Experience Report), in Proc. of OOPSLA'92, pp. 178–183 (1992).