ソフトウェア保守作業における プログラムの理解支援に関する研究

中村健二† 田中成典‡ 古田均‡ 高橋亨輔†

関西大学大学院総合情報学研究科† 関西大学総合情報学部‡

1．はじめに

近年，ソフトウェアの大規模化や複雑化に伴い，ソフトウェア保守作業の負担が増大[1]している．ソフトウェア保守作業とは，納入後のソフトウェアに対して，バグ修正や性能改善などプログラムを改訂する作業である．プログラムの改訂は，部分的な改訂であっても不具合を埋め込む危険性[2]がある．そのため，改訂前には，改訂箇所に関わるプログラム全体を理解する必要があるが，次の理由により，プログラムの理解には膨大な時間がかかるという問題がある．

- 開発者が以前の作業内容を忘れている場合や開発者と保守担当者が異なる場合は，再度プログラムを理解する必要がある．
- ソースコードを繰り返し修正することで，ソースコードとドキュメントの内容が乖離する．
- ソースコードのみでは，プログラム全体の構成や関係の把握が困難である．

これらの問題に対して，既存研究[3]では，ソースコードから UML（Unified Modeling Language）のクラス図を自動生成し，クラス間の構成を可視化することで，プログラムの理解を支援するものがある．しかし，既存研究で生成されるクラス図は，クラス間の関係やクラスの重要度を考慮せずにすべてのクラスを等間隔で配置するため，クラス図の可読性が低いという問題がある．そこで，本研究では，ソースコードから UML のクラス図を自動生成する際に，クラス及び関連に重要度を付与する手法を提案する．そして，重要度に基づいてクラス図を生成することで，可読性を高め，プログラムの理解を支援する．

Research on Support for Comprehending Program in Software Maintenance
†Kenji Nakamura, Kyosuke Takahashi
Graduate School of Informatics, Kansai University, 2-1-1 Ryouzenji-cho Takatsuki-shi, Osaka 569-1095, Japan
‡Shigenori Tanaka, Hitoshi Furuta
Faculty of Informatics, Kansai University, 2-1-1 Ryouzenji-cho Takatsuki-shi, Osaka 569-1095, Japan

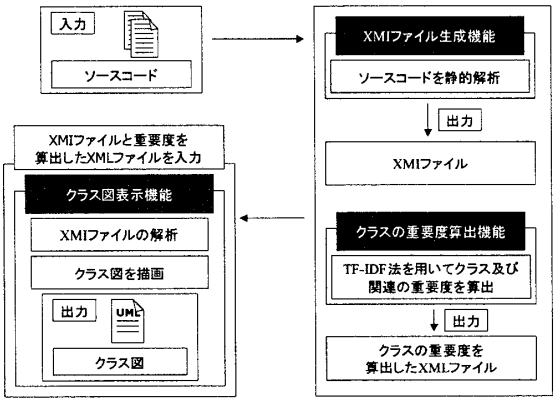

図 1　システムの流れ

2．研究の概要

本研究では，ソースコードからクラス及び関連の重要度を考慮したクラス図を生成する手法を提案する．本システムの流れを図 1 に示す．

本システムは，XMI（XML Metadata Interchange）ファイル生成機能，クラスの重要度算出機能とクラス図表示機能で構成される．XMI は，XML（eXtensible Markup Language）形式でメタデータを交換する標準規格であり，UML モデリングツール間のデータ交換に標準として利用されている．XMI ファイル生成機能では，入力したソースコードを静的解析し，クラス図生成に必要な情報を集め，XMI ファイルを生成する．クラスの重要度算出機能では，他クラスへのアクセスを単語，クラスを文章と考え，TF-DF 法[4]を用いてクラスの重要度を算出し，XML 形式のファイルで出力する．クラス図表示機能では，XMI ファイル生成機能で生成した XMI ファイルとクラスの重要度算出機能で出力した XML ファイルを解析し，クラス図を表示する．クラス図の表示は，関連するクラスを近くにまとめて配置し，重要度の高いクラス上位 30%に色づけを行う．

3．実証実験

3．1　実験内容

本実験では，被験者が実際にソフトウェア保守作業を行い，本システムを利用した場合と既存システムを利用した場合の作業時間を比較した．既存システムとして，オープンソースのUML モデリングツールである IIOSS を利用した．そして，実験終了後，本システムの有効性に関するアンケート調査を行った．実験の対象者は，Java によるオブジェクト指向システムの開発経験が３年程度の開発者６名とした．保守作業の対象は，酒屋問題[5]の仕様に基づいたソフトウェアとした．実装には Java を使用し，ソフトウェアの規模はクラス数 23，ソースコードは約 1,000 行である．実験では，被験者を３名ずつの２グループに分け，１つのグループは本システムより生成したクラス図，もう１つのグループは既存システムにより生成したクラス図を利用し，保守作業を行った．作業内容は，表１に示すバグ修正の課題 A と機能追加の課題 B，課題 C の合計３件である．被験者には，自然文による課題の説明と入出力の例を与え課題を提示した．課題提示後は，特に指示は出さず，作業方法は，被験者の判断に委ねた．

3．2　結果と考察

各作業に要した平均作業時間を表２に示す．バグ修正の課題 A は，本システムと既存システム間で作業時間の違いはほとんどみられなかった．これは，課題 A は，単一メソッド内の修正であり，クラス図を利用する効果が低かったためである．機能追加の課題 B と課題 C では，本システムは既存システムよりも短時間で作業を完了できた．これは，機能追加の課題は，複数クラスの参照が必要となり，クラス間の関係情報の把握にクラス図を利用する効果が高かったためである．また，表２の実験結果に対して，有意な差が認められるかを確認するために検定を行った．ここで，実験結果は正規分布をとらないため，Wilcoxon の符号付順位和検定を実施した．その結果，有意水準１%で平均値に差があるという結果が出た．このことから，本システムは，既存システムよりソフトウェア保守作業に有効なクラス図を生成できることがわかった．

本システムの有効性に関して，評価値５を最良とする５段階評価によるアンケートを実施した結果を表３に示す．表３の結果，本システムはクラス図の可読性という点で既存システムより高評価であった．このことから，本システムにより生成したクラス図は，既存システムと比較して可視化の点で有効であることがわかった．

表１　作業内容

課題番号	内容
A	ライブラリ使用方法の誤り修正
B	商品一覧画面の機能追加
C	空コンテナ一覧画面の機能追加

表２　平均作業時間（分）

課題番号	本システム	既存システム
A	15	10
B	20	32
C	13	41

表３　アンケート調査の項目と結果

内容	本システム	既存システム
システムにより生成したクラス図は，可読性のあるクラス図か	3.5	2.7
システムにより生成したクラス図は，プログラムの理解支援として有効か	4.0	3.2

4．おわりに

本研究では，クラス及び関連に重要度を付与し，ソースコードから可読性の高いクラス図を自動生成する手法を提案した．実証実験の結果より，本システムの有効性が証明された．特に，複数クラスを参照する保守作業において，本システムの利用は有効であることがわかった．今後は，ソースコードを動的解析し，プログラム実行時のクラス間の関係情報を集め，より可読性の高いクラス図の自動生成を目指す．

参考文献

[1] Yip, S. and Lam, T. : A Software Maintenance Survey, Proceedings 1st Asia-Pacific Software Engineering Conference, pp.70-79, 1994. 12.

[2] Hetzel, W. : The Complete Guide to Software Testing, John Wiley and Sons Incorporated, 1984.12.

[3] 齊藤寿和，海尻賢二：オブジェクト指向プログラムのリバースエンジニアリング，ソフトウェア工学研究会研究報告，情報処理学会，Vol.2002，No.23，pp.119-126，2002.3.

[4] Salton, G. and Yang, C. : On the Specification of Term Values in Automatic Indexing, Journal of Documentation , Emerald Group Publishing Limited , Vol.29, No.4, pp.351-372, 1973.6.

[5] 山崎利治：共通問題によるプログラム設計技法解説，情報処理，情報処理学会，Vol.25，No.9，pp.934-935，1982.9.