氏　　名	王　　林			
学位の種類	博士(情報科学)			
学位記番号	博情第 317 号			
学位授与年月日	平成 27 年 3 月 20 日			
論文題目	Improving Transformation of Object-Oriented Program to Aspect-Oriented Program with Static Analysis （静的プログラム解析に基づくオブジェクト指向プログラムからアスペクト指向プログラムへの変換改善手法）			
論文審査委員	主査	鈴木　正人	北陸先端科学技術大学院大学	准教授
		二木　厚吉	同	特任教授
		緒方　和博	同	教授
		青木　利晃	同	准教授
		増原　英彦	東京工業大学	教授

論文の内容の要旨

Separation of concerns is expected to be supported by development methodology or programming languages through enabling encapsulation of each different concern in its own unit of modularity. Unfortunately, current object-oriented languages and development methodology fail to provide a complete and effective support for the separation of concerns. Undesirable phenomena such as code scattering and code tangling occur.

Aspect-Oriented Programming (AOP) technique supports the separation of crosscutting concerns. Aspect-orientation proposes a technique to obtain better software modularity in a practical situation where object-oriented development/programming and associated design patterns are not appropriate. There is a prospect of aspect-oriented programming becoming a mainstream technology in the near future. The questions of how to deal with existing legacy object-oriented software system when aspect-oriented becomes standard practice and how to introduce aspects to an existing legacy object-oriented software system are emerged. The traditional object-oriented programming technique has known limitations, which it could not modularize the implementation of crosscutting concerns in existing software systems. Thus, applying AOP technique to transform the legacy object-oriented software system is one way to obtain the benefits of aspect-oriented programming technique.

In this dissertation, we explore the approach to resolve two obstacles in the transformation of object-oriented (OO) program to aspect-oriented (AO) program with static program analysis techniques. First, we propose a new heuristic algorithm for optimizing the

combination of input metrics for clustering based aspect mining. Aspect mining is the first phrase in the aspect-oriented refactoring (AOR). Aspect mining aims at detecting the code which is likely to implement a crosscutting concern. Aspect refactoring is the second phrase in the AOR, while it is a way to remove the code implementation of crosscutting concerns from an object-oriented program into aspects. Second, After aspect refactoring, the pointcuts is created with pointcut abstraction rules. The pointcuts are always implemented as name-based or enumeration pointcuts, however, these pointcuts are known as fragility against program evolution. We propose a framework Nataly to solve the fragile pointcut problem by (1) automatic inferring the intention properties from the join points matched by the given name-based pointcuts and (2) generating a method to check whether the given join point satisfies the properties or not. We also give a deep discussion of aspect interference problem in the AOR. In this discussion, we propose an idea to check the aspect interference by using static control flow analysis.

The result of experiments revealed that the metric selection approach increased the accuracy of aspect mining in the aspect mining. The fragile pointcut problems are also alleviated, because the pointcuts which are generated automatically by our framework are more robust than the name-based or enumeration counterpart.

Key Words: Aspect-oriented refactoring, Aspect mining, Crosscutting-concern, Analysis-based pointcut, Static analysis

論文審査の結果の要旨

本論文はオブジェクト指向プログラム(OOP)からアスペクト指向プログラム(AOP)への変換に関する新しい手法とその効果について述べたもので英文 8 章から構成される。背景では OOP の問題点として機能の実現が複数の箇所(クラス)に散逸し保守性が低下する点、これを AOP に変換するために必要なアスペクトの発見とその記述、検証が困難である点を述べ、プログラムの静的解析技術を用い作成者の負担を軽減したうえでこれらの問題を解決するというアプローチを述べている。成果は大きく２つあり、前半は OOP 中から共通モジュールとして分離可能な性質の発見(Aspect Mining)、後半は分離した性質を適用した結果として OOP を AOP へ再構成する手法(Aspect Refactoring)に関するものである。前半は複数の単位(クラス)に共通する性質を静的なプログラム指標(メトリクス)を測定することで抽出するが、従来はこの指標の選択が困難でありまた手作業で行っていたため効率、信頼性の点で問題があった。論文では QeA-SOM および QAHSSS という発見的手法を組み合わせることによって有効な指標を少ない計算量で導き出

すアルゴリズムを提唱し、実験によりその効果を確認している。また後半ではプログラムの共通モジュール(横断的関心事, Cross Cutting Concern)を適用する箇所を指定する際に、従来手法では変数や関数(メソッド)の名前により検索を実施していたために(Name-Based Pointcut)追加や変更に対して脆弱であった(fragile pointcut problem)。この問題に対してプログラム中の名前でなく特定の構造をキーとして検索と適用を可能にする新しい手法(Analysis-Based Pointcut)を採用し、構造を 6 個のプリミティブとその合成手法を定義している。またこれらの成果を統合して自動変換ツール Nataly を実装、実用規模(最大 3 万行)のオープンソースを対象に実験を行い、NBP が検索/適用に失敗する割合(最大約 20%)を ABP では最大 5%以下に抑制することに成功している。この成果は多くの審査員から高く評価された。また複数の CCC を適用する際 CCC の選び方によってはその適用順序によりプログラムの意味が変わってしまう問題(Aspect Interference)が指摘されているが、議論の章で本論文で提唱した静的解析に基づく手法と定理証明器を組み合わせることによって問題を起こす CCC の組み合わせを発見する手法について述べている。完成度は不十分であるもののこれも高く評価されている。

以上本論文は OOP から AOP への変換を(半)自動的に行う手法について理論的、実践的に述べたもので、ツールも実装されており学術的、工学的価値も高い。よって博士(情報科学)の学位論文としてふさわしいものであると認める。