論文内容の要旨

〔 題　　名 〕 ソースコードの静的解析によるソフトウェア保守支援に関する研究

学位申請者　　小　堀　一　雄

社会におけるソフトウェアの重要性が高まってきた現在では，特に社会基盤や企業の基幹業務を担う大規模かつ複雑なソフトウェアの品質を保つことが重要である．特に，ソフトウェア保守はソフトウェアの全ライフサイクルにかかるコストのうち，多くの割合を占めるため，ソフトウェア保守を支援することが重要になっている．

ソフトウェア保守を効率的に進めるためには，保守対象のプログラムの性質や振る舞いを開発者が理解する必要がある．しかし，ソフトウェアの大規模化や複雑化が進むにつれて，人手で十分な理解を行うことが難しくなっている．一方で，コンピュータの計算能力は近年著しく向上しており，コンピュータによるソフトウェア保守の支援を目的としたソースコード静的解析が盛んに研究されている．

本論文では，ソースコード静的解析手法のうち，以下の2つの手法を提案する．

1. アクセス修飾子過剰性に関する解析
2. ソフトウェア部品間における類似性計測

1. については，ソフトウェアを解析し，フィールド及びメソッドの呼び出し関係をグラフ化することで実際の呼び出し元の範囲と，当該フィールド及びメソッドに宣言されているアクセス修飾子の呼び出し範囲の乖離を分析する手法を提案し，この乖離を自動的に解析・修正するツールModiCheckerを開発した．これにより，開発者は意図せずアクセス修飾子を過剰に広く設定してしまったフィールドやメソッドを検知でき，第三者が誤ってアクセスすることを事前に防止できる．また，同じソフトウェアの複数のバージョン間における過剰なアクセス修飾子の数の変化量を比較分析することで，過剰なアクセス修飾子がどのように発生し，どのように修正されていくのかを分析した．

2. については，ソースコード部品におけるJava言語の予約語の出現回数に関するメトリクスと複雑性に関するメトリクスを計測し，2つのソフトウェア部品間でこれらのメトリクス値の比較を行うことで高速に類似部品を検出する手法を提案する．これにより，従来の文字列比較を用いた手法にくらべて解析時に扱う情報量が格段に低減されるため，解析コストを低く抑えることが可能となる．提案した手法は類似性計測ツールLuigiとして実現し，従来の文字列比較による類似分析を実装したツールSMMTと，今回開発したLuigiを用いて，同じソフトウェア群に対して類似分析を行い，類似測定に関する精度とコストを比較評価することで，提案手法の優位性を示した．

様式7

論文審査の結果の要旨及び担当者

氏　名　（　小　堀　一　雄　）		
	（職）	氏　名
論文審査担当者	主 査　教授	井上　克郎
	副 査　教授	萩原　兼一
	副 査　教授	楠本　真二
	副 査　准教授	松下　誠

論文審査の結果の要旨

本論文では，Javaのアクセス修飾子の冗長性と，ソフトウェア類似性に関するソフトウェア静的解析技術を扱っている．これらの技術は，ソフトウェアのライフサイクルにおいてコストの面で3分の2以上を占めるといわれる「ソフトウェア保守」の支援を目的としており，ソフトウェア保守を支援する様々な技術の中から，これらの技術を選択した理由として，ソフトウェア開発現場への導入容易性や期待される効果などを総合的に判断していることや，技術の重要性を論文中に示している．これは，自らの研究の立ち位置や選択理由などを客観的に説明する能力があることを示していると考える．

次に，Javaのアクセス修飾子の冗長性解析に関する研究では，Javaのメソッドとフィールドに宣言されているアクセス修飾子と，実際の呼び出し範囲の差異に注目し，これをアクセス修飾子に関する冗長性を表すメトリクス「AE（Accessibility Excessiveness）」として定義している．本論文では，AEが存在すると，設計上は呼び出されるはずのない範囲からメソッドやフィールドが直接呼び出されることが可能になり，潜在バグとなりうることを，例をあげて説明することで，解決したい課題を明確にしている．過去の類似研究では，この冗長性に関する直接的な対策を打っていない，もしくは部分的な解決しかしていないことを調査しており，研究の新規性についても記述がなされている．さらに，本論文ではAEを網羅的に解析し，修正を支援する手法にとどまらず，提案した手法を実現するためのツールModiCheckerを作成して２つの応用実験も行っている．

1つ目の応用実験は，ModiCheckerを7つのOSSに対して適用することで，一度，アクセス修飾子が設定されると，その後アクセス修飾子が変更されるケースは少なく，用途が変わる場合は，フィールド/メソッド自体を削除して作成しなおす傾向にあることを考察している．

2つ目の応用実験では，ModiCheckerを利用すべきタイミングを示すために，OSSであるAntの22種類のバージョンアップにおけるAEの増減を調査した．その結果，多くのAEにおいてメジャーバージョンアップ時のほうがマイナーバージョンアップ時に比べて多くのAE変化量が見られることが分かった．これにより，ModiCheckerを用いてAE解析を行うのに適したタイミングはメジャーバージョンアップであることを示している．

これらのことから，単に手法を提案するだけではなくツール化し応用実験を行うことで，導入を容易とし，また手法の適用対象に関する有用性の高い考察を導き出すことができていると考える．

また，ソフトウェア類似性に関する研究においては，既存の文字列比較を用いた手法に対して大幅な高速化を目的として類似度メトリクスを複数提案し組み合わせることにより，数値比較のみで類似性を比較する手法を提案している．さらに，類似度メトリクスの一部をハッシュ値に変換してデータベースに保存しておくことで，新規の類似比較を行う際にハッシュ値が合わない部品を除外することで比較回数も削減する工夫を取り入れている．その結果，既存手法に比べ，150倍の高速化を実現した．

これらの研究成果は，ソフトウェア工学分野において有用なものであり，新規性をもつものである．よって，博士（情報科学）の学位論文として価値のあるものと認める．