

論文の内容の要旨

論文題目 プログラムの依存関係分析による遺産ソフトウェアの解析とアスペクト指向プログラムへの展開に関する研究

氏名 四野見 秀明

1 研究の背景と目的

ソフトウェアは開発された後に修正が行われなければならないという事はありえない。バグの修正、機能拡張、パフォーマンス改善、保守性向上のためのリファクタリング、プラットフォーム移行などのために、必ずプログラムの修正が行われる。修正のためのプログラムの理解や影響把握の生産性向上がライフサイクル全体のコスト削減に大きく寄与する。修正ではなくシステム自体を別のプラットフォーム上で作り直す場合も、既存システムを理解する必要がある。その際にもプログラムの理解が必要である。

プログラムの理解となると仕様書が役立つと考えられるが、仕様書は度重なるプログラムの修正で、プログラムと仕様書は乖離していたり、必要となる情報が仕様書に残されていない場合も多い。プログラマが、プログラムを正確に把握するためには、プログラム自体を解析する必要がある。

プログラム解析に有効な技術としてプログラムスライシング技術がある。プログラムスライシング技術は、その手法やアルゴリズムについての研究は報告されているが、実世界のプログラムに適用した例は多くない。また、新しい方法論であるアスペクト指向プログラミングにおいての適用は研究が始まったばかりである。本研究では、新旧プログラムに対するプログラムスライシング技術を適用するためのプログラム解析モデルの考案と、その応用によるプログラム理解による再構築支援や、修正に伴う影響分析の方式について提案した。対象とするプログラムは、COBOL、PL/I による遺産ソフトウェアから、オブジェクト指向言語 Java に対して近年注目をされているアスペクト指向の機能拡張をした AspectJ によって書かれたものまでを対象にしている。

2 プログラムスライシング技術とプログラム保守 / 理解

Weiser によるプログラムスライス概念の提案後、Horwitz らによってスライシングを効率良く計算するためにプログラム依存グラフを利用するアルゴリズムが提案された。

その後は、プログラム依存グラフを構成する文の間のデータ依存関係と制御依存関係を利用して文の部分集合を計算することが、主にプログラムスライシング技術の適用と呼ばれている。テスト、デバッグ、保守などの適用例がある。

スライスには、プログラムの実行を伴わない静的スライスと、実行による情報を使う動的スライスがある。また、プログラムの制御の流れにそったフォワードスライスとバックワードスライスがある。本研究では、プログラム理解や影響分析を狙いとすることから、それに適した静的スライスを対象とする。スライスの方向としてはフォワード、バックワードの両スライスを実現できるように考慮している。

3 プログラム依存関係分析モデルと遺産プログラムへの適用

COBOL や PL/I を対象として、プログラムスライシング技術におけるデータ / 制御依存関係の解析結果を抽出できるプログラム解析モデルを構築し、API として設計・実現した。遺産プログラムの多くは、ソフトウェア工学的には望ましくないコーディングが行われている。それにより正確なデータフローが計算できないために、正確なスライシングが行われていなかった。本解析モデルと API を適用すれば、そのようなプログラムにおいても正確なスライシングを実現できる。

その API の適用例として、データ / 制御依存関係に基づく解析結果を帳票化したツールを実現した。そのツールは、データ項目の使用法を一貫性を持たせるためのシステム再構築の現場でプログラムの理解に実際に利用された。また、実現されたツールは実装済みのビジネスロジック（例、保険の約款）の検証に役立つことも分かった。

既存の COBOL や PL/I で書かれたシステムを Web 化、J2EE 化するなどの必要性というレガシー現代化が注目されている現在、遺産プログラムの理解のための正確なプログラム解析は重要である。

4 アスペクト指向への展開と織り込みによる影響波及解析への適用

プログラム解析モデルを、アスペクト指向言語 AspectJ で書かれたプログラムに適用するための拡張も行った。

アスペクト指向に基づくプログラムは、アスペクトとそれが織り込まれるベースプログラムとから成る。アスペクト指向における織り込みは、ベースプログラムに対するアスペクトによるある種の「修正」と見なすことができる。ベースプログラムが既にある程度安定して稼動している場合、そのアスペクトの織り込みによりベースプログラム中のどの範囲まで影響があるかを把握することが有効である。たとえば、織り込みによりバグが挿入された場合の問題解決、織り込み後の回帰テストの必要な範囲の把握などに役立つ。

アスペクト指向に対応したプログラム依存グラフに基づくスライシングをおこなうこと

により、織り込みの影響が、ベースプログラムに与える影響の範囲を解析する手法を提案した。その影響分析の結果、準備されていたテストケースの中で、どの範囲までテストすべきかを限定できることを示した。また、実現したツールを公開されている AspectJ プログラムに適用し、その有効性を検証した。この織り込みによる影響分析のツールは、2006 年 3 月に開催される Fifth International Conference on Aspect-Oriented Software Development (AOSD.06) へのデモンストレーション・プロポーザルが受理され、デモを行うことが決定している。

アスペクト指向に対応したプログラム依存グラフとその上でのスライシングを提供する部分は、Eclipse のプラグインとして実装され、Jade と名付けられた。Jade は、プログラムスライシング技術を利用するツールの実現を容易にするプラグインとして、オープンソースで公開予定である。

5 結論と今後の研究課題

前時代的なコーディングがなされているために、正確なスライスが計算できなかった COBOL と PL/I による遺産プログラムが、本研究によるプログラム解析モデルにより正確なスライスが実現できるようになった。そのモデルを実現した API を利用したツールの有効性は、システム再構築の現場で確認された。

アスペクト指向プログラム上でのプログラムスライシングを実現するためのモデルが提案された。それによりアスペクト指向における織り込みによる影響の分析が可能になった。AspectJ において Jade に相当するツールは、今のところ他にない。また、織りこみによってベースプログラムに与える影響を解析するという発想に基づくツールも同様のものは他にない。

今後の研究課題として、AspectJ と AspectCobol の解析モデルの共通化の可能性や COBOL プログラムへのアスペクト織り込みを利用した動的解析の可能性がある。また、AspectJ に関しては、今後公開することが予想される現実的な AspectJ プログラムに本研究による織り込みによる影響分析ツールを適用し、アスペクトの使用法の傾向を統計的に調査するという可能性があるだろう。

Jade により、Java と AspectJ のプログラムに対してスライシング技術を利用したツールを実現するための道具が提供された。それを利用して、既存プログラムからアスペクトを抽出するアスペクト・マイニングや、形式的仕様記述との組み合わせでプログラム自体の検証を行なう研究などが今後考えられる。