バイトコードによる Java servlet プログラムの解析

野口 建仁† 松井 藤五郎‡ 大和田 勇人‡
†東京理科大学大学院理工学研究科経営工学専攻 ‡東京理科大学理工学部経営工学科

1 序論

1.1 背景

楽天や Amazon で知られるショッピングサイトなどの Web システムは、ユーザーが安心して使える高い信頼性を確保することが容易ではない。困難な点として Java servlet、JSP、PHP 等、複数の開発言語で構成されるシステム全体の整合性把握やユーザーから共感を得るための Web サイトデザイン構築に伴う、頻繁に変わるコンテンツやサイト構造の正確な把握が挙げられる。

この課題の解決策となる解析ツールとして、NRI の Ridual や富士通の Interstage が存在する。ただ、これらのような静的プログラム解析に基づく解析は Java servlet などの実行時決定要素を含むような Web システムの解析を十分に行えないため、動的なプログラム解析手法を用いた Web システムの解析が必要となる。

1.2 本論文の目的

本論文の目的は、既存研究では十分に解析を行うことができない Java servlet を用いて実現された Web システムに対して、解析網羅率を保つ静的なプログラム解析手法と、解析結果精度の高い動的プログラム解析手法を適用することにより Web システムの解析を行うことである。

また本論文においては Soot[2] を用いて変換した、Web システム中における Java を解析するにはもっとも相性がよいと考えられる Java バイトコード [1] による解析手法を提案する。

2 提案手法

本システムの構造を図 1 に示す。本システムの入力は、Java servlet プログラムをコンパイルして生成された Java プログラム実行形式の class ファイルである。静的、動的プログラム解析による解析結果からサイト構造およびデータフローを把握するために、MySQL サーバにこれら解析結果をそれぞれ整理して格納し、対象システムのデータフローを視覚化した HTML を出力する。

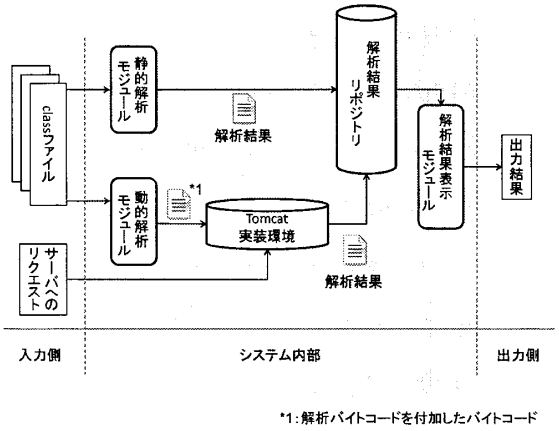

図 1: システムの構造

2.1 静的プログラム解析手法

静的プログラム解析手法により、実際にプログラムを動かすことなくソースコード上からこれから起こりうるバグを発見する。また静的プログラム解析を行うことにより、プログラム解析の網羅率を高く維持することを目的としている。静的プログラム解析手法の方針は以下に示す通りである。

1. HttpServletRequest、HttpServletResponse、HttpSession に定義されている全 11 種類のメソッドの引数に指定される値及び変数の解析を行う
2. 引数が変数である場合、バイトコード中のアサーション処理の解析を行う
3. 開発者が定義したパッケージ内のクラスの参照が発生する場合、該当するクラスに関しても 1. と同様の解析を行う

この手順を全てのプログラムに行うことにより静的プログラム解析を達成する。

2.2 動的プログラム解析手法

動的プログラム解析手法は静的プログラム解析手法と違い、実際にプログラムを動かして初めてわかるバグを発見する。動的プログラム解析を行うことにより、プログラム解析の細粒度を向上させることを目的とする。

Dataflow analysis between Java servlet programs by bytecode
†Takehito Noguchi ‡Tohgoroh Matsui ‡Hayato Ohwada
†Department of Industrial Administration, Graduate School of Science and Technology, Tokyo University of Science
‡Department of Industrial Administration, Faculty of Science and Technology, Tokyo University of Science

```
1   getParameterValues_args_value
2   "shop","ultra"
3   setAttribute_args
4   "shopName","ultra"
5   getParameterValues_args_value
6   "com","0"
7   getParameterValues_args_value
8   "cat","0"
9   getParameterValues_args_value
10  "categoryID","12"
```

図 2: 動的プログラム解析結果例

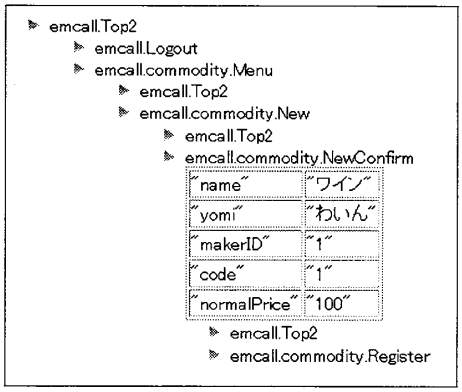

図 3: 本システムの出力結果

動的プログラム解析手法では静的プログラム解析手法と同様に HttpServletRequest、HttpServletResponse、HttpSession クラスに定義されている全 11 種類のメソッドの返値を取得する。ここで動的プログラム解析では、Getter クラスというデータ形式を変換する機能とファイルストリームへの入出力機能を持つクラスを使用し、バイトコードの挿入を行い、プログラム内に定義されるデータを出力する。

3 実験

3.1 実験データ

実験対象として、クラス数 4341、総プログラム行数 60 万からなる大規模な商用 Web システムを解析した。より詳細な実験データに関する情報は以下に示す通りである。

実験データ	
クラス数	4341
プログラム行数	605162
パッケージ数	122
クラスファイルの総バイト数	24.9[Mbyte]

3.2 実験例

図 2 は動的プログラム解析結果ファイルの中身である。静的プログラム解析結果と動的プログラム解析結果は同じ形式で出力されるため、ここでは動的プログラム解析の結果を示す。図 2 のデータにおいて奇数が、解析したメソッド名を示し、メソッド名の次の行、すなわち偶数行がメソッドの引数の値となる。1 行目の getParameterValues メソッドのような引数を二つ以上とるメソッドの場合はカンマ (,) を区切り文字として第一引数から順に引数の値が解析結果として出力される。動的プログラム解析の結果では実際に実行時決定要素である引数の値を取得できているのがわかる。

図 3 に示したのは今回対象とした Web システムのデータフローを含め解析した結果の一部である。本システムでは、対象システムの全体像に加えて各プログラム間のデータフローを見ることができる。

4 結論

本論文では、既存研究では難しかった Java serlvet に対するプログラム解析を行った。解析に当たり提案したことは三つである。一つ目に Java バイトコードを利用したアプローチにより解析を行う。二つ目に静的プログラム解析手法、動的プログラム解析手法両方を併せて行う解析手法を提案した。三つ目にこの解析手法により、システムのバグ候補となるプログラム欠如を発見できる。本システムにより、約 60 万行も有するプログラムに対して一つ一つ目を通すことなく視覚的にデータフローを知ることができ、バグも発見した。

参考文献

[1] Raja Valle-Rai and Laurie J.Hendren. BIT: Jimple: Simplifying java bytecode for analyses and transformations. *In Sable Technical Report 1998-4. Sable Reserch Group, McGill University, 1998.*

[2] Raja Valle-Rai, Laurie Hendren, Vijay Sundaresan, Patrick Lam, Etienne Gagnon, and Phong Co. Soot - a java optimization framework. *In Proceedings of CASCON 1999, pp.125-135, 1999*

[3] 福安直樹, 山本晋一郎, 阿草清滋. 細粒度リポジトリーに基づいた case ツール・プラットフォーム sapid. 情報処理学会論文誌, 第 *39* 巻, *pp1990–1998, 1998.*