外部から見た振る舞いとアクセス手段に注目したソースコード類似性

佐々木　裕介[†]　石尾　隆[†]　山本哲男[††]

本稿ではソースコード類似性に関するアンケートを作成する上で，どのような意図で設問を用意したかについて言及している．我々は再利用促進を目的としたソフトウェア部品検索システムの研究を行っており，再利用という観点でソースコード類似性を判断する際には，外部から見たときのアクセス手段と振る舞いに注目している．

1. はじめに

開発者のソフトウェア部品の再利用を促進するために，ソフトウェア部品検索システムが研究されている[1)]．検索対象となるソフトウェア部品の収集時には，プロジェクト間で複製され再利用されたもの，同一部品の異なるバージョンなども蓄積されていく．このため，検索結果の上位をそのような類似部品群が占めてしまい，利用者が部品を探す作業の効率が悪化することがある．

そこで我々は，ソフトウェア部品検索システムにおいてこの冗長性を解消するために，再利用性の観点から類似コードをグループ化して表示するための研究を行っている．類似判断の基準としては，あるコードと同じグループに属する別のコードを入れ替えたとき，同じように利用可能であることを類似条件としている．また今後の展望として，単なる再利用以外の用途，例えば1つのコードにまとめることが可能なコード群の管理，管理しているコードのバグ検出などにもソフトウェア部品検索の技術を応用できないかと考えているため，アンケートでも様々な観点から考察した回答を求めている．

2. アンケート回答

第一著者の回答の結果は表1のようになった．各観点において何を基準にして類似・非類似を判断したかは以下に記述する．

観　点　A

1つにまとめるのが容易であるコード片の組のみ類似であるとみなしている．1つにまとめるのが容易であるとは即ち，共通する記述を抽出したときに外部から見た振る舞いが変わることなく，かつ2者の実装方法（importしているクラスも含む）も類似している場合を指す．ただし定数や引数の型など一部のパラメータのみ異なるものについては，テンプレートをスーパークラスとして作成することで1つのクラスへまとめることが可能と考え，類似とみなしている．

観　点　B

同じ構成の記述が多いほど同じバグを含んでいる可能性が高いと考え，類似とみなすことにした．同じ構成の記述が多いとは即ち，差分として認識できるトークン数が全体のトークン数と比べて少ないことを指す．

観　点　C

置き換えが可能であるためには，外部から見た振る舞いと，外部からのアクセス手段が同じでなければならないと考え，これら2つを基準とした．振る舞いについてはパフォーマンスの違いは考慮せず，メソッドの実行が終わったときに外部から見たときのクラスの状態が同じであれば類似と判断する．

観　点　D

観点Dについては，具体的なシナリオとして類似性を用途別に分類して管理することを想定し，他の観点A-Cにおいて一つでも類似であると判断されている場合は類似とみなしている．例えばクラスX,Yが観点Aにおいて類似とみなされていれば，X,Yが類似であることと観点Aにおいて類似判断したこと両方を記録する．

3. 各設問の意図

3.1 ソース No.1 について

クラス名が異なると外部からアクセスされるときのプロトコルも変わるため，クラス名のみ異なるコード

† 大阪大学
Osaka University
†† 立命館大学
Ritsumeikan University

表 1 設問への回答

ソース No.	A	B	C	D	X
1	yes	yes	yes	yes	-
2	no	yes	no	yes	-
3	no	yes	no	yes	-
4	no	yes	no	yes	-
5	no	yes	yes	yes	-
6	no	yes	no	yes	-
7	yes	yes	yes	yes	-
8	no	no	yes	yes	-
9	yes	yes	no	yes	-
10	yes	yes	yes	yes	-
11	no	no	no	no	-

片の組を取り上げた．ただし名前の違いは極めて表面的なものであり，クラス自体の振る舞いには影響を与えない．

3.2 ソース No.2 について

提示したコード片の組は，メソッド changeIndent のアクセス修飾子以外は同じである．しかしアクセス修飾子の異なるメソッドを単純に 1 つにまとめると，外部からアクセスする場合のクラスの振る舞いが元のものと別のものに変わってしまうので，回答者の観点によって意見が分かれると考えられる．

3.3 ソース No.3 について

インタフェースでは具体的な実装はほとんど行っていない．しかし，引数の型の間違いなどはクラスと同様起こりうる．また，インタフェースの定義が異なると外部からアクセスする方法も異なるため，回答者によっては意見が異なると考えられる．

3.4 ソース No.4 について

提示した例では，IPv4 を処理するコードの代わりに IPv6 を処理するコードを用いることは可能だが，その逆は不可能である．よって観点 C からこれらのコード片の類似性を考察する場合，類似であるとみなすかどうかは類似の判断結果を利用する状況に依存する．例えばグループ化により類似コードを分類する際に，No.4 のコード片の組を同じグループに含めた場合，そのグループ内のコードは必ずしも置き換え可能ではないということになる．

3.5 ソース No.5 について

外部から見たときに最終的に果たされる役割が同じであるという点では，2 者の入れ替えは可能であると考えられる．しかし利用しているライブラリの実装の差異のため，性能は異なったものになる．

3.6 ソース No.6 について

No.6 のコード片の違いはフィールド states のインタフェースが異なることである．そのため非類似とする理由の一つとして，外部オブジェクトが states へ代入できるオブジェクトの範囲が異なることが挙げられる．またコードを単なるトークンの系列とみなしたとき，異なるトークンの数自体は少量とみなせるが，全体として異なる割合，つまり (差分トークン数) / (総トークン数) は比較的大きい数値になる．

3.7 ソース No.7 について

処理の抽出のように，クラスの振る舞いを変えない簡単なリファクタリングであっても，コードをトークン系列とみなして違いを定量的に計算する場合には，この差が大きな違いとして検出される可能性がある．

3.8 ソース No.8 について

コードの記述方法やパフォーマンスに注目して類似判定を行う場合には，アルゴリズムの違いは大きな差と考えられる．一方でコードの実行結果や振る舞いに注目する場合には，アルゴリズムの違いは類似判定に大した影響を与えない．

3.9 ソース No.9 について

提示したコード片のように，記述の違いが型とメソッド及びフィールドのパラメータのみであっても，クラスの機能が全く異なるケースが存在する．

3.10 ソース No.10 について

No.10 はコードをトークン系列とみなしたとき，差分として検出されるトークンの割合はトークン総数と比較すると少ないが，トークンの数そのもの（絶対数）が多い例である．

3.11 ソース No.11 について

No.11 はコードをトークン系列とみなしたとき，差分として検出されるトークンの割合はトークン総数と比較すると多いが，トークンの数そのもの（絶対数）が少ない例である．

トークンの差分を用いて類似判定を行う際には，No.10 のようなケースと No.11 のようなケース両方を考慮する必要がある．

4. 謝　　辞

本研究は，日本学術振興会科学研究費補助金基盤研究（A）（課題番号:21240002）の助成を得た．

参 考 文 献

1) 横森 励士, 梅森 文彰, 西 秀雄, 山本 哲男, 松下 誠, 楠本 真二, 井上 克郎.: “ Java ソフトウェア部品検索システム SPARS-J ”, 電子情報通信学会論文誌 D-I, VolJ87-D-I, No.12, pp1060-1068, 2004.