

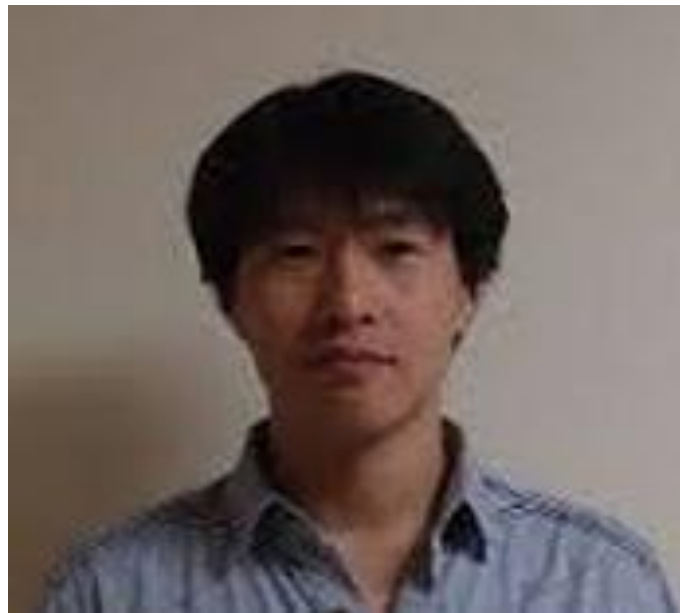
プログラム依存グラフを用いた
Template Method パターン適用による
コードクローン集約支援

s161053
前田欣耶



堀田圭佑

大阪大学 楠本研究室 研究員



肥後芳樹

大阪大学 情報科学研究科
助教授



楠本真二

大阪大学 情報科学研究科 教授

目次

- 1、背景
- 2、提案方法
- 3、実装
- 4、評価

1、背景

コードクローンはソフトウェアの保守性を低下させる要因の一つである。
Template Methodパターン適用によるコードクローン集約方法は高い効果が得られるが、課題
その課題点をプログラム依存グラフを用いて改善した。

コードクローン

ソースコード中の同一、あるいは類似するコード片。

主にコピーアンドペーストで発生し、その存在はソフトウェアの保守を困難にする。

コードクローンを自動的に検出あるいは集約を支援する方法が多数提案されている。

Template Methodパターン

- ・ 共通の処理を親クラスで定義
- ・ 異なる処理を subclasses で定義

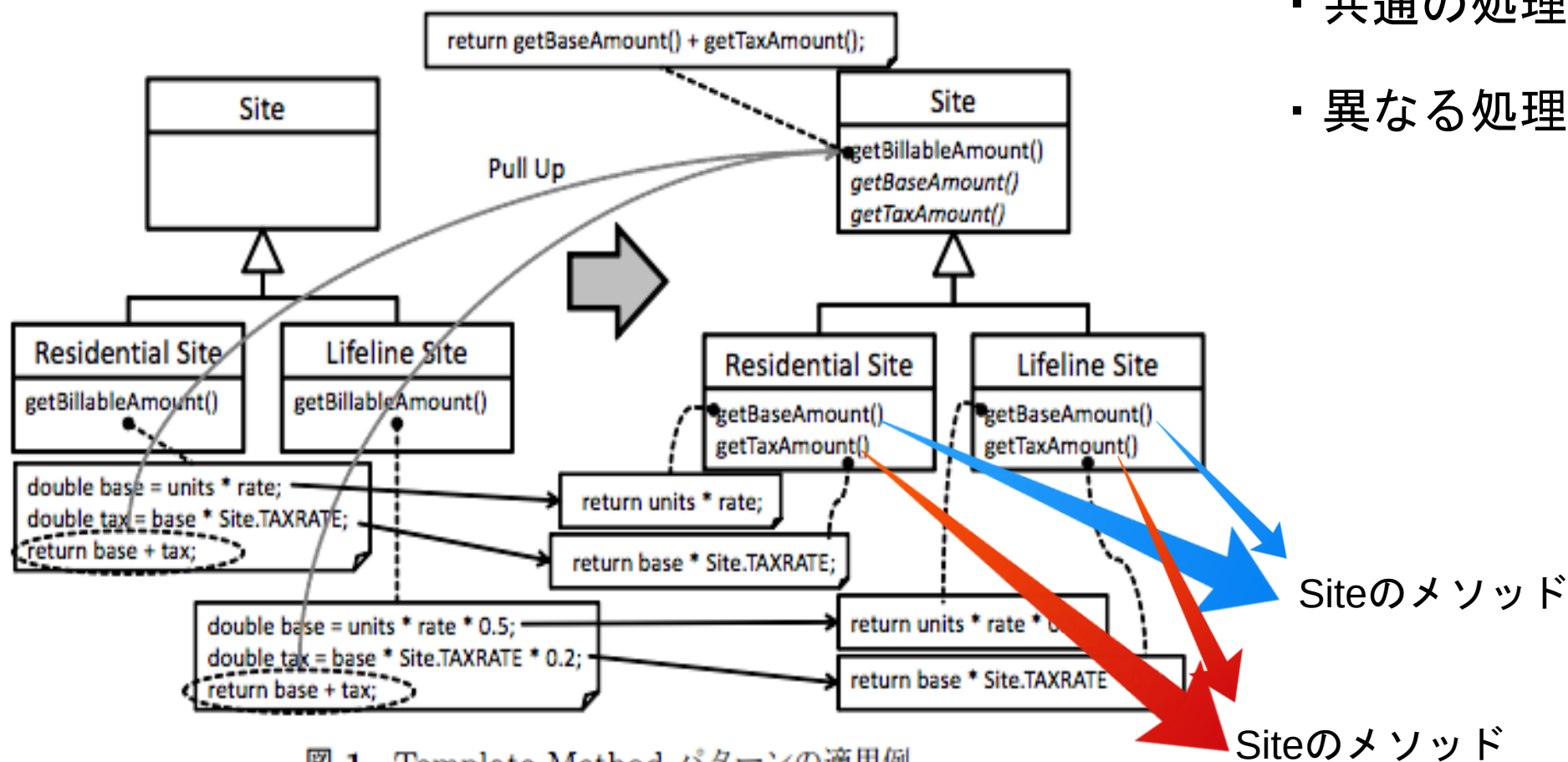


図 1 Template Method パターンの適用例

Fig. 1 An Example of the Application of Template Method Pattern

(Program Dependence Graph(PDG))

```
1: x = 0;  
2: y = 0;  
3: z = MAX;  
4: while (y < z) {  
5:   y = x + 1;  
6: }  
7: println(y);
```

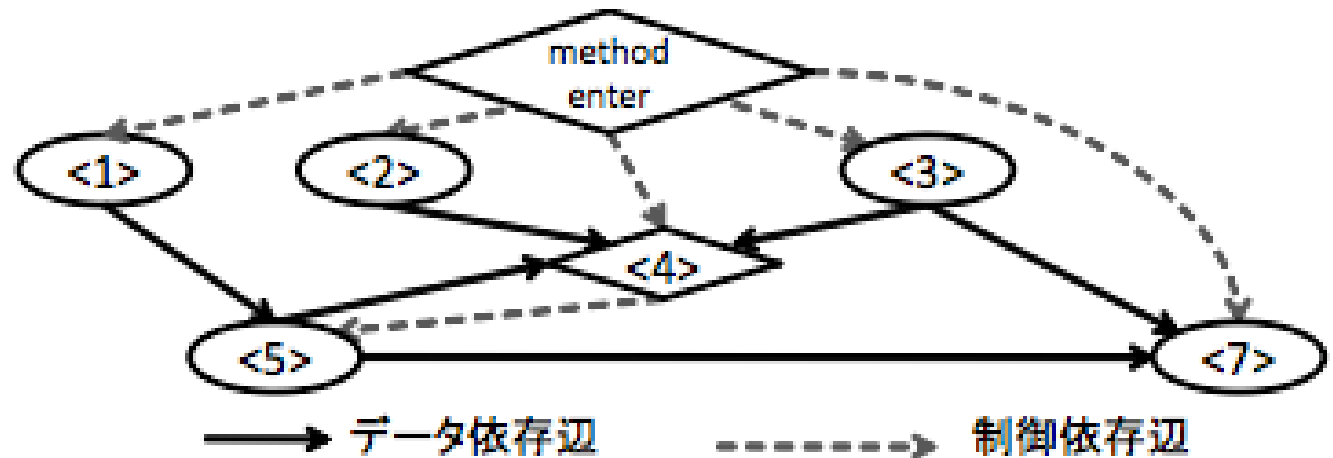


図 2 PDG の例

Fig. 2 An Example of PDG

- ・ 4行目で変数y,zの値が参照されるため、2,3,5行目から4行目
- ・ 5行目は4行目によって実行するかしないか決まるため、4行

2、提案方法

概要

入力

対象となるメソッドの対、各メソッドのPDG、メソッド間のコードクローン情報。

出力

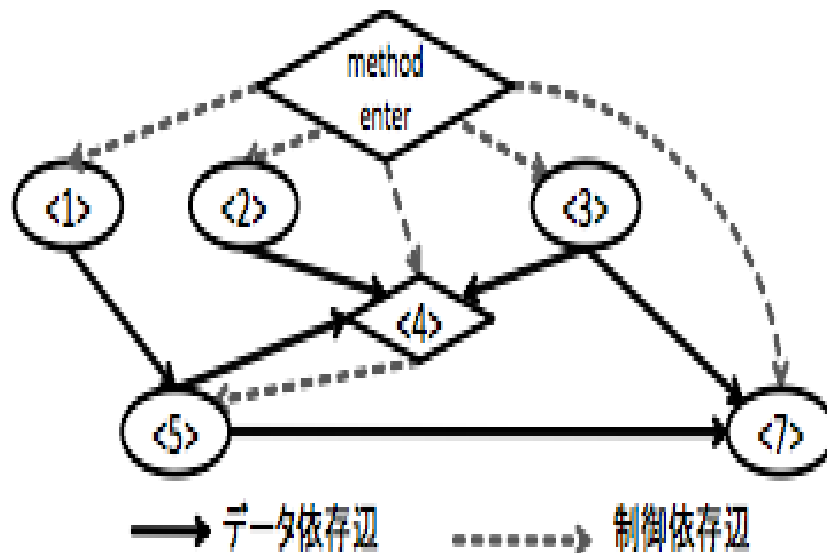
各メソッドの親クラスに引き上げる頂点集合、子クラスに残すべき頂点集合(一つのメソッド

用語の定義

定義1

メソッドm0,m1が同じクラスに実装されていなく、m0を含むクラスの先祖のクラスたちがm1

定義2



BackwardEdges(n)...nを到達頂点とする

ForwardEdges(n)...nを出発頂点とする

fromnode(e)...ある辺eについてeの出発頂点

tonode(e)...ある辺eについてeの到達頂点

定義 3

$\text{Nodes}(m)$...メソッド m の頂点集合、 $\text{Edges}(N)$...頂点集合 N の辺集合。

定義 4

$\text{ClonePairs}(m_0, m_1)$...対象メソッドペア (m_0, m_1) 間の:

定義 5

対象メソッドペアの共通頂点集合 CommonNodePairs

対象メソッドペア (m_0, m_1) の各メソッドについてその

$$\text{CommonNodePairs}(m_0, m_1) := \{(s, t) : \exists cp(N_0, N_1) \in \text{ClonePairs}(m_0, m_1) \\ ((s \in N_0) \wedge (t \in N_1))\}$$

$$\text{CommonNodes}(m_i) := \{n \in \text{Nodes}(m_i) : (\exists (s, t) \in \text{CommonNodePairs} \\ (m_0, m_1) ((s = n) \vee (t = n)))\}$$

定義 6

対象メソッドペア(m0,m1)の各メソッドについて、そ

$$DiffNodes(m_i) := \{n \in Nodes(m_i) : n \notin CommonNodes(m_i)\}$$

この中に含まれるNode(m0)のみ集合、
あるいはNode(m1)の集合

Node(m0)

Node(m1)

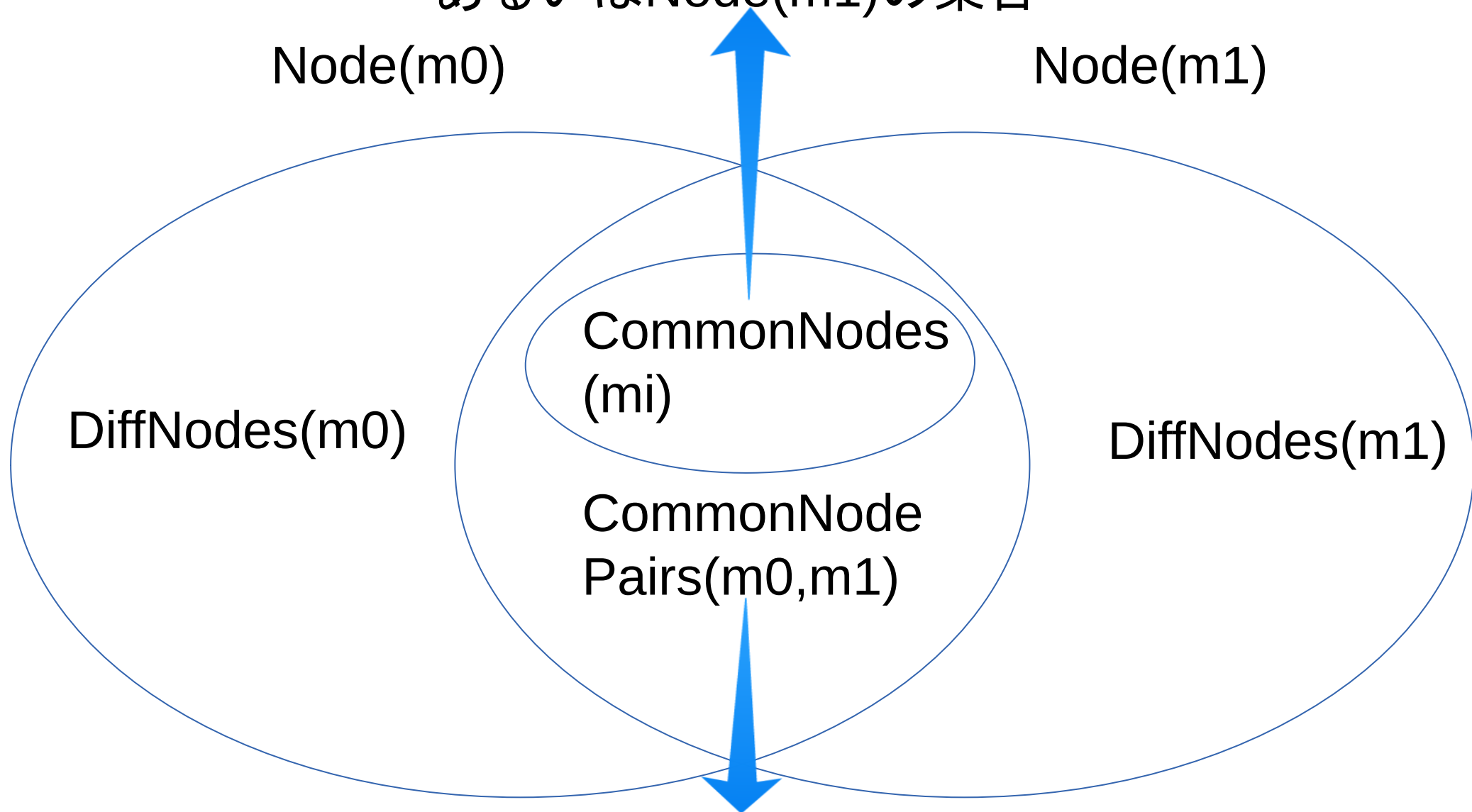
DiffNodes(m0)

CommonNodes
(mi)

DiffNodes(m1)

CommonNode
Pairs(m0,m1)

ClonePairsに含まれる部分(コードの一部)



処理

Step1:必要条件の解決...Template Methodには満たさなければならない必要条件がある。

その内容

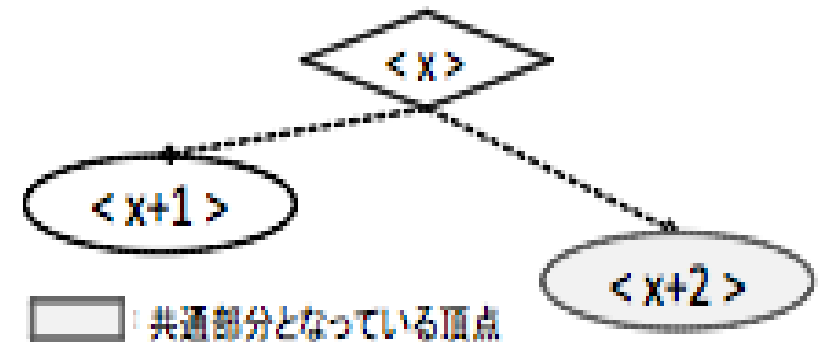
条件 1:差異頂点集合がreturn文を含まない。

return文を含む差異頂点集合が新規メソッドとして抽出された場合、そのメソッドと親クラス

条件2: 差異頂点集合からメソッド内共通頂点集合への制御依存が存在しない。

こ
た
x : if (this.getVerbose()) {
x+1: long endTime = System.currentTimeMillis();
x+2: logStats(startTime, endTime, (int) totalLength);
x+3: }

 : 共通部分となっているコード



を新規

図 3 差異部分から共通部分への制御依存

.Step2:抽出頂点集合、抽出頂点集合群の特定

.対象メソッドペア(m0,m1)の各メソッドについて、差異頂点集合から一つの新規メソッドとし

.Step3:抽出頂点集合同士の対応付け

.DiffGroup(m0)とDiffGroup(m1)で各要素同士で対応とり抽出頂点集合ペアDiffGroupPairs(m0,m

3、実装

- javaを用いて(MASU,Scorpioはjavaのみ対応)実装。PDG情報とコードクローン情報の特定はScorpio
- javaに特化した条件
- 条件1:戻り値を一つしか指定できないので、特別な処理をかまして、複数の戻り値を指定して返す
- 条件2:抽出頂点集合が条件式を含まないブロック文(try文など)をまたぐ。

候補の提示

- ・ソースコード表示では、共通部分をハイライト表示し、差異部分については一つのメソッド内
- ・対象メソッドペアにメソッド間の類似度などのメトリクスをいくつか用意し、そのメトリクス

4、評価

Apache Antに対して適用実験を行った。

適用
メソ

表 1 実験対象と所要時間
Table 1 Target Software System and Execution Time

行数	ファイル数	所要時間と実験環境
212,401	829	110s (CPU: Xeon 2.67GHz(4 core), メモリ: 4GB)

設計

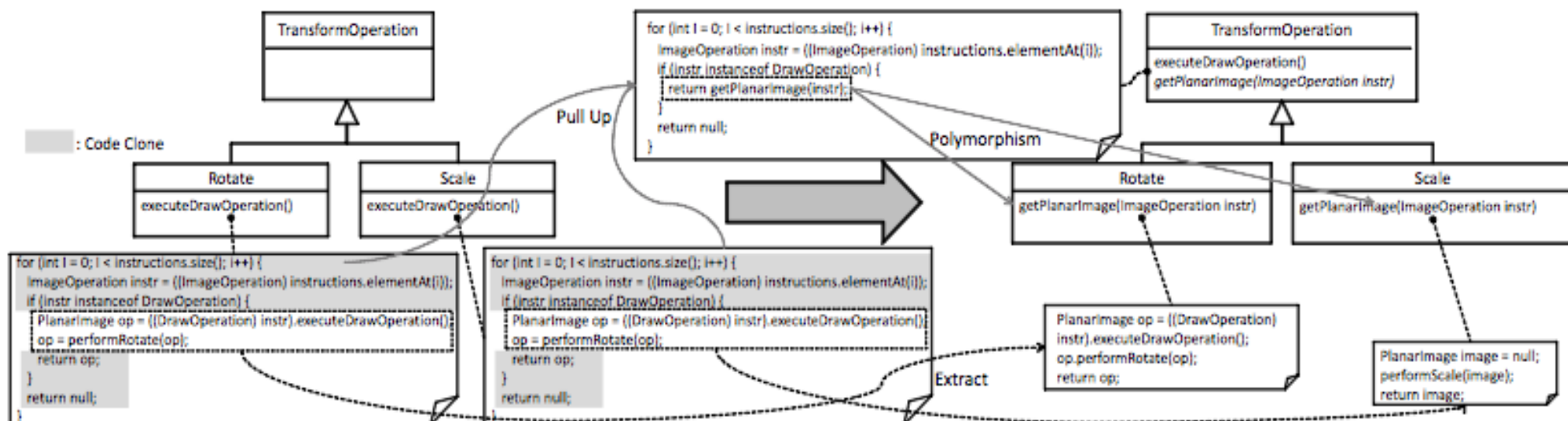


图 5 適用例

Fig. 5 An Example of Application of Template Method Pattern with Proposed Method