# コードクローンの分類に基づくリファクタリングパターン集の提案に向けて

徳永将之[†1] 吉田則裕[†2] 吉岡一樹[†1]
松下誠[†1] 井上克郎[†1]

リファクタリングはコードクローンを取り除く代表的な手法である．我々は，コードクローンのリファクタリング作業を支援することを目的とし，詳細なコードクローンの分類に基づいたコードクローンのリファクタリングパターン集の作成を計画している．本稿では，我々の目指すリファクタリングパターン集の作成方法について述べる．

# Towards Collection of Refactoring Patterns Based on Code Clone Classification

MASAYUKI TOKUNAGA,[†1] NORIHIRO YOSHIDA,[†2]
KAZUKI YOSHIOKA,[†1] MAKOTO MATSUSHITA[†1]
and KATSURO INOUE [†1]

Refactoring is a well-known way to remove code clones. Currently, we are planning to write a collection of refactoring patterns for removing code clone based on code clone classification. In this paper, we describe a method to develop our intended collection of refactoring patterns.

## 1. はじめに

コードクローンとは，互いに一致または類似したコード片を指し，プログラムの保守作業を困難にする一因として指摘されている[1)]．コードクローン関係にある二つのコード片（クローンペア）およびコード片の集合（クローンセット）は，リファクタリングにより一つにまとめることで取り除くことが出来る．リファクタリングする際には，リファクタリングの適用方法をまとめたリファクタリングパターンを利用することが多い．しかし，既存の書籍では，コードクローンの特徴が詳細に分類されておらず，パターンの適用条件およびリファクタリング作業の記述が不十分であるため，コードクローンのリファクタリング作業は困難なものとなっている．そこで我々は，コードクローンの詳細な分類の作成およびそれに基づいたリファクタリングパターン集の提案を計画している．本提案によって，経験の少ない作業者には困難なリファクタリング作業を支援し，ソフトウェア保守の効率向上を目指す．本稿では，我々の目指すリファクタリングパターン集の作成方法について述べる．

†1 大阪大学
Osaka University
†2 奈良先端科学技術大学院大学 Nara Institute of Science and Technology

## 2. Fowler のリファクタリングパターン

コードクローンのリファクタリングをする際に参照される代表的なパターンとして，Fowler の提案するものが挙げられる[2)]．Fowler はコードクローンの特徴を，クローンの位置関係とコードの差異に関して大まかに 5 つに分類している．

- 同一クラス内にある場合
- 兄弟クラス内にあり，完全に一致する場合
- 兄弟クラス内にあり，似通っている場合
- 兄弟クラス内にあり，異なるアルゴリズムの場合
- 関係のないクラス内にある場合

Fowler は上記の 5 分類に対して適用可能なリファクタリングパターンを提案している．しかし，5 種類という分類数では，リファクタリングパターンを選択するための条件およびリファクタリング作業の手順が曖昧なものとなり，作業者の判断に依存する部分が多くなる．そのため，リファクタリング作業経験の少ない者にとって，その作業は困難なものとなる．

## 3. 提案するパターンの作成手法

本節では，提案するリファクタリングパターンの作成手法に関して述べる．まず，コードクローンの分類

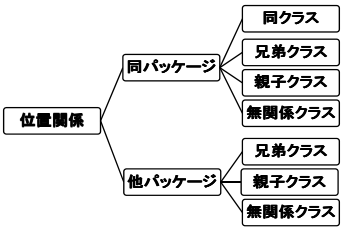


**図 1** 分類観点「クローンペアの位置関係」に関する分類木
Fig. 1 Classification tree for the classification viewpoint on "position relations between a clone pair"

| 要約 | |
|---|---|
| 分類 | クローンペアの差異 |
| | クローンペアの位置関係 |
| | クローン部の長さ |
| | 引数となるオブジェクトの種類 |
| | 戻り値の数 |
| | 制御構造要素の有無 |
| | instanceof の有無 |
| 手順 | |
| 実例 | |

**図 2** リファクタリングパターン記述項目
Fig. 2 Description framework of the proposed refactoring pattern

観点を定義する．次に，定義した分類観点毎に分類木を作成する．最後に，各分類木の分類の組合せに対してリファクタリングパターンを記述する．

### 3.1 分類観点の定義

コードクローンを分類する観点（以降，分類観点）を定義する．分類観点は，開発者がリファクタリング対象のコードクローンに適用するパターンの選択に用いる．本稿では，以下の 7 つの分類観点がパターンの選択に必要であると考える．

- クローンペアの差異
- クローンペアの位置関係
- クローン部の長さ
- メソッド抽出の際に必要な引数オブジェクトの種類
- メソッド抽出の際に必要な戻り値の数
- 制御構造要素の有無
- instanceof の有無

### 3.2 分類木の作成

定義した分類観点毎の分類を表す分類木を作成する．そして，各分類木の分類の組合せは，一つのコードクローンの状態を表す．分類木を思索して作成する場合，頻出するコードクローンの特徴を特定する事が難しいため，実際のコードを参考に作成する．作成方法は，まず，CCFinder を利用して Java 言語で記述された 11 種類のオープンソースからクローンセット群を検出する．次に，検出されたクローンセット群から一定数[⋆1]のクローンセットをランダムで抽出する．最後に，抽出したクローンセットの特徴を基に各分類木を作成する．図 1 に分類木の例を示す．

### 3.3 リファクタリングパターンの作成

作成した各分類木毎の分類の組合せに対して，コードクローンの特徴とリファクタリング方法をまとめたリファクタリングパターンを作成する．リファクタリングパターンの記述は，リファクタリング作業の要約，コードクローンの特徴，作業手順および実例を記述する (図 2)．また，頻出するコードクローンの特徴からリファクタリングパターンを提案していき，リファクタリングパターンが提案されていない分類をリファクタリング出来ない分類とする．

## 4. 今後の課題

今後の課題として，3 節で示した作成手法によるリファクタリングパターンの考案，および評価実験を行う必要がある．評価実験は，作成した分類木およびリファクタリングパターンに関する評価を考えている．分類木の評価は，全てのコードクローンの特徴に対する分類木の網羅率を計測する．具体的には，オープンソース以外のコードからランダムで選び出した一定数のコードクローンに対して分類木の網羅率を計測する．一方，リファクタリングパターンの評価は，リファクタリングパターンの妥当性を評価するために，リファクタリングパターン適用前後で動作の一貫性が保たれているか否かの調査を行う．また，リファクタリングパターンの有効性を評価するために，リファクタリングパターン適用に要する時間を計測する．加えて，提案するリファクタリングパターンと Fowler のパターンの適用，およびリファクタリングパターンの有無によるバグの修正に要する時間の変化を調査する．

## 参 考 文 献

1) T. Kamiya, S. Kusumoto, and K. Inoue : CCFinder: A Multilinguistic Token-Based Code Clone Detection System for Large Scale Source Code , *IEEE Transactions on Software Engineering* , vol.28 , No.7 , pp. 654–670 , 2002.
2) M. Fowler, K. Beck, J. Brant, W. Opdyke and D. Roberts : *Refactoring: Improving the Design of Existing Code* , Addison-Wesley Professional, 1999.

⋆1 本稿では，100 個のクローンセットを利用する．