B-004

利用シーンを考慮したコードクローン情報提示方法の提案
A proposal of Code Clone Information and Method for Use Scenes

中谷 和希† 垣谷 広輝† 平山 雅之† 菊地 奈穂美‡
Kazuki Nakatani Kouki Kakiya Masayuki Hirayama Nahomi Kikuchi

1. はじめに

ソースコードの複製によって生まれるコードクローンについては検出法も含め数多く議論されている[1]．コードクローンは保守性をはじめとしてソフトウェアの品質に様々な影響を及ぼすと考えられる．

例えば，コードクローンはソフトウェアの保守性を低下させる原因となりうるという報告もある[2].

クローンの元となるオリジナル部分に不具合がある場合，それを複製して生まれた箇所にも不具合がある可能性もあり，クローンを介して不具合が伝播してしまうという問題も考えられる．

これらの例からもわかるようにソフトウェアの保守性，信頼性向上の視点からコード内にクローンを検出した場合にどのように対処するか考える必要がある．このために，本論文では実開発の各段階において関与する開発者，マネージャに対してコードクローン情報をどのような形で提示していくかを提案し，コードの改良につなげていく方策を考える．

2. 研究背景

コードクローンに関しては，神谷らの CCFinder 等様々な研究が行われてきた．これらの研究の多くはクローン検出やその精度に重点を置いたものや，変更履歴等を考慮した保守性面からの議論が中心となっている[3]．一方で，実際の大規模ソフトウェアの場合，コード内には多量のクローンが含まれるケースもあり，それらをすべて検出したとしても具体的な対処方法を決めることができないといった問題がある．例えばあるコードについて保守性面から，まったく同一のクローンをモジュール化するという判断をしたとしても，それにより内部ロジックの誤りを誘発してしまうという可能性もあるため，画一的な判断は難しい．

上記のような場合，検出したクローンをさらに特徴的に分類して開発者やマネージャの判断に必要な情報を提示することが求められる．具体的にはプログラミングによって作成したコードだけでなく，設計段階で再利用するコードやテスト対象となるコードなどについてもクローン情報を提示し活用することが有効と考えられる．

3. クローン情報を提示する利用シーンの分類

2 章のような背景より，本研究では開発の各段階とその中での開発者，マネージャの役割を考慮したクローン情報の提示法を提案する．

3.1 利用シーン

ソフトウェア開発における段階を大きく分けると要求仕様を決定する段階と設計，プログラミング，テスト の 4 つに分けられる．この内，コードの再利用等も考えると，開発者，マネージャがクローン情報を活用するのは設計，プログラミング，テストの各段階であると考えられる．これら各段階で利用シーンを考え各シーンに応じたクローン情報の提示内容とタイミングを考えていく．

a) 設計段階

再利用するコード内のクローンの状況を確認し，そのコードの再利用が適切かを判断する．

b) プログラミング段階

実装したコードの保守性向上や不具合の除去を行う視点からクローン部分の確認を行う．

c) テスト段階

テスト戦略を立案する際にクローン部分をどう扱うかを判断する．

3.2 開発途中におけるクローン情報の提示方法

上記シーンのうち，設計，テスト段階ではコンパイル済みのコードを対象とするが，プログラミング段階でのクローン情報提示について考えてみるとコンパイル前のものも対象となり得る．

実際の開発現場ではコンパイル済みの完成コードに対してコードクローンの情報を示されたとしても，コードに手を加えることが難しいと考える人は多い．コンパイル前であればクローンを減らすような改造をするなどの対処はより容易であると考えられる．

このプログラミング段階では誤りを含むコードや，記述途中のコードでもクローン検出を可能にする手法を利用する必要がある．[4]の手法では構文解析を用いず，コードの形状からクローンの推定を行うことからコンパイル前のコードにも利用可能である．

4. 利用シーン毎の情報提示法

今回は 3.1 に示したシーンのうち，設計段階とプログラミング段階において具体的にどのようにクローン情報を提示すべきかを述べる．

4.1 設計段階

再利用したコード内にクローンが含まれる場合，修正時にオリジナル部分に手を加えると，そのクローン部分にも手を加えなければならないことも多い．この場合コードの修正には，想像以上に工数を要するといった問題が生じる．このため，この問題を視覚的に捉える指標として、[5]で提案されたクローン含有率を計測する．クローン含有率は(1)式となる．

$$\text{クローン含有率} = \frac{\text{クローンコード行数}}{\text{ソースコード総行数}} \times 100 \quad (1)$$

設計段階で再利用を考えているコードのクローン含有率を知っておくことで今後改変の際の工数や手間を見通すことができる．さらに，具体的に設計段階では再利用対象コードについてのクローン行数規模別含有率(図 1)と規模別出現回数(表 1)を提示する．図 1 はコードファイル内の

†日本大学 Nihon University
‡沖電気工業株式会社 Oki Electric Industry Co., Ltd.

40%近くをクローンが占めていることが分かる．小規模クローンが多く含まれているコードを再利用するとその後のコードの改変の際に影響が出そうであると推測できる．

表 1 では規模別に出現回数を提示している．出現回数とクローン種類数を提示することで 1 種類ごとの大まかな出現回数の情報が得られる．また，出現回数が多く種類数が多いクローンはマネージャが再利用時にどれだけの工数を要するかといった判断の参考になると考えられる．

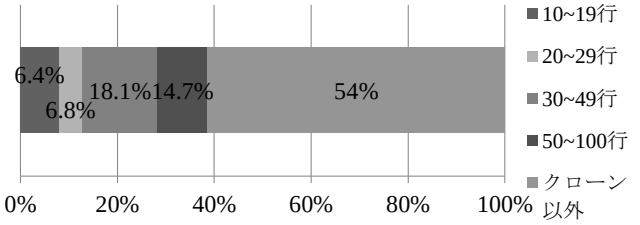

図1　クローン行数規模別含有率

表 1　クローン行数規模別出現回数と含有率

総行数	5000			
コード規模[行数]	出現回数	クローン種類数	クローン出現回数[回/種類]	クローン含有率[%]
10	20	5	4.00	4.0
15	8	1	8.00	2.4
20	12	3	4.00	4.8
…	…	…	…	…
50	4	1	4.00	4.0
80	6	2	3.00	9.6

4.2　プログラミング段階

プログラミング段階では実装中のコードの保守性や信頼性面からの完成度を確認し必要に応じて改善することが求められる．

a)　保守性

保守性面からは主にコード断片の複製回数に着目する．規模の小さなコード断片でも多数複製されていて，コード内に散乱しているとコード全体としては保守性が低下すると考える．

b)　信頼性

信頼性面では複雑なコード断片が複製されるとコード全体の複雑性が増加するため信頼性が低下する可能性がある．

例えば，保守性面からは図 2 の提示を提案する．図 2 はクローン出現回数別の種類数を提示している．出現数が多く，種類数が多い場合はクローン部分のコードに誤りが見つかった際の修正箇所が多いことを表しており，プログラミング段階でコンパイル前に大まかな保守工数を知ることができる．

更に対象を組込みソフトウェアなどのメモリ制約を受ける場合を考えると，上記の保守性，信頼性を考慮しながらコードの圧縮の検討が求められることがある．ここでは(2)式に示すコード削減可能量を提示することを考える．

$$削減可能量 = (コード規模) \times (出現回数 - 1) \quad (2)$$

(2)式で求めた値が大きいほど，コード品質への影響が強いということになる．削減可能量は図 3 のように提示する．図 2 ではコードのオリジナル部分の規模とそのクローン部分の出現回数を基に削減可能量を示している．図 3 のクローン A を例に，コード内に 10 行のクローン部分があった場合，その出現回数がオリジナル部分を含め 20 回であれば削減可能な箇所はオリジナル部分を除いた 19 箇所である．削減可能量は(2)式より求め，削減可能量は 190 となる．また，図 3 は削減可能量が大きいクローンを優先的に表示することで利用者にどのクローンから手を付ければいいかを知らせている．

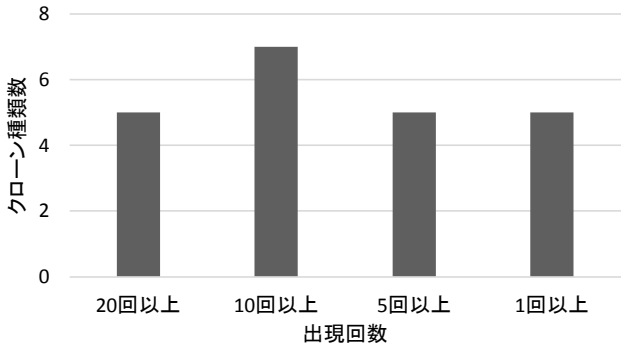

図2　クローン出現回数とその種類数

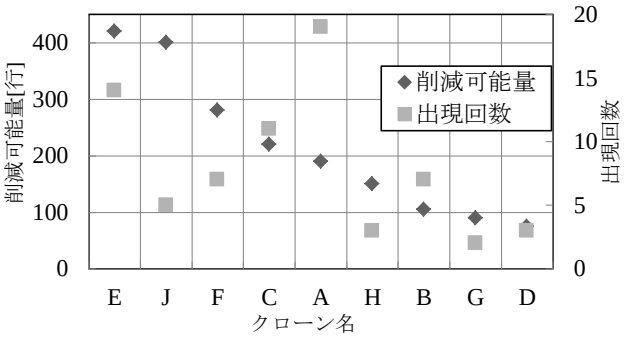

図3　コード削減可能量の提示例

5.　まとめ

本論では開発における設計段階とプログラミング段階でのコードクローン情報の提示方法を提案した．今後はテスト段階での提示方法について精査を行っていき，クローンの削減に有効な提案を考えていく．

参考文献

[1]肥後 芳樹，楠本 真二，井上 克郎 “コードクローン検出とその関連技術”, 電子情報通信学会論文誌. D, 情報・システム J91-D(6), 1465-1481, 2008-06-01

[2]門田 暁人，佐藤 慎一，神谷 年洋，松本 健一 “コードクローンに基づくレガシーソフトウェアの品質の分析”,情報処理学会論文誌,Vol.44 No.8,Aug. 2003

[3]神谷 年洋，肥後 芳樹，吉田 則裕 “コードクローン検出技術の展開”，コンピュータソフトウェア，Vol.28，No.3，pp.29-42 (2011)

[4]垣谷 広輝，安藤 優作，平山 雅之，菊地 奈穂美 "ソースコードを構成する処理ブロックの特徴に着目したコードクローン推定技術", 情報処理学会研究報告ソフトウェア工学研究会, Vol.2013-SE-182(29), 2013.

[5]馬場 慎太郎，吉田 則裕，楠本 真二，井上 克郎 “Fault-Prone モジュール予測へのコードクローン情報の適用”， 電子情報通信学会論文誌 D, Vol.J91-D, No.10, pp.2559-2561