選択的インライン展開に基づくメソッド境界を越えたAPI利用パターンの抽出とその活用

相澤 遥也[†1] 小林 隆志[†1]

コードの静的特性及び設計意図に基づく選択的インライン展開を用いて，複数のモジュールに跨るAPI利用パターンを抽出するアプローチを提案し，その概要と利用法について議論する．

Finding API Usage in Multiple Modules Based on Selective Inlining

Yuya Aizawa[†1] and Takashi Kobayashi[†1]

We propose a method to extract API usage patterns in multiple modules. Our approach applies selective inlining based on design intents and static characteristics of the code. We explain the outline of our approach and discuss applications of the patterns.

1. はじめに

ソフトウェア開発において，開発者は様々なライブラリ，フレームワークを利用する．ライブラリに付属されるドキュメントは，ライブラリを利用する開発者がその機能や利用方法を知る基礎的な資料であるが十分な情報を含んでいないことが多く [1]，ライブラリに修正や機能追加があっても，ドキュメントは適切に修正されず不整合を含むことさえある [2].

この問題に対し，API利用パターンと呼ばれる典型的なコードの書き方を既存のソフトウェアから自動的に抽出する手法が存在する．これらの手法にはソフトウェアのソースコードを静的解析して得られる情報からAPI利用パターンを抽出する手法 [3] や，ソフトウェアの実行情報を解析して得られる情報から抽出する手法 [4]，さらには，Web上の情報から発見する方法 [5] や，テストコードを解析する [6] 方法などがある．本研究では静的に解析する手法に着目する．

適切に設計されたソフトウェアは，機能独立性の高いモジュールに分割されて実現されている．すなわち，1つの機能を実現するコードが長くなる場合，それをそのままソースコード上で連続して記述するのではなく，一定のまとまったサブ機能ごとに新しい関数として抽出し，分割してコードを記述する．コードの分割が行われ，連続するコードが短くなると開発者にとっては可読性や保守性が向上する．しかし，静的解析によるAPI利用パターン抽出手法の多くは，連続して記述されたコードを抽出候補としている．このため，複数のモジュールにまたがって利用されているAPIの利用方法はパターンとして抽出できない．

我々は以前に，分割されたコードにおいても継承，委譲などによる特定の分割形態を検出し，コードを集約することでより長いAPI利用パターンを抽出できることを示した [7]．この際に，頻出系列数の著しい増大，パターンの有用性などの点で課題があった．本研究では，デザインパターンに基づいたコード分割の集約によって開発者がコードに込めた意図を汲み取ることを目指す．また，コードを集約する方法，抽出したAPI利用パターンの利用法について議論し，今後の研究の方針について述べる．

2. API利用パターンの抽出

以下の操作によってAPI利用パターンを抽出する．

(1) ソースコードをメソッド単位に分割
(2) 静的特徴，デザインパターンによる分割を検出
(3) 検出した分割形態に基づき，選択的インライン展開を用いてコードを集約
(4) 制御構文とメソッド呼び出しを抽出，抽象化してアイテム系列を生成
(5) 系列からAPI利用パターンを抽出

(1) と，(2) のデザインパターンの検出についてはDPdT [8] などの既存のツールを用いる．静的特徴については主に以前の研究の方法 [7] を用いる．(3) にお

†1 東京工業大学 大学院情報理工学研究科 計算工学専攻
Dept. of Computer Science, Grad. School of Information Sci. & Eng., Tokyo Institute of Technology

いてコードの分割形態に着目して選択的にインライン展開を行うことで，画一的な展開よりも少ない計算量で，分割された一連の機能の集約を実現する．(4) の抽象化では制御構文を loop，branch の 2 群の要素に集約することで，パターンの分解能を向上させつつ，実装の細かな違いを吸収しパターン数を確保する．(5) では飽和系列マインニングを用いることで計算コストの爆発的増加を防ぐ．このようにしてソースコードから API 利用パターンを，集約されたメソッド単位で，メソッド呼び出しと抽象化された構文要素の系列として抽出する．

3. API 利用パターンの応用

抽出された API 利用パターンを応用した開発者支援の方法について述べる．

バグ検出及び補完推薦

2 節のようにして集約した系列は，一連の機能の中で順番に呼ばれるべきメソッド (例えば入出力ストリームのオープンとクローズなど）をより多く含むことが期待される．故に，開発者の編集中のコードが抽出したパターンと食い違うような場合（クローズのないオープン）にこれを検出し報告，或いは補完を推薦する（クローズを挿入する）といった用途に有利である．

サンプルコードの提示

開発者の興味のあるメソッドを含む API 利用パターンを抽出し，その系列を生成するコードを提示すれば，開発者の理解を助けることができる．デザインパターンに基づくコード分割を集約，抽出して得た API 利用パターンは，従来のメソッド単位で抽出したものと比べて，メソッド或いはクラスを跨ぐ大局的なライブラリの利用法を含んでいるため，サンプルコードとして利用できる可能性がある．

デザインパターン利用推薦

デザインパターンに基づいて集約された系列が API 利用パターンとして抽出された場合，そのデザインパターンはライブラリメソッドと相性が良い可能性が高い．開発者のコード中から，抽出した API 利用パターンと類似の系列を検出し，デザインパターンを推薦すれば，開発者はその場でより適切なコード分割を行える可能性がある．

4. 今後の課題

コードの分割形態による集約の方法

デザインパターンに基づく集約はデザインパターン毎に固有の方法を定義する必要がある．この際留意すべき点として以下が挙げられる．

- 実行時にどの具象メソッドが呼ばれても，系列と矛盾しないこと
- 動的束縛されるメソッドの候補による組み合わせ爆発が起こらないこと
- デザインパターンを用いずに同様の機能を実現するコードから生成した系列と類似すること

例として State パターンであれば，Context クラスからの抽象 State クラスのメソッド呼び出しを集約する．この際，各具象 State のメソッドを候補として複数の系列を作るのではなく，全体を switch 文とみなす系列を生成する方法を検討している。

系列からコードへの逆変換

サンプルコードを生成するためには，API 利用パターンからコードへの逆変換が必要である．コードを系列に抽象化するときに，メソッドを利用したインスタンスの関係やメソッドの引数と戻り値の関係の情報は失われるため，これらをサンプルコードでどのように補完し，開発者に伝えるかという課題がある．

参 考 文 献

1) Robillard, M.P. and DeLine, R.: A field study of API learning obstacle, *Empir Software Eng*, Vol.16, No.6, pp.703–732 (2011).
2) Lethbridge, T.C., Singer, J. and Forward, A.: How Software Engineers Use Documentation: The State of the Practice, *IEEE Softw.*, Vol.20, No.6, pp.35–39 (2003).
3) Zhong, H., Xie, T., Zhang, L., Pei, J. and Mei, H.: MAPO: Mining and Recommending API Usage Patterns, *Proc. ECOOP2009*, pp. 318–343 (2009).
4) Pradel, M. and Gross, T.: Automatic Generation of Object Usage Specifications from Large Method Traces, *Proc. ASE2009*, pp. 371–382 (2009).
5) Sahavechaphan, N. and Claypool, K.: XSnippet: Mining For Sample Code, *Proc. OOPSLA'06*, pp.413–430 (2006).
6) Zhu, Z., Zou, Y., Xie, B., Jin, Y., Lin, Z. and Zhang, L.: Mining API Usage Examples from Test Code, *Proc. ICSME2014*, pp. 301–310 (2014).
7) 大塲光明，渥美紀寿，小林隆志，阿草清滋：メソッド境界を越えた呼び出しパターン抽出のためのコールグラフ探索戦略，情処研報 2012-SE-175 No.18，情報処理学会 (2012).
8) Tsantalis, N., Chatzigeorgiou, A., Stephanides, G. and Halkidis., S.T.: Design pattern detection using similarity scorin, *IEEE TSE*, Vol.32, No.11, pp.896–909 (2006).