情報伝播によるオブジェクト指向プログラム理解支援の提案

田中 昌弘† 石尾 隆† 井上 克郎†
大阪大学大学院情報科学研究科†

1. はじめに

Java などのオブジェクト指向言語による開発では，オブジェクトの実装とインターフェースを分離して実装の詳細を隠蔽し，外部モジュールとの結合を弱めることでモジュール単位の再利用が可能となる．

しかし，このような情報隠蔽が進むほど，デバッグやインスペクション作業で詳細な実行の過程を追う作業が困難になるという問題がある[1]．ソフトウェアの各機能の実行に対して，オブジェクト指向特有のインターフェースを介したメソッド呼び出しや，複数のオブジェクトによる協調動作が行われるので，大規模なソースコードの実装の詳細を把握することは極めて困難である[2]．

この問題を解決するために，メソッド呼び出し先のソースコードを呼び出し元で表示するという研究がある[3]．これにより複雑に断片化したソースコードを逐次的に読むことができるが，クラス継承や多相性が大規模になっている場合には対応できないという問題がある．

そこで本稿では，大規模なソースコードにも対応でき，効率的に実装の詳細を読解することを目的とした，ソースコードへのタグ付け手法を提案する．ユーザはソースコードに関する付加情報をタグとして記述して，ソースコード上に配置する．本手法はそれらのタグを関連する場所に自動的に伝播・表示する．これにより，実装に関する情報をその場で確認しながらソースコードを追って読むことができ，コメントや仕様書がほとんど無い状況でも効率的な理解が可能となることが期待される．

2. 提案手法

2.1. 概要

本手法では，ユーザの入力や自動解析によってソースコード上にタグを配置し，現在ユーザが参照しているソースファイル上の関連する箇所にタグを伝播して表示する．タグはソースコードと独立して管理されるため，ユーザはソースコードを更新することなくタグを配置し，編集することができる．

本手法におけるタグとは，ソースコード上の識別子である変数やメソッドに対する付加情報であり，一時的な識別子の値や型などといった情報を扱う．このタグを T として，以下のように定義する．

T(n, c, P)

n は識別子が位置する，ソースコードの構文木上の節であり，タグの位置を意味する．c は付加情報であり，ユーザが記入する．P は構文木上の節の集合であり，後述のタグ伝播ルールにおける伝播元のタグの位置を記録するものである．多相性や制御分岐などにより伝播元のタグが複数存在することが考えられるので，記録する節は集合として定義する．

ユーザがタグを配置すると，タグ T が関連する箇所に伝播され，情報伝播された識別子からも付加情報 c を参照することができる．

以下，タグの内容とその伝播ルールについて説明する．

2.2. タグに記述する情報

付加情報 c には，変数の型やその時点での値，もしくはその説明のためのコメントがユーザによって記入される．本研究では，Javadoc などと同様に自然言語と属性記述を組み合わせた形式を採用する．例えば変数の型については，type = Rectangle || Circle のように記述する．

2.3. タグの伝播

開発者が作成したタグは，同一の値となる識別子へ伝播される．ユーザが付加情報 c を入力して作成したタグは T(n, c, φ)となり，以下に示す 4 種類のデータフロー関係に従って新しいタグ T'(n', c, {n})を自動的に生成していく．

1. データフローグラフ上での参照
2. 1 対 1 の代入
3. メソッド呼び出しの引数
4. メソッドの戻り値

1. は，データフロー上での定義・参照の関係にある識別子の間での伝播である．例えば図 1 のように，1 行目の変数 i で作成したタグが，データフロー上で参照の関係にある 3 行目の i に伝播される．

Code Reading Support with Propagating Information Tags
†Masahiro TANAKA, †Takashi ISHIO, †Katsuro INOUE
†Graduate School of Information Science and Technology, Osaka University

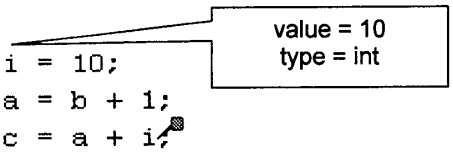

図1　データフローの参照関係

2. は例えば a = b;のように，1 つの識別子から 1 つの識別子に代入される場合である．この場合，右辺の識別子のタグが，左辺の識別子に伝播される，なお，右辺に演算子や他の識別子が含まれている場合は，伝播を行わない．

3. はメソッド呼び出しの実引数からメソッドの仮引数への伝播である．この場合，実引数の識別子のタグが仮引数の識別子に伝播される．2. と同様に，実引数は 1 つの識別子である(演算子などを含まない式である)ことを伝播条件する．

4. はメソッドの戻り値が左辺として代入される場合であり，メソッドの実装先の return 文にある識別子のタグが，呼び出し元のメソッドに伝播される．この場合も同様に return の値は 1 つの識別子であることが条件である．

このような伝播を行うとき，伝播先の識別子には，タグ T(n, c, P)がすでに付加されている場合がある．ここで同じ識別子 n に新たに付加したいタグ T'(n, c', P')があったときには，もし c と c'が同じであるときは，タグ同士の衝突と見なして，単一のタグ T(n, c, PUP')へとマージする．c と c'が異なるときは，相互に独立したタグとして，別個に伝播を行う．

以上のような方式で伝播が行われるが，同一の情報がすでに存在する識別子にはタグを伝播しないというルールと，識別子の個数が有限であることから，伝播処理は有限時間で終了する．ユーザが必要に応じてタグを追加すると，その都度，タグ伝播処理を実施する．

この伝播を行う際に必要となるのが，データフロー解析である．構文木情報のみを必要なときに探索することで計算量の削減を図る[4]．

タグの伝播において問題となるのが，メソッド呼び出しにおいて，多相性によって，複数のメソッドを実行しうる場合である．通常は，実行されうる各メソッドに対してタグを伝播させることとなるが，ユーザがオブジェクトの型情報を指定するタグを付与した場合，その情報をもとに多相性を解決することを考えている．このような仕組みを用いれば，ユーザが想定する実行状況でのプログラムの振る舞いに関する情報をソースコード上に表示することができ，これによりソースコードの詳細な実行過程をその場で記録しながら理解できる環境を実現することが期待される．

3. 手法の実装計画

手法は Eclipse プラグインとして実装し，対象言語は Java で行う．タグの作成は，エディタ上で識別子を選択して行う．タグは，図 2 のようにマーカーとしてソースコード上に配置され，クリックなどによってタグの内容が吹き出しとして表示される．

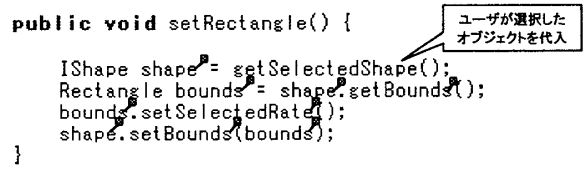

図2　ソースコード上のタグ

4. 考察

本稿では実装の詳細把握を目的とした，ソースコードにタグを付けて伝播させる手法を提案した．本手法はデータフロー解析に基づいているが，多くの解析手法はソースコード全体を対象にしており，ソースコードの一部だけを個別に解析できる技術は少ない．本研究におけるタグ伝播は，全体解析を省略しており，ユーザにとって満足できる応答速度を実現する．一方，たとえば多相性の解決では，多くの統合開発環境でのメソッド呼び出しの解決と同様，どのメソッドが実際に呼び出される場合を知りたいかをユーザが対話的に入力する．このようなユーザの意思決定も，タグとして記録していく．

今後の課題としては，本手法をソフトウェア保守作業に適用し，ユーザが与える入力データ量と，ユーザが得られる出力とのバランスの評価が挙げられる．

謝辞

本研究は，科学研究費補助金若手研究(スタートアップ)(課題番号:19800021)の助成を得た．

参考文献

[1] D. Spinellis, *Abstraction and Variation.* IEEE Software, Vol.24, No.5, pp.24-25, 2007.
[2] N. Wilde et al., *Maintenance Support for Object-Oriented Programs.* IEEE TSE, Vol.18, No.12, pp.1038-1044, 1992.
[3] M. Desmond et al., *Fluid Source Code Views for Just In-Time Comprehension.* SPLAT 2006.
[4] A. Hajnal et al., *Demand-Driven Def-Use Chaining Algorithm.* CSMR 2002, pp. 77-86.