5L-5

対話的なソースコード検索ツールの提案

平山 拓朗 † 丸山 一貴 ‡ 寺田 実 †
Takuro Hirayama Kazutaka Maruyama Minoru Terada

1 はじめに

プログラミングにおいて, ソースコード検索は重要なツールである. ソースコード検索を行うことで, 着目している事柄に関する手がかりを得られる.

Starke らは, プログラマが Eclipse を用いてデバッグ作業をしている際のソースコード検索の利用状況を観察した [1]. その結果, プログラマはある一つの事柄を探すために検索条件の試行錯誤を繰り返すという特徴が見受けられた. 具体的には,

1. あいまいな単語で検索を行う（バージョン管理システムにおける commit など）
2. Java 言語の構文を考慮する Java 検索があるにも関わらず, 単純なテキスト検索を多用する

などがあった.

原因として, 検索開始時点ではプログラマが検索すべき具体的な条件・キーワードをよく把握していないことが考えられる. その結果として, 関連性が低いものを含む検索結果が多数提示されてしまい, 条件を変えて再び検索を行わなくてはならなくなってしまう.

2 目的

本研究では, 検索キーワードに関連する識別子を提示してユーザに選択させることで, 対話的な検索条件の試行錯誤を支援するシステムを提案し, その有用性を考察する.

3 提案システム

図 1 に使用イメージを示す. 初期状態では全ソースコード中の識別子リストとメソッド名リストが表示される（図 1, 1）. ここで, 識別子リストから検索したい識別子を選択して（図 1 , 1 - 2）検索を行うと, 選択した識別子が出現するメソッドに限定された識別子リストとメソッド名リストの画面に更新される（図 1 , 2）. この操作を繰り返すことで, 着目している事柄に関連する記述を絞り込んでいくことができる（図 1 , 2 - 2, 3）.

識別子リスト内での識別子の選択の際には, リストにおけるキーワード検索や, 表示する言語要素の設定を行うことができる（図 1 , a）.

4 実装

4.1 AST 解析部

EclipseJDT[*1]を用いて Java ソースコードの抽象構文木（AST）を構築して, 情報収集, 記録を行う Java プログラムである.

4.2 検索システム部

ユーザインタフェース表示, ユーザの操作に応じた画面更新を行う. サーバ側は Java, クライアント側は JavaScript の Web アプリケーションである.

5 考察

ソフトウェアのデバッグ作業において検索を行うことを想定し, 本システム使用時と Eclipse の検索機能使用時を比較, 考察を行った.

5.1 対象ソフトウェアとシナリオ

1. Commons CLI[*2]（20 ファイル, 4739 行）
   概要 コマンドライン引数実装支援ライブラリ
   シナリオ 特定コマンドライン引数を必須（required）とする機能に関する記述の検索
2. 手書き Wiki[*3]（29 ファイル, 5697 行）
   概要 リンクによる複数シート間の双方向結合を特徴とした電子ホワイトボードシステム
   シナリオ マウス右ボタンドラッグでシート間リンクを作成する処理に関する記述の検索

5.2 テキスト検索と比較して

5.1 の 1 にて全ソースコードを対象に require というキーワードで通常のテキスト検索を行うと, 126 行

---

* Proposal of interactive source code inspection tool
† 電気通信大学, The University of Electro-Communications
‡ 明星大学, Meisei University

[*1] Eclipse Java development tools (JDT) : http://www.eclipse.org/jdt/
[*2] http://commons.apache.org/proper/commons-cli/
[*3] http://pr.ice.uec.ac.jp/~terada/tegakiwiki/

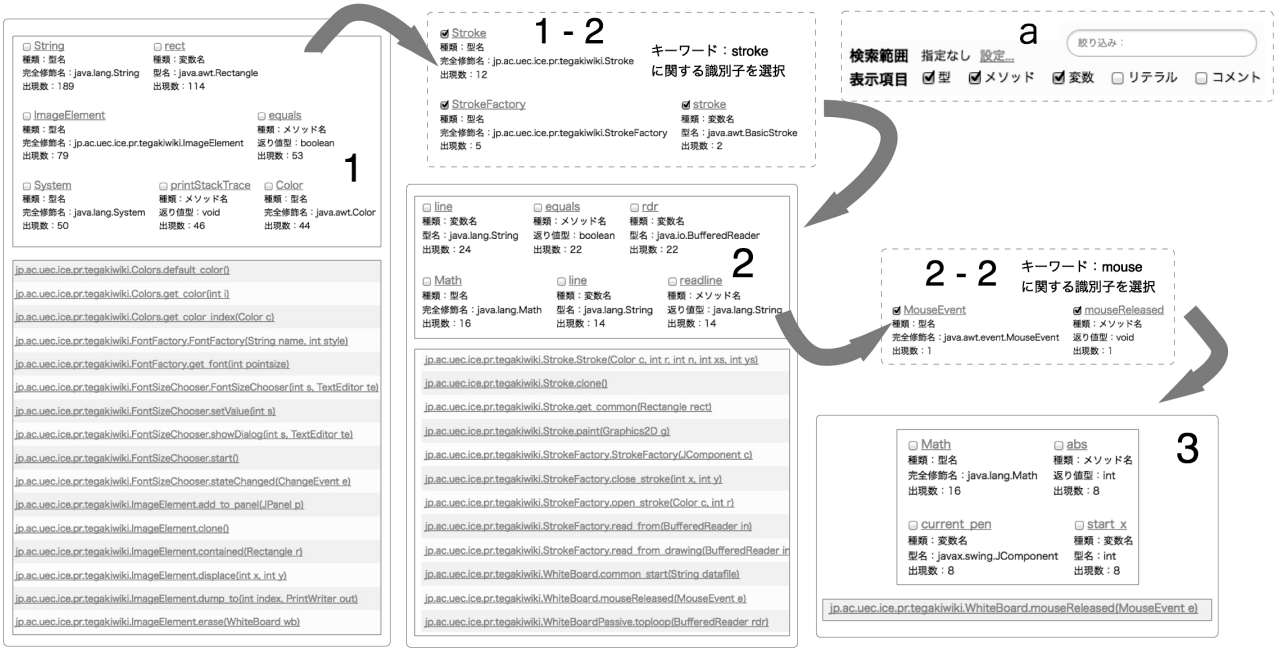

図 1　提案システム使用イメージ

もの該当箇所が全 20 ファイルに散在している結果となった．内容を観察すると，インポート文やドキュメンテーションといったノイズが多い，同一の識別子が複数回参照されている分だけ冗長に表示される，といった傾向が見受けられた．その結果，ファイルを開いて広い範囲を見渡す必要性が生じる．

本システムの場合，AST 解析後に言語要素の検索を行う．また，特定の言語要素を非表示にできる．そのため，コメントとリテラルを非表示にして同様に require というキーワードで検索を行うと，19 個の該当する識別子が 20 個のメソッドに存在しているという結果になった．キーワードに関する識別子と出現するメソッド名がわかるので，内容の吟味にかかる手間を軽減することができる．

5.3　Java 検索と比較して

Java 検索の場合，検索実行前に検索対象とする言語要素を指定する．この言語要素の指定は排他的であり，同時に複数の言語要素を選択することができない．例として 5.1 の 2 において，押されたマウスボタンが右クリックかを判別する条件の記述は，`isButton3(event)` や `event.getButton() == MouseEvent.BUTTON3` など複数考えられる．このとき，button3 というキーワードで Java 検索を行う場合，メソッドと変数について個別に検索を行い，それぞれの検索結果を吟味する必要がある．

本システムの場合，検索結果として表示する項目を対話的に複数選択することができるため，そのような手間が生じることがない．

6　関連研究

プログラム理解支援のために，ソースコード中の語彙に着目した研究として，Cultivate[*4]や Code Gestalt [2] が挙げられる．これらのシステムではソースコード中に出現する英単語を分析して，タグクラウドのような可視化図を作成できるが，単語を選択してコードナビゲーションを行うといったインタラクティブ性は備えていない．

7　まとめ

本研究では，検索キーワードに関連する識別子を提示し，選択することで，対話的な検索を行うシステムを提案し，有用性を考察した．その結果，検索にかかる手間を軽減できることを示せた．

参考文献

[1] Starke J. , Luce C. , Sillito, J. “Searching and Skimming: An Exploratory Study” , Software Maintenance, ICSM 2009, pp. 157–166, Sep. 2009.

[2] Christopher Kurtz. “Code Gestalt: a software visualization tool for human beings” , CHI EA ’11, pp. 929–934, 2011.

[*4] `http://sewiki.iai.uni-bonn.de/research/cultivate`