特別研究報告

題目

メソッド名とその周辺の識別子の相関ルールに基づく メソッド名変更支援手法

指導教員
井上 克郎 教授

報告者
柏原 由紀

平成 25 年 2 月 12 日

大阪大学 基礎工学部 情報科学科

平成 24 年度 特別研究報告

メソッド名とその周辺の識別子の相関ルールに基づくメソッド名変更支援手法

柏原 由紀

内容梗概

ソースコード中の識別子が不適切である場合，プログラムの理解に時間がかかることが知られている．開発者が識別子からその役割を正しく推測できるようにするため，オブジェクト指向プログラミングにおけるメソッドに，その動作と対象を表すような名前を識別子として用いることが推奨されている．しかし，一般的なガイドラインにはメソッドに対してどの単語を使って命名したほうがよいかということは明記されておらず，開発者がドメイン知識や経験をもとに用いる語を選択する必要がある．

そこで本研究では，ソースコード中の既に記述されたメソッドの名前を変更しようとする開発者に，そのメソッドに対してつけられる可能性が高いメソッド名の変更候補をリストとして提示する手法を提案する．既存のソースコードからメソッド本体の内容に対応するメソッド名を学習し，開発者が名前を変更したいメソッド本体の内容に対応したメソッド名を命名できるよう支援する．メソッド本体とメソッド名の対応付けには相関ルールマイニングを用いており，作成したルールの情報を候補の生成とその順位づけに用いた．

手法が適切なメソッド名を提示できることを確かめるために 2 つのソフトウェアに対する適用実験を行った．その結果，実験に用いたソフトウェアに対して，メソッド名の動詞については，87.1%のメソッドで生成でき，その 52.9%がリストの高い位置に提示できることがわかった．メソッド名の目的語については，21.9%のメソッドで生成でき，その 50.0%がリストの高い位置に提示できることがわかった．

主な用語

メソッド名
命名支援
相関ルールマイニング

目 次

1　はじめに

ソースコードの読解においてソースコード中の識別子が不適切であるとプログラムの理解に時間がかかることが知られている [7]．保守作業においてソースコードの読解がそのコストの大半を占めており [10]，プログラム理解に時間がかかることは保守コストの増大にもつながっている．

ソースコードの読解では，ソースコード中に出現する識別子からその役割を推測することがある．保守作業において，開発者はソースコードを読解することで修正個所を特定するが，このときの手掛かりの 1 つが識別子である [8]．また，識別子の命名においてその役割を過不足なく表す命名をすることが重要であると複数のガイドライン [9, 11] で主張されていることからも，適切な識別子をつけることが重要であることがわかる．

オブジェクト指向プログラミング言語の識別子には，代表的なものとしてクラス名，変数名，メソッド名が挙げられる．変数名は変数が保持しているデータの意味を表す名前，メソッド名はその動作と対象を表す名前をそれぞれつけることが重要であり，クラス名はクラス内で定義されている変数やメソッドを包括するような名前をつけることが重要であると考えられている [9]．

これらの識別子の中でも，本研究ではメソッドの命名が難しいことに着目する．オブジェクト指向プログラムのメソッド名は一般に動詞や名詞などを組み合わせて命名される [9] が，一般的なガイドラインにはメソッドに対してどの単語を使って命名したほうがよいかということは明記していない．メソッド名に使われる一部の動詞については，メソッドの中でどのような処理が行われているかが調査されている [3] が，これ以外の動詞については調査されていない．また，動詞に対する目的語については開発者がドメイン知識や経験をもとに選択する必要がある．

そこで本研究では，ソースコード中の既に記述されたメソッドの名前を変更しようとする開発者に，そのメソッド本体に対してつけられる可能性が高いメソッド名の変更候補をリストとして提示する手法を提案する．既存のソースコードからメソッド本体の内容に対応するメソッド名を学習し，開発者が名前を変更したいメソッドに対して，本体の内容に対応した命名を行えるよう支援する．メソッド本体とメソッド名の対応付けには相関ルールマイニングを用いており，作成したルールの情報を候補の生成とその順位づけに用いた．

手法を実現するツールを統合開発環境である Eclipse のプラグインとして実装した．本ツールは，まず指定されたメソッドを特徴づける識別子を取得し，事前に作成された命名相関ルールと既についているメソッド名を用いて，メソッド名の変更候補を開発者に提示する．具体的には，Java エディタ上で，名前を変更したいメソッドにカーソルを合わせてツールを起動すると，ビュー上にメソッド名の候補がリストで提示される．

手法が適切なメソッド名を提示できることを確かめるために，評価実験を行った．その結果，実験に用いたソフトウェアに対して，メソッド名の動詞については，87.1%のメソッドで生成でき，その52.9%がリストの高い位置に提示できることがわかった．メソッド名の目的語については，21.9%のメソッドで生成でき，その50.0%がリストの高い位置に提示できることがわかった．

本稿における貢献は以下のとおりである．

- 既存のソースコード群の命名事例から命名相関ルールを作成する手法を提案した
- 作成した命名相関ルールを用いてメソッド名を変更する手法を提案した
- 提案したメソッド名の結果が良いものであることを示した

以降，2章では背景について述べ，3章では提案手法について説明する．その後，4章で提案手法を実装したツールについて述べ，5章では評価実験について説明する．さらに6章では関連研究を示し，最後に7章でまとめと今後の課題について述べる．

2　背景

本研究の手法の背景として，プログラミング言語の識別子について，一般的な識別子の記法と，オブジェクト指向プログラミング言語である Java の識別子に対する命名法を説明し，さらに識別子の中でもメソッドの命名における難しさを説明する．次に，メソッド名の命名支援に関する研究として，メソッド本体を記述した後にメソッド名の動詞の変更を支援する手法と，今から記述するメソッドを命名しようとしているときにメソッド名の命名支援を行う手法の 2 つの研究を紹介する．最後に，本研究で提案する手法で用いる相関ルールマイニングについて説明する．

2.1　識別子

本節では，一般的な識別子の記法を説明してから，Java に登場する識別子について識別子の種類ごとに命名の特徴を説明する．

2.1.1　一般的な識別子の記法

一般に識別子は 1 つの単語もしくは複数の単語の列を用いて命名される．識別子の中では，単語境界に空白文字を用いることができないため，複数の単語をひと綴りで記述する．空白を用いずに複数の単語をひと綴りで表現する記法には camelCase，PascalCase，snake_case などがある．camelCase は単語の列の先頭文字を小文字に，単語境界となる単語の先頭文字を大文字で表して単語をつなげる記法である．一方，PascalCase は単語の列の先頭文字および単語境界となる単語の先頭文字をどちらも大文字で表して単語をつなげる記法である．また，snake_case は単語が大文字表記か小文字表記かにかかわらず，単語間をアンダースコアでつなげる記法である．「example」と「identifier」の 2 つの単語をつなげる場合，camelCase の記法では exampleIdentifier，PascalCase の記法では ExampleIdentifier となり，snake_case の記法では example_identifier となる．

2.1.2　Java に登場する識別子の種類と命名

Java に出現する識別子は，クラス名・変数名・定数名・メソッド名の 4 種類に大別できる．それぞれの分類における識別子について，Java の命名規約 [11] に基づいて説明する．

クラス名

クラス名に分類される具体的な識別子にはクラス名とインタフェース名があり，一般に PascalCase で表現した単語もしくは単語の列として命名される．これらの単語には，クラスの役割を表現しているような，略語ではない名詞が使われる．

変数名

変数は，フィールド変数やローカル変数などをさし，その名前は一般に camelCase で表現した 1 つ以上の単語の列として命名される．for 文などで使われるローカル変数がアルファベット 1 字の命名が許されることを除いて，役割を表現する単語もしくはその略語を用いて命名する．

定数名

定数は，final 修飾子によってプログラム実行中に値が変化しない変数のことをさし，その名前はすべて大文字の単語を snake_case で結合した 1 つ以上の単語の列として命名することが多い．変数名と同様に，役割を表現する単語やその略語を用いて命名する．

メソッド名

メソッド名は一般的には小文字で始まる動詞を先頭単語として camelCase で表現された 1 つ以上の単語の列で構成され，メソッド名全体で自然言語の動詞と目的語のような関係をもつことが多い [13]．また，メソッド名の目的語に用いられる単語の列は，メソッドの動作の対象となる識別子を用いることが多いと考えられている [9]．

メソッドの命名には様々な守られるべきルールが存在する．例として，フィールドの値を取得するメソッド (Getter) は動詞 get に取得するフィールド名を組み合わせるというルール，フィールドの値を設定するメソッド (Setter) は動詞 set に設定するフィールド名を組み合わせるというルールがある．Getter および Setter は広く知られているルールであり，Eclipse でも標準機能として get および set の動詞を用いたメソッドを補完する機能が提供されている．

さらに，特殊なメソッド名の命名パターンとして，目的語-動詞の順に並べて命名することや動詞を使わずに命名することもある．目的語-動詞の順に単語を並べたメソッド名は，マウスの動きやボタンのクリックなどを監視して，動作が発生したときに呼び出されるメソッドの名前としてよく出現しており，例として，java.awt.event.ActionListener インタフェースの actionPerformed メソッドがある．また，動詞を使わない命名の例としては，java.lang.Object クラスの toString メソッドなどの前置詞から始まるメソッドが挙げられる．

2.2 メソッド名の命名支援に関する研究

動詞や目的語の膨大な組合せがある中で，それらをどう組み合わせるかを推薦することでメソッドの命名を支援する研究が行われている．既に命名されたメソッド名の動詞に対して変更支援を行う lancelot[5] というツールと，新規に命名しようとしているメソッド名に対して命名支援を行う MethodFather[12] というツールについてそれぞれ順に説明する．

2.2.1 lancelot[5]

lancelot は，開発者が記述したメソッドの動作に対して適切なメソッド名の動詞を提示するツールである．lancelot は，既に記述されたメソッドと，メソッド内の動作とメソッド名に使うべき動詞を対応付けたルールを入力として，現在のメソッド名の動詞がメソッドの動作に対して不適切であるときに，適した動詞を理由とともに提示する．このツールで用いているルールは既存のソースコードの事例からツールの製作者によって手作業で綿密に作成されている．

lancelot は，動作に対して動詞が適切であるかどうかの判定を非常に厳しく行っているため，動作に対して非常に高い精度で適切なメソッド名の動詞を提示することができる．ただし，用意されているルールの数が少ないため提示できる動詞の数が少なく，また，手作業でルールを作成しているため，ルールの追加によって提示できる動詞の数を増やすには，大きな手間が必要となる．

2.2.2 MethodFather[12]

MethodFather は，開発者が新規にメソッドを命名するときに開発者にメソッド名の候補を提示するツールである．MethodFather は，開発中のソースコード内に既に記述されている識別子と動詞-目的語関係を収録した辞書を入力として，動詞と目的語の組で構成されたメソッド名の候補を提示する．このツールで用いている動詞-目的語関係を収録した辞書は，既存のソースコードを入力として，それらのソースコード中の動詞-目的語関係を定式的にすることで生成されている [13].

MethodFather は，辞書に含まれる動詞-目的語関係の数を増やすことで，非常に多くのメソッド名を開発者に提示することができる．ただしこの手法は，メソッドを記述する前にメソッド名を提示することを想定しているため，メソッド本体が記述されているかどうかにかかわらず，メソッドの動作を考慮したメソッド名の候補を提示できないという課題がある．

2.3 相関ルールマイニング [2]

相関ルールマイニングとは，大量のアイテム集合 (トランザクション) の集合を入力として，あるアイテムがトランザクションに出現したときに，別のアイテムもまた同じトランザクションに出現する可能性が高いという関係を抽出する手法である．相関ルールマイニングによって抽出されるアイテム間の関係を相関ルールという．相関ルールをアイテム集合 X, Y を用いて，式 (1) のように表す．このとき，式 (1) における X を条件部，Y を帰結部といい，X，Y は入力に与えたトランザクションの部分集合となる．

$$X \rightarrow Y \tag{1}$$

入力として用いたトランザクションの集合に対して抽出した相関ルールがどのくらい強い相関を持っているかを評価する指標として，支持度と確信度という値がある．支持度は，すべてのトランザクションに対する条件部と帰結部がともに部分集合となるトランザクションの割合である．支持度に類似した指標として，条件部と帰結部がともに部分集合となるトランザクションの数を表す頻度がある．確信度は，条件部が部分集合となっているトランザクションのうち，帰結部もまた部分集合となっているトランザクションの割合を表す．支持度や頻度が高いとその相関ルールを満たすトランザクションの数が多いことがわかり，確信度が高いと，条件部が部分集合となるトランザクションは帰結部のアイテムを含んでいることが多いことがわかる．相関ルールマイニングでは，支持度や確信度の値がある閾値以上であるときに相関があるとみなすが，この閾値は相関ルールマイニングを行うときに自由に決めることができる．

3　提案手法

本研究では，開発者がメソッド名の変更を検討する際に参考にできるように，メソッド名の変更候補をリストで提示する手法を提案する．本手法は，既存のソースコードからメソッドを特徴づける識別子とメソッド名の関係を表すルール (**命名相関ルール**) を作成し，開発者が指定したメソッドのコンテキストに合致する命名相関ルールを用いてメソッド名を提示する．2.1.2 節で示したようにメソッド名にはさまざまな形があるが，本手法では，大部分のメソッド名の構造である動詞のみのメソッド名または動詞と目的語の組になっているメソッド名のみを提示する．

提案手法の概要を，図 1 に示す．本手法は大きく分けて命名相関ルールの抽出を行うステップ 1 と，命名相関ルールを用いたメソッド名の生成を行うステップ 2 から成る．ステップ 1 の命名相関ルールの抽出では，既存のソースコード群を入力として相関ルールマイニングを行い命名相関ルールを抽出する．ステップ 2 のメソッド名の変更候補の生成では，指定されたメソッドのコンテキストを用いて命名相関ルールを検索し，得られた命名相関ルールと指定されたメソッドの名前からメソッド名の変更候補リストを得る．

3.1　ステップ 1:命名相関ルールの作成

本ステップでは，既存のソースコード群を入力として相関ルールマイニングにより命名相関ルールを作成する．このステップは以下の 2 つのサブステップからなる．

サブステップ 1-1 既存のソースコード群からメソッド名とそのコンテキストを収集する

サブステップ 1-2 サブステップ 1-1 の出力から命名相関ルールを作成する

以降の節で，各サブステップについて順に説明する．

3.1.1　サブステップ 1-1：ソースコード群から情報取得

本サブステップでは，ソースコード群からメソッドごとにメソッド名の構成要素とコンテキストを取得する．メソッド名の構成要素とは，メソッド名に使われる動詞と目的語の組である．メソッドのコンテキストとは，1 つのメソッド周辺に出現した識別子とその種別の組の集合を表すものであり，1 つのメソッドに対して一意に定まる．コンテキストに含まれる要素は，具体的には以下の通りである．

クラス名 メソッドが定義されているクラスの名前

親クラス名 メソッドが定義されているクラスの親クラスの名前

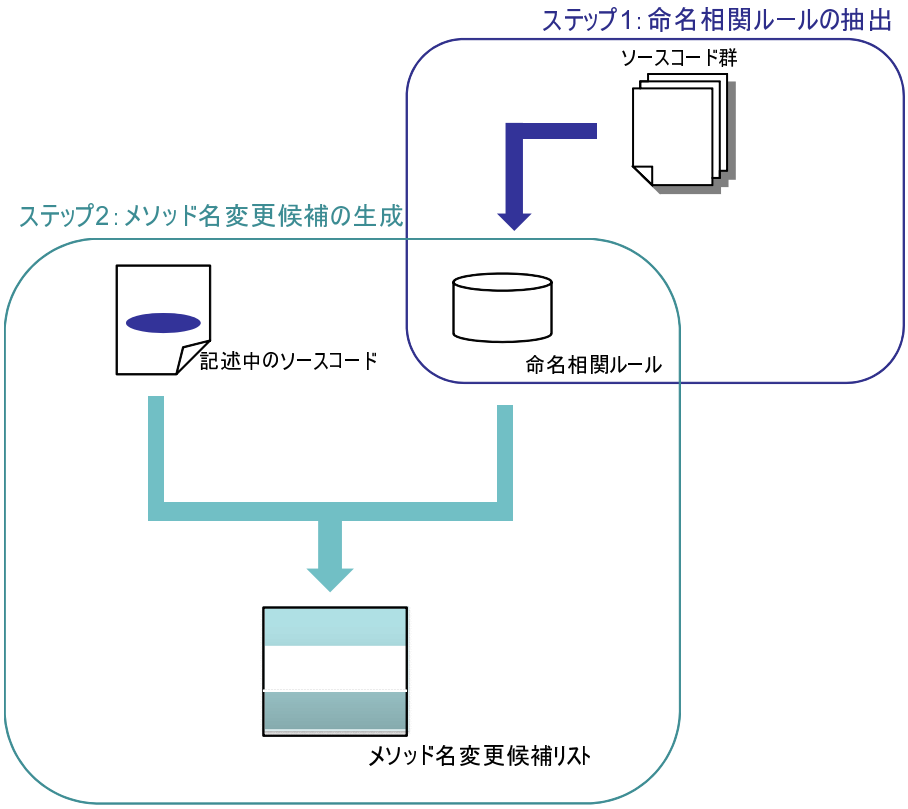
ステップ1: 命名相関ルールの抽出
ソースコード群
ステップ2: メソッド名変更候補の生成
記述中のソースコード
命名相関ルール
メソッド名変更候補リスト

図 1: 提案手法の概要図

インタフェース名 メソッドが定義されているクラスのインタフェースの名前

返り値の型 メソッドの返り値の型名

引数の型 メソッド引数に使われている型名

引数の名前 メソッド引数に使われている名前

フィールド名 メソッドの中でアクセスしているフィールドの名前

呼び出しメソッド名 メソッドの中で呼び出しているメソッドの名前

コンテキストの例を図 2 に示す．この図の (a) のソースコードには，findName，setName という 2 つのメソッドが含まれており，findName からは (b) が，setName からは (c) がコンテキストとして得られる．

本サブステップでメソッド名の構成要素とコンテキストを収集するメソッドは，メソッド名が動詞と目的語の組となっているメソッドのうち，以下に示すメソッド**以外**の全てのメソッドを収集対象とする．

- main メソッド
- コンストラクタ
- 内部クラス内および匿名クラス内に定義されているメソッド
- Getter および Setter

解析対象のメソッド集合を M，あるメソッド m のコンテキストを $c(m)$，m のメソッド名に含まれる動詞を v，目的語を obj とすると，本ステップの出力 S は次のような集合として記述できる．

$$S = \bigcup_{m \in M} \{c(m) \cup \{\text{メソッド名の動詞} : v, \text{メソッド名の目的語} : obj\}\} \tag{2}$$

メソッド名の構成要素を収集する際に，動詞や目的語の判定には品詞解析ができる自然言語処理ツールを利用しており，実装では OpenNLP[1] を用いた．ただし，to，new，init，calc，cleanup，setup の 6 つの単語については，品詞解析の結果にかかわらず，本手法では動詞として扱う．これらの単語は，メソッドの命名において慣習的に動詞として用いられているからである．

```
import java.io.Serializable;
import java.util.LinkedList;

public class NameList implements Serializable {
  LinkedList<String> namelist;
  int size;

  public String findName(String n){
    if (namelist.contains(n)) {
      return n;
    }
    return null;
  }

  public void setName(Integer size) {
    this.size = size;
  }
}
```

(a) ソースコード

クラス名：NameList
インタフェース名：Serializable
返り値の型：String
引数の型：String
引数の名前：n
フィールド名：namelist
呼び出しメソッド名：contains

(b) メソッド findName のコンテキスト

クラス名：NameList
インタフェース名：Serializable
返り値の型：void
引数の型：Integer
引数の名前：size

(c) メソッド setName のコンテキスト

図 2: メソッドのコンテキストの例

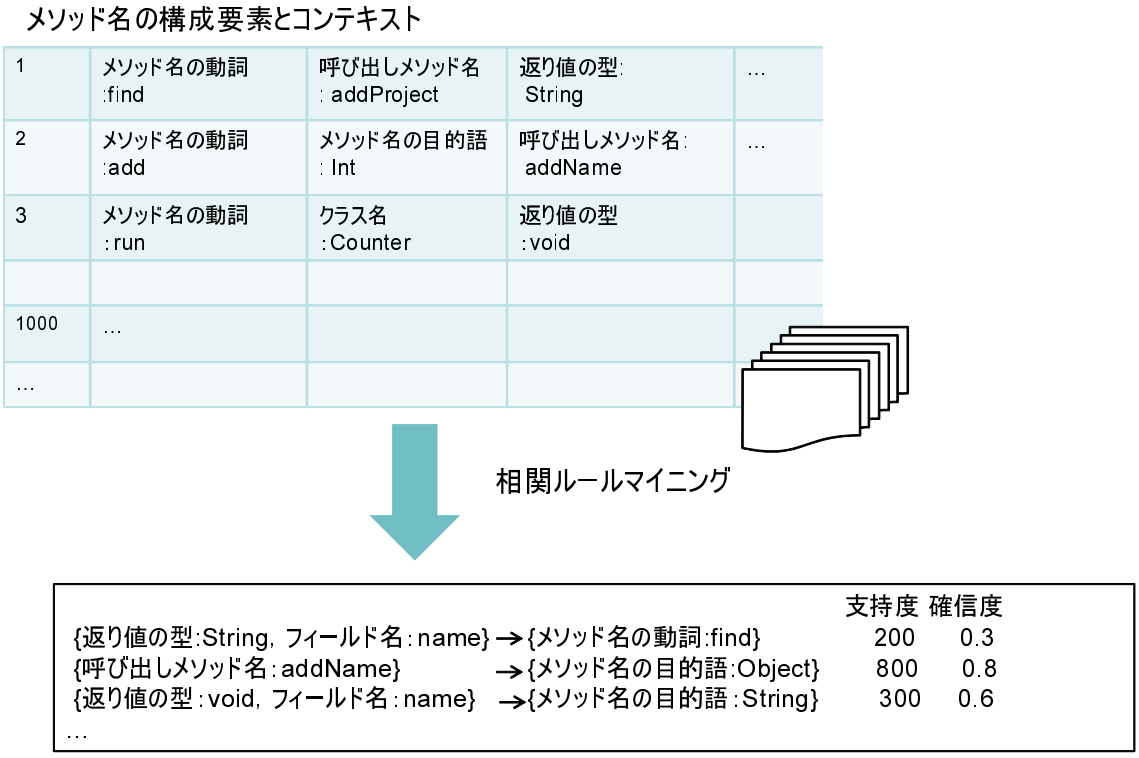

図 3: 命名相関ルールの生成

3.1.2 サブステップ 1-2 : 命名相関ルール作成

サブステップ 1-1 で収集した情報を入力として，相関ルールマイニングを行い命名相関ルールを作成する．命名相関ルールとは，条件部にメソッドのコンテキストとなる要素の集合が，帰結部にメソッド名の構成要素となる集合が出現している相関ルールである．命名相関ルールにおける支持度は，条件部と帰結部の要素がともに出現するメソッドの数となる．確信度は，条件部がコンテキストの部分集合となっているメソッドのうち，帰結部もメソッド名の構成要素の部分集合となっているメソッドの割合となる．相関ルールマイニングの閾値は支持度 100 以上，確信度 0 以上とした．抽出されたルールのうち，条件部にメソッドのコンテキストの集合のみを含み，帰結部に「メソッド名の動詞」と「メソッド名の目的語」の高々2 種類が含まれているルールをメソッドのコンテキストと名前を対応付けるルールとして採用する．図 3 で命名相関ルール抽出の例を示す．

3.2 ステップ 2：命名相関ルールを利用したメソッド名変更候補リストの生成

本ステップでは，ステップ 1 で抽出した命名相関ルールと，開発者が名前の変更を意図して指定したメソッドの名前の構成要素とそのコンテキストを用いて，メソッド名の変更候補リストを生成する．このステップは図 4 に示すように，4 つのサブステップからなる．

サブステップ 2-1 編集中のソースコードから指定されたメソッドの名前の構成要素とそのコンテキストを収集する

サブステップ 2-2 収集したコンテキストを用いて適用可能な命名相関ルールを抽出する

サブステップ 2-3 命名相関ルールから得た情報と，サブステップ 2-1 で収集したメソッドの構成要素から，メソッド名の変更候補の集合を生成する

サブステップ 2-4 生成したメソッド名の変更候補の集合の要素を並び替え，メソッド名変更候補リストを生成する

以降，各ステップを順に説明する．

ただし，次に挙げるメソッドは本手法の対象外とし，開発者によって変更を意図して指定された場合でもメソッド名の変更候補リストの提示は行わない．

- main メソッド
- コンストラクタ
- 匿名クラス内に定義されているメソッド

3.2.1 サブステップ 2-1：ソースコードからの情報取得

開発者がメソッド名の変更を意図して指定したメソッドの名前の構成要素とそのコンテキストを取得する．ただし，指定されたメソッドの名前が動詞と目的語の組となっていない場合，名前の構成要素は取得しない．また，コンテキストについては，サブステップ 1-1 と異なり，以下に挙げたメソッドが指定された場合でも取得を行う．

- 内部クラス内に定義されているメソッド
- Getter および Setter

3.2.2 サブステップ 2-2：命名相関ルールの検索

メソッド名変更候補を提示するために必要な命名相関ルールを検索する．サブステップ 2-1 で取得したメソッドのコンテキストを検索条件として，ステップ 1 で抽出した命名相関ルールのうち，条件部がコンテキストの部分集合となる命名相関ルールをすべて取得する．

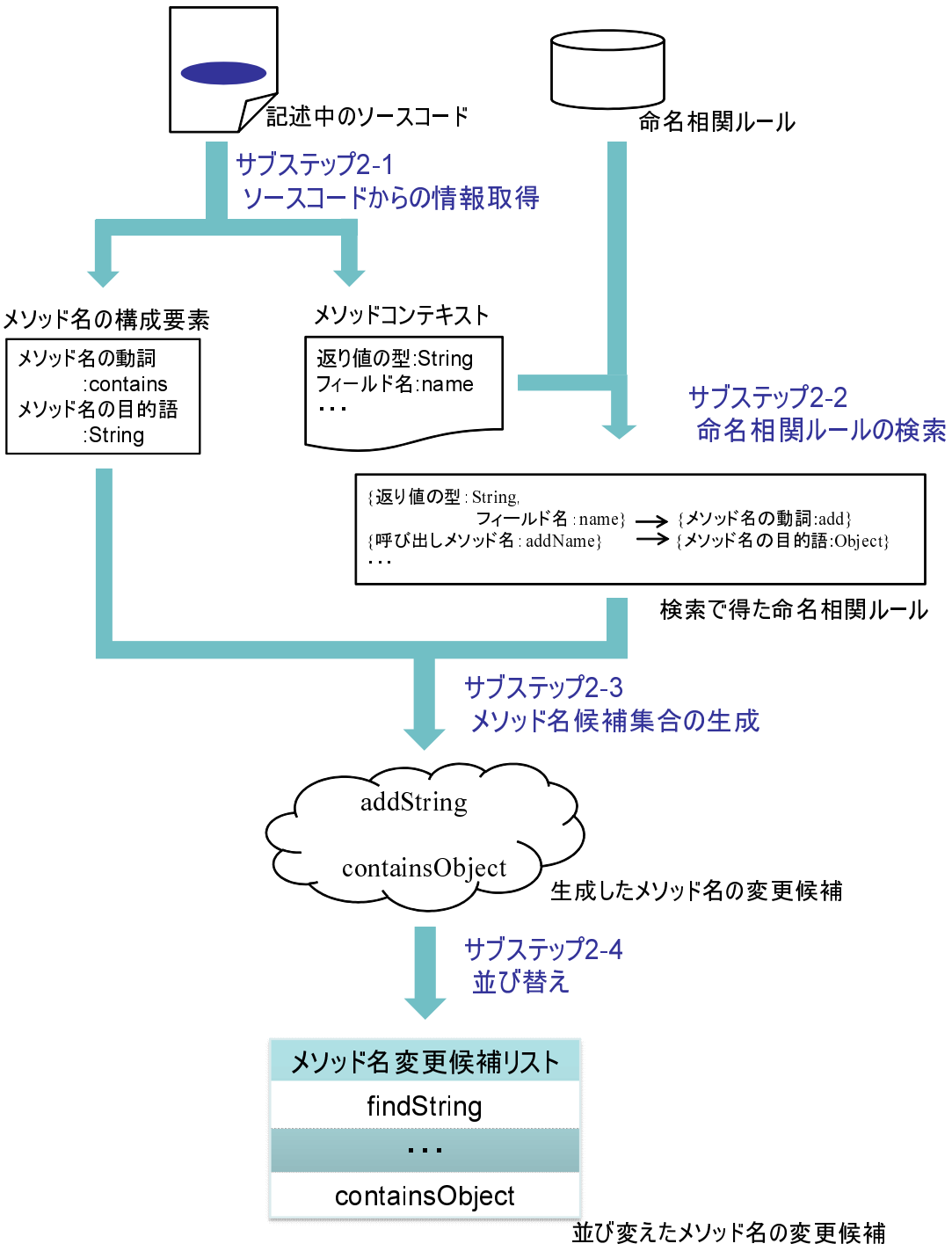

図 4: メソッド名生成部の概要図

3.2.3 サブステップ 2-3：メソッド名候補集合の生成

サブステップ 2-2 で得た命名相関ルールの帰結部と，サブステップ 2-1 で取得したメソッド名の構成要素を用いてメソッド名の変更候補の集合を生成する．まず，サブステップ 2-2 で得た命名相関ルールを，帰結部に動詞と目的語が両方出現しているもの，帰結部に動詞が出現し，かつ命名相関ルールの取得元のメソッド名がその動詞のみであるもの，それ以外の 3 つに分類し，それぞれの命名相関ルールの集合に対し以下に示す方法でメソッド名の変更候補を生成する．

1 つ目に分類される命名相関ルールについては，同じ帰結部から得られる動詞と目的語を組み合わせて 1 つのメソッド名を生成し，動詞のみのメソッド名として変更候補とする．

2 つ目に分類される命名相関ルールについては，帰結部から得た動詞を目的語と組み合わせずに，メソッド名の変更候補とする．

3 つ目に分類される命名相関ルールについては，帰結部とサブステップ 2-1 で取得したメソッド名の構成要素を総当たりで組み合わせてメソッド名を生成する．具体的には，帰結部に出現する動詞の集合を V, 帰結部に出現する目的語の集合を O, メソッド名の構成要素中の動詞を m_v, 目的語を m_o とおいた時，以下の組み合わせで作成される名前をすべて列挙してメソッド名の変更候補とする．ただし，以下の式において,「+」は単語の連結を意味するものとする．

- $v + o (v \in V + o \in O)$
- $m_v + o (o \in O)$
- $v + m_o (v \in V)$

ただし，命名相関ルールにもメソッドの構成要素にも目的語が出現しない場合など，動詞と組み合わせる目的語が存在しない場合には，目的語を「*」とする．同様に，目的語に対して組み合わせる動詞がない場合も動詞を「*」とする．

3.2.4 サブステップ 2-4：並び替え

生成されたメソッド名候補の集合の要素を，メソッドに対して適切であるメソッド名が上位に提示されるように並び替え，メソッド名変更候補リストとする．具体的には，3.3 節で述べる各メソッド名候補に対する評価式から算出される値の降順に並び替えてリストとする．

3.3 メソッド名候補の評価式とパラメータチューニング

本節では，メソッド名候補を並び替える際に用いる評価式と，評価式中で基準値に対する重みとして用いられている定数の値を決定するパラメータチューニングについて説明する．

3.3.1 メソッド名候補の評価式

メソッド名候補の評価式は，適切なメソッド名の基準に沿って設定した複数の値に対してそれぞれ適当な重みを与え，それらを線形に組み合わせた式である．

適切なメソッド名の基準には以下の 2 つを設定した．

基準 1 よい命名相関ルールが使われているかどうか

基準 2 生成されたメソッド名が適切であるかどうか

基準 1 に沿った値として，以下の値を用いる．

C： メソッド名候補の生成に用いた命名相関ルールの確信度

S： メソッド名候補の生成に用いた命名相関ルールの支持度

Z： サブステップ 2-2 で用いた検索条件の数に対する，命名相関ルールの条件部の検索条件と合致した要素数の割合

なお，メソッド名候補が複数の命名相関ルールの組み合わせで生成された場合は，C, S, P は動詞と目的語の元となったすべての命名相関ルールから得られる値の相加平均を用いる．

基準 2 に沿った値として，以下の値を用いる．基準 2 に沿った値はすべて真偽値であり，真の場合 1，偽の場合 0 の値をとる．

E： 動詞と目的語が同一の命名相関ルールから得られたかどうか

V： 生成したメソッド名候補の動詞が，メソッド本体の呼び出しメソッド名の動詞に使われているかどうか

O： 生成したメソッド名候補の目的語が，メソッド本体に出現する識別子の命名に用いられているかどうか

R： 動詞と目的語の組としてメソッド名候補が生成されているかどうか

以上までで挙げた値に適当な重みを与えて線形に組み合わせて，評価式 (3) を作成した．ただし，S は他の値と分布の形が異なるため，自然対数を取ることで値を調整した．式中の c, s, z, e, v, o, r は，それぞれ，C, S, P, E, V, O, R に対する重みであり，3.3.2 節で述べるパラメータチューニングによって決定される (0,1] の範囲で値を持つ定数である．

$$(c \times C) + (s \times \log_e S) + (z \times Z) + (e \times E) + (v \times V) + (o \times O) + (r \times R) \qquad (3)$$

3.3.2 パラメータチューニング

式 (3) で用いる重みを決定するために行うパラメータチューニングについて説明する．本パラメータチューニングには Simulated Annealing[6] を用いる．Simulated Annealing は金属工学における焼きなましを模した大域的最適化問題への確率的メタアルゴリズムであり，この手法によって広大な探索空間内に与えられた関数の大域的最適解の近似を得ることができる．

本パラメータチューニングでは，既存のソースコードに記述されているメソッドを対象に，メソッド名を一旦変更して提案手法を用いてメソッド名候補の集合を生成して並び替えを行ったときに，元のメソッド名，または，元のメソッド名で用いられている動詞か目的語を用いているメソッド名が上位に提示できるように重みの最適値の近似解を求める．なお，パラメータチューニングで用いるソースコードは命名相関ルールの生成に用いていないオープンソースコード群とする．さらに，パラメータチューニングの際に対象とするメソッドは，メソッド名が小文字の動詞のみであるメソッド，および，メソッド名が camelCase で表現された動詞と目的語の組となっているメソッドのみとする．

パラメータチューニングにより重みの最適値の近似解を求める具体的な手順は以下のとおりである．なお，w を現在のパラメータ (c, s, z, e, v, o, r) のベクトル，$f(w)$ を式 (3) の重みとして w を設定した式，$E(w)$ を w の評価値とおく．

1. w 中の c, s, z, e, v, o, r に (0,1] の範囲でそれぞれに適当な値を与え，現在の重みとする．
2. 現在の重み w の近傍 w' を設定する．w' は w に含まれる c, s, z, e, v, o, r のうち，どれか 1 つをランダムに選択し，微小値 Δw 増加させるか減少させるかしたものである．
3. $E(w) > E(w')$ なら $w \leftarrow w'$ とする．$E(w) \leq E(w')$ の場合は遷移確率 p に従い，$w \leftarrow w$ または $w \leftarrow w'$ とする．
4. p が十分に小さければ現在の w を最適値の近似解とする．それ以外の場合，p を微小値 Δp 減少させて 2 に戻る．

重み w の評価値 $E(w)$ は，パラメータチューニング用に与えられたソースコードに含まれるメソッドの集合 M に対して，$m \in M$ である各メソッド m に対する重みの評価値 $E_m(w)$ の総和である．$E_m(w)$ は以下の手順で求める．なお，メソッド m はその名前 n_m を動詞 v_m と目的語 o_m に分解できるものとする．

1. 提案手法を用いて以下の 3 つのメソッド名候補のリストを生成する．3 つのリストはそれぞれ，メソッド名に使われている動詞が悪かったときの変更候補，メソッド名に

使われている目的語が悪かった場合の変更候補，メソッド名全体が悪かった場合の変更候補，に対応している．

L_1：メソッド名に使われている動詞を「$」に変更して生成したメソッド名候補集合を，$f(w)$ を用いて並び替えたリスト

L_2：メソッド名に使われている目的語を「$」に変更して生成したメソッド名候補集合を，$f(w)$ を用いて並び替えたリスト

L_3：メソッド名全体を「$」に変更して生成し，並び替えたメソッド名候補集合を，$f(w)$ を用いて並び替えたリスト

2. 各リストに対して，評価値 $E'_m(w, L_j)$ を求める．$E'_m(w, L_j)$ は，次の 3 つの値の総和である．

   - 動詞に v_i が出現するメソッド名が L_j に最初に出現する順位の逆数（L_j に出現しない場合は 0）
   - 目的語に o_i が出現するメソッド名が L_j に最初に出現する順位の逆数（L_j に出現しない場合は 0）
   - n_i が L_j に最初に出現する順位の逆数（L_j に出現しない場合は 0）

3. $E_m = \sum_{j=1..3}^{j} E'_m(w, L_j)$ を求める．

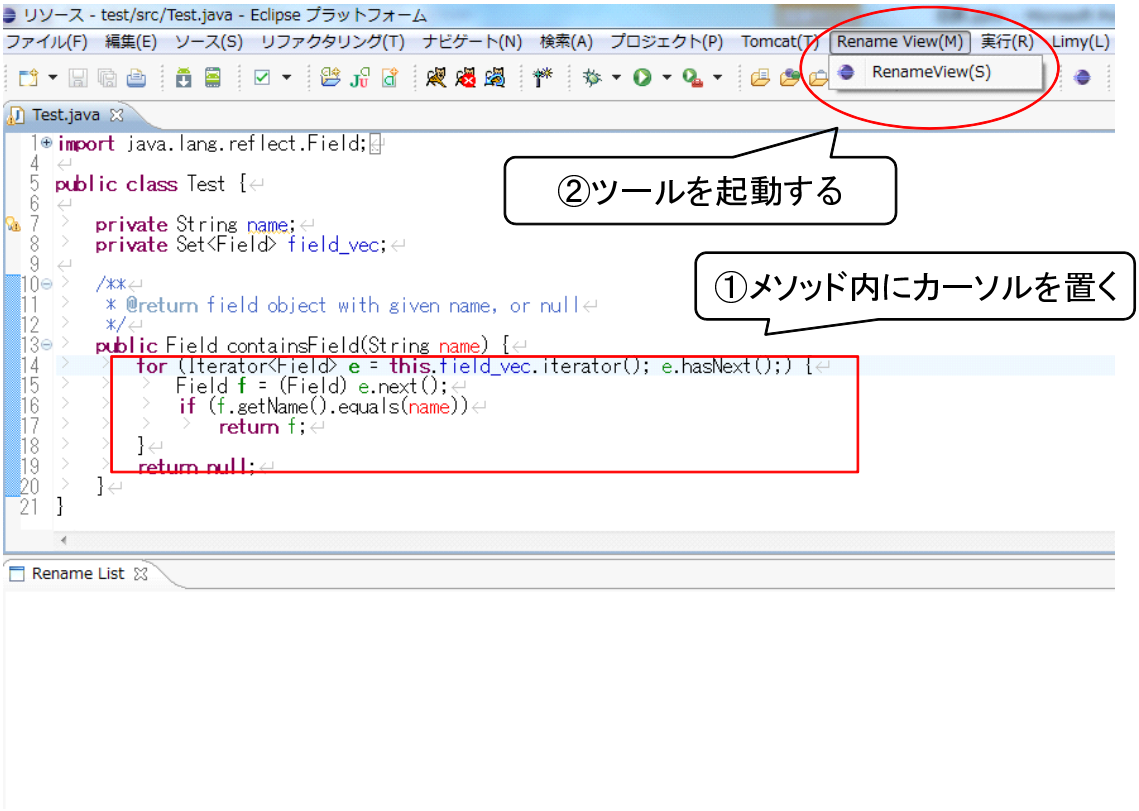

図 5: Eclipse の Java エディタでメソッド定義位置にカーソルがある状態

4 ツールの実装

本手法を，統合開発環境 Eclipse 上のプラグインとして，プログラミング言語 Java を対象に実装を行った．本ツールは，開発者が名前を変更しようとしているメソッドを指定してツールを起動すると，メソッド名の変更候補がリストで表示される．

開発者は，以下の手順でツールを利用する．まず，ツールの開発者がメソッド名の変更候補を見たいときに，Java エディタ上に記述されている対象とするメソッド内部にカーソルを置いてツールを起動する (図 5)．これにより，メソッド名の候補リストがビュー上に表示される (図 6)．次に開発者はビュー上に提示されたメソッド名候補を見てメソッド名の変更を行う．開発者は，ビュー上のメソッド名を元に別の名前に変更するか，ビュー上のメソッド名に変更する．ビュー上に表示されたメソッド名に変更する場合，ビュー上のメソッド名をダブルクリックすることで，選択したメソッド名がテキストボックスに記述された状態で Eclipse の標準機能である名前変更機能のダイアログが表示される (図 7)．これにより開発者はすぐに名前の変更を行うことができる．

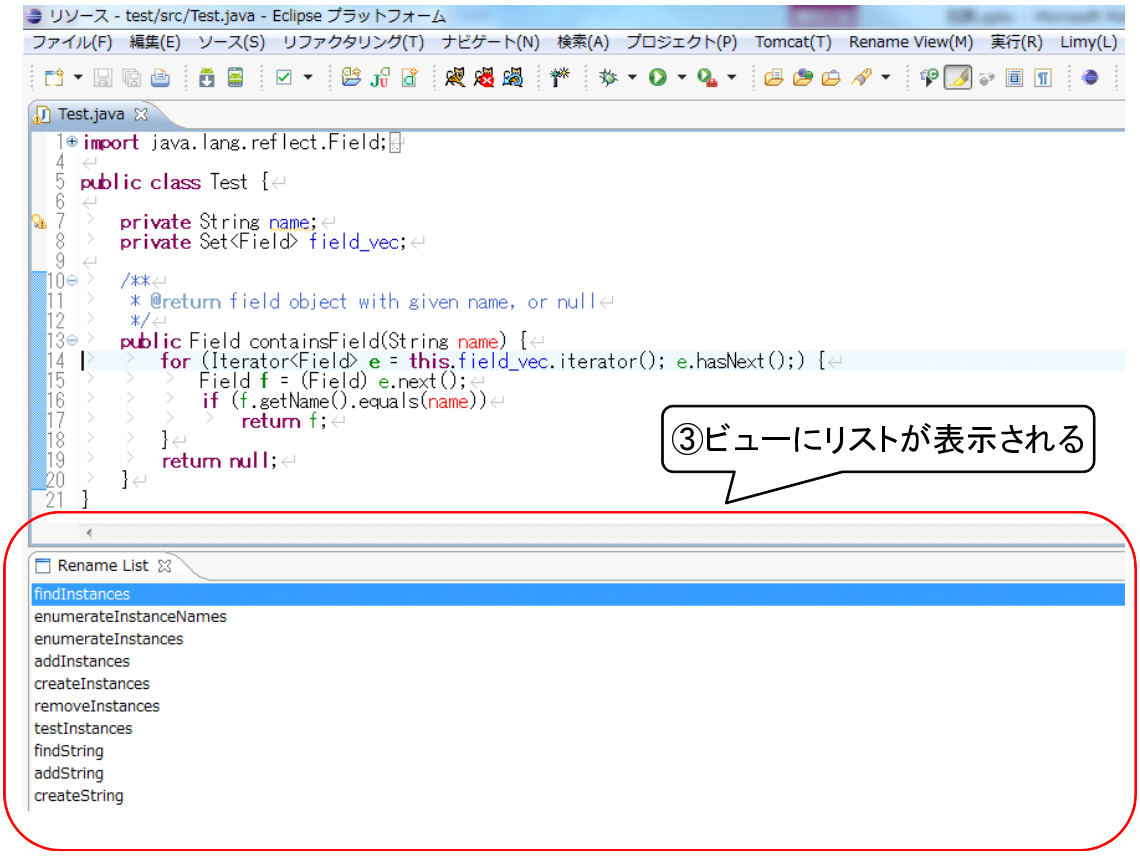

図 6: メソッド名の変更候補がビュー上にリストで表示された状態

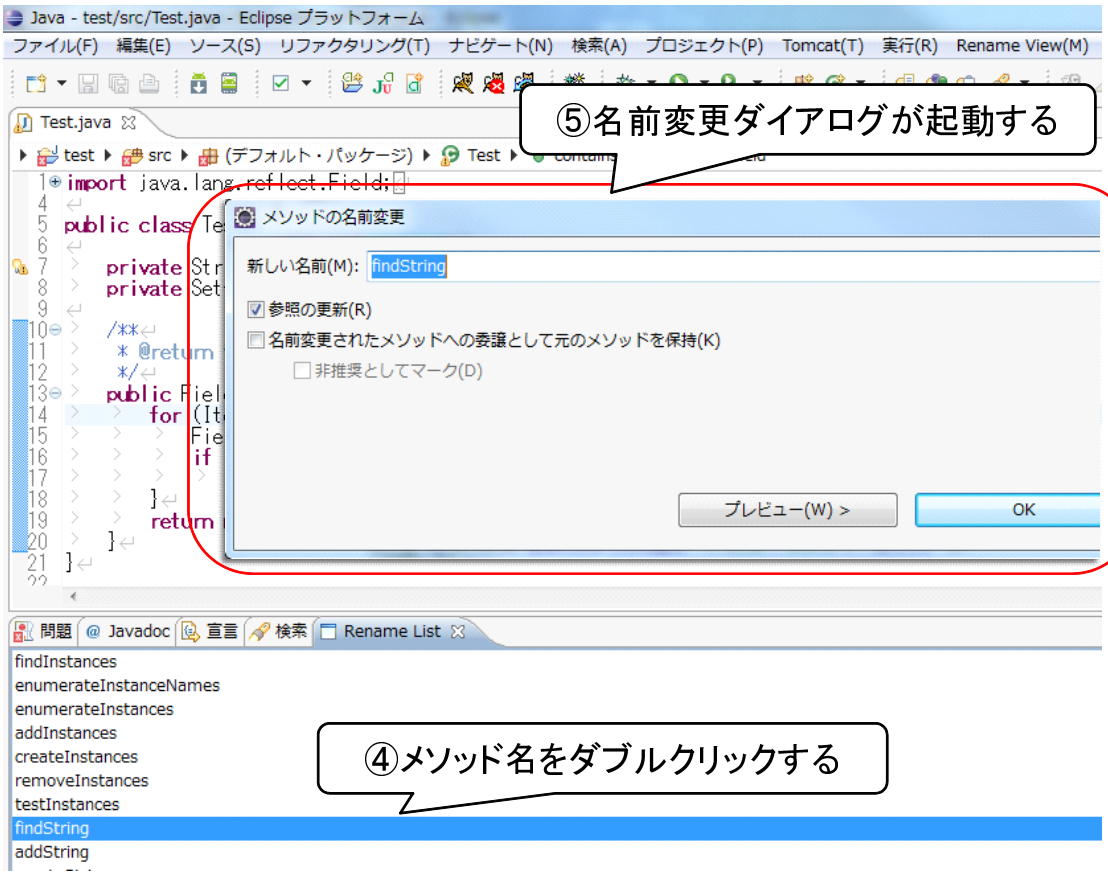

図 7: リスト上のメソッド名候補をダブルクリックして命名変更のダイアログが起動した状態

5　評価実験

提案手法が適切なメソッド名を提示しているかどうかを確認するために，実験を行った．評価は，本手法で生成したメソッド名の変更候補リスト中での正解メソッド名の出現位置を調査することで行った．正解メソッド名として，著名なオープンソースソフトウェアで命名されたメソッド名を採用した．このようなメソッド名を正解とした理由は，著名なオープンソースソフトウェアは多くの人の目に触れていると考えられるためである．本手法で提示できる動詞のみのメソッド名および動詞と目的語の組として命名されているメソッド名のみを評価実験の対象とした．

5.1　実験準備

評価実験の準備のために，評価実験で用いる命名相関ルールの作成とパラメータチューニングを行った．以降それぞれについて説明する．

5.1.1　命名相関ルールの作成

命名相関ルールを作成するのためのソースコード集合として，付録に掲載している全 445 個のソフトウェアのソースコードを用いた．作成された命名相関ルールの総数は，165285 個である．

5.1.2　パラメータチューニング

命名相関ルール作成後，3.3.2 節で述べたパラメータチューニングを行った．パラメータチューニングの際に用いたソースコード集合は，Axion 1.0 Milestone 2 である．遷移確率 p の初期値は 0.99 であり，試行 1 回あたりの変動値 Δp は，その時点の p の値によって $\Delta p = 0.01p$ とした．重みを変動させる際の微小値 Δw は初期値を 0.3 として，その後 1 回試行を行うごとに，次回試行のための微小値 $\Delta w'$ を，$\Delta w' = 0.99\Delta w$ として求めた．最終的に得られた重みは表 1 のとおりである．結果，906 回の試行が行われた．

5.2　評価方法

提案手法を用いて生成したメソッド名の変更候補の提示リストに含まれる正解メソッド名の位置を調べることで手法を評価する．

評価実験の対象として，ArgoUML と jEdit の 2 つを用いた．表 2 に，対象とする 2 つのソフトウェアについての詳細を示す．これらは著名なオープンソースソフトウェアであり，他の研究でも評価に用いられているため，評価実験で対象にすることにした．

表 1: パラメータチューニングの結果得られた重み

c	0.9235493974691869
s	0.47452412788335385
z	0.6138267759466827
e	0.42950352851679796
v	0.7932739582490955
o	0.6077403953426064
r	0.3946858036161451

表 2: 評価に用いるソフトウェア

	バージョン	総行数	メソッド総数	対象メソッド数
ArgoUML	0.28.1	367052	15008	5542
jEdit	4.3 .1	176556	6299	2161

対象としたメソッドは，本手法で生成できる形のメソッド名がついていて，かつ，そのソフトウェア内で命名されている可能性が高いものとした．具体的には，対象ソフトウェアの開発者がつけたと考えられる，動詞のみ，もしくは，動詞と目的語の組となっている名前がついているメソッドを対象にする．外部で定義された不適切な名前がついているメソッドをオーバーライドしている可能性があるメソッドを除去するために，今回の評価実験では対象ソースコード中の@Override の修飾子がついたメソッドは対象から除外した．また，本研究の手法では動詞が get および set のメソッド名は提示しないので，メソッド名の動詞が get または set のメソッドも評価実験の対象としない．

実験ではまず，3.3.2 節で説明したパラメータチューニングと同様に，メソッド名を以下の 3 通りの方法で変更し，メソッド名の候補リストを生成する．

L_1： メソッド名に使われている動詞を「$」に変更して生成したメソッド名候補集合を，$f(w)$ を用いて並び替えたリスト

L_2： メソッド名に使われている目的語を「$」に変更して生成したメソッド名候補集合を，$f(w)$ を用いて並び替えたリスト

L_3： メソッド名全体を「$」に変更して生成し，並び替えたメソッド名候補集合を，$f(w)$ を用いて並び替えたリスト

これらのリストを対象に，適切なメソッド名が生成できているかどうか，適切なメソッド名が上位に提示できているかを確認する．

適切なメソッド名を生成できているかどうかを確かめるために，以下の評価値を設定した．

$FOUND_V$ リスト中に元のメソッド名の動詞が含まれているメソッドの数

$FOUND_O$ リスト中に元のメソッド名の目的語が含まれているメソッドの数

$FOUND_N$ リスト中に元のメソッド名そのものが含まれているメソッドの数

適切なメソッド名が上位に提示できているかを確かめるために，以下の評価値を設定した．

$RANK_V$ 元のメソッド名の動詞がリスト内で最初に出現する順位

$RANK_O$ 元のメソッド名の目的語がリスト内で最初に出現する順位

$RANK_N$ 元のメソッド名そのものがリスト内で最初に出現する順位

5.3 結果

入力に与えたメソッドに対して生成したメソッド名の候補数を ArgoUML については図 8 に，jEdit については図 9 に示す．平均で，ArgoUML で 62109 個，jEdit で 81476 個のメソッド名が生成される．ArgoUML で 89.2%，jEdit で 91.9%以上のメソッドが 1000 以上のメソッド名の候補を生成していることがわかる．これは，生成の際に動詞と目的語を総当たりで組み合わせるからだと考えられる．

適切なメソッド名を提示できているかどうか

2 つのソフトウェアについて $FOUND_V$，$FOUND_O$，$FOUND_N$ を示す．表 3 には元のメソッド名の動詞を変更した場合，表 4 には元のメソッド名の目的語を変更した場合，表 5 は元のメソッド名の動詞と目的語を変更した場合をそれぞれ表す．

示した表から，入力に用いた 2 つのソフトウェアにおいて，同様の傾向が得られたことがわかる．表 3 から，変更前のメソッド名の動詞が正しくないときでも，87.1%のメソッドで正解のメソッド名の動詞を生成できることがわかる．また同様に，表 4 から，目的語が正しくないときでも 22.2%のメソッドで元のメソッド名の目的語を生成できていることがわかる．表 5 から，動詞も目的語も正しくないメソッド名がついていた場合，19.0%のメソッドで正解のメソッド名を生成できていることがわかる．

入力に与えたメソッドのうち，元の動詞は変更前の動詞がない場合でも 87.1%の動詞を生成できたが，目的語やメソッド名全体だと，元のメソッド名が生成できた割合は 19.0%，目的語が生成できた割合は 21.9%であった．これは，動詞が多くのソフトウェアに共通した意

表 3: 対象としたメソッドのうち，元のメソッド名の動詞を「$」に変更して生成した提示リストに正解が含まれている数

	$FOUND_V$	$FOUND_O$	$FOUND_N$
ArgoUML	4871(87.9%)	5402(97.5%)	4825(87.1%)
jEdit	1842(85.2%)	2144(99.2%)	1799(83.2%)

表 4: 対象としたメソッドのうち，元のメソッド名の目的語を「$」に変更して生成した提示リストに正解が含まれている数

	$FOUND_V$	$FOUND_O$	$FOUND_N$
ArgoUML	5388(97.2%)	1048(18.9%)	1046(18.9%)
jEdit	2133(98.7%)	637(29.5%)	628(29.1%)

味を持って使われており，目的語に用いられやすい名詞に対して動詞の数は少なく，メソッドの命名に用いられる動詞は限られているからだと考えられる．目的語が動詞に比べて生成できていないのは，ドメインやプロジェクトによって用いられる単語の傾向が異なるため，生成した命名相関ルールがメソッド名で用いられる目的語を網羅できていないためだと考えられる．また，目的語が命名相関ルールから生成できていないことで元のメソッド名も生成できていないと考えられる．

適切なメソッド名が上位に提示できているかどうか

評価値 $RANK_V$，$RANK_O$，$RANK_N$ についてソフトウェアとメソッド名の変更方法ごとに図に示す．元のメソッド名の動詞を変更した場合の ArgoUML での結果を図 10 に，jEdit での結果を図 11 に示す．同様に，メソッド名の目的語を変更した場合の結果を図 12 と図 13 に，メソッド名の動詞と目的語の両方を変更した場合の結果を図 14 と図 15 に，それぞれ

表 5: 対象としたメソッドのうち，元のメソッド名の動詞と目的語をそれぞれ「$」に変更して生成した提示リストに正解が含まれている数

	$FOUND_V$	$FOUND_O$	$FOUND_N$
ArgoUML	4871(87.9%)	1048(18.9%)	928(16.7%)
jEdit	1842(85.2%)	637(29.5%)	539(25.0%)

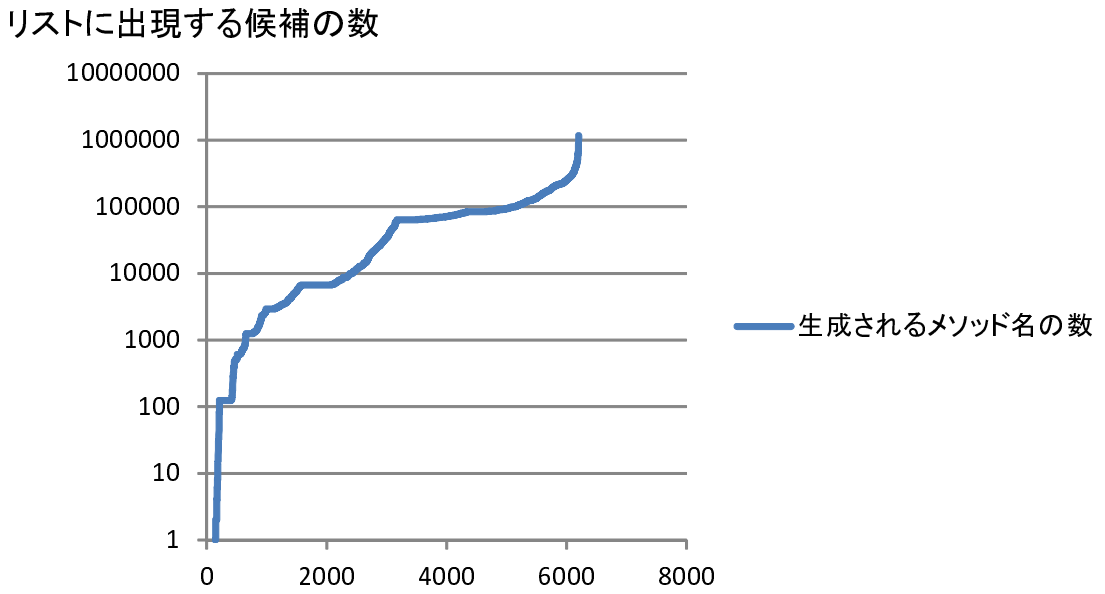

図 8: ArgoUML の対象メソッドに対してメソッド名の候補を生成したときの候補数

示す．それぞれの図は，提示リストに含まれる正解の出現順位を求めておき，x 軸に正解の出現順位の昇順ソートした各メソッドを並べ，y 軸に正解の出現順位をプロットしたものである．

入力に与えた 2 つのソフトウェアについて，同様の傾向が得られた．図 14，図 15 から，元のメソッド名はリスト中のほとんどが 100 位以下に提示されているものの，元の動詞がリスト内に提示されているメソッドのうち 52.9%，入力に用いたメソッドの 46.1%がリストの 100 位までに提示されていることがわかる．

結果として，メソッド名の動詞に関しては，正しくない動詞がついている場合でも入力全体の 46.1%のメソッドで正しい動詞が 100 位以上に提示できていることがわかった．メソッド名の候補が平均でも 60000 以上生成されていることから，100 位以上は十分高いと考える．

メソッド名の目的語に関しても，正しくない目的語がついている場合でも入力全体の 11.0%が 100 位以上に提示されている．目的語は命名相関ルールから元の目的語が生成されることが少ないため，生成できる数，100 位以上に提示できる数がともにが少なくなると考えられる．

元のメソッド名は，100 位以上に提示できていない．元のメソッド名自体は生成できているので，パラメータチューニングの入力データの影響で候補の並び方が偏っているのではないかと考える．

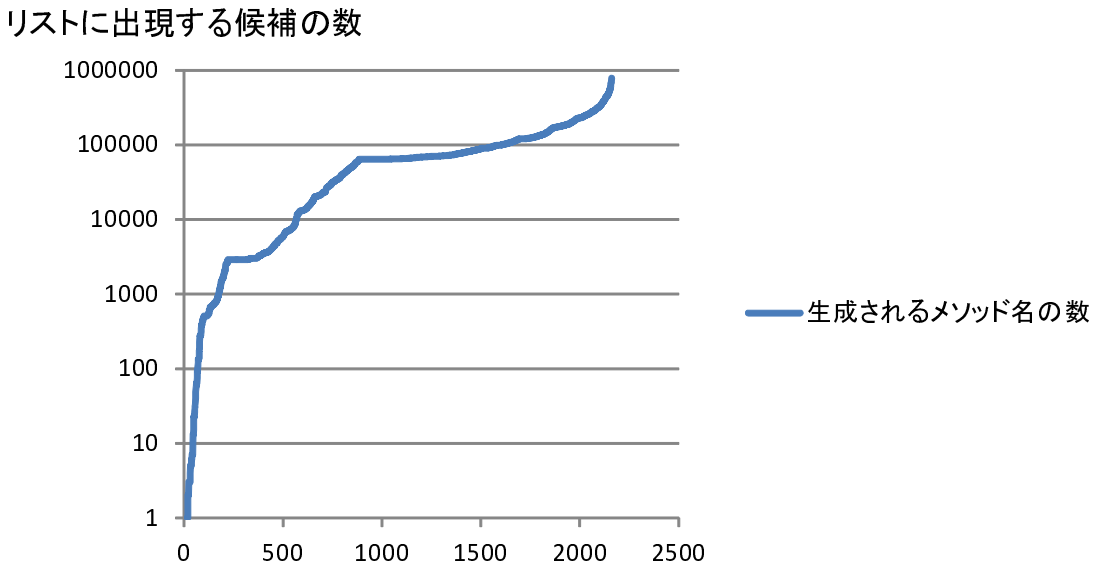

図 9: jEdit の対象メソッドに対してメソッド名の候補を生成したときの候補数

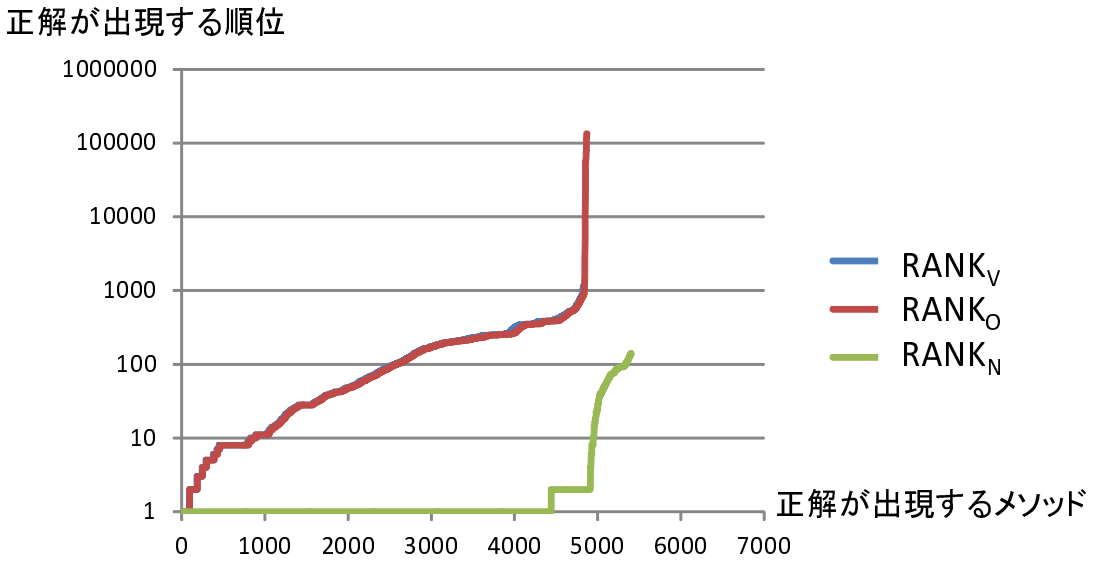

図 10: ArgoUML の対象メソッドに対してメソッド名の動詞を「$」に変更したとき，正解が出現する順位の分布

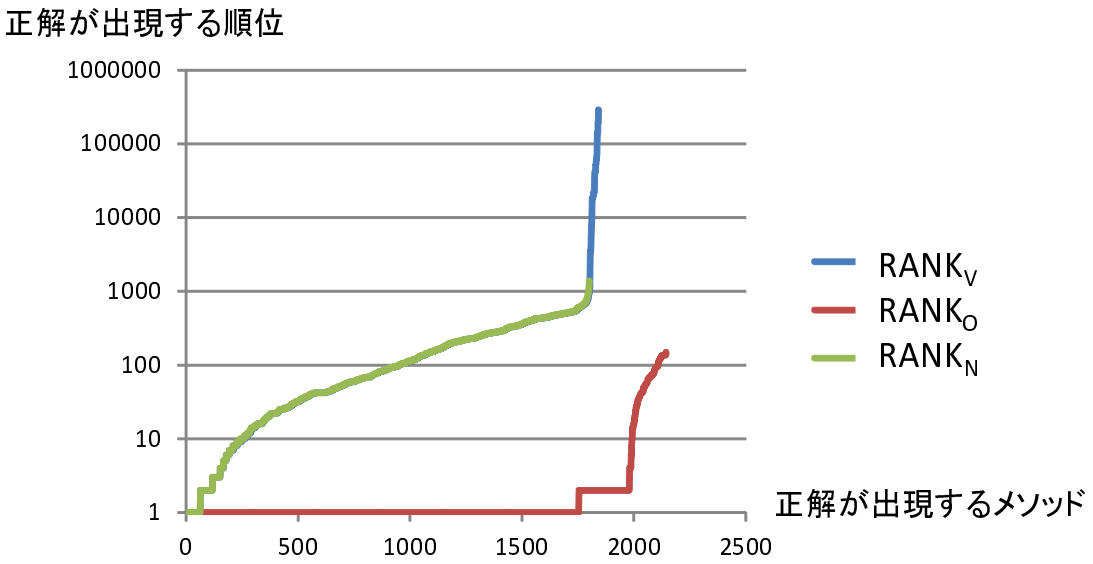

図 11: jEdit の対象メソッドに対してメソッド名の動詞を「$」に変更したとき，正解が出現する順位の分布

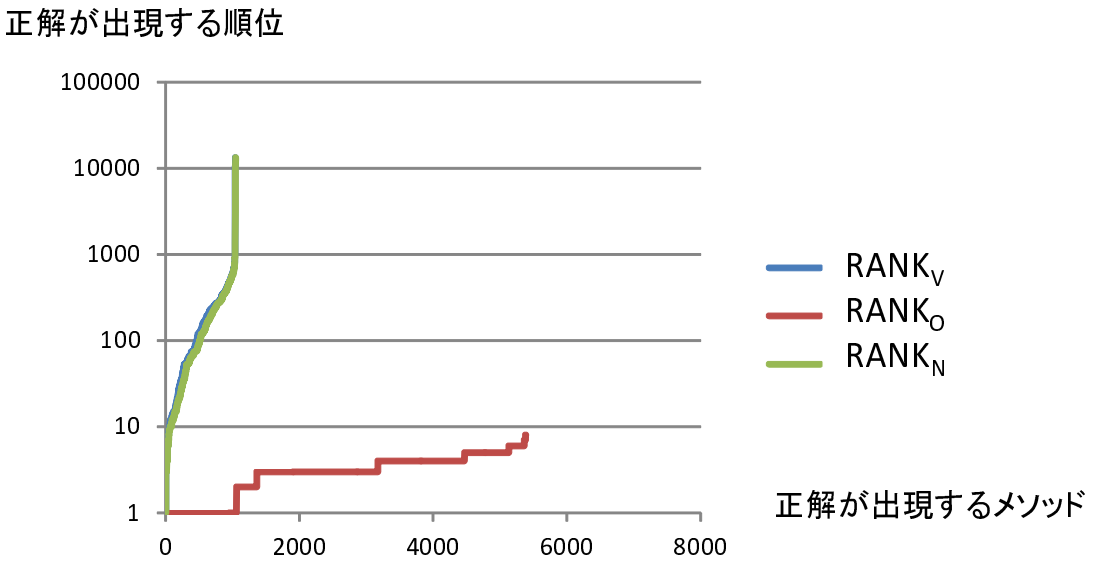

図 12: ArgoUML の対象メソッドに対してメソッド名の目的語を「$」に変更したとき，正解が出現する順位の分布

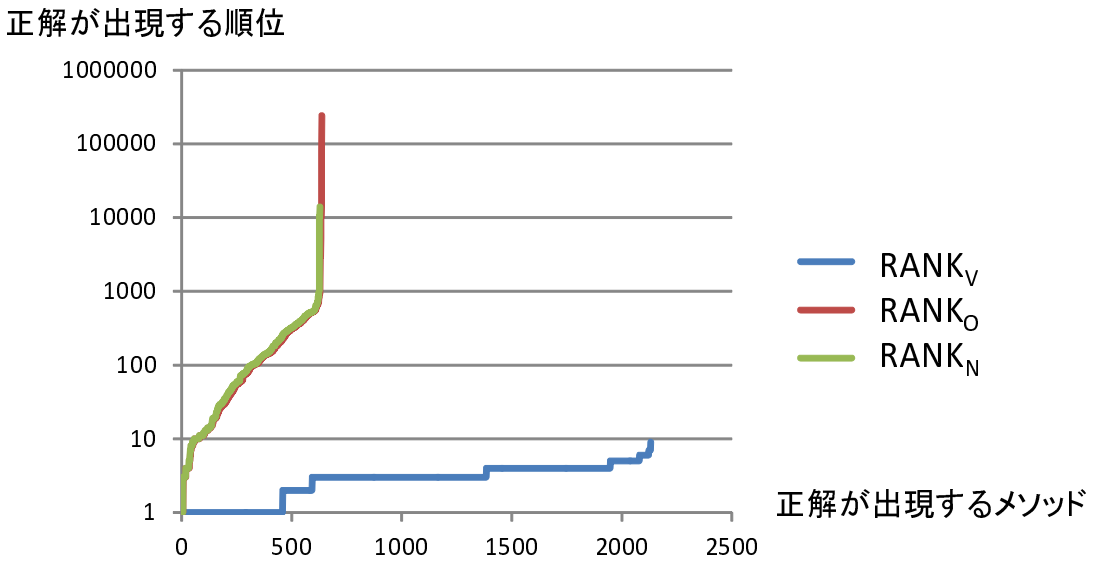

図 13: jEdit の対象メソッドに対してメソッド名の目的語を「$」に変更したとき，正解が出現する順位の分布

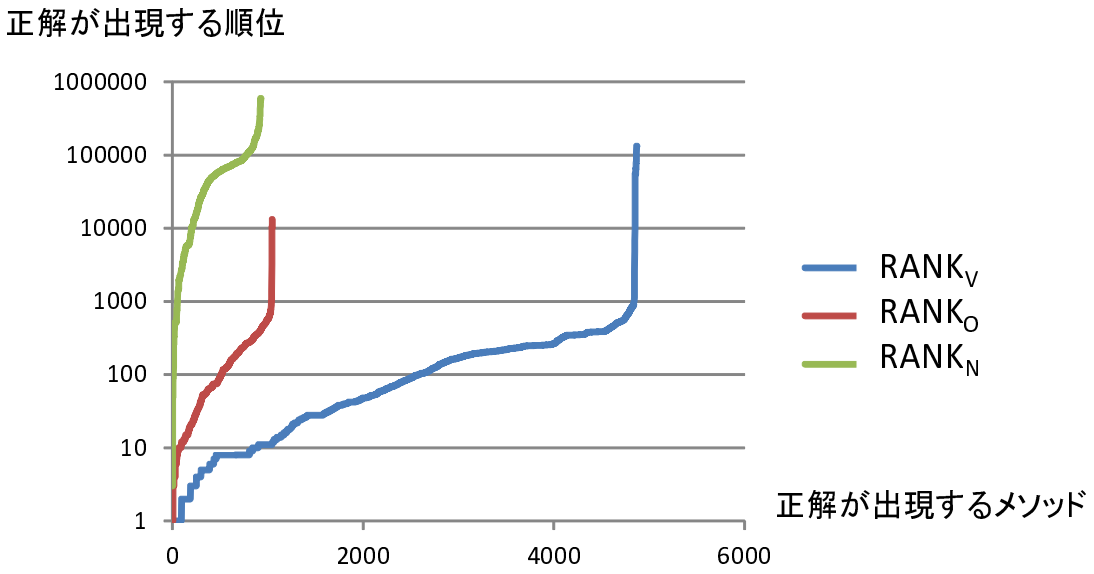

図 14: ArgoUML の対象メソッドに対してメソッド名の動詞と目的語をそれぞれ「$」に変更したとき，正解が出現する順位の分布

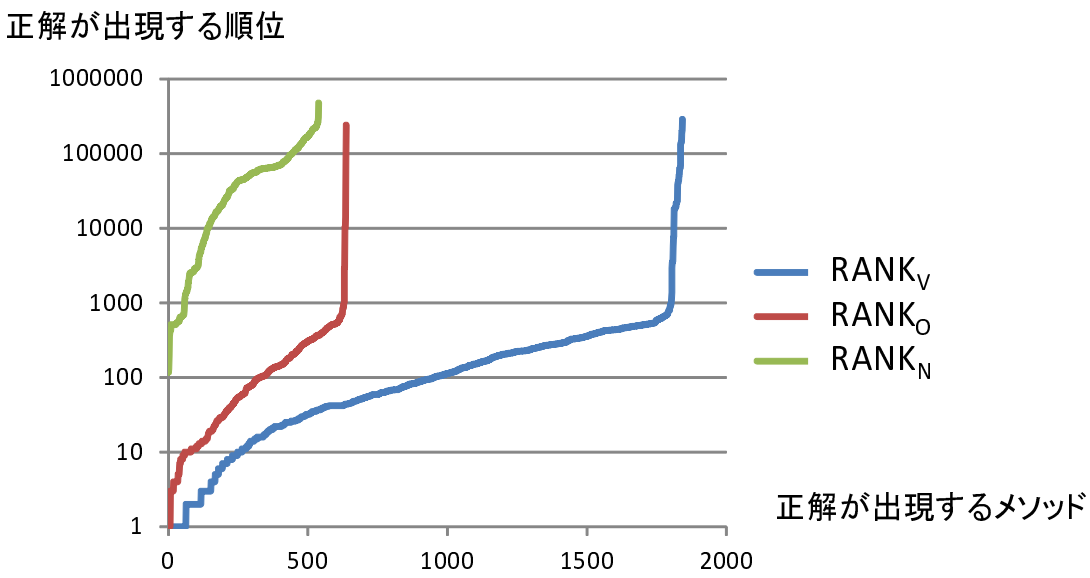

図 15: jEdit の対象メソッドに対してメソッド名の動詞と目的語をそれぞれ「$」に変更したとき，正解が出現する順位の分布

5.4 妥当性の脅威

本研究で行った評価実験に関する妥当性の脅威を検証する．

命名相関ルールの作成には，特定のソースコード集合を用いた．ソースコード集合は，多様性を高めるように選択したが，意図しない偏りがある可能性がある．そのため，作成に用いるソースコード群を変更した場合，実験結果が大きく変わるかもしれない．

並び替えの評価式に与える重みを決定するためにパラメータチューニングを行った．パラメータチューニングで用いたソースコード群は Axion 1.0 Milestone 2 であるが，入力に与えるソースコード群を大きくしたり変更したりすることでパラメータチューニングの結果が変わる可能性がある．

評価実験では，ArgoUML と jEdit を対象に手法を評価した．2 つのソフトウェアはどちらも著名なオープンソースソフトウェアであり，他の研究でも評価に用いられているため評価実験の対象とした．しかし，そのどちらも GUI アプリケーションであり，対象としたドメインが偏っている可能性がある．また，対象が 2 つのソフトウェアであるため，より多くのソフトウェアを対象とすることで結果が変わる可能性がある．

評価実験では，対象ソフトウェア内で命名されている可能性が高いメソッド名を対象とするために，@Override の修飾子がついたメソッドを対象外とした．これにより，外部で定義された不適切な名前がついているメソッドをオーバーライドしている可能性があるメソッド

を除去することができる．しかし，全てのオーバーライドしているメソッドに@Override の修飾子がついているとは限らないので，オーバーライドしているメソッドを全て取り除けていない可能性がある．

適切なメソッド名が手法により生成したリストの上位に提示できているかどうかを確かめる方法として，正解と定めたメソッド名やその動詞，目的語がリスト内で最初に出現する位置を評価した．評価のために生成したリスト内には，異なる命名相関ルールから生成した同じメソッド名の重複を許して提示している．したがって，重複を許さず提示した場合，生成されるメソッド名の数や順位が変わる可能性がある．

6 関連研究

メソッドの命名が難しいという問題を解決する手法が提案されている．本節ではそれらの関連研究について述べる．

Høst らは，メソッド名の動詞に着目して，不適切な動詞がついているメソッド名に対して，適切な動詞を理由とともに提示して開発者のメソッドの命名を支援する手法を提案している [4]．この手法は，あらかじめメソッド名の動詞と本体の内容を対応付けたルールを作成しておき，既についているメソッド名の動詞と本体がこのルールを満たしていない場合，不適切である理由とともにメソッド本体に対応した動詞を提示する．また，手法を実現したツールを作成してオープンソースプロジェクトとして公開している [5]．この手法で用いられるルールは人の手で作られているため，ルールの作成には手間と時間がかかる．また，メソッド本体が記述されているメソッドのうち，このルールに記載されていないメソッド名の動詞を用いて命名されているメソッドはメソッド名を提示することができない．この研究に対して本手法では，メソッド本体とメソッド名を対応付ける命名相関ルールを相関ルールマイニングにより自動で作成するため，ルールの数を増やすことが容易である．作成した命名相関ルールから，多様な動詞と目的語の組を生成する．生成するメソッド名が適切であるかどうかを判定するかわりに，評価式を用いて並び替えを行う．また，メソッド名の変更候補は，本体が記述されているという条件を除いて制約なく提示できる．

鹿島らは，オブジェクト指向プログラムのソースコードに出現するメソッド名から動詞-目的語関係を抽出して辞書を作成し [13]，鬼塚らはその辞書を用いて，新規に記述するメソッドの命名を支援する手法を提案した [12]．本手法と比較すると，鬼塚らの手法はメソッドの本体が記述されていない状態でもメソッド名を提示できるという利点があるが，メソッド本体が記述されている状態でもメソッド本体の内容を反映したメソッド名を生成することができない．本研究では，この研究から開発者がメソッド名の変更を行うとき参考にできるような多くのメソッド名を提示するというアイデアを得た．

7　まとめと今後の課題

本研究では，あらかじめ作成した命名相関ルールを用いて開発者がメソッド名の変更を検討するとき参考にできるように，メソッド名の変更候補をリストで提示する手法を提案した．命名相関ルールは，既存のソースコード群から相関ルールマイニングにより作成する．

オープンソースソフトウェアを用いた評価実験では，特にメソッド名の動詞に対して適切なメソッド名を生成できていることがわかった．また，そのうちのおよそ50%が100位以上に提示できていることを確認し，この手法により，適切なメソッドの候補を開発者に提示できる可能性があることがわかった．

今後の課題としては，提示される目的語の数の精度を高めること，適切なメソッド名を上位に提供すること，実際のソフトウェア開発へ適用することが挙げられる．

本手法では，命名相関ルールの作成の際，入力に用いるソースコードの種類や，出力される動詞や目的語を区別していない．しかし，評価実験から，動詞に比べて目的語が生成できていないことがわかる．これは，動詞に比べて目的語の数が多く，目的語に対して十分な命名相関ルールを生成できていないと考えられる．したがって，多様性が高い目的語に対応できるような手法を考える必要がある．

適切なメソッド名を上位に提示するために，並び替えにおけるパラメータチューニングの入力に用いるソースコード群を変更する方法が考えられる．パラメータチューニングに用いるソースコード群を変更することで並び替えの結果が変わると考えられる．適切なメソッド名をより上位に提示するために，パラメータチューニングの入力に用いるソースコード群の検証を行う必要がある．

また，本手法が実際のソフトウェア開発へどのくらい適用できるかの検証も重要な課題である．本手法は，開発者に対する支援を行うので，開発者にとってどのくらい役に立つ手法であるかを検証する必要がある．

謝辞

本研究において, 常に適切な御指導および御助言を賜りました大阪大学大学院情報科学研究科コンピュータサイエンス専攻井上克郎教授に心より深く感謝いたします.
本研究において, 適切な御指導および御助言を賜りました大阪大学大学院情報科学研究科コンピュータサイエンス専攻松下誠准教授に深く感謝いたします.
本研究において, 逐次適切な御指導および御助言を頂きました大阪大学大学院情報科学研究科コンピュータサイエンス専攻石尾隆助教に深く感謝いたします.
本研究において, 常時適切な御指導および御助言を頂きました筑波大学システム情報系早瀬康裕助教に深く感謝いたします.
本研究において, 数多くの御指導および御助言を頂きました大阪大学大学院情報科学研究科コンピュータサイエンス専攻眞鍋雄貴特任助教に深く感謝いたします.
本研究において, 様々な御指導および御助言を頂戴しました大阪大学大学院情報科学研究科コンピュータサイエンス専攻鹿島悠氏に深く感謝いたします.
本研究において, 様々な御指導および御助言を頂きました大阪大学大学院情報科学研究科コンピュータサイエンス専攻鬼塚勇弥氏に心から感謝いたします.
最後に，その他様々な御指導，御助言等を頂いた大阪大学大学院情報科学研究科コンピュータサイエンス専攻井上研究室の皆様に深く感謝いたします.

参考文献

[1] OpenNLP. `http://opennlp.sourceforge.net/`.

[2] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pp. 207–216, 1993.

[3] Einar W. Høst and Bjarte M. Østvold. The Programmer's Lexicon, Volume I: The Verbs. In *Proceedings of the 7th IEEE International Working Conference on Source Code Analysis and Manipulation*, pp. 193–202, 2007.

[4] Einar W. Høst and Bjarte M. Østvold. Debugging Method Names. In *Proceedings of the 23rd European Conference on Object-Oriented Programming*, pp. 294–317, 2009.

[5] Edvard K. Karlsen, Einar W. Høst, and Bjarte M. Østvold. Finding and fixing Java naming bugs with the Lancelot Eclipse plugin. In *Proceedings of the ACM SIGPLAN 2012 workshop on Partial Evaluation and Program Manipulation*, pp. 35–38, 2012.

[6] Scott Kirkpatrick, Charles Daniel Gelatt, and Mario Ponte Vecchi. Optimization by simulated annealing. *Science*, Vol. 220, pp. 671–680, 1983.

[7] Dawn Lawrie, Christopher Morrell, Henry Feild, and David Binkley. What's in a name? a study of identifiers. In *Proceedings of 14th International Conference on Program Comprehension*, pp. 3–12, 2006.

[8] Andrea De Lucia, Massimiliano Di Penta, Rocco Oliveto, Annibale Panichella, and Sebastiano Panichella. Using IR methods for labeling source code artifacts: Is it worthwhile? In *Proceedings of 20th International Conference on Program Comprehension*, pp. 193–202, 2012.

[9] Steve McConnell. *Code Complete, Second Edition*. Microsoft Press, Redmond, WA, USA, 2004.

[10] Gail C. Murphy, Mik Kersten, Martin P. Robillard, and Davor Čubranič. The emergent structure of development tasks. In *Proceedings of European Conference on Object-Oriented Programming*, pp. 33–48, 2005.

[11] Inc. Sun Microsystems. *Code Conventions for the Java Programming Language*, 1997.

[12] 鬼塚勇弥, 早瀬康裕, 石尾隆, 井上克郎. ソースコード中に出現する動詞-目的語関係を利用したメソッド名の命名支援手法. 信学技報, Vol. 111, No. 481, pp. 1–6, 2012.

[13] 鹿島悠, 早瀬康裕, 真鍋雄貴, 松下誠, 井上克郎. メソッドに用いられる動詞-目的語関係を収録した辞書構築手法の提案. 情報処理学会研究報告, Vol. 2010-SE-168, No. 12, pp. 1–8, 2010.

付録

実験に使用する命名相関ルールを作成するために用いたソースコード集合一覧

表 6: sourceforge.net

プロジェクト名	java ファイル数	総行数
acqlite/	2737	458733
adempiere/	4143	1155868
adito/	1982	306562
advanced-gwt/	111	19334
agilewiki/	726	89373
antlrv3ide/	197	20842
aria/	1272	266199
armedbear-j/	797	199355
artifactory/	272	30348
atunes/	520	94001
avr-eclipse/	210	44985
bt747/	324	87566
cdk/	210	47586
coopnet/	122	25729
cruisecontrol/	912	137566
daimonin/	20	4864
dapper/	271	39001
datacrow/	1764	348198
davmail/	42	8304
dbunit/	486	67912
dimdim/	1219	174027
dita-op/	69	8320
dita-ot/	125	22789
donnerlaparole/	170	27928
dooble/	469	124880
dozer/	568	41919
eclim/	308	35199
eclipsesql/	432	54875
ejbca/	1255	221922
erlide/	677	122251
findbugs/	1828	220400
firebird/	711	227708
fireflyclient/	207	31052
follow/	70	8333
freecol/	485	144748
freelims/	31	18289
freemarker/	377	60990
frostwire/	2626	547046
fuber/	238	30440
ganttproject/	1112	187868
gdcm/	5	650
geneontology/	1436	226718
gep/	123	13936
gham/	798	88893
gpsmid/	202	36448
hibernate/	3896	445894
hibernate4gwt/	270	25333

hipergate/	553	178003
hyperic-hq/	3097	521323
iplist/	7	2770
j2s/	1898	463020
jabref/	601	122809
jailer/	77	27115
java-ml/	258	34839
jaxodraw/	307	70495
jfreechart/	1130	288757
jgnucashlib/	204	93576
j-interop/	205	41578
jitterbit/	485	37527
jmol/	439	158875
jmri/	2249	352202
jnetpcap/	174	39751
josso/	472	60650
jpen/	57	5822
jpwsafe/	119	17327
jspx-bay/	115	15973
jupload/	80	20688
jvlt/	185	21349
jython/	837	276114
keros/	1	11
kmlcsv/	91	7713
kolmafia/	206	83327
liquibase/	643	53680
logicaldoc/	298	44638
makagiga/	835	167531
makumba/	388	67444
matrex/	1082	92132
megamek/	1328	322713
megameklab/	33	14266
microemulator/	310	44598
microlog/	105	14285
nekohtml/	38	12683
obpm/	1100	169144
octave/	79	18853
ogre4j/	1241	272848
omegat/	244	49305
openbravopos/	483	65343
opencsv/	17	2714
openmodeller/	10	716
opennms/	2600	480144
openqrm/	1564	240107
openxava/	2483	285838
opproject/	578	141513
osmius/	640	766786
oswing/	960	191123
pcgen/	2976	587511
pdfsam/	490	82032
phex/	809	167087
pmd/	1501	179756
posterita/	785	166038
redmin-mylyncon/	57	8259
regexpeditor/	30	2910

reprap/	204	64229
rodin-b-sharp/	4645	750102
romaframework/	1155	90594
saxon/	580	118117
sblim/	4923	1612974
schemaspy/	61	11605
smartfrog/	2954	433726
soapui/	902	127764
staf/	382	124210
storybook2/	220	40331
supercsv/	135	8894
sweethome3d/	161	59663
symmetricds/	271	31461
taverna/	3532	488151
tvbrowser/	961	198106
ucdetector/	148	10453
unicore/	5196	927124
unitils/	386	57919
veryquickwiki/	227	32662
villonanny/	53	7281
vnc-tight/	42	11554
webadmin/	26	9401
webcamstudio/	420	59181
wiki2xhtml/	92	18841
wintvcap-gui/	94	33390
wsmostudio/	114	27796
xbrlapi/	408	45933
xmm/	421	144993
xpstudio/	71	3907
yale/	2610	381842
zk1/	1266	177381
zkforge/	669	147504

表 7: apache.org

プロジェクト名	java ファイル数	総行数
ant/	1238	258385
beehive/	2933	369641
cayenne/	2989	352995
cocoon/	2844	494003
commons-attributes/	56	6887
commons-beanutils/	225	67779
commons-betwixt/	393	47654
commons-chain/	97	17702
commons-cli/	103	18952
commons-codec/	48	12784
commons-collections/	469	110889
commons-compress/	62	10457
commons-configuration/	199	65918
commons-csv/	20	3990
commons-daemon/	12	1928
commons-dbcp/	88	22454
commons-dbutils/	41	6386
commons-digester/	171	30527

commons-discovery/	47	6127
commons-el/	62	12321
commons-email/	24	7952
commons-fileupload/	41	8767
commons-finder/	11	1285
commons-flatfile/	61	5744
commons-functor/	309	32326
commons-i18n/	32	3494
commons-id/	62	8578
commons-io/	157	34210
commons-javaflow/	93	9314
commons-jci/	74	8274
commons-jelly/	551	63241
commons-jexl/	108	16432
commons-jnet/	4	256
commons-jxpath/	213	38801
commons-lang/	247	100805
commons-launcher/	17	4120
commons-logging/	65	13288
commons-math/	626	120572
commons-modeler/	43	11070
commons-monitoring/	99	9322
commons-nabla/	126	10560
commons-net/	204	43051
commons-openpgp/	17	1785
commons-performance/	18	3094
commons-pipeline/	80	9254
commons-pool/	50	17365
commons-primitives/	457	54651
commons-scxml/	144	24751
commons-transaction/	60	12340
commons-validator/	128	26566
commons-vfs/	310	47747
db-derby-code/	2649	1003507
db-torque-runtime/	107	30547
excalibur/	941	155897
felix/	2210	354850
forrest/	232	33001
hadoop-core/	1434	298206
hivemind/	782	73101
httpcomponents-httpclient/	312	51192
httpcomponents-httpcore/	409	66452
httpcomponents-oac.hc3x/	277	64597
jackrabbit/	2471	451862
jakarta-bcel/	434	63584
jakarta-bsf/	149	20199
jakarta-cactus/	296	47984
jakarta-ecs/	391	70793
jakarta-jcs/	476	87872
jakarta-jmeter/	828	146384
jakarta-oro/	98	17197
jakarta-regexp/	14	5587
james-server/	535	102620
lenya/	3622	602967
logging-chainsaw/	112	24626

logging-log4j/	302	54620
lucene/	1037	215926
maven-components/	584	82245
mina/	549	84625
myfaces-tobago/	773	83636
ode/	1249	142460
poi/	2003	349983
shale/	5197	891069
struts1/	671	148326
struts2/	1035	120877
synapse-java/	654	85217
tapestry/	1759	212616
tomcat/	1204	340150
turbine-core/	452	97846
velocity-anakia/	15	2564
velocity-dvsl/	20	3501
velocity-engine/	381	73101
velocity-texen/	20	3056
velocity-tools/	147	40082
webservices-scout/	119	17226
xerces/	814	253364
xmlbeans/	997	229874
xml-commons/	523	62346
xmlgraphics-commons/	304	53767
xmlgraphics-fop/	1282	227278
xml-security/	415	79035
xml-xindice/	469	72310

表 8: eclipse.org

プロジェクト名	java ファイル数	総行数
org.apache.lucene/	10	3799
org.eclipse.ant.core/	36	6889
org.eclipse.ant.tests.core/	20	2538
org.eclipse.ant.tests.ui/	48	7844
org.eclipse.ant.ui/	311	53760
org.eclipse.compare.examples.xml/	28	5832
org.eclipse.compare.examples/	3	267
org.eclipse.compare.tests/	16	2828
org.eclipse.compare/	177	41662
org.eclipse.core.applicationrunner/	2	222
org.eclipse.core.commands/	79	12814
org.eclipse.core.contenttype/	34	5210
org.eclipse.core.databinding.beans/	14	3610
org.eclipse.core.databinding/	183	23878
org.eclipse.core.expressions.tests/	18	1999
org.eclipse.core.expressions/	45	5315
org.eclipse.core.filebuffers.tests/	23	5034
org.eclipse.core.filebuffers/	39	7444
org.eclipse.core.filesystem/	22	3991
org.eclipse.core.jobs/	31	7135
org.eclipse.core.net/	29	4421
org.eclipse.core.resources.compatibility/	54	7996
org.eclipse.core.resources.spysupport/	2	98

org.eclipse.core.resources/	254	60944
org.eclipse.core.runtime.compatibility.auth/	8	1094
org.eclipse.core.runtime.compatibility.registry/	6	542
org.eclipse.core.runtime.compatibility/	36	7737
org.eclipse.core.runtime.osgi/	100	18798
org.eclipse.core.runtime/	23	8151
org.eclipse.core.tests.harness/	35	4637
org.eclipse.core.tests.net/	4	718
org.eclipse.core.tests.resources.saveparticipant/	4	823
org.eclipse.core.tests.resources.saveparticipant1/	1	196
org.eclipse.core.tests.resources.saveparticipant2/	1	207
org.eclipse.core.tests.resources.saveparticipant3/	2	256
org.eclipse.core.tests.resources/	255	61797
org.eclipse.core.tests.runtime/	108	21163
org.eclipse.core.tools.resources/	29	5103
org.eclipse.core.tools/	78	8485
org.eclipse.core.variables/	16	1987
org.eclipse.debug.core/	177	27370
org.eclipse.debug.examples.core/	27	3760
org.eclipse.debug.examples.ui/	52	4530
org.eclipse.debug.ui/	735	127334
org.eclipse.equinox.http.jetty/	5	851
org.eclipse.help.appserver/	6	654
org.eclipse.help.base/	128	20720
org.eclipse.help.tests/	36	2550
org.eclipse.help.ui/	100	19617
org.eclipse.help.webapp/	52	8806
org.eclipse.help/	82	8852
org.eclipse.jdt.apt.core/	164	25650
org.eclipse.jdt.apt.pluggable.core/	14	1395
org.eclipse.jdt.apt.pluggable.tests/	24	2215
org.eclipse.jdt.apt.tests/	133	12482
org.eclipse.jdt.apt.ui/	14	3246
org.eclipse.jdt.compiler.apt.tests/	73	8523
org.eclipse.jdt.compiler.apt/	44	9576
org.eclipse.jdt.compiler.tool.tests/	5	1795
org.eclipse.jdt.compiler.tool/	8	2741
org.eclipse.jdt.core.manipulation/	51	7474
org.eclipse.jdt.core.tests.builder/	24	12416
org.eclipse.jdt.core.tests.compiler/	175	342261
org.eclipse.jdt.core.tests.model/	4462	437806
org.eclipse.jdt.core.tests.performance/	18	44737
org.eclipse.jdt.core/	1183	428132
org.eclipse.jdt.debug.jdi.tests/	95	11777
org.eclipse.jdt.debug.tests/	365	117814
org.eclipse.jdt.debug.ui/	394	60556
org.eclipse.jdt.debug/	465	72365
org.eclipse.jdt.junit.runtime/	29	2608
org.eclipse.jdt.junit/	112	21974
org.eclipse.jdt.junit4.runtime/	6	364
org.eclipse.jdt.launching.j9/	25	5413
org.eclipse.jdt.launching.macosx/	15	1917
org.eclipse.jdt.launching/	102	22797
org.eclipse.jdt.text.tests/	161	23925
org.eclipse.jdt.ui.examples.javafamily/	42	4508

org.eclipse.jdt.ui.examples.projects/	5	684
org.eclipse.jdt.ui.tests.refactoring/	6851	120466
org.eclipse.jdt.ui.tests/	220	99518
org.eclipse.jdt.ui/	2031	495198
org.eclipse.jface.databinding/	79	11290
org.eclipse.jface.examples.databinding/	65	10039
org.eclipse.jface.snippets/	72	14567
org.eclipse.jface.tests.databinding.conformance/	29	3748
org.eclipse.jface.tests.databinding/	198	26982
org.eclipse.jface.text.tests/	19	3680
org.eclipse.jface.text/	321	74182
org.eclipse.jface/	377	100673
org.eclipse.jsch.core/	17	1797
org.eclipse.jsch.tests/	1	60
org.eclipse.jsch.ui/	16	3583
org.eclipse.ltk.core.refactoring.tests/	27	3296
org.eclipse.ltk.core.refactoring/	138	23238
org.eclipse.ltk.ui.refactoring.tests/	10	776
org.eclipse.ltk.ui.refactoring/	127	20658
org.eclipse.pde.p2.ui/	7	882
org.eclipse.platform/	4	598
org.eclipse.releng.basebuilder/	43	11343
org.eclipse.releng.eclipsebuilder/	1	524
org.eclipse.releng.tests/	2	1544
org.eclipse.releng.tools/	47	7807
org.eclipse.scripting.adapters.javascript/	1	129
org.eclipse.scripting.examples/	4	1389
org.eclipse.scripting.tests/	2	155
org.eclipse.scripting/	216	38384
org.eclipse.sdk.tests-feature/	117	20615
org.eclipse.search.tests/	22	2728
org.eclipse.search/	152	24096
org.eclipse.swt.examples.browser.demos/	5	751
org.eclipse.swt.examples.browser/	2	79
org.eclipse.swt.examples.controls/	3	116
org.eclipse.swt.examples.launcher/	5	1012
org.eclipse.swt.examples.layouts/	2	81
org.eclipse.swt.examples.ole.win32/	4	1110
org.eclipse.swt.examples.paint/	2	157
org.eclipse.swt.examples/	139	36312
org.eclipse.swt.graphics.text/	46	35701
org.eclipse.swt.opengl.examples/	17	4481
org.eclipse.swt.opengl/	17	4200
org.eclipse.swt.snippets/	308	23486
org.eclipse.swt.tests/	241	55513
org.eclipse.swt.tools/	48	13905
org.eclipse.swt/	1679	645217
org.eclipse.team.core/	172	27526
org.eclipse.team.cvs.core/	187	34854
org.eclipse.team.cvs.ssh/	13	3057
org.eclipse.team.cvs.ssh2/	7	726
org.eclipse.team.cvs.ui/	322	64171
org.eclipse.team.examples.filesystem/	101	12030
org.eclipse.team.ftp/	18	1985
org.eclipse.team.tests.core/	20	1765

org.eclipse.team.tests.cvs.core/	83	18025
org.eclipse.team.ui/	310	61080
org.eclipse.team.webdav/	14	1477
org.eclipse.test.performance.data/	1	73
org.eclipse.test.performance/	56	9883
org.eclipse.test/	4	1009
org.eclipse.text.tests/	32	12894
org.eclipse.text/	127	25160
org.eclipse.tomcat/	8	1479
org.eclipse.ua.tests/	118	12560
org.eclipse.ui.browser/	51	7287
org.eclipse.ui.carbon/	1	454
org.eclipse.ui.cheatsheets/	124	17990
org.eclipse.ui.cocoa/	2	573
org.eclipse.ui.console/	56	9397
org.eclipse.ui.editors.tests/	5	595
org.eclipse.ui.editors/	124	29933
org.eclipse.ui.examples.components/	25	2133
org.eclipse.ui.examples.contributions/	31	2537
org.eclipse.ui.examples.fieldassist/	8	1440
org.eclipse.ui.examples.javaeditor/	39	3315
org.eclipse.ui.examples.job/	16	1656
org.eclipse.ui.examples.multipageeditor/	3	378
org.eclipse.ui.examples.navigator/	7	777
org.eclipse.ui.examples.presentation/	22	3560
org.eclipse.ui.examples.propertysheet/	16	2864
org.eclipse.ui.examples.rcp.browser/	11	1196
org.eclipse.ui.examples.readmetool/	31	3577
org.eclipse.ui.examples.undo/	14	1711
org.eclipse.ui.examples.views.properties.tabbed/	94	20005
org.eclipse.ui.externaltools/	38	6962
org.eclipse.ui.forms.examples/	28	3158
org.eclipse.ui.forms/	80	21332
org.eclipse.ui.ide.application/	6	2423
org.eclipse.ui.ide/	531	120756
org.eclipse.ui.internal.r21presentation/	14	6420
org.eclipse.ui.intro.universal/	23	4444
org.eclipse.ui.intro/	86	19597
org.eclipse.ui.navigator.resources/	34	5471
org.eclipse.ui.navigator/	137	23976
org.eclipse.ui.net/	14	1676
org.eclipse.ui.presentations.r21/	14	7335
org.eclipse.ui.tests.browser/	11	870
org.eclipse.ui.tests.forms/	10	1091
org.eclipse.ui.tests.harness/	22	2930
org.eclipse.ui.tests.navigator/	40	3818
org.eclipse.ui.tests.performance/	74	7844
org.eclipse.ui.tests.rcp/	22	2907
org.eclipse.ui.tests.views.properties.tabbed/	105	7136
org.eclipse.ui.tests/	696	91142
org.eclipse.ui.tutorials.rcp/	23	853
org.eclipse.ui.versioncheck/	2	112
org.eclipse.ui.views.properties.tabbed/	36	6235
org.eclipse.ui.views/	33	5832
org.eclipse.ui.win32/	2	814

org.eclipse.ui.workbench.compatibility/	2	392
org.eclipse.ui.workbench.texteditor.tests/	13	2421
org.eclipse.ui.workbench.texteditor/	160	39850
org.eclipse.ui.workbench/	1309	287745
org.eclipse.ui/	2	353
org.eclipse.update.configurator/	29	6868
org.eclipse.update.core/	276	49144
org.eclipse.update.examples/	11	1262
org.eclipse.update.scheduler/	8	1006
org.eclipse.update.tests.core/	77	9235
org.eclipse.update.ui/	100	17321
org.eclipse.vcm.core.cvs.ext/	3	135
org.eclipse.vcm.core.cvs.ssh/	10	2654
org.eclipse.vcm.core.cvs/	103	9300
org.eclipse.vcm.core/	63	8324
org.eclipse.vcm.tests.core.cvs.ssh/	6	316
org.eclipse.vcm.tests.core.cvs/	7	932
org.eclipse.vcm.tests.core/	20	2802
org.eclipse.vcm.ui.cvs/	33	4427
org.eclipse.vcm.ui/	102	13966
org.eclipse.webdav/	120	22315

表 9: sourceforge.net

プロジェクト名	java ファイル数	総行数
acqlite/	2737	458733
adempiere/	4143	1155868
adito/	1982	306562
advanced-gwt/	111	19334
agilewiki/	726	89373
antlrv3ide/	197	20842
aria/	1272	266199
armedbear-j/	797	199355
artifactory/	272	30348
atunes/	520	94001
avr-eclipse/	210	44985
bt747/	324	87566
cdk/	210	47586
coopnet/	122	25729
cruisecontrol/	912	137566
daimonin/	20	4864
dapper/	271	39001
datacrow/	1764	348198
davmail/	42	8304
dbunit/	486	67912
dimdim/	1219	174027
dita-op/	69	8320
dita-ot/	125	22789
donnerlaparole/	170	27928
dooble/	469	124880
dozer/	568	41919
eclim/	308	35199
eclipsesql/	432	54875
ejbca/	1255	221922

erlide/	677	122251
findbugs/	1828	220400
firebird/	711	227708
fireflyclient/	207	31052
follow/	70	8333
freecol/	485	144748
freelims/	31	18289
freemarker/	377	60990
frostwire/	2626	547046
fuber/	238	30440
ganttproject/	1112	187868
gdcm/	5	650
geneontology/	1436	226718
gep/	123	13936
gham/	798	88893
gpsmid/	202	36448
hibernate/	3896	445894
hibernate4gwt/	270	25333
hipergate/	553	178173
hyperic-hq/	3097	521323
iplist/	7	2770
j2s/	1898	463020
jabref/	601	122809
jailer/	77	27115
java-ml/	258	34839
jaxodraw/	307	70495
jfreechart/	1130	288757
jgnucashlib/	204	93576
j-interop/	205	41578
jitterbit/	485	37527
jmol/	439	158875
jmri/	2249	352202
jnetpcap/	174	39751
josso/	472	60650
jpen/	57	5822
jpwsafe/	119	17327
jspx-bay/	115	15973
jupload/	80	20688
jvlt/	185	21349
jython/	837	276114
keros/	1	11
kmlcsv/	91	7713
kolmafia/	206	83327
liquibase/	643	53680
logicaldoc/	298	44638
makagiga/	835	167531
makumba/	388	67444
matrex/	1082	92132
megamek/	1328	322713
megameklab/	33	14266
microemulator/	310	44598
microlog/	105	14285
nekohtml/	38	12683
obpm/	1100	169144
octave/	79	18853

ogre4j/	1241	272848
omegat/	244	49305
openbravopos/	483	65343
opencsv/	17	2714
openmodeller/	10	716
opennms/	2600	480144
openqrm/	1564	240107
openxava/	2483	285838
opproject/	578	141513
osmius/	640	76786
oswing/	960	191123
pcgen/	2976	587511
pdfsam/	490	82032
phex/	809	167087
pmd/	1501	179756
posterita/	785	166038
redmin-mylyncon/	57	8259
regexpeditor/	30	2910
reprap/	204	64229
rodin-b-sharp/	4645	758507
romaframework/	1155	90594
saxon/	580	118117
sblim/	4923	1612974
schemaspy/	61	11605
smartfrog/	2954	433726
soapui/	902	127764
staf/	382	124210
storybook2/	220	40331
supercsv/	135	8894
sweethome3d/	161	59663
symmetricds/	271	31461
taverna/	3532	488151
tvbrowser/	961	198106
ucdetector/	148	10453
unicore/	5196	927124
unitils/	386	57919
veryquickwiki/	227	32662
villonanny/	53	7281
vnc-tight/	42	11554
webadmin/	26	9401
webcamstudio/	420	59181
wiki2xhtml/	92	18841
wintvcap-gui/	94	33390
wsmostudio/	114	27796
xbrlapi/	408	45933
xmm/	421	144993
xpstudio/	71	3907
yale/	2610	381842
zk1/	1266	177381
zkforge/	669	147504