社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2007－ＳＥ－156 (10)
2007－EMB－ 5 (10)
2007／5／29

ソースコード中の識別子に基づくカテゴリ階層構築手法

宮崎 宏海† 早瀬 康裕† 市井 誠† 松下 誠† 井上 克郎†

† 大阪大学 大学院情報科学研究科
〒 560-8531 大阪府豊中市待兼山町 1-3

カテゴリ検索は，あらかじめ用意された階層的な分類（カテゴリ）を選択することで目的を段階的に絞り込んでいく情報検索手法の１つである．本研究では，カテゴリ検索をソフトウェア部品検索に適用する．カテゴリの階層構造は，利用者が段階的な絞り込みが行えるようにカテゴリ名の意味を考慮して構築されなければならないが，人手で行うにはソフトウェア部品に対する膨大な知識と多大なコストを要するため，自動化が不可欠である．そこで，ソースコード中の識別子に基づいたカテゴリ階層構築手法を提案する．具体的には，ソースコード中の識別子から単語間の上位下位関係を表すシソーラスを作成し，そのシソーラスを用いてカテゴリ階層を構築する．また，提案手法を実装したツールを用いて実験を行った結果，カテゴリと部品の対応と，カテゴリ間の親子関係が適切であることが分かり，システムの有用性が確認できた．

An approach to constructing category tree based on source code

Hiromi Miyazaki† Yasuhiro Hayase† Makoto Ichii† Makoto Matsushita† Katsuro Inoue†

† Graduate School of Information Science and Technology, Osaka University
1-3 Machikaneyama-cho, Toyonaka, Osaka 560-8531, Japan

Category search systems enable users to narrow down documents by selecting a category from the prepared hierarchical classification (i.e category tree). We apply category search to software component search. A category tree should be constructed according to the meaning of the category name. If a search system administrater constructs a category tree by hand, encyclopedic knowledge of software components and substantial efforts required. Therefore, we propose a method to construct a category tree based on identifiers in the source code. The method creates a category tree from a thesaurus of super-sub relation extracted from the source code. Furthermore, the method is implemented and evaluated. The result of the evaluation confirms the correctness of inclusion relation between a category and a component and super-sub relation between categories.

1 まえがき

近年，高品質なソフトウェアを短時間で製作するために，ソフトウェア開発においてソフトウェア部品の再利用が頻繁に行われている．ソフトウェア部品の再利用とは，開発のコスト削減や製品の品質向上を目的として，既存のソフトウェア部品を他のソフトウェアで利用することである [9, 2].

一方でインターネットの普及により，多くのオープンソースソフトウェアが公開されるようになり，大量のソースコードを容易に入手することが出来るようになった．

これらの公開されているソースコードの量は膨大であるため，目的のソフトウェア部品を入手するためには検索システムが用いられることが多い．このため，目的の部品をより的確に表示する検索システムを実現することで，ソフトウェア部品の再利用を促進することが出来る．ソフトウェア部品の検索システムには SPARS-J [5]，Koders [7] が挙げられる．

本研究では，ソフトウェア部品の検索手法の 1 つであるカテゴリ検索に着目する．カテゴリ検索とは，検索の問い合わせのためにキーワードを入力するキーワード検索とは異なり，あらかじめ用意された階層状のカテゴリを辿ることで目的の文書を探す検索手法で，検索者が適当なキーワードが思い付かない場合でも，漠然とした目的から検索を行うことができる．

一般に，カテゴリ検索に用いられるカテゴリは，カテゴリに含まれる文書が適当であるかどうかの判断が必要であるため，手作業で維持されることが多く，その作業に多大な時間と手間を要する．特に，ソフトウェア部品を対象とした場合，追加もしくは更新しようとするソフトウェア部品がどのカテゴリに属するのが適当であるかを判断するために幅広い専門知識を必要とするため，適切なカテゴリ階層を維持するのは非常に困難である．

従って，ソフトウェア部品を対象としたカテゴリ検索には，カテゴリへの分類とカテゴリの維持の自動化は不可欠だといえる．

カテゴリの階層構造について考えた場合，検索者が段階的な絞り込みを行なえるように，その構造はカテゴリ名の意味的な関係に基づいて構築されているべきである．すなわち，親子関係を構築している 2 つのカテゴリの内，親のカテゴリ名には子のカテゴリ名を抽象的に表したものが付くべきである．

カテゴリ名の関係に関連するデータとして単語間の関係を記述した辞書であるシソーラスがある．シソーラスには単語間の類義関係，反義関係，上位下位関係などが記されている．この中で，上位下位関係とは単語の概念による包含関係を意味する．

本研究では単語の上位下位関係を記述したシソーラスを用いて，カテゴリ階層を構築する．しかし，既存のシソーラスはソフトウェア部品のカテゴリ階層の構築には適当ではない．なぜならば，既存のシソーラスは自然言語についての単語間の関係を記述しているが，ソフトウェア部品では自然言語とは異なる用途で単語を用いたり，複合語を用いることが多いためである．そこで，提案手法ではソフトウェア部品に適した新たなシソーラスを自動で作成し，それを用いてカテゴリ階層を構築する．

さらに，本手法を実現するシステムを作成する．そして，作成したシステムを用いて実際にソフトウェア部品の集合からカテゴリ階層を構築し，カテゴリ階層の評価を行う．

以降，2 節で既存のソフトウェア部品検索とカテゴリ検索におけるカテゴリ階層について述べる．3 節では Java のクラス間の利用関係からシソーラスを作成し，それを利用してカテゴリ階層を構築する手法について述べる．4 節では提案手法を実装したシステムを用いた実験とその結果について説明し，5 節でまとめと今後の課題について述べる．

2 ソフトウェア部品のカテゴリ検索

本節では，ソフトウェア部品検索について説明した後，カテゴリ階層とシソーラスの関連性と関連研究について述べる．

2.1 ソフトウェア部品検索

ソフトウェア部品検索とは，部品を再利用するために大量のソフトウェア部品の集合の中から目的の部品を見つけ出す作業のことである．自然言語文書の検索システムでは，検索者が目的の文書に関するキーワードを入力して検索するのが一般的であるが，ソフトウェア部品検索システムでは部

品に関するメトリクスや記述言語，求める機能の抽象表現を検索の問い合わせに用いる場合もある．

2.2 カテゴリ階層とシソーラス

シソーラスとは同義関係，類義関係，反義関係，上位下位関係などの単語間の関係を記した辞書である．自然言語の分野ではロジェのシソーラス [3] や WordNet [1] などが電子化されたシソーラスとして開発されている．

カテゴリ階層とはカテゴリ間の親子関係によって構成される階層構造である．本研究では，カテゴリ階層を 1 つのカテゴリに複数の親カテゴリを認める有向グラフ状とする．カテゴリ検索では漠然とした目的から段階的な絞り込みを行うため，あるカテゴリのカテゴリ名はその子カテゴリより抽象的な名前がつき，その親カテゴリよりも具体的なカテゴリ名が付いていることが望ましい．そこで，単語間の包含関係を表す上位下位関係のシソーラスをカテゴリ階層として用いることを考える．

しかし，既存のシソーラスは自然言語について記述したものであるため，ソフトウェア部品のカテゴリ階層の構築にそのまま用いることは出来ない．これは，ソフトウェアにける単語の意味が一般的な単語の意味と異なるものがあるためである．例えば，Java には List という型が Collection の型のサブクラスとして存在する．このため，Java のソースコード中では，List は Collection の下位語であるといえる．しかし，WordNet などの自然言語について記述したシソーラスには，このような関係は記述されていない．このような意味の異なる単語が存在するため，ソフトウェア部品のカテゴリ階層を構築するためには，一般的なシソーラスではなく，ソフトウェア部品用の新たなシソーラスを作成する必要がある．

2.3 関連研究

ソフトウェアを対象とした分類手法として，川口らの LSA に基づく手法 [6] が挙げられる．この手法では，ソフトウェアのソースコード中に出現する識別子に対して LSA[8] を適用し，その結果を用いて識別子によるソフトウェアの非排他的クラスタリングを行っている．これにより，前提知識を用いずにソフトウェアの集合を機能やライブラリといった複数の視点による自動分類を実現している．

また，ソフトウェア部品をカテゴライズした手法としては仁井谷らのソースコード中の特徴語に基づく手法 [11] がある．この手法では，ソースコード中の単語に対して出現位置による重みを付け，さらに複合語や利用関係を考慮することで，部品の特徴を表す特徴語を選出し，その特徴語をカテゴリ名とすることで自動分類を行っている．

自然言語のシソーラスの自動構築手法としては，Hearst による構文パターンのマッチングによる単語間の上位下位関係を取得する手法 [4] がある．この手法では，特定の構文パターン中の単語には上位下位関係が生じやすい事を利用している．

また，構造化文書を対象としたシソーラス構築手法として，新里らの HTML 文書の構造を利用した手法 [10] がある．新里らは，箇条書きやリストボックスなどのように文書中に繰り返して出現する要素に使用される単語は意味的に類似しており共通の上位語を持ちやすいことに着目している．さらに，単語の係り受けを調べることで共通の上位語と各下位語の類似度を調べて，精度を高めている．

3 提案手法

本節では，カテゴリ階層を構築する手法について述べる．まず概要を述べ，続いて手法の詳細を手順ごとに述べる．

3.1 概要

提案する手法の概要を図 1 に示す．まず Java クラスの集合を入力として，単語間の上位下位関係を持つシソーラスを作成する．次に，同じ Java クラスの集合を入力として Java クラスのクラスタリングを行い，得られたクラスタに特徴語を付ける．最後に，得られたシソーラスとクラスタからカテゴリ階層を構築する．以降，それぞれの手順について詳しく説明する．

3.2 シソーラスの作成

本研究では，シソーラスを「単語を頂点とし，単語間の上位下位関係を上位の頂点から下位の頂点に引いた有向辺で表した有向グラフ」であると定義する．そして，Java クラスの特定の利用関係に注目することで単語間の上位下位関係を取得する．

図 2 にシソーラスの例を示す．図 2 では，シソー

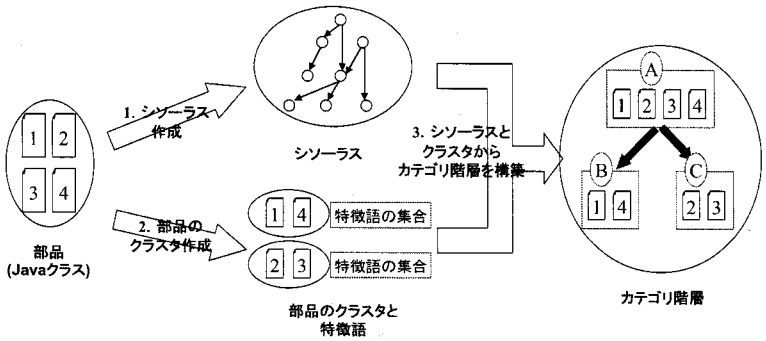


図 1: 提案手法の概要

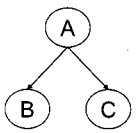


図 2: シソーラスの例

ラスには A, B, C の 3 つの単語が含まれ，A は B および C の上位であることが表されている．

まず Java クラスの利用関係から単語の組と上位下位関係を取得する．用いる利用関係を表 1 に示す．ただし，クラス名やインタフェース名は名前空間を除いたものを取得する．例えば，java.util.List が java.util.Collection を継承していることから，Collection を上位，List を下位とする上位下位関係を得る．

次に，取得した単語の表記をキャメルケースの形式で統一する．キャメルケースとは複数の単一語で構成されている複合語に対して単一語の先頭文字を大文字で書き，先頭以外の文字を小文字で書く表記法である．具体的には，全ての単語に対して以下の処理を加える．

- 1 文字目を全て大文字に変換する
- "_"（アンダーバー）の次の 1 文字を大文字に変換して"_"(アンダーバー) を全て削除

表 1: 取得する利用関係

関係	上位	下位
継承	親クラス名	子クラス名
	親インタフェース名	子インタフェース名
実装	インタフェース名	実装するクラス名
型と変数	フィールド変数の型名	フィールド変数の名前

また，取得した単語の組の内，以下のいずれかの条件を満たす単語を含む組を削除する．

- 1 文字の単語
- 数字のみで構成されている単語

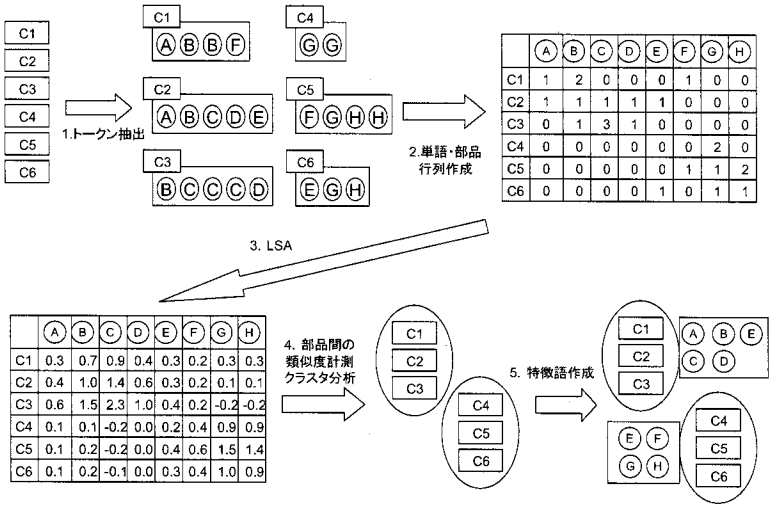
上記の条件を満たす単語は部品の機能や特徴を表すことがほとんどなく，部品のカテゴリ階層名の構築に用いられることを前提としたシソーラスに含まれる単語として不適切であると考えられる．

取得する利用関係は上位下位関係の期待が出来るものを選択したが，単語間に必ずしも上位下位関係が成立するとは限らない．そこで，取得した関係が上位下位関係かどうかを出現回数で判断する．出現回数が多い単語の組ほど上位下位関係の確からしさが高いと考えられる．

3.3 Java クラスのクラスタと特徴語の決定

本研究では，川口ら [6] により提案されたソフトウェアの分類手法に基づく手法を用いて部品のクラスタリングを行う．この手法で得られたクラスタには 10 個の特徴語がついている．ただし，川口らの手法に若干の変更を加えている．図 3 および以下にその概要を示す．

まず部品のソースコードからトークンの抽出を行い，単語の出現数を要素とする単語×部品の行列を作成する．次に，作成した行列に対して LSA を適用する．LSA の結果を用いて，部品間の類似度を計測する．その類似度を用いてクラスタ分析を行い，部品を排他的に分類する．得られた全てのクラスタに対して特徴的な単語を 10 個抽出する．



	A	B	C	D	E	F	G	H
C1	1	2	0	0	0	1	0	0
C2	1	1	1	1	1	0	0	0
C3	0	1	3	1	0	0	0	0
C4	0	0	0	0	0	0	2	0
C5	0	0	0	0	0	1	1	2
C6	0	0	0	0	1	0	1	1

	A	B	C	D	E	F	G	H
C1	0.3	0.7	0.9	0.4	0.3	0.2	0.3	0.3
C2	0.4	1.0	1.4	0.6	0.3	0.2	0.1	0.1
C3	0.6	1.5	2.3	1.0	0.4	0.2	-0.2	-0.2
C4	0.1	0.1	-0.2	0.0	0.2	0.4	0.9	0.9
C5	0.1	0.2	-0.2	0.0	0.4	0.6	1.5	1.4
C6	0.1	0.2	-0.1	0.0	0.3	0.4	1.0	0.9

図 3: ソフトウェア部品のクラスタリングの概要

3.4 カテゴリ階層の構築

作成したシソーラスと Java クラスのクラスタリング結果からカテゴリ階層を構築する．

まず，クラスタリング結果からカテゴリを作成する．クラスタの特徴語がシソーラスに登録されていれば，その特徴語をカテゴリ名とするカテゴリを作成する．図 4 では A，B，E，F のカテゴリが作成される．そして，クラスタに含まれている部品をカテゴリに割り当てる．図 4 では，特徴語 A と特徴語 B を持つクラスタに含まれる部品はカテゴリ A および B に，特徴語 A と特徴語 E を持つクラスタに含まれる部品はカテゴリ A および E に，特徴語 E と特徴語 F を持つクラスタに含まれる部品はカテゴリ E および F に，それぞれに分類される．

次に，以下の条件を満たす単語をカテゴリ名とするカテゴリを作成する．

- クラスタの特徴語に含まれない
- シソーラスにおける下位の単語のうち，2 個以上の単語がクラスタの特徴語に含まれる

図 4 では D のカテゴリが作成される．この処理で作成されるカテゴリには部品が含まれないが，複数の子カテゴリを持っているため絞り込みには有用である．

3.2 節で述べた方法により作成したシソーラスを用いて，カテゴリ間の親子関係を作成する．具体

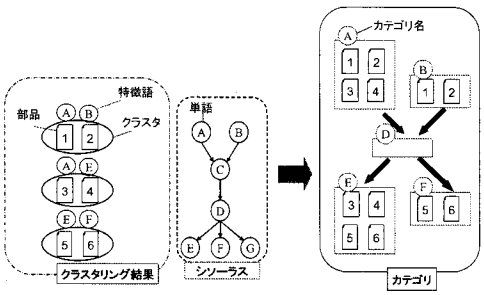


図 4: カテゴリ階層の構築

的には，シソーラス上にカテゴリ名に対応する 2 つの頂点間に経路が存在し，その経路の始点と終点以外にカテゴリ名に対応する頂点が存在しない場合，始点を親，終点を子とするカテゴリ間の親子関係を作成する．

図 4 では頂点 A と頂点 D の間には経路が存在し，A と D の経路上にある頂点 C をカテゴリ名とするカテゴリは作成されていないため，カテゴリ A とカテゴリ D の間に親子関係を作成する．

一方，頂点 A と頂点 E の間にも経路が存在するが，A と E の経路上にある頂点 D をカテゴリ名とするカテゴリが作成されているため，カテゴリ A とカテゴリ E の間には親子関係を作成しない．

4 実験

提案するシステムが作成するカテゴリ階層の有用性を確認するために実験を行う．実験では，作成したシステムを用いて実際のソフトウェアに対するカテゴリ階層を構築し，カテゴリ間の親子関係の妥当性とカテゴリに割り当てられた部品の妥当性を評価する．

4.1 実験内容

システムへの入力は JDK1.4 に含まれる全てのクラスとした．JDK1.4 に関する情報は表 2 の通りである．JDK1.4 を入力とした理由は，以下の通りである．

- Java の標準ライブラリとして作られているため，クラス名や変数名などの識別子に適切な名前付けが期待できる
- 規模が小さ過ぎない

出現した識別子の組は，出現回数が表 3 に記した値以上のものをシソーラスに登録した．クラス名は外部に公開することが多いため適切な名前が期待できると判断し，継承関係と実装関係の閾値は 1 回とした．一方，フィールド変数は外部に公開しない場合もあるため，閾値を 10 回以上とした．

システムが出力したカテゴリ階層の概要を表 4 に示す．

4.2 実験 1：カテゴリ名と部品の適合率

まず，部品の割り当てが適切かを評価するために，カテゴリ名と部品の適合率を求める．出力した 1109 個のカテゴリの中からランダムに 50 個のカテゴリを取得し，そのカテゴリ名と割り当てられた部品 324 個の適合率を計測した．ここで評価する適合率は，カテゴリ名に適合する部品の数をカテゴリに割り当てられている部品の数で割った値とした．

表 2: JDK1.4 に関する情報

総ファイル数	6024 個
総クラス数	10123 個
総行数	2066967 行

表 3: 閾値の設定

利用関係	閾値 (出現数)
継承関係	1 回
実装関係	1 回
フィールド変数の型と変数名	10 回

また，以下に示す 5 つの条件のうち，いずれか 1 つに該当した場合に，適合する部品であると判断した．

- カテゴリ名と部品名が同じである
- カテゴリ名が部品名の複数形や省略語である
- カテゴリ名が部品名の部分語に含まれている
- カテゴリ名が部品名の類似語である
- カテゴリ名が部品の特徴の一部を表している

適合率は 83.6 % (271 組/324 組) であった．適合条件別の適合度は表 5 のようになった．

4.3 実験 2：カテゴリの親子関係の適合率

次に，階層構造が適切かを評価するために親子関係を構築しているカテゴリの親子関係の適合率を計測する．出力した 2501 組の関係の中からランダムに 200 組の関係を抽出し，その親子関係の適合率を計測した．ここで評価する適合率は，親子関係が適合するカテゴリの組の数を取得したカテゴリの組の数で割った値とした．

親カテゴリ名を A，子カテゴリ名を B としたとき，B が A の一種である場合に，その親子関係が適合するとした．

適合率は 64.0 % (128 組/200 組) となった．表 6 に適合例を，表 7 に不適合例を示す．

4.4 考察

カテゴリ名と部品の適合率は 83.6 % と高い値であったため，ソフトウェア部品のカテゴリへの割り当ては適切に行われている．

しかし，カテゴリ名と部品が 1 つも適合していないカテゴリも存在した．これは，特徴語をどのクラスタに対しても 10 個の単語としてしまったため，特徴語に部品の特徴を表す単語として不適切なものが含まれてしまったことが原因と考えられる．

カテゴリの親子関係は 64.0 % と実用的といえるほどの高い値は得られなかった．この原因としては，入力としたソフトウェア部品の集合である

表 4: 出力

総カテゴリ数	1109 個
カテゴリ間の親子関係	2501 組
カテゴリに割当てられた部品数	6000 個
カテゴリに割当てられた延べ部品数	18583 個
カテゴリに含まれる部品数 (平均)	16.75 個

表 5: カテゴリ名と部品の適合率

カテゴリ名と部品名の関係	適合率 (適合数/総数)	適合例 (カテゴリ名：部品名)
同じ	7.1 % (23/324)	Receiver：javax.sound.midi.Receiver.java
複数系や省略語	0.3 % (1/324)	Channel：java.nio.cahnnels.Channels.java
部分語	23.1 % (75/324)	Holder：org.omg.CORBA.CharHolder.java
類似語	0.9 % (3/324)	PipedWriter：java.io.PipedOutputStream.java
特徴の一部を表す	52.2 % (169/324)	ItemEvent：java.awt.Checkbox.java
合計	83.6 % (271/324)	

JDK1.4 の中で，継承や実装が正しく用いられていなかったことが原因であると考えられる．例えば，ある部品の特定の機能を使用するために継承を用いる場合があった．このような場合には，子クラスの名前が親クラスの名前の一種とならないため，シソーラスに上位下位関係として不適切な関係が抽出されてしまう．

また，フィールド変数の型と変数名の組の出現回数はすべて閾値以下であったため，フィールド変数の型と変数名の組がシソーラスに登録されることは無かった．閾値を 10 回から 2 回に変更した場合には，フィールド変数の型と名前からいくつかの関係がシソーラスに抽出された．これらの抽出された関係の中には上位下位関係として適切なものも存在したが，多くはクラス名とその省略語であるなど，不適切なものであった．

5 まとめと今後の課題

本研究では，Java クラスの利用関係を用いたシソーラスの作成と，作成したシソーラスと Java クラスのクラスタリング結果を用いてカテゴリ階層の構築を行う手法を提案した．

また，システムを実装して実験を行い，提案手法によって意味的な絞り込みに適したソフトウェア部品のカテゴリ階層を自動で構築できることを示した．

表 6: 親子関係の適合例

上位	下位
AbstractMap	HashMap
Cursor	CustomCursor
Member	Method

表 7: 親子関係の不適合例

上位	下位
DefaultListCellRenderer	UIResource
TableModelListener	JTable
Thread	Daemon

今後の課題として，カテゴリ階層中の親カテゴリと子カテゴリの親子関係の適合率を改善することが挙げられる．具体的には，ソフトウェア部品中で用いられている識別子の中でも，その部品をその部品の外から利用するために必要な名前には適切な名前付けがなされていると考えられることから，シソーラスへの登録の閾値を private, public 等のアクセス制御の種類によって変化させることなどが考えられる．他に，カテゴリ間の関係に親子関係以外の関係を導入することでカテゴリ検索の有用性を高めることも考えられる．カテゴリ検索を行うためのユーザーインターフェースの開発も重要な課題である．

参考文献

[1] A.Miller, R.Beckwith, C.Fellbaum, D.Gros and R.Tengi: Five Papers on WordNet, *Technical Report CSL Report 43* (1990).

[2] C.Braun: Reuse, *Encyclopedia of Software Engineering* (Marciniak, J., ed.), Vol. 2, pp. 1055–1069 (1994).

[3] Chapman.R: L.R.Roget's International Thesaurus, *Thomas Y.Crowell Company Inc.* (1977).

[4] Hearst.M.A: Automatic acquistiotion of hyponyms from large text corpora., *In Proceedings of the 14th International Conference on Computational Linguistics*, pp. 539–545 (1992).

[5] Inoue, K., Yokomori, R., Yamamoto, T., Matsushita, M. and Kusumoto, S.: Ranking Significance of Software Components Based on Use Relations, *IEEE Transactions on*

Software Engineering, Vol. 31, No. 3, pp. 213–225 (2005).

[6] Kawaguchi, S., Pankaj.K.Garg, Matsushita, M. and Inoue, K.: MUDABlue: An automatic categorization system for Open Source repositories, *The Journal of Systems and Software*, Vol. 79, pp. 939–953 (2006).

[7] Koders: . `http://koders.com/`.

[8] T.K.Landauer, P.W.Foltz and D.Laham: An Introduction to latent Semantic Analysis, *Discourse Processes*, Vol. 25, pp. 259–284 (1998).

[9] V.R.Basili, G.Caldiera, F.McGarry, R.Pajerski, G.Page and S.Waligora: The software engineering laboratory - an operational software experience, *Proceedings of 14th International Conference on Software Engineering*, Melbourne, Australia, pp. 370–381 (1992).

[10] 新里圭司，鳥澤健太郎：HTML 文書からの単語間の上位下位関係の自動取得，自然言語処理学会， Vol. 1, No. 1, pp. 539–545 (2005).

[11] 仁井谷竜介：ソースコードの特徴語を用いた Java ソフトウェア部品の分類システム，情報処理学会研究報告， Vol. 149, No. 75, pp. 49–56 (2005).