社団法人 電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE
SS2002-47(2003-1)

WWWを対象としたソフトウェア検索エンジンの構築

亀井 俊之† 門田 暁人† 松本 健一†

†奈良先端科学技術大学院大学情報科学研究科 〒630-0192 けいはんな学研都市
E-mail: †{toshi-k, akito-m, matumoto}@is.aist-nara.ac.jp

あらまし 過去に多くのソフトウェアが開発されてきたが，その知識は必ずしも有効に蓄積・利用されていない．一方，WWW にはソースコードや関連ドキュメントなど，ソフトウェア開発に役立つ情報が数多く存在している．これらの情報の一部は一般的な web 検索エンジンによっても獲得できるが，これらのエンジンはソフトウェア資源を必ずしもデータベースに含んでおらず，ソフトウェアの特徴を考慮した検索方法が提供されていない．そこで本稿では，WWW 空間を巡回してソフトウェア資源を収集し，解析してユーザに提供する検索システムの構築について述べる．ユーザはクエリにメトリクスなどのソフトウェア特有の値を含めて検索することができる．本稿では，試作したシステムの実装と検索例についても述べる．

キーワード ソフトウェア再利用，ソフトウェアメトリクス，巡回ロボット

The Development of a Software Search Engine for the World Wide Web

Toshiyuki KAMEI† Akito MONDEN† and Ken'ichi MATUMOTO†

† Graduate School of Information Science Nara Institute of Science and Technology Kansai Science City,

630-0192 Japan
E-mail: †{toshi-k, akito-m, matumoto}@is.aist-nara.ac.jp

Abstract Although many software systems had been developed in the world, the knowledge of developments has not been effectively accumulated and reused. On the other hand, there are many information sources on the WWW that are useful for software development, such as source codes and their related documents. Some of useful information can be retrieved by the common web search engines, however, these engines do not necessarily contain software itself in the database. Moreover, they do not have a functionality suitable for retrieving software resources. This paper describes the design of a software search engine, which have an ability to crawl the WWW space, to collect and analyze software resources, and to provide them to users. The user of our engine can search software resources using a query including symbols and values in software, such as product metrics. This paper also describes an implementation of a prototype system and an example of retrieval.

Keyword Software reuse, Software Metrics, Web Crawler

1. はじめに

過去に多くのソフトウェアが開発されてきたが，その知識は必ずしも有効に蓄積・利用されていない．似たようなソフトウェアが様々な場所で開発されていたり，同じようなミスでソフトウェア開発が滞ることがあるなどの現状を考えると，ソフトウェアの知見に関するデータベースと検索エンジンが必要であると考える．

一方，WWW には多くの情報が共有されており，その中にはソフトウェア開発の際に得られた知見や，開発に役立つ情報も多く含まれている．たとえばソースコードそのものやそのコメント，開発日記や Tips などが利用可能である．現在このような情報を得るために web 検索エンジンを用いることが多いが，ソフトウェア資源の特徴を生かした検索が可能な検索エンジンが存在せず，ユーザは時間と手間をかけて検索しなければならない．

そこで本稿では，WWW に存在するソフトウェア開発に関する知見や情報を検索するための検索エンジンの構築を目的とする．本検索エンジンでは，ソフトウェアメトリクスや，パッケージ名，クラス名などを指定できるインタフェースを持ち，コードや関連ドキュメントなどの情報を検索結果としてユーザに提供する．

このような検索を可能とするために，提案するソフトウェア検索エンジンは，次の三つの機能を持つ．

・ソフトウェア資源収集

巡回ロボットにより，WWW 上のソフトウェア資源を効率的にかつ大量に収集する．ロボットは既存検索エンジンを利用して WWW の様々な地点に起点となる URL を選択し，各起点 URL からソフトウェア資

源が存在すると思われる巡回路を選び探索を行う．

・ソフトウェア資源解析

収集した資源のインデクシングを行う．圧縮されているファイルならば解凍してソースコードを発見し，そのクラス名，メソッド名やソフトウェアメトリクスなどを解析して検索のためのインデックスを作成する．

・ソフトウェア検索

ブラウザを通してユーザからクエリを受け取り，データベースから対応する資源を提供する．

以上の機能により，ユーザは大量のソフトウェア資源に対してその特徴を生かした柔軟な検索が可能となる．本稿ではシステムの提案とその試作を通して，考察を行う．

2. ソフトウェア検索

2.1. ソフトウェア検索による開発者支援

ソフトウェアを検索する動機として，開発者の立場と利用者の立場の大きく分けて２つの立場で考える．本稿で対象としているのは，開発者の立場からのソフトウェア検索であり，たとえば以下のような検索要求が考えられる．

- あるプログラムが作りたくて，参考になりそうなコードを読みたい．
- あるクラス，パッケージの典型的な使い方が知りたい．
- あるアルゴリズムについて，コメントが多く書いてあるコードを読みたい．
- 特定の型の変数を引数と返り値に持つメソッドを知りたい．

このような要求を満たすには，ソースコードを集め，そのメトリクス値を測定するなど，事前にインデクシングを行っておかなければならない．また，ユーザから受け取るクエリとして，メトリクス値や，どこを対象に検索するか（クラス名なのか，パッケージ名なのか，など）記述できなければならない．このようなクエリが記述できれば，開発者の希望通りのソースコードを取り出すことが可能となる．

2.2. ソフトウェア検索の現状

現在でも WWW におけるソフトウェア検索は存在するが，ソフトウェアの利用者の立場にのみ立った検索エンジンが多い．

たとえば窓の杜 [9] や Vector[11]，DOWNLOAD.COM[7] などに代表される登録型ソフトウェアライブラリのサービスは，管理者が選んだオンラインソフトなどに説明文をつけてデータベースに登録しておき，ユーザはクエリとしてキーワードを与え，登録されているソフトウェアの名称とその説明文を対象に全文検索を行う．

また，SourceForge[10] に代表される登録型プロジェクトライブラリでも検索を行うことができる．このサービスは，開発者に対して開発に必要なさまざまなリソースを無料で提供する開発支援サービスであり，クエリとしてキーワードを与え，登録されているプロジェクトの名称や内容説明文を対象に全文検索を行う．

これらのサービスは，ソフトウェアの名称やだいたいの働きなどが分かっており，ソフトウェアの利用者として検索するならば充分な能力を持っているといえるが，ソフトウェアの開発者として利用するには不充分であるといえる．また，これらのサービスのデータベースは人手で登録しており，WWW 全体を対象に検索できるわけではない．

WWW に存在しているソフトウェア資源を得るために，現在主に使われている手段は web 検索エンジンを用いた方法であると考えられる．しかし今のところ，WWW を対象にしたソフトウェア検索専用に設計された検索システムは存在しない．通常の検索エンジンにそれらしいクエリ，たとえば java 言語のソーティングに関するソースコードを検索したいときは「sort java」と入力し，提供された結果の周辺を人間がブラウジングして情報を得るといった方法が一般的であろう．検索エンジンを用いれば WWW 全体を対象に検索することができるが，たとえばソースコードそのものを検索したりすることはできないため，時間と手間がかかってしまう．開発者支援のためには，web ページのみを検索対象とするのではなく，ページ上のソフトウェア資源そのものを検索対象とする検索エンジンの構築が望まれる．

3. 提案システム

本稿で提案する手法は，WWW からソフトウェア資源を収集してダウンロードし，インデクシングを行ってデータベースを作成して，ユーザのクエリによって検索できるようにするといった方法をとる．こうすることによって WWW 全体を対象にキーワードだけでなくソフトウェア特有のメトリクスをキーとして検索できるシステムが実現可能となる．

しかし，WWW を巡回して資源を探すのは手間と時間のかかる作業であり，一般的にコストが高いため，様々な工夫をしてコストを抑える努力を行う．

本稿で対象とするソフトウェア資源とは，ソースコードを中心にその付属ドキュメント，ページで公開さ

れている Tips など，ソフトウェア開発に役立ちそうで WWW に公開されているものすべてを指す．本稿ではソースコードを中心に話を進める．

提案システムのおおまかな構成を図 1 に示す．

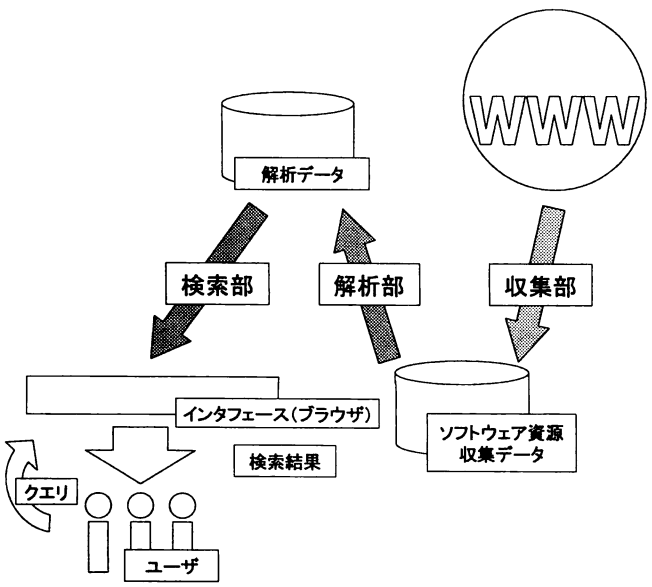

図 1　システム構成

収集部では WWW を巡回してソフトウェア資源の URL を収集し，解析部で集めた資源のインデクシングを行う．収集した資源の中には圧縮されたファイルも多く，解析部ではその解凍も行う．また，集めた資源が実際に有用なものかどうかを判断しながら巡回路の設定を行うので，巡回と解析は同時に並行して行われる．解析部でインデクシングされたデータは，検索部のインタフェースを通してユーザに提供される．

3.1. 収集部

近年，WWW はその規模を拡大し，爆発的に成長し続けている．一般的に検索エンジンは，WWW 空間からデータを収集してユーザからの検索に備えるが，巨大な WWW 空間を隅々まで巡回するのは不可能であるといえる．

検索エンジン Google では，一万台以上の PC を運用して巡回しているが，巡回ロボットが巡回路を一巡し，巡回を開始したページに戻ってくるまで数ヶ月かかるといわれている．この数ヶ月の間にも WWW は変化し続けており，有用な情報が失われるといったことも起こってしまう．

そこで本稿では，ソフトウェア資源がありそうなところを重点的に巡回し，なさそうな部分は全く巡回しないようにすることにより，巡回ロボットの巡回路を短くして収集の効率を上げる．

また，ソフトウェア資源を発見すると，その周囲を資源の説明として取得する．資源へのリンク文字列や見つけたページのタイトルタグ，ページの重要単語や近くのタイトルタグなどを資源の説明として収集する．

データの収集には巡回ロボットを用いる．一般的な巡回ロボットのアルゴリズムを図 2 に示す．

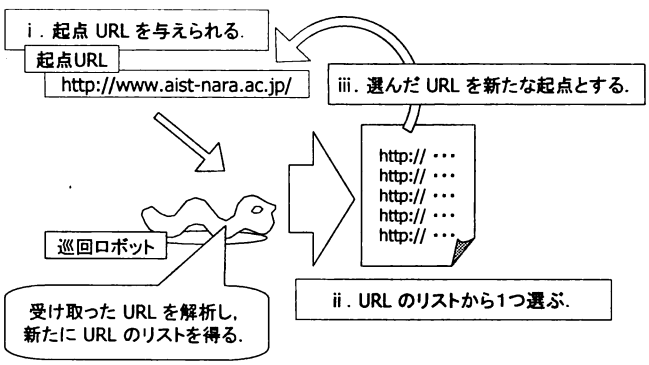

図 2 巡回ロボットの基本的なアルゴリズム

i.　**起点 URL の選択**

起点となる URL を入力する．

ii.　**新たな URL の選択**

その URL をダウンロードし，HTML 構造を解析してリンクタグを抜き出し，新たな URL のリストを得る．その中から次の起点を選ぶ

iii.　**再帰**

ii.で選んだ URL を起点として i.から再帰的に繰り返す．

各過程で工夫を施し，多くの資源を短時間で収集できるようにする．

i.　起点 URL の選択

WWW ページはリンクによってページ間が連結されている．一般的に，このリンクで連結されたページは共通のトピックを持つ傾向にあり，巨視的に見れば共通のトピックを持った集合が形成されているといえる．WWW にはこのような共通のトピックを持つ集合（コミュニティ）が無数に存在している．たとえば料理のレシピを公開している人たちは同じような料理のサイトにリンクを張る傾向があり，またソフトウェアに関する情報を扱っているサイトは同じような傾向のサイトにリンクを張る傾向がある．

このようなコミュニティ内を巡回すれば，そのコミュニティの話題になっている情報を多く集めることが

できる[1][2]が，本稿で収集の対象としているソフトウェア資源を収集するために，料理のレシピを公開しているコミュニティを巡回しても意味がないので，うまくソフトウェア資源を扱っているコミュニティを発見しなければならない．コミュニティの一部を発見し，そこを起点に巡回すればそのコミュニティ内の情報を取得できるが，人手でページを発見し与えるのは効率が悪い．

そこで，起点 URL を発見するために，既存 web 検索エンジンにクエリを与え，その結果を起点として巡回する．クエリは目的のコミュニティ内でよく出現する特徴的単語から選ぶ．このようにすることによってウェブ全体に分散して存在するコミュニティを巡回することができる．

既存検索エンジンには様々な種類のものがあり，そのデータベースの性格もいろいろなので，複数の検索エンジンを並行的に用いる，メタ検索方式を採用する．

ii. 新たな URL の選択

ページのダウンロードと HTML 文書の解析，リンクの抽出を再帰的に繰り返すと，巡回する URL がキューに貯まっていく．この中から，もっとも資源を発見できそうな URL を選択し，巡回しなければならない．

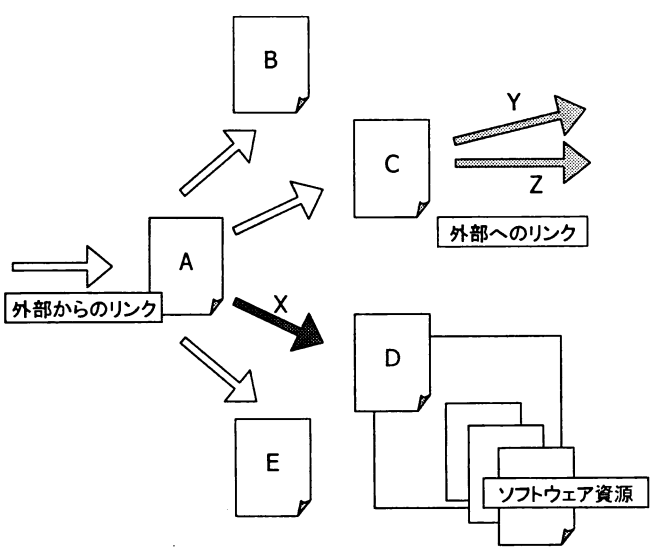

図 3 想定するページ構造

本稿では，図 3 に示すような，トップページからいくつかの同一サーバ上にあるコンテンツにリンクを張っているような構成のサイトを次々に巡回することを想定している．このようなサイトに入り，ソフトウェア資源を収集し，速やかに次のサイトに移るということを繰り返して収集を行う．巡回ロボットは A のページに到達すると，B，C，D，E へのリンクと，そのリンクの説明を得る．ふつうに巡回すると次は B，C，・・と巡って D で資源を発見するが，X のリンクに手がかりがあれば B，C より先に D を巡回することができる．D のページはソフトウェアを配布しているページを想定しているが，そのページへのリンクには特徴があることが多い．たとえば，リンクの URL（http://www.xxx.com/download.html など）や，リンクの文字列（product，download など）から手がかりを得る．

このようにサイト内のソフトウェア資源を予想して収集し，Y，Z のような外部へのリンクで外に出て，次のサイトへ移る．

iii. 再帰

既存検索エンジンから起点となる URL をいくつか得ることができるが，そのすべてがコミュニティの一部をさしているとは限らない．また，コミュニティの一部をさしているとしても，その大小で巡回の深さの制御を行うことにより資源のないところはできるだけ巡回しないようにする．

巡回は通常，決められた階層だけ行うが，ある程度進んだ時点で簡単な見積もりを行い，探索する階層を制御する．

3.2. 解析部

収集したソフトウェア資源を解析してインデクシングを行う．圧縮されているファイルは解凍し，ソースコードや関連ドキュメントを取り出した後，ソフトウェアメトリクスなどを測定する[4][5]．

ソフトウェアメトリクスとは「ソフトウェアとその開発／利用課程を対象とした定量的評価尺度」であり，コード行数（LOC）やサイクロマティック数，演算子数やクラス数などのプロダクトメトリクスと，設計工数やテスト時間などのプロセスメトリクスがある．本稿では，ツールによる機械的な評価が比較的容易なプロダクトメトリクスを測定する[3]．

ソフトウェアメトリクスのほかに，パッケージ名やクラス名，メソッド名とメソッドの引数の数や型，返り値の型も取得してデータベースに納め，検索の際の指針とする．

3.3. 検索部

ユーザからクエリを受け取り，データベースからソフトウェア資源を提供する．本システムでは検索対象

のソフトウェア資源として，コードそのものと関連ドキュメントを想定しているので，クエリの形としては，関連ドキュメントはキーワードによる全文検索，コードに対しては，キーワード＋メトリクス（コードの規模，コメントの規模，引数の型・数…）の形とする．

キーワードやメトリクスは，検索対象によって意味が変わってくるので，対象を限定できるようにクエリを作成する．たとえば「あるパッケージの典型的な使い方が知りたい．」として検索する場合，パッケージ名だけを対象に検索できなければならない．

ほかにもソースコードを入力として与えることにより，そのメトリクスを測定し，類似するようなコードを検索することができる．似たようなコードを読むことにより，作成しているプログラムの問題点やその解決法が得られることが期待できる．

4. 実装

4.1. システムの試作

ソフトウェア検索エンジンの試作を行った．今試作では，巡回ロボットと解析部は Java 1.4.1_01，データベースは mysql 3.23.52，検索インタフェース部分は Java Servlet(tomcat 4.1.18) で実装した．メトリクス測定には Javancss 21.41[8] を用いた．

収集するソフトウェア資源は拡張子で判断し，java, zip を対象に収集した．圧縮ファイルは解析課程で仮解凍し，その中に java のソースファイルが含まれているとディスクに展開してそのメトリクスを測定した．

また，今発表のプロトタイプで用いたソフトウェアメトリクスは以下の通りである．

- 名前（パッケージ名，クラス名，メソッド名，メソッド数）
- 引数と返値の数と型
- 行数（コード行数，コメント行数）
- コメント(javadoc 形式でメソッドの説明が書かれているか)

インタフェースは図4のようになる．

測定の終わったソースコードは著作権の関係により，URL と解析結果だけ残して消去する．

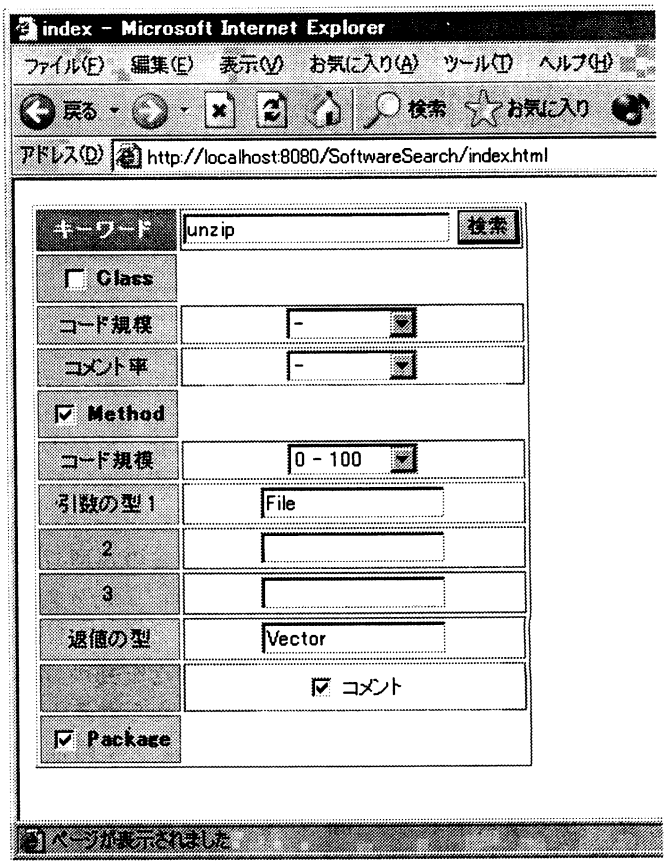

図4 インタフェース

4.2. 結果と考察

プロトタイプにてクエリ「sort」で検索した画面の一部を図5に示す．クラス名とメソッド名に「sort」が含まれている一覧が示されている．クラスは取得日時，クラス名，パッケージ名，メソッドの数，コード行数，コメントや空行・かっこだけの行を抜かした行数 NCSS(Non Commenting Source Statements)，Javadoc 形式でコメントの書かれたメソッドの割合，Javadoc 形式で書かれたコメントの行数，付属ドキュメントがあればそのドキュメントへのリンクが示されている．クラス名は WWW 空間の実際のファイルへのリンクとなっている．

また，メソッドは取得日時，メソッド名（引数の型や数も含む），返り値の型，そのメソッドを持つクラス名，パッケージ名，コード行数，NCSS，Javadoc 形式で書かれたコメントの行数，コメントが Javadoc 形式で書かれているか，付属ドキュメントが示されている．表示する順番は単に取得日時が新しい順となっている．

No	Date	ClassName	Package	# functions	LOC	NCSS	Javadoc	Javadoc Line	付属 document
1	2002-12-04	SortApplet		3	55	37	0%	0	-
2	2002-12-04	Sort		8	41	23	0%	0	-
3	2002-12-04	ElemSort	org.apache.xalan.templates	14	386	50	100%	214	-
4	2002-12-04	ProcessorTemplateElem	org.apache.xalan.processor	3	171	36	100%	38	-
5	2002-12-04	SortingFocusTraversalPolicy	javax.swing	15	473	48	100%	168	-

No	Date	Method	返り値の型	Class	Package	LOC	NCSS	Javadoc Line	Javadoc	付属 document
1	2002-12-04	sortIndexMap()	void	IndexBuilder	com.sun.tools.doclets	5	3	2	○	-
2	2002-12-04	sort(int[] a, int fromIndex, int toIndex)	void	Arrays	java.util	4	3	18	○	-

図 5 検索結果

収集できた圧縮ファイルの傾向として，製品のアップデートファイル（ java 開発環境ソフトなど）などのあまり有用でないものも含まれていたが，その 8 割以上がソースコードを含んでおり，通常の巡回ロボットに見られない効果が得られた．しかしリンク文字列や URL の先読みから資源が見つかった例は 1 割以下にとどまり，検索エンジンにより起点を見つける方式の効果のほうがより重要であると考えられる．

プロトタイプでの収集には以下のような傾向があった．まず，「java download」や「java sample」などのクエリで検索すると，sun のページなど有名なページに陥って，毎回同じような経路で巡回してしまい，似たようなものしか収集できないことがあった．本稿の巡回方法では，資源が収集できたら次サイトに抜けようと試みるので，sun のように大規模なサイトから脱出できないということはないが，抜けた後のルートが似たようなものになってしまうので，たとえば登録しておいたサイトは巡回しないなどの制限を加える必要があるかもしれない．

本実装と実験は一台のパソコンで行ったが，実験前には圧縮ファイルを解凍してメトリクスを計測することはとてもコストが高く，処理が追いつかない可能性があったが，それほど多くの資源が収集できるわけでもなく解凍などが大きな負担となることはなかった．しかし，もっと多くの台数で巡回したり，また並列化したりすることなども考えると，明らかに解凍しなくても良いものは放置するなど，効率を見直す必要が出てくるだろう．

5. まとめと今後の課題

本論文では，WWW を対象にメトリクスなどをキーとして検索することのできる検索エンジンの構築法について提案を行い，プロトタイプを実装して想定した検索が行えることを確認した．

現段階では検索エンジンに与えるクエリについてあまり検討できていない．たとえばソフトウェア資源をうまく発見できたページの周辺の特徴的単語を加えたり，実績がなかった単語をはずしたりなどの工夫を行えば収集の効率は上がるものと考える．

ユーザに提供する際にも，ある程度の評価を行いランク付けして表示することも必要であろう．たとえば利用実績や，ソースコード間の関係から利用頻度の高そうなソースコードを推薦する [6] などの方法も考えられる．

謝辞 本研究は，平成 14 年度科学技術振興事業団計算科学技術活用型特定研究開発推進事業(ACT-JST)の支援を受けている．

文 献

[1] C. Aggarwal, F. Al-Garawi, P. Yu, "Intelligent Crawling on the World Wide Web with Arbitrary Predicates", World Wide Web 2001.

[2] G. W. Flake, S. Lawrence, C. L. Giles, "Efficient Identification of Web Communities", Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp.150-160, 2000.

[3] 井上克郎，松本健一，飯田元，"プロセスと環境 トラック ソフトウェアプロセス"，共立出版株式会社，2000.

[4] L. Elshoff, "An investigation into effects of the computing method used on software ecience measurements, "SIGPLAN Notices, Vol.13, No.2, pp.30-45, 1978.

[5] S. R. Chidamber and C. F. Kemerer, "A metrics suite for object-oriented design, "IEEE Transactions on Software Engineering, Vol20, No.6, pp.476-493, 1994.

[6] 横森励士，山本哲男，楠本真二，藤原晃，松下誠，井上克郎，"ソフトウェア部品間の利用関係を用いた再利用性評価手法の提案"，ソフトウェア・シンポジウム 2002 論文集，pp.216-225, 2002.

[7] DOWNLOAD.COM, http://www.download.com/

[8] JavaNCSS - A Source Measurement Suite for Java, http://www.kclee.com/clemens/java/javancss/

[9] 窓の杜，http://www.forest.impress.co.jp/

[10] SourceForge, http://sourceforge.net/

[11] Vector, http://www.vector.co.jp/