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.

Java ソフトウェア部品解析・検索システム SPARS-J の構築

西　秀雄† 梅森　文彰† 横森　励士† 山本　哲男†† 松下　誠†
楠本　真二† 井上　克郎†

†† 独立行政法人科学技術振興機構　〒 332-0012 埼玉県川口市本町 4–1–8
† 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
E-mail: †{h-nisi,umemori,yokomori,t-yamamt,matusita,kusumoto,inoue}@ist.osaka-u.ac.jp

あらまし　本論文では，Java ソフトウェア部品の解析・検索システム SPARS-J(Software Product Archive, analysis and Retrieval System for Java) の構築を行う．SPARS-J はキーワードを検索キーとして，検索キーと関連したソフトウェア部品のソースコード，および再利用支援のために有益な情報を検索結果として提供する．本システムを用いることで，再利用性の高いソフトウェア部品を再利用することができるため，結果として生産性と品質を改善しコストを削減できる．

キーワード　ソフトウェア部品，再利用，Java

Java Software Component Analysis and Retrieval System SPARS-J

Hideo NISHI†, Fumiaki UMEMORI†, Reishi YOKOMORI†, Tetsuo YAMAMOTO††, Makoto MATSUSHITA†, Shinji KUSUMOTO†, and Katsuro INOUE†

†† Japan Science and Technology Agency
4–1–8, Honmachi, Kawaguchi-shi, Saitama 332–8531, Japan
† Graduate School of Information Science and Technology, Osaka University
1–3, Machikaneyama-cho, Toyonaka-shi, Osaka 560–8531, Japan
E-mail: †{h-nisi,umemori,yokomori,t-yamamt,matusita,kusumoto,inoue}@ist.osaka-u.ac.jp

Abstract　Reusing the software components that have high reusability improve the software productivity, quality and cost. In this paper, we develope SPARS-J(Software Product Archive, analysis and Retrieval System for Java). A component searcher will give SPARS-J queries by keywords, and get matched source codes or efficient information for reuse. Using SPARS-J, we can save a lot of cost by reusing software component with high reusability.

Key words　Software component, Reuse, Java

1. はじめに

近年のソフトウェア開発の大規模化と複雑化に伴い，高品質なソフトウェアを一定期間内に効率良く開発することが重要な課題となっている．それらの課題を解決すべく様々なソフトウェア工学技術が提案されており，代表的な技術の一つとしてソフトウェア部品の再利用がある．

再利用はライブラリ中の既存のソフトウェア部品を同一システム内や他のシステムで用いることであると定義されている [1]．一般にソフトウェアの再利用は生産性と品質を改善し，結果としてコスト削減する報告が多く出されている [2].

再利用の効果を最大限に得るためには，開発者がライブラリに関する十分な知識を持つことが必要である．しかしながら，開発プロジェクト内の多くのメンバー間で知識の共有を行なうことは非常に困難で，コストのかかる作業である．このため，再利用可能なソフトウェア部品がライブラリ中に存在するにも関わらず，同種の部品が独立して開発されていることが多々ある．

このとき，過去のソフトウェア開発の遺産を効果的に再利用するために，大量のソフトウェア部品を解析しデータベースに蓄積する解析システムが必要である．また，データベースから開発者の必要としている機能を持つ部品を検索し，その部品がどのように使われているかなどといった再利用に有益な情報を提供する検索システムを構築する必要がある．

構築したソフトウェア部品解析・検索システムを用いることで，開発プロジェクトで品質や機能が高く将来において再利用

することが望ましいソフトウェア部品が生産された場合に，それらを効率良く整理して保存し，必要に応じて参照可能となる．これによって，ライブラリの知識が無い開発者も有用なソフトウェア部品やそれに付随する有益な情報を平易に入手することができるため，効果的な再利用が期待できる．

一方で，インターネットの普及により，大量のソフトウェア部品を容易に入手することが可能となった．SourceForge [3] などのソフトウェア開発コミュニティでは，大量のプログラムソースコードを容易に入手できる．これらのソフトウェア部品から要求を満たす部品を検索することができれば，効果的な再利用が可能となると考えられる．

我々は以前の提案 [4] をもとに，本論文において，Java を対象としたソフトウェア部品解析・検索システム SPARS-J(Software Product Archive, analysis and Retrieval System for Java) を構築する．SPARS-J は Java ソフトウェア部品の容易な検索，参照を実現するためのシステムである．キーワードを検索キーとして，検索キーと関連したソフトウェアのソースコードを効率良く検索することが可能である．さらに，検索結果表示の際にソフトウェア部品に関する詳細な情報を併せて提供する．

以降，2. 節では，SPARS-J における基本的な概念について簡単に説明する．また，関連研究の紹介を行なう．3. 節では，構築した SPARS-J システムの構成について説明する．4. 節では，システムの使用例を示す．最後に，5. 節で本研究のまとめと今後の課題について述べる．

2. 準　　備

本節では，ソフトウェア部品や類似部品群，検索結果の順位付け手法など，SPARS-J の根幹をなす概念について簡単に説明する．

2.1 ソフトウェア部品と部品群

一般にソフトウェア部品 (Software Component) は再利用できるように設計された部品とされる [5]．しかし，部品の集合にはコピーした部品や，コピーして一部変更した部品が多く存在する．

そこで，SPARS-J では類似した部品をまとめることにより，部品の集合をいくつかの部品群に分類する．部品の集合を部品群に分類するために，任意の部品間の類似度 (Similarity) をメトリクスを用いて定量的に評価し，閾値以上の類似度をもつ部品を同一部品群として分類する．

また一般的に，単体の部品間には互いに利用する，利用されるという利用関係が存在する．そこで，ある部品群に属する部品が，他の部品群に属する部品を利用している場合には，その 2 つの部品群間には利用関係が存在するとみなしている．

2.2 順位付け手法

一般に部品検索をキーワードによって行なうと，検索キーと合致する索引キーを持つものが多数存在する場合がある．そのため，検索結果を提示する際に適当な順位で表示することが必要となる．

テキスト文書を対象とした情報検索システムで一般的に用いられている手法では，検索対象となる文書集合から各文書の特徴を表すキーワードである索引語を抽出し，索引語の集合によってその文書の内容を近似する．登録するそれぞれの文書の特徴を的確に表すように付与された，索引語の集合などからなる情報のことを索引キーと呼ぶ．検索キーは検索者の要求の内容を近似しているため，検索キーと索引キーを用いて検索キーと文書の適合度を測り，順位付けするのが一般的である．

しかし，ソフトウェア部品を対象として再利用のための部品を検索する場合は，検索者の要求の内容と適合する部品であるかどうかのほかに，その部品がよく利用されている部品かどうかに関しても考慮する必要がある．検索キーと適合度が高い部品よりも，よく利用されている部品を上位に提示した方が，利用例も多く出る可能性があり，部品の再利用をスムーズに行なうことが可能な場合もある．そのため，利用しやすい部品かどうか定量的に評価するための指標を導入し，検索キーと部品の適合度も併せて，両面を考慮した部品の順位付けを行なう必要がある．

索引語の重み付けによる適合度の評価

索引語の中には部品の内容と密接に関係したものもあれば，関係の薄いものも存在する．抽出された索引語が部品の内容を表すうえでどれだけの重要度を持っているか測ることができれば，より精度の高い検索を実現できると考えられる．このために用いられるのが索引語の重み付けである．索引語の重みを利用することによって，同じ索引語を含む部品でもその索引語の各部品中での重要度を考慮して，検索キーに対する部品の適合度を計算し部品を順序付けすることが可能になる．検索者が与えた検索キーがある部品の索引キーにヒットしたとき，その索引キー中の索引語の重みの総和を検索キーと部品の適合度とする．SPARS-J における索引語の重み付け手法として，情報検索の分野で一般的に用いられる TF-IDF 法 [6] を用いて各部品における重み付けを行なう．TF-IDF 法は，任意の部品中における特定の索引語の出現頻度 TF(Term Frequency)，および特定の索引語を含む部品数の逆数 IDF(Inverse Document Frequency) の値を正規化して重みを算出する．TF は部品内で出現頻度の低い索引語と高い索引語を差別化し，部品をより特徴付ける語を選別するためのものである．IDF は部品集合内の他の部品の索引語の分布について考慮するためのもので，ある索引語が，どの程度その部品に特徴的に現れるのかという特定性を示す．検索キーと部品の適合度の高い順に順位付けすることを，後述する CR 法に対して KR 法 (Keyword Rank 法) と呼ぶことにする．また，測定した適合度の値を KR 値と呼ぶ．

利用関係による評価

我々はこれまでに，利用関係からソフトウェア部品の利用実績を測定し，順位付けし，評価する手法 (Component Rank 法, CR 法) を提案している [7]．CR 法では，十分な時間が経過し利用関係が収束した部品の集合に対して，各部品間に存在する利用関係に基づいてグラフおよび行列を構築し，構築された行列に対して繰り返し計算を行う事で各部品を評価する．求められる値は，開発者が利用関係に沿って参照を行うと仮定した場合の各部品の参照されやすさを表しており，よく利用される部品や，重要な部品から利用される部品の順位は高くなる．

2.1 で述べたように，実際の部品の集合には多数のコピーや類似した部品が存在している．異なるシステムをまたいで全く同じ，もしくはほとんど似た部品が現れる場合，それらの部品が再利用されたものと推測できる．そのため，CR 法では部品群を部品の単位としてみなす．それぞれの類似した部品への利用関係が一つの部品群への利用関係とみなされるため，コピーされた部品への評価を高くすることが可能となる．

2.3 関連研究

これまでにソフトウェア検索を行なうために種々の手法が提案されている．庭山ら [8] はセマンティック Web の手法を利用してソフトウェアのソースコードにメタデータを付与し，検索ワードをオントロジ変換したものとのマッチングを行なうことで，ソフトウェア部品を意味に基づいて検索する手法を提案している．また，鷲崎ら [9] は独立して再利用可能なコンポーネントを，検索および試行可能な検索システムの提案を行なっている．提案システムでは，クラス間の依存関係を解析することで，要求された機能を実現するような複数のソースコードをコンポーネントとして抽出し，ユーザに提供する．

他に，WWW のソフトウェア資源を自動収集し，それらを解析することでデータベースに分類・索引付けして蓄積し，作成したデータベースを対象に検索を行なう形式のソフトウェア検索システムに [10] [11] [12] がある．Agora [10] はサーチエンジン AltaVista の SDK を利用してソフトウェア資源の自動収集を行なう．CORBA オブジェクトである ORB や Java Beans を検索可能である．jCentral [11] は IBM が開発した Java のソフトウェア資源検索システムである．jCentral では Java ソースコードの他，アプレット，Java Beans，ニュース，FAQ，チュートリアルなどを検索することが可能である．亀井ら [12] の提案したシステムでは，検索キーにソフトウェアメトリクスなどの特有の値を含めてソフトウェアを検索することができる．

3. SPARS-J システム

本節では我々が構築した Java ソフトウェア部品解析・検索システム SPARS-J について説明する．まず，システムの機能と構成の概要を述べ，その後個々の構成部品について詳しく説明する．

3.1 システムの機能と構成

SPARS-J におけるソフトウェア部品とは，Java のクラスまたはインタフェースを単位とするソースコードを指す．以降，単に部品と呼ぶ．本システムが持つ機能は以下の通りである．

- キーワード検索
  ユーザが要求したキーワードを検索キーとして部品の検索を行なう．
- 部品間の利用関係の提示
  部品間の利用関係を提示することで，部品の使用方法に関する情報を得ることを可能にしている．
- パッケージブラウザ
  クラス階層構造をハイパーリンクで表現することでパッケージ内に含まれるクラスの一覧を取得できる．

本システムは以下のような構成をとる．構成を図 1 に示す．

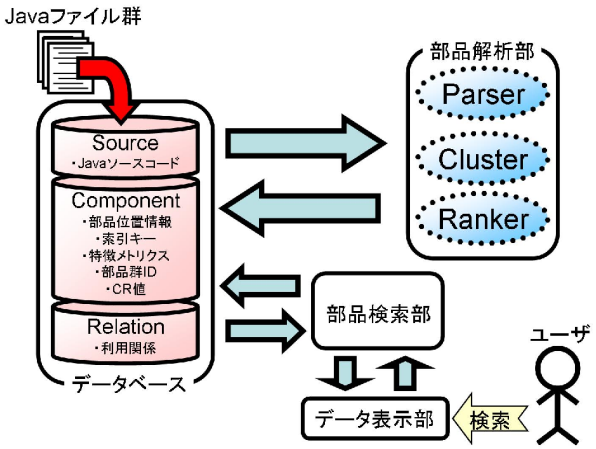

図 1　SPARS-J の構成

- データベース (Source リポジトリ, Component リポジトリ, Relation リポジトリ)
- 部品解析部 (Parser, Cluster, Ranker)
- 部品検索部
- データ表示部

記述言語として C 言語および C++言語を，データベースに BerkeleyDB [13] を用いてシステムの実装を行なった．ソースコードの規模は，データベース約 7800 行，部品解析部約 13000 行，部品検索部・データ表示部約 6500 行，合計約 27000 行であった．

3.2 データベースの説明

3.2.1 Source リポジトリ

Source リポジトリは Java で記述されたソースコードのファイルをそのままの形で保存しておく貯蔵庫である．解析対象である Java ファイルは，まず最初に Source リポジトリに格納する．格納する際には，一意なファイル ID を与える．

3.2.2 Component リポジトリ

Component リポジトリには，Source リポジトリ中の Java ファイルを解析することによって得られた，各部品に関する情報が格納される．詳細は以下の通りである．

- 部品のファイル中の位置情報
- 部品の索引キー
- 部品の特徴メトリクス計測値
- 部品が所属する部品群 ID
- 部品の CR 値

構文解析した結果と特徴メトリクスの値を部品 ID と関連付けし，格納する．また，Component リポジトリは検索の際にも使用するため，各解析結果と部品 ID の逆引きの表を作成する．

3.2.3 Relation リポジトリ

各部品間の利用関係を格納するリポジトリである．利用関係は，利用関係の種類と関係を結ぶ部品 ID の組から構成される．逆に，任意の部品 ID を選んだ際にどの部品 ID とどのような関係が存在するかを高速に検索するために，部品 ID の数だけ逆引きの表を作成する．

3.3 部品解析部の説明

部品解析部は，Java ファイルを解析することで，検索時に必要な情報を抽出し，それらの情報を各種リポジトリに格納する．

3.3.1 Parser

Source リポジトリに追加された Java ファイル群は Parser で解析される．Parser は Java ファイルの構文解析を行ない，検索に必要とされる種々の情報を抽出する．まず，ファイル中からクラスまたはインタフェースを見つけ出し部品として切り出し一意な部品 ID を与える．部品は記述されているファイル ID やファイル中の行番号をファイル位置情報として Component リポジトリに格納する．

さらに，各部品の索引付けを行なう．検索時に利用するために各部品中に含まれる識別子やコメント中の語句を索引語として抽出し，種類 (クラス宣言名やコメントなど) 毎に出現頻度をカウントする．それらを索引キーとして Component リポジトリに格納する．

また，部品間の利用関係の抽出を行う．本システムで利用関係として抽出するものを以下に示す．

- クラスの継承関係
- インタフェースの実装関係
- 抽象クラスの実装関係
- メソッドの呼び出し関係
- フィールドの参照関係
- インスタンスの生成関係
- 変数宣言関係

上記の内，実行時に動的に決定される利用関係については静的な構文解析のみで一意に特定する事は不可能である．しかし，全てのソフトウェア部品について実行を行うのは多くのコストがかかり，実行する際に入力が必要な場合もあるので，自動化することが難しい．そのため，静的解析を行い，宣言されている部品に対して利用関係を抽出する．抽出した利用関係は Relation リポジトリに格納する．

Parser では，その他に特徴メトリクスの計測を行い，後述する Cluster によって部品群化を行なう．計測したメトリクス値は Component リポジトリに格納する．部品の特徴メトリクスを計測し比較することで，類似部品をまとめた部品群を作成することが可能になる．すべてのメトリクスは構文解析時に計測可能であるため，Parser で特徴メトリクスの計測だけを行う．

3.3.2 Cluster

Parser で求めた特徴メトリクスを用いて類似部品を部品群にまとめる．我々は文献 [14] において，Java プログラム間類似度測定手法の提案を行なっており，同手法に基づいて類似部品と判断された部品同士を同一部品群にまとめる．この際，各部品の利用関係は部品群毎に取得できるようにまとめている．

3.3.3 Ranker

Relation リポジトリに格納された部品群間の利用関係から CR 値の計算を行う．最初に，Relation リポジトリに格納されている利用関係を取り出す．部品の重要度評価値を求める計算は，行列の固有値計算に帰着され，利用頻度の順位を決定した後，部品 ID と対にして Component リポジトリに格納する．

索引語の種類	重み	索引語の種類	重み
クラス定義名	200	メソッド定義名	200
インタフェース名	50	パッケージ名	50
インポートパッケージ名	30	呼出しメソッド名	10
参照フィールド変数名	10	生成したクラス名	10
変数の型名	10	参照する変数名	1
コメント (/*...*/)	30	文書コメント (/**...*/)	50
行末コメント (//...)	10	文字列リテラル	1

表 1　索引語の種類とその重み

3.4 部品検索部

部品検索部は，検索者によって与えられた検索条件から検索の種類を判別し，要求と一致する部品を検索する．また，検索結果の部品を評価値が高い順に順位付けして，データ表示部に渡す．検索条件には以下の項目を指定することができる．

- 複数のキーワードからなる検索キー
- 検索対象とするパッケージ名
- 検索対象とする索引語の種類
- 順位付けの方法
- 検索方法 (AND-OR 検索) や，検索キーの大文字小文字の区別の有無

部品検索部は，検索条件に応じて Component リポジトリの適切な部分のみを検索し，ヒットした部品の部品 ID のリストを作成する．ヒットしたリスト中の部品 ID を持つ各部品について，索引キー中の索引語の重みの総和を検索キーと部品の適合度とし，KR 法を用いて順位付けする．索引語の重みを算出する際に，種類によって異なる重みを与れば，Java ソフトウェア部品の特性を視野に入れた順位付けが可能となる．

SPARS-J が区別する索引語の種類と重み付けの例を表 1 に示す．どの種類の索引語にどの程度の重みを与えると良い結果が得られるかは知られていないが，ここではクラス定義名やメソッド定義名など，その部品の概念を象徴した名前が付けられる傾向の索引語に対して大きな重みを設定している．

部品 c における索引語 t の重み w_{ct} を以下の式で求める．

$$w_{ct} = \left(\sum^{\text{索引語の種類数}} kw_i \cdot TF_i \right) \cdot IDF$$

kw_i は索引語の種類 i の重みを指し，その値は表 1 に従う．例えば，$kw_{\text{クラス定義名}}$ は 200 である．さらに，TF_i は c における索引語の種類 i と一致する索引語の出現頻度，IDF は $(\frac{\text{データベース中の総部品数}}{t \text{ を含む部品数}})$ とする．

算出した索引語の重みの総和を KR 値として算出する．その後，求めた KR 法による評価値と，Ranker で求めた CR 法による評価値をもとに検索条件に指定した方法でリストを再度順位付けし，結果をデータ表示部に渡す．指定できる順位付けの方法として，KR 法のみ，CR 法のみ，および KR 法と CR 法の合成手法を現在実現している．現在の KR 法と CR 法の合成手法では，KR 法と CR 法それぞれの順位を用いて統合順位を求める．各々の部品の KR 法と CR 法それぞれにおける順位をそのまま点数として足し合わせ，合計点数が少ないものから昇順に順位付けする．

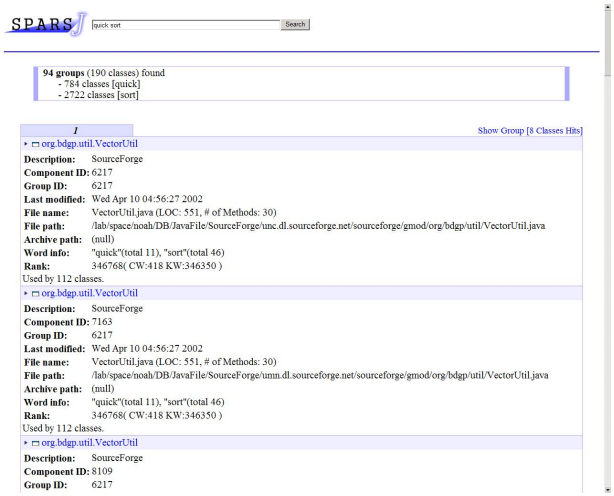

図 2　検索結果表示

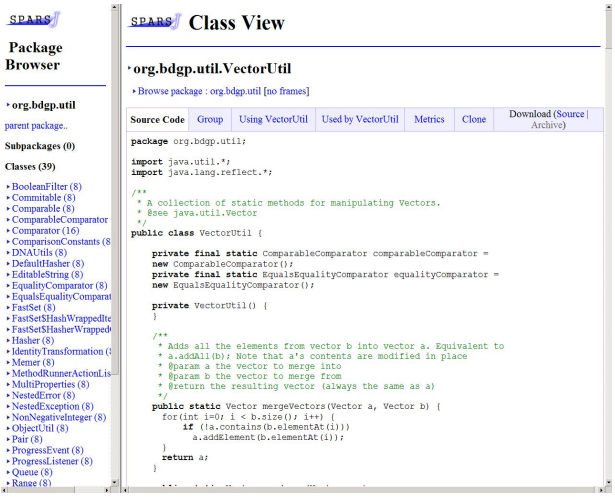

図 3　詳細データ表示

3.5　データ表示部

データ表示部は部品検索部と検索者を結ぶ Web インタフェースである．SPARS-J は Microsoft の Internet Explorer などの Web インタフェースを通して検索機能を提供する．検索者が指定した検索条件を部品検索部に渡し，部品検索部が返してくる検索結果の部品リストを受け取って表示する．

検索結果表示では，ヒットした部品を部品リストの順位で表示する．検索結果表示の様子を図 2 に示す．各部品の行数や部品中で定義しているメソッド数，部品の利用実績を併せて表示することで，再利用に有益な情報を検索者に提供する．詳細を知りたい部品を選択することで，部品の詳細データ表示が可能である．

部品の詳細データ表示では，部品のソースコードや，同一部品群に属する部品，パッケージブラウザによるクラス階層構造，部品間の利用関係，などといった各部品の詳細情報を閲覧することができる．詳細データ表示の様子を図 3 に示す．

4.　適　用　例

本システムを使用した適用例の結果を紹介し，それらについて考察する．比較対象として代表的なキーワード検索型の WWW 検索エンジンである Google [15] を用いる．本システムの検索結果の上位 10 部品と Google の検索結果の上位 10 ページを調べ，目的の情報と適合するかどうかユーザが判断し，検索結果の適合率を算出する．

準備として，我々が過去に開発した Java ソフトウェアや，インターネットから取得してきた Java ソフトウェアをあらかじめ解析しデータベースに登録する．このようにして約 15 万個の部品をデータベースに登録し，適用例に使用している．

適用例として，Java プログラミング習熟度が低いユーザが，電卓アプレットおよびサーバ・クライアント方式のチャットプログラム作成を行なう場合を考える．Java プログラミング習熟度が低いとは，教材を通して一通り Java について学んだがコーディングを行なったことがない程度のユーザの習熟度を指す．また，ユーザは本システムについて十分な説明を受けており，使用において問題がないこととする．本システムの検索条件には以下の条件を与える．

- 検索対象は全種類の索引語
- 順位付けの方法は KR 法と CR 法の合成手法とする
- 検索方法は AND 検索で大文字小文字は区別しない

本システムはソースコード集合を対象に検索を行なうので，WWW 検索エンジンである Google は不利となる．その点を解消するために以下のように決める．

- Google の検索キーは SPARS-J の検索キーにキーワード ”java source” を追加したものを与える．
- Google の検索結果 Web ページから 1 リンク以内に目的に応じたソースコードが存在すれば適合とみなす．
- SPARS-J の利用関係提示機能とパッケージブラウザ機能は使用しない．

4.1　適用例１:電卓アプレット作成

ユーザが電卓を GUI で実現したい場合を考える．電卓機能から連想されるキーワードとして ”calculator”を ，GUI で実現することからキーワード ”applet” をユーザが連想すると想定し，SPARS-J の検索キーに ”calculator applet” を与える．この検索キーは，Java プログラミング習熟度が低いユーザでも思い浮かぶ，妥当なものであると考えられる．また，Google の検索キーには ”calculator applet java source” を与える．検索した結果，SPARS-J でヒットしたのは 9 件であった．結果を表 2 に示す．なお表中の各順位における適合率は，適合率 $=$ $\frac{\text{現在の順位までで適合した数}}{\text{現在の順位}}$ で与えられる．

4.2　適用例２:チャットプログラム作成

続いてユーザがサーバー・クライアント方式のチャットプログラムを実現する場合を考える．SPARS-J には検索キーとして ”chat client server” を与え，Google には ”chat client server java source” を与えた．検索した結果，SPARS-J でヒットしたのは 69 件であった．結果を表 3 に示す．

4.3　適用例に関する考察

適用例１，２いずれも SPARS-J は適合する部品が上位に示されていることがわかる．適用例１では 9 件の部品中，適合する部品 7 件が全て不適合の部品よりも上位に表示され，不適合な部品は下位に表示されている．適用例２ではヒットした 69

	SPARS-J		Google	
順位	適合	適合率	適合	適合率
1	○	1	○	1
2	○	1	×	0.5
3	○	1	○	0.67
4	○	1	×	0.5
5	○	1	○	0.6
6	×	0.83	○	0.67
7	○	0.86	×	0.57
8	×	0.75	○	0.63
9	○	0.78	×	0.56
10	-	-	×	0.5

表 2　適用例 1 の結果

	SPARS-J		Google	
適合	適合率	適合	適合率
○	1	×	0

表 3　適用例 2 の結果

件の部品中，適合する部品は 57 件であり，適合する部品が上位に来て不適合な部品は下位に集中しているため，適合率は 1 を保っている．それに対して Google では，ソースコードが存在せずアプレットそのものが直接表示されるだけの場合や，製品の紹介などが行なわれているだけの場合が多く，再利用に適するものは少なかった．すなわち，適合しないものが上位に表示される場合が多く，ソースコードの検索において SPARS-J の順位付けが有効であることがわかる．

SPARS-J は Java ソースコードのみの集合から検索を行なうため，WWW 検索エンジンである Google と比較して SPARS-J が優れているのは妥当な結果である．Google の結果では，ソースコードが存在せずとも再利用に有益な情報を記している Web ページが存在するかもしれないが，今回そのような情報は不適合としている．よって，将来的には純粋に再利用支援という観点から見た場合の有効性の検証が必要であると思われる．しかしながら，今回見ることの無かった，利用関係による部品の使用方法に関する情報取得や，パッケージブラウザによる有効性の向上も視野に入れると，本システムは有効であると考えられる．

5. まとめと今後の課題

本研究では，Java ソフトウェア部品の解析・検索システム SPARS-J を構築した．SPARS-J を用いることで，インターネット上で入手可能な大量のプログラムソースコードから，要求を満たす部品を検索することが可能である．

適用例では代表的な WWW 検索エンジンである Google と比較を行ない，得られた結果に対して考察を行なった．上位 10 件の適合率は SPARS-J が高く，ユーザの要求する部品が上位に表示されていることがおり，本システムの有効性を確認することができた．

今後の課題として以下が挙げられる．

- より効果的な索引付け
  現在の索引付けは対象が Java プログラムであることを考慮していない．Java プログラムに特化した索引付け手法を適用することで，検索精度の向上ができると考えられる．
- 協調フィルタリング [16] による検索精度の向上
  協調フィルタリングは，検索者の行動を保存し，それをもとに過去の検索者との行動の類似性から検索者の要求を推測することである．協調フィルタリングを用いることで，より適切な情報を検索者に提供することができると考えられる．
- 順位付け手法に関する検討
  KR 法の索引語の種類による重みの調整を行なうことで，より良い順位が得られる可能性がある．CR 法と KR 法の合成手法についても調整を行なう必要がある．
- 本システムの詳細な実験評価
  本システムの実験評価にとして，利用関係による部品の使用方法に関する情報取得や，パッケージブラウザによる有効性の向上などを視野に入れた，より詳細な再利用支援システムとしての評価が必要である．

謝辞　本研究は，独立行政法人科学技術振興機構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. Braun: Reuse, in John J. Marciniak, editor, *Encyclopedia of Software Engineering*, Vol. 2, John Wiley & Sons, pp. 1055-1069 (1994).
[2] V. R. Basili, G. Caldiera, F. McGarry, R. Pajerski, G. Page and S. Waligora: "The software engineering laboratory - an operational software experience", *Proc. of ICSE14*, pp. 370-381 (1992).
[3] "SourceForge",http://sourceforge.net/
[4] 山本, 横森, 松下, 楠本, 井上: "利用頻度に基づくソフトウェア部品の解析・検索システムの提案", 電子情報通信学会技術研究報告, SS2002-17, Vol. 102, No. 329, pp. 13-18, (2002).
[5] I. Jacobson, M. Griss and P. Jonsson: Software Reuse, *Addison Wesley*, (1997).
[6] 北, 津田, 獅々堀: 情報検索アルゴリズム, 共立出版, (2002).
[7] 横森, 藤原, 山本, 松下, 楠本, 井上: "利用実績に基づくソフトウェア部品重要度評価システム", 電子情報通信学会論文誌 D-I, Vol. J86-D-I, No. 9, pp. 671-681, (2003) , and Technical Report of SE Lab, Dept. of Computer Science, Osaka University, SEL-Nov-21-2002, Nov. (2002).
[8] 庭山, 松尾, 萩原, 金田: "セマンティック Web を利用したソフトウェア検索システムの提案", 電子情報通信学会技術研究報告, SS2002-57, Vol. 102, No. 704, pp. 27-31, (2003).
[9] 鷲崎, 深澤: "オブジェクト指向プログラムのためのコンポーネント抽出型検索システム", オブジェクト指向 2003 シンポジウム (OO2003), (2003).
[10] R. C. Seacord, S. A. Hissam, K. C. Wallnau: "Agora: A Search Engine for Software Components", *IEEE Internet Computing*, Vol. 2, No. 6, pp. 62-70 (1998).
[11] M. H. Aviram: "Code-centric search tool strives to reduce Java development time", *Java World*, June (1998).
[12] 亀井, 門田, 松本: "WWW を対象としたソフトウェア検索エンジンの構築", 電子情報通信学会技術研究報告, SS2002-47, Vol. 102, No. 617, pp. 59-64, (2003).
[13] "BerkeleyDB",http://www.sleepycat.com/
[14] 小堀, 山本, 松下, 井上: "類似度メトリクスを用いた Java ソースコード間類似度測定ツールの試作", 電子情報通信学会技術研究報告, SS2003-2, Vol. 103, No. 102, pp. 7-12, (2003).
[15] "Google",http://www.google.com/
[16] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, J. Riedl: "GroupLens: an open architecture for collaborative filtering of netnews", *Proc. of the 1994 CSCW*, pp. 175-186 (1994).