データベースシステム 119－49
（１９９９．７．２２）

仕様に基づいたRDBクエリ検索システム

村田 美友紀 † 掛下 哲郎 ‡

DBアプリケーション構築の際には数多くのクエリが必要になる．これらのクエリの中には，入れ子クエリやビューを用いたクエリのように実現が難しいものも含まれる．したがって，効率的なアプリケーションの開発を促進するためには，クエリの再利用が不可欠である．本稿では，仕様に基づいたRDBクエリ検索システムを提案する．本システムは，RDBクエリの仕様を属性，組，組のペアを要素とする集合を用いて表現し，集合要素から構成されるサンプルを用いて検索を行なう．ここで，任意のクエリを唯一に検索するサンプルを作成するために要素辞書を用いる．また効率的にサンプルを作成するための利用者支援機構として，フィルタリング，検索，重複度別分類を提案する．

A Specification Based Retrieval System for RDB Queries

Miyuki Murata† Tetsuro Kakeshita‡

A DB application contains number of queries which may be nested or composed of views. In general design of a query containing subqueries is not easy. Reuse of query thus is required for effective construction of a DB application. A specification based query retrieval system then becomes necessary. We propose a sample based RDB query retrieval system. Query specification is represented using a set whose members are projection attributes, tuples and tuple pairs. We introduce the element dictionary in order to construct a sample which can identify a query in the repository. Furthermore we propose three mechanisms in order to assist sample construction. They are filtering, element retrieval and element weight.

1 はじめに

DBアプリケーション構築の際には，多様な利用目的に対応したフォームやレポートを作成するため，多くのクエリが必要になる．その中には入れ子クエリやビューを基に作成されたクエリもある．このようなクエリを作成するためには，サブクエリへの分割が必要である．サブクエリ数が多い場合やサブクエリの入れ子が深い場合には，適切な分割は困難になる．したがって，既存クエリの再利用が不可欠である．また，クエリの再利用はDBアプリケーションの開発や改良の効率化を促進するためにも重要である．

クエリを再利用するためには，仕様に基づいた検索機構が必要になる．本稿では，仕様に基づいたRDBクエリ検索システムを提案する．本システムでは，集合を用いてクエリの仕様を表現し，集合要素から作成したサンプルを用いてクエリを検索する．また，クエリを格納したリポジトリに対して要素辞書 [1] を定義する．要素辞書を用いるとリポジトリに格納されている任意のクエリを特定するサンプルが構成できる．サンプルの作成は利用者が行なうが，サンプルに追加する要素を要素辞書から選択するための支援機構として，フィルタリング，検索，重複度別分類を提案する．これらを組み合わせることで，効率的にサンプルを作成できる．

ソフトウエア工学分野では，ソフトウエアコンポーネントを再利用するために，仕様に基づいた検索方式が研究されている．Rittri [2] や Zaremski [3] によって，シグネチャマッチングによる検索方式が提案されている．これは引数や戻値の型を関数検索の条件として用いる．この方式では詳細な検索条件の記述が困難である．Fisher[4] や Mili[5] は，利用者が記述した仕様にマッチングするコンポーネントを検索する方式を提案している．この方式では利用者が仕様記述言語を習得する必要がある．また，theorem prover を用いて検索を実現するため，大規模リポジトリに対する検索の効率化が難しい．Podgurski [6] は入力や出力の値をサンプルに用いた関数検索機構を提案しているが，サンプル作成は利用者に任されており，利用者に対する支援機構は提供されていない．

2 サンプルを用いたクエリ検索

本節では，サンプルを用いたクエリ検索機構を定義する．それに先立ち，クエリの仕様を集合を用いて表現する．

†八代工業高等専門学校 情報電子工学科
Department of Information and Electronics Engineering,
Yatsushiro National College of Technology
‡佐賀大学理工学部 知能情報システム学科
Department of Information Science, Saga University

2.1 クエリ仕様の集合表現

クエリの仕様は，クエリの実行結果を用いて表現できる．クエリの実行結果は，属性，組，組のペアの集合で表現できる．属性集合はクエリの射影属性から構成される．射影属性には，関係スキーマで定義された属性だけでなく，集計や関数を用いて計算された属性も含まれる．組の集合はクエリが検索する組から構成される．ソート条件を含むクエリの仕様を表現するためには，検索される組の表示順序を指定する必要がある．そこで，組のペアから構成される集合を用いて組の表示順序を表現する．クエリの仕様の集合を用いた表現の例を以下に示す。

例 1 図 1(A) に示す 2 つの関係を考える．これらに対して，図 1(B) のクエリ q_1を実行した結果が図 1(C) である．よって，q_1の仕様は以下の集合を用いて表現できる．

{ 学生, 科目, 点, ⟨ 森, 英語, 43⟩,
⟨ 森, 英会話, 49⟩, ⟨ 鈴木, 英会話, 45⟩,
[⟨ 森, 英語, 43⟩, ⟨ 森, 英会話, 49⟩],
[⟨ 森, 英語, 43⟩, ⟨ 鈴木, 英会話, 45⟩],
[⟨ 森, 英会話, 49⟩, ⟨ 鈴木, 英会話, 45⟩] } □

クエリの仕様を表現する集合をリポジトリに直接格納するためには，大きな格納領域が必要になる．このため，リポジトリにはクエリ自身を格納する．クエリの仕様を表現する集合は，クエリの実行結果を用いて定義されるため，DB の状態によって変化する．そこで，通常の DB 操作では変更されない検索用 DB を導入する．検索用 DB においてクエリ q_iの仕様を表現する集合を g_iと表記する．ここで，リポジトリに格納されている任意のクエリ $q_i, q_j (q_i \neq q_j)$ に対して，$g_i \neq g_j$となるように検索用 DB の状態を決定する．このような検索用 DB の構築は任意のクエリ集合に対して可能である．

2.2 サンプルを用いたクエリ検索機構

リポジトリを $R = \{q_1, \cdots, q_n\}$ とする．サンプル Sはリテラルの集合である．リテラルは要素 $e \in \cup_{i=1}^{n} g_i$に対して正例または負例のいずれかを明示したものである．サンプル S中のすべての正例 e に対し $e \in g_i$，かつ S中のすべての負例 $\bar{e}$に対し$\bar{e} \notin g_i$ であるとき， Sは q_iを検索すると定義する．

e が属性のとき， q_iの射影属性の集合に e が含まれるならば $e \in g_i$である．e が組のとき， q_iを検索用 DB 上で実行し，検索された組の集合に e が含まれるならば $e \in g_i$である．e が組のペアのとき，q_i を検索用 DB 上で実行した結果に e が示す順序で e を構成する 2 つの組が含まれるならば $e \in g_i$ である．

リポジトリ Rについて，$Set(R, S) = \{q \in R | S$は qを検索する $\}$ を定義する． $Set(R, S)$ は Sによる Rの絞り込みの結果である．サンプル Sと $Set(R, S)$ には以下の性質が成り立つ．

性質 1 リポジトリ Rと 2 つのサンプル S_1, S_2について，以下の式が成り立つ．

$$Set(R, S_1 \cup S_2) = Set(R, S_1) \cap Set(R, S_2 - S_1)$$ □

性質 2 リポジトリ Rと 2 つのサンプル S_1, S_2について，以下の式が成り立つ．

$$Set(R, S_1 \cup S_2) = Set(Set(R, S_1), S_2 - S_1)$$ □

性質 1より，絞り込み検索のためにはサンプルにリテラルを追加すればよいことが分かる．性質 2より，サンプルにリテラルを追加した後のクエリ検索は，前段階で検索されたクエリ集合に対してのみ行えばよい．

以上の議論よりサンプルを用いたクエリ検索は以下の手順で行なう．

1. サンプルを空にする
2. サンプルにリテラルを追加する．
3. サンプルを用いたクエリ検索
4. 所望の結果が得られるまで，ステップ (2), (3) を繰り返す．

以下にサンプルを用いたクエリ検索の例を示す．

例 2 図 1(A) に示す検索用 DB とクエリ q_1，図 2に示すクエリ $q_2, \cdots, q_5$を考える．リポジトリを $R = \{q_1, \cdots, q_5\}$ とする．まず，属性を追加したサンプル $S_1 =$ { 点 } を用いて検索すると，$Set(R, S_1) = \{q_1, q_2, q_4\}$ となり射影属性に点を含むクエリが検索される． S_1に組 ⟨ 森, 英語, 43⟩ を追加したサンプル S_2を用いてクエリを検索すると，$Set(R, S_2) = \{q_1, q_2\}$ となる．さらに 組のペア [⟨ 鈴木, 英会話, 45⟩, ⟨ 森, 英会話, 49⟩] を負例として追加したサンプル S_3を用いて検索すると $Set(R, S_3) = \{q_1\}$ となる． □

3 要素辞書

サンプルを作成する際には，所望のクエリと他のクエリを区別するリテラルを追加する必要がある．このような要素を利用者が直接指定するのは困難である．本節では，リポジトリに格納されているクエリ間を区別する要素の集合を要素辞書 [1] として定義する．要素辞書を用いると，所望のクエリを唯一に検索するサンプルが常に作成できる．

講義

科目	教官	時間数
英語	山田	16
DB	山下	16
SW	田中	15
英会話	山田	10

成績

学生	科目	点	欠課数
森	英語	43	0
斎藤	英語	90	0
鈴木	英語	65	2
森	DB	76	5
斎藤	DB	78	3
森	英会話	49	0
鈴木	英会話	45	2

(A)

q_1

```
SELECT   学生, 科目, 点
FROM     成績, 講義
WHERE    (講義. 科目 = 成績. 科目)
         AND (教官 = '山田')
         AND (点 < 50)
```

(B)

学生	科目	点
森	英語	43
森	英会話	49
鈴木	英会話	45

(C)

図 1: サンプルスキーマ

q_2 : 山田教官が担当する教科の得点が 50 点以下の学生を得点順に検索する.

学生	科目	点
森	英語	43
鈴木	英会話	45
森	英会話	49

q_3 : ある科目の欠課率が 10% 以上の学生を検索する.

学生	科目	欠課率
鈴木	英語	12.5
森	DB	31.3
斎藤	DB	18.8
鈴木	英会話	20.0

q_4 : 英語の成績が平均点以下の学生を検索する.

学生	点
森	43
鈴木	65

q_5 : 得点が 50 点以下の学生が存在する科目を検索する.

科目	教官
英語	山田
数学	田中
英会話	山田

図 2: クエリの例

定義 1 リポジトリを $R = \{q_1, \cdots, q_n\}$ とする. R の要素辞書 D_R は $\cup_{i=1}^{n} g_i$の部分集合であり, 任意のクエリ $q_i, q_j (q_i \neq q_j)$ に対して $g_i \cap D_R \neq g_j \cap D_R$を満足する. □

図 1, 図 2に示すクエリ $q_1, \cdots, q_5$が格納されているリポジトリ Rの要素辞書は, { 点, 教官, ⟨ 森, 英語, 43⟩, [⟨ 鈴木, 英会話, 45⟩, ⟨ 森, 英会話, 49⟩] } である.

要素辞書 D_R中でクエリ $q_i \in R$の仕様を表現する集合 $g_i \cap D_G$は互いに異なっているため, D_Rの要素をリテラルとして用いれば Rに属する任意のクエリを特定するサンプルが作成できる.

R中のクエリ q_i, q_jについて, $diff(q_i, q_j) = (g_i - g_j) \cup (g_j - g_i)$ を定義すると以下の補題が成り立つ.

補題 1 R中の 2 つのクエリ $q_i, q_j (q_i \neq q_j)$ について, $D_R \cap diff(q_i, q_j) \neq \phi$ が成立する. □

検索が進むにしたがって検索されるクエリは類似してくるため, それらの区別は容易でない. クエリの再利用のためにはクエリ間の違いを明確にし, 最適なクエリを特定することが重要である. このような場合には, $diff(q_i, q_j) \cap D_R$ に含まれる要素を検索すればよい.

Rの要素辞書 D_Rは一般に複数存在する. D_Rの要素数が少ないほどサンプル作成は容易になる. n 個のクエリを格納するリポジトリ Rに対して, 要素数が高々$n-1$の要素辞書が存在することが証明されている. 我々は, リポジトリ Rに対して, 要素数が $n-1$ 以下の要素辞書を作成するための要素辞書構成アルゴリズムを提案した. このアルゴリズムでは, リポジトリへのクエリの追加削除にともなって, 既存要素辞書を再構成する.

4 サンプル作成支援機構

本節では, サンプル作成のための手順を提案する. このための利用者支援機構として, フィルタリング, 検索, 重複度別分類を提案する. これらを組み合わせると効率的にサンプルが作成できる.

4.1 サンプル作成手順

利用者は要素辞書を用いてサンプルを作成する. サンプルの作成は, リテラルを追加することで行なう. サンプル Sに追加すると $Set(R, S)$ 中のクエリ数が減少する要素を, Sに追加して意義のある要素と呼ぶ. Sに追加して意義のある要素 e は, 以下の条件を満足する.

$$\exists q_i, q_j \in Set(R, S), e \in diff(q_i, q_j) \cap D_R$$

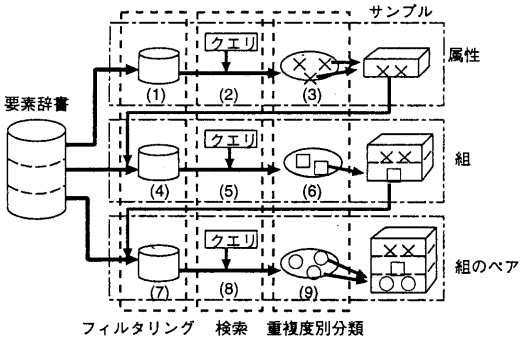

図 3: サンプル作成手順

ここで，Rはリポジトリ，D_RはRの要素辞書である．Sに追加して意義のある属性の集合を$E_S(A)$，組の集合を$E_S(T)$，組のペアの集合を$E_S(P)$と記述する．

効率的にサンプルを作成するためには，追加して意義のある要素をサンプルに追加することが重要である．しかし，リポジトリの大規模化にともない要素辞書も大きくなる．そこで，追加して意義のある要素を要素辞書から選択するための利用者支援として，図 3に示す手順を提案する．

サンプルは，リテラルを属性，組，組のペアの順で追加することで作成する．属性の追加によって，クエリが検索する組のスキーマによる検索ができる．組の追加は，検索条件に基づいたクエリの絞り込みに対応する．組のペアはORDER BY 句で指定された組の順序を明示するために追加する．

サンプル作成手順の各ステップは，フィルタリング，検索，重複度別分類の 3 つに分類できる．(1)，(4)，(7) では要素辞書のフィルタリングを行なう．(1) では，射影属性のみを抽出する．(4)，(7) では，既存のサンプルSについて，$Set(R,S)$中のいずれかのクエリの実行結果である組，表示順序を表現する組のペアを抽出する．(2)，(5)，(8) はフィルタリングの結果に対して，利用者が作成したクエリを用いて要素を検索する．(3)，(6)，(9) では，重複度別分類を用いて，検索された要素の中からSに追加して意義のある要素のみを抽出し，絞り込み効果が最も高い要素の選択を支援する．

図 3は標準的な手順を示しているが，利用者は必要に応じて各手順をスキップしたり，繰り返し実行できる．

	条件
属性	eが属性である．
組	eが組である，かつS中の射影属性について，すべての正例の属性値がeに含まれ，いずれの負例もeに含まれない．
組のペア	eが組のペアである，かつeに含まれる 2 つの組がいずれもSにおいて負例でない．

表 1: フィルタリングの抽出条件

4.2 フィルタリング

フィルタリングは，これまでに作成したサンプルSを用いて検索の候補となる要素の集合$E'_S(X)(X = A,T,P)$を要素辞書から抽出する．フィルタリングはシステムによって自動的に実行される．各ステップにおけるフィルタリング条件を表 1に示す．

各ステップにおいて検索された要素の集合$E'_S(X)$が$E'_S(X) \supseteq E_S(X)$を満足することを証明する．$X = A$の場合は自明である．$X = T$の場合を考える．$e \in E_S(T)$より，eは組である．また，$e \in diff(q_i,q_i)$を満足する$q_i,q_j \in Set(R,S)$が存在する．このとき一般性を失うことなく$e \in g_i$である．g_iはSに含まれる射影属性について，すべての正例を含み，いずれの負例も含まない．ゆえに，q_iによって検索される組eは，Sに含まれる属性について，すべての正例の属性値を含み，いずれの負例の属性値も含まない．したがって，$e \in E'_S(T)$が成立する．$X = P$の場合もこれと同様に証明できる．

なお，要素$e \in E'_S(X)$がすべてSに追加して意義のある要素とは限らない．$E'_S(X)$には，任意のクエリ$q_i \in Set(R,S)$について，$e \in g_i$を満足するeが含まれることがある．このようなeをSに追加しても$Set(R,S)$中のクエリ数は減少しない．

4.3 検索

本手法は，フィルタリングで求めた要素の中から，利用者が作成するクエリを用いてサンプルに追加するリテラルを検索する．作成するクエリを要素検索クエリと呼ぶ．要素検索クエリは通常の RDB クエリとは異なり，組だけでなく，属性や組のペアも検索対象とする．また，RDB クエリが検索する組のスキーマは同一であるが，要素検索クエリは，スキーマが異なる組を検索する場合もある．

図 3(1)，(4)，(7) ではそれぞれ属性，組，組のペアを検索する．各ステップで作成する要素検索クエリで指定する条件を表 2に示す．

利用者が作成した要素検索クエリが$Set(R,G)$中のすべてのクエリq_iについて，$e \in g_i$を満足する要素eを検索す

	条件
属性	eの値が指定された属性aを用いて計算されている． eの値が指定された関数fを用いて計算されている． eが指定された関係rに含まれている．
組	eが指定された式expを満足する．expはSQLのWHERE句に記述できる式である．
組のペア	eが指定された組tを含む．tは既存サンプルにおいて正例である．

表 2: クエリで指定する条件

要素	重複度
点	3
教官	1
⟨森, 英語, 43⟩	2
{⟨鈴木, 英会話, 45⟩, ⟨森, 英会話, 49⟩}	1

表 3: 要素の重複度

る場合がある．よって，本方法で求められた要素がすべてサンプルに追加して意義のある要素とは限らない．サンプルに追加して意義のある要素を求めるために，重複度別分類を用いる．

4.4 重複度別分類

重複度別分類は，要素辞書中の要素をそれをサンプルに追加したときの絞り込みの効果によって分類する．重複度別分類を用いると，サンプルに追加し定義のある要素の中から絞り込み効果が最も高い要素を選択できる．これより，要素の重複度を定義する．

定義 2 クエリの集合を$R' = \{q_1, \cdots, q_n\}$とする．要素$e$について$R$に対する$e$の重複度$W = |\{q_i \in R' | e \in q_i\}|$を定義する． □

例としてリポジトリ$R = \{q_1, \cdots, q_5\}$の要素辞書の各要素の重複度を表3に示す．

重複度がWである要素eを正例としてサンプルに追加すると，検索後のクエリ数はWに減少する．一方eを負例としてサンプルに追加すると，検索後のクエリ数は$n - W$に減少する．

システムは，検索された要素について重複度を計算し，重複度順にソートした要素のリストを利用者に提示する．重複度が0とnの要素はリストに含めない．提示された要素をサンプルに追加すると検索後のクエリ数が減少する．よって，リストにはサンプルに追加する意義のある要素のみが含まれる．

利用者は提示されたリストの中から要素を選択する．絞り込み効果が高い要素を選択するためには，正例を追加する場合は重複度が小さいものを選択し，負例を追加する場合は重複度が大きいものを選択すればよい．

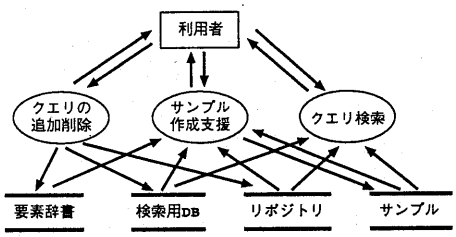

図 4: システム構成図

重複度の計算時間は，重複度別分類の対象となる要素数と既存のサンプルが検索するクエリ数に比例する．このため，本手法はフィルタリングや検索によって，要素数を絞り込んだ後に使用する．ただし，属性を用いて初期サンプルを作成する際に重複度計算コストの問題は発生しない．属性の重複度の計算にはクエリ実行の必要がないためである．

5 クエリ検索システム

クエリ検索システムの構成 (DFD) を図 4に示す．本システムは，クエリの追加削除，サンプル作成支援，クエリ検索から構成される．利用者は，クエリの追加削除を用いてリポジトリを更新する．これと同時に要素辞書も再構成される．サンプルを作成する際には，サンプル作成支援を用いる．サンプル作成支援は，属性リテラル追加，組リテラル追加，組のペアリテラル追加に詳細化されている．さらにそれぞれについて，フィルタリング，検索，重複度別分類に詳細化される．クエリ検索では，作成したサンプルを用いてクエリの検索を行なう．

クエリ検索システムのユーザインタフェースを図 5に示す．メインウィンドウでサンプル作成ボタンを押すとサンプル表示とサンプル作成支援が開く．サンプル表示は現在のサンプルを表示する．利用者はサンプル作成支援で，サンプルに追加するリテラルの検索方法を選択する．図 5には属性選択のステップを示す．属性/フィルタリングボタンを押すと属性のフィルタリングに結果が表示される．また，フィルタリングにより求められた属性を含む関係や値の計算に用いられる関数も表示される．これらは要素検索クエリを作成する際に参考になる．属性 /検索ボタンを押すと属性の検索が開く．利用者はそこで要素検索クエリを作成する．属性の検索ボタンまたはサンプル作成支援の属性 /重複度別分類ボタンを押すと属性の重複度別分類が開き，要素検索クエリによって検索された要素の中でサンプルに追加して意義のある要素のリストが表示される．正負

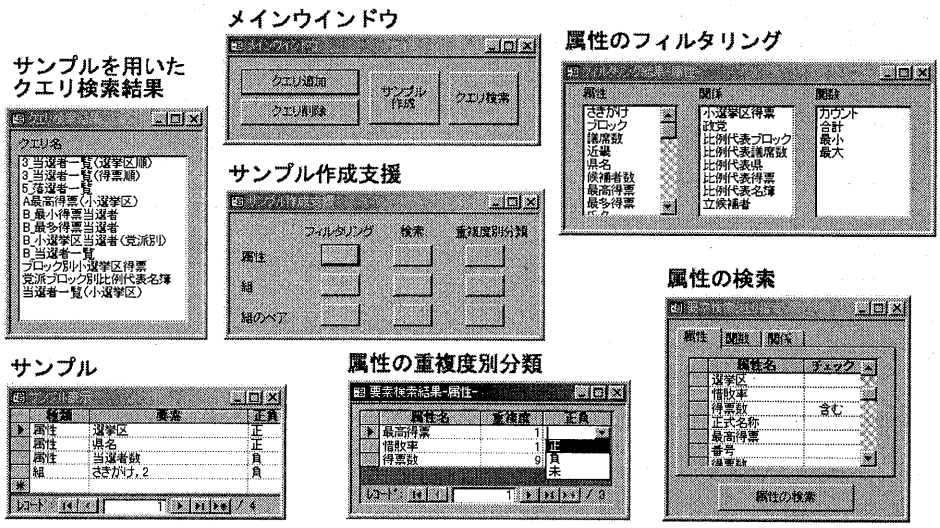

図 5: ユーザインタフェース

フィールドで正または負を選択すると要素は正のリテラルまたは負のリテラルとしてサンプルに追加される．未を選択すると既存サンプルより要素が削除される．クエリの検索は，メインウィンドウでクエリ検索ボタンを押すと起動される．サンプルを用いたクエリ検索結果が開き，検索されたクエリの一覧が表示される．

6 おわりに

本稿では，RDB クエリを仕様に基づいて検索するためのシステムを提案した．本システムを用いることにより，所望のクエリを唯一に検索できる．射影属性はクエリのシグネチャであることから，属性から構成されるサンプルを用いた検索は，シグネチャマッチングに対応する．

リポジトリ Rへクエリ q_iを追加する際に，検索用 DB において $g_i = g_j$ となるクエリ q_jが存在する場合がある．q_i と q_jを区別する要素を要素辞書に登録するためには，検索用 DB を更新する必要がある．ところが検索用 DB の状態が変化するとクエリの実行結果も変化するため，更新後の検索用 DB において要素辞書が定義を満足することが保証されない．これを解決するためには検索用 DB のバージョン管理が必要になる．

本システムにおいては，検索クエリのみを対象としていた．RDB クエリには，パラメータクエリ，追加クエリ，削除クエリ，更新クエリなどが存在する．今後はこれらのクエリについても適用できるようにシステムを拡張する予定である．

参考文献

[1] 村田, 掛下. 集合間の相違を明確にする要素辞書. 情報処理学会論文誌データベース, Vol.40, No. SIG3 (TOD 1), pp. 60–67, 1999.

[2] M Rittri. Using types as search keys in fuction libraries. In *Proc. Conf. on Functional Programming Languages and Computer Architectures, Reading, Mass., Addison-Wesley*, 1989.

[3] A. M. Zaremski and J. M. Wing. Signature matching: A tool for using software libraries. *ACM Trans. on Software Engineering and Methodology*, Vol. 4, No. 2, pp. 146–170, 1995.

[4] B. Fischer, M. Kievernagel, and G. Snelting. Deduction-based software component retrieval. In *IJCAI Workshop on Reuse of Proofs, Plans and Programs, Montreal, Canada*, June 1995.

[5] R. Mili, A. Mili, and R. T. Mittermeir. Storing and retrieving software components: A refinement based system. *IEEE Trans. on Software Engineering*, Vol. 23, No. 7, pp. 445–460, 1997.

[6] A. Podgurski and L. Pierce. Retrieving reusable software by sampling behavior. *ACM Trans. on Software Engineering and Methodology*, Vol. 2, No. 3, pp. 286–303, 1993.