시소러스를 이용한 재사용 컴포넌트 검색 시스템

김 귀 정
건양대학교

Reusable Component Retrieval System using Thesaurus

Kim gui-jung
Division of IT Konyang Univ.
E-mail : gjkim@konyang.ac.kr

요 약

본 논문은 컴포넌트의 재사용을 위한 컴포넌트 검색 시스템을 구축하였다. 컴포넌트 검색을 위해서는 클래스의 상속관계를 이용한 시소러스로 구축하였고, 이를 통하여 질의를 이용한 컴포넌트 검색이 가능하도록 하였다. 또한 검색결과는 우선순위로 보여줌으로서 질의에 대한 보다 빠른 검색이 되도록 하였다. 검색된 컴포넌트는 원시코드, 컴포넌트 정보, 클래스 다이어그램 등을 제공함으로서 효율적인 컴포넌트 재사용이 가능하도록 하였다.

Abstract

This paper constructed component retrieval system for reusability of component. Constructed by thesaurus that use inheritance relation of class for component retrieval, and did so that component retrieval that use queries may be available. Also, the retrieval result did to become faster retrieval about queries as that show by priority. Retrieved components made efficient component reusability to be possible as that support source code, component information, class diagram etc.

I. 서론

본 논문에서는 객체지향 컴포넌트의 효율적인 검색을 위하여 클래스의 상속관계를 이용하여 개념을 분류하였고, 퍼지 논리를 적용한 객체지향 시소러스 검색시스템을 구축하였다. 시소러스 구축은 클래스의 상속관계를 이용하여 개념들 사이의 관계를 의미를 갖는 구조로 확장하였다.

따라서 본 연구는 기존의 정보 검색 시스템에서 사용한 시소러스 방법과는 다른 객체지향 컴포넌트를 잘 표현할 수 있는 시소러스를 구축하는데 그 목적을 두었다. 본 시스템은 컴포넌트 검색을 위한 인터페이스와 질의 처리기를 구현하였다. 또한 사용자를 위하여 시소러스 정보를 제공하고 검색된 컴포넌트의 재사용을 위하여 소스 코드와 클래스 다이어그램을 제공할 수 있도록 하였다. 그리고 검색된 컴포넌트는 유사도 순으로 검색되기 때문에 원하는 컴포넌트를 효율적으로 찾을 수 있는 재사용성이 높은 컴포넌트 검색시스템이다.

II. 기존의 시소러스 기반 검색 시스템

ACM의 CRCS(Computing Reviews Class-ification Structure)[1]는 5단계 계층구조의 트리 형태를 가지고 1000여 개의 키워드로 구성되어 있으며 키워드 사이의 관계는 'IS-A' 관계를 사용하고 있다. CRCS의 단점으로는 시소러스의 지식을 잘 활용하여 이용자가 기술하는 개념을 보다 잘

적용시킬 수 있는 그룹화 방법이 필요하며, 컴포넌트의 변화에 따라 전체적으로 시스템 구축방법을 재구성해야 하므로 확장이 어려운 단점이 있다. FIRMS(Fuzzy Information Retrieval and Management System)[2]은 퍼지 객체를 처리하기 위해서 개발된 시스템으로 퍼지 속성을 가진 객체의 표현방법을 제안하고 객체를 검색하는 데 그 목적이 있다. FIRMS는 속성을 기본 속성, 연산 속성 그리고 집합 속성으로 구분하여 객체의 속성값을 퍼지 정도로 표현하였다. 그러나 이 시스템은 유사 용어의 관계를 연결하기 위하여 용어들의 개념적인 정의를 해야 하는 어려움이 있을 뿐만 아니라 전문가가 용어의 속성과 가중치를 분류해야하는 단점이 있다. 계층적 시소러스 시스템[3]은 객체지향 코드에서 계층적 분류가 이루어지도록 범주를 설정하고, 소프트웨어 컴포넌트가 행위적 특성에 따라 분류되는 방법을 제안하였다. 특성에 따라 계층적으로 분류된 용어의 쌍은 퍼지 시소러스를 이용하여 유의어 사전을 구축하게 된다. 그러나 이 시스템은 클래스가 증가할수록 멤버함수가 기하급수적으로 증가하기 때문에 중복된 멤버함수와 파라메터로 인하여 노이즈가 많다는 단점이 있다.

Ⅲ. 객체지향 시소러스 구축

1. 컴포넌트 구조와 클래스 가중치

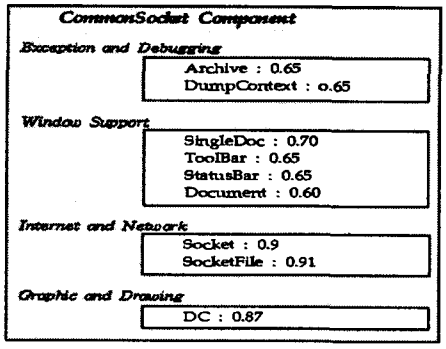

그림1. CommonSocket 컴포넌트

본 연구에서는 패싯분류의 개념을 사용하여 컴포넌트를 하나 이상의 범주(CCC)로 분류하였다. CCC는 컴포넌트와 도큐멘트 정보로부터 구축되며, 구성단위는 컴포넌트의 기능을 포괄적이고 모듈별로 제공해주는 클래스가 된다. 각 클래스는 컴포넌트와의 가중치를 가지고 있다. 컴포넌트의 구조는 다음 그림 1과 같다.

클래스의 가중치는 컴포넌트와 클래스간의 관련 정도를 나타내는 척도이다. 한 컴포넌트의 행위와 더 관련이 깊고 유일하게 나타나는 클래스 일수록 가중치는 높은 값을 가지게 된다. i번째 컴포넌트의 k번째 클래스 가중치는 다음과 같은 식에 의해서 계산된다.

$$w_{i,k} = \frac{v_{i,k}\log(\frac{N}{n_k})}{\sqrt{\sum_{z=1}^{F}(v_{i,z}\log(\frac{N}{n_z}))^2}}$$

$w_{i,k}$: i 번째 컴포넌트에 대한 k 번째 클래스 가중치
$v_{i,k}$: i 번째 컴포넌트에 나타나는 k 번째 클래스 빈도
N : 전체 컴포넌트의 수
n_k : k 번째 특징이 나타난 컴포넌트의 수

2. 시소러스 구축

각 클래스와 CCC에 대한 매칭도와 비매칭도를 비교함으로써 이들 사이의 퍼지 정도를 계산하여 시소러스를 구축한다. CCC와 클래스 사의의 관련성과 클래스 상속에 따른 CCC의 상속에 초점을 두어 시소러스를 구축하였다. 과정은 다음과 같다.

① *CCC* 발생횟수 계산

클래스가 직접적으로 포함된 CCC뿐 아니라 상속받은 CCC도 해당 클래스와 관련이 있으므로 직접 포함된 CCC와 상속받은 CCC 모두를 표현할 수 있어야 한다. 발생횟수 '1'은 클래스가 해당 CCC에 직접적으로 포함된 경우이며, '1/2'은 상위 클래스가 포함된 CCC를 상속받은 것이며, '1/4'은 두 단계 상속받은 CCC를 나타낸다. 이 값들은 CRV를 계산할 때, 클래스 발생횟수로써 사용된다.

② *j*번째 CCC와 *l*번째 클래스에 대한 $CRV_{l,j}$ 계산

CCC 관련값(CCC Relevance Value:CRV)으로 이루어진 테이블을 생성한다. CRV 계산의 기본적인 가정은 다음과 같다. 「한 컴포넌트를 구성하는 CCC에 대하여, *i*번째 CCC에 속한 클래스의 수는 컴포넌트와 *i*번째 CCC와의 관련성을 암시해 준다[3].」

$$CRV_{i,j} = \frac{CCC\ j\text{에서 } i\ class\text{의 발생횟수}}{CCC\ j\text{에서 모든 } class\text{의발생횟수}} \times \frac{CCC\ j\text{에서 } i\ class\text{의 발생횟수}}{\text{모든}CCC\text{에서 } i\ class\text{의전체발생횟수}}$$

③ 클래스와 클래스사이의 매칭도, 비매칭도 계산

위에서 계산된 각 클래스에 대한 CRV를 이용하여 클래스와 클래스 사이의 매칭도, 비매칭도를 계산하는 과정이다.

$$j\text{ 번째 }CCC\text{에 대한 매칭도}\begin{cases} m_j=\frac{1}{1+|a_j-b_j|} & ,a_j>0 \text{ and } b_j>0 \\ 0 & ,a_j=0 \text{ or } b_j=0 \\ & \ a_j=0 \text{ and } b_j=0 \end{cases}$$

클래스 A,B의 매칭도 : $M=\sum_{j=1}^{C} m_j$

$$j\text{ 번째 }CCC\text{에 대한 비매칭도}\begin{cases} m_j^m=\frac{|a_j-b_j|}{1+|a_j-b_j|} & ,either\ a_j=0 or b_j=0 \\ 0 & ,a_j=0 \text{ and } b_j=0 \\ & \ a_j\neq 0 \text{and } b_j\neq 0 \end{cases}$$

클래스 A,B의 비매칭도 : $M^m=\sum_{j=1}^{C} m_j^m$

④ 두클래스와 CCC사이의 매칭도, 비매칭도 계산

한 CCC에 대해 두 클래스가 동시에 나타나거나, 나타나지 않거나 하는 정도를 퍼지 집합을 사용하여 계산하는 방법이다. 두 클래스에 대한 CCC 사이의 매칭도(M'), 비매칭도($M^{m'}$) 계산은 클래스 사이의 매칭도, 비매칭도 계산과 유사하다.

⑤ 두 클래스의 퍼지 유의값 계산

클래스와 클래스 사이에서 계산된 매칭도, 비매칭도, 그리고 두 클래스와 CCC 사이의 매칭도, 비매칭도를 이용하여 두 클래스의 최종적인 퍼지 유의값을 생성할 수 있다.

$$f=\max(0,\frac{1}{2}\times(\frac{M-M^m}{M+M^m}+\frac{M'-M^{m'}}{M'+M^{m'}})) \quad 0\le f\le 1$$

⑥ 모든 클래스에 대한 퍼지 유의어 테이블 생성

①에서부터 ⑥까지의 과정을 시소러스 내에 있는 모든 클래스에 대해 수행함으로써 모든 클래스들 간의 유의어 값을 생성한다.

Ⅳ. 검색 시스템

1. 컴포넌트 검색

컴포넌트를 우선순위로 검색하기 위하여 질의와 컴포넌트들의 유사도를 계산한다[4]. 유사도 계산 과정을 예를 들어 알아본다.

Query
ToolBar : 0.7 AND SocketFile : 0.9
Gopher Component
CommandLine : 0.95, DumpContext: 0.81,
ToolBar : 0.50, InternetFile : 0.90,
HttpFile : 0.88, String: 0.79, StatusBar : 0.80

그림 2. "Gopher" 컴포넌트의 유사도 계산

① 질의와 컴포넌트 간의 동치관계 계산

다음은 동치관계(Equivalence)를 계산하기 위한 식이다. 이 식은 질의에 나타난 클래스와 컴포넌트에 있는 각 클래스 간의 유의값을 반환한다.

$$Eq(Query(u),Comp(v))=SYNON(Query(u),Comp(v))$$

u : 질의에 있는 질의어 갯수
v : 컴포넌트에 있는 클래스 갯수

표 1. 질의에 대한 "Gopher"컴포넌트의 동치관계

	Command -Line	Dump -Context	ToolBar	Internet -File	Http -File	String	Status -Bar
ToolBar	0.769	0.894	1.000	0.654	0.580	0.753	0.912
SocketFile	0.752	0.874	0.942	0.901	0.930	0.872	0.890

② 질의와 컴포넌트 간의 함축관계 계산

함축관계식에 의해서, 질의에 설정한 질의 중요도가 함축관계에 의해 계산되어진 값보다 작거나 같을 경우에 질의와 컴포넌트의 각 클래스에 대한 교환이 가능함을 의미한다.

$$Imp(Query(u),Comp(v))=\max\{w(Query(u)),w(Comp(v))\}\ [Eq(u,v)]$$

표 2. 질의에 대한 "Gopher" 컴포넌트의 함축관계

	Command -Line	Dump -Context	ToolBar	Internet -File	Http -File	String	Status -Bar
ToolBar	0.731	0.724	0.700	0.589	0.510	0.595	0.730
SocketFile	0.714	0.787	0.848	0.811	0.837	0.785	0.801

③ 질의와 컴포넌트 클래스의 만족도 계산

만족도(satisfaction value)는 질의어와 컴포넌트의 각 클래스가 얼마나 호환성이 있는가를 계산하는 과정이다.

$$Sat(Query,Comp(v))=\frac{[\sum_{u=1}^{U} Imp(u,v)\times Eq(u,v)]}{U}$$

$$Sat(Query,Comp(v))=\{0.550,0.668,0.725,0.558,0.537,0.567,0.690\}$$

④ 질의와 컴포넌트 간의 유사도 계산

③에서 만들어진 만족집합에 컴포넌트의 가중치 벡터를 적용하고 모든 요소를 더함으로써 최종적인 질의와 컴포넌트 간의 유사도를 계산한다.

$$Sim(Query,Comp)=\sum(Sat\times W)=$$
$$\sum(\{0.550,0.668,0.725,0.558,0.537,0.567,0.690\}\times$$
$$\{0.169,0.144,0.089,0.160,0.156,0.140,0.142\})=0.604$$

질의에 대해 'Gopher' 컴포넌트는 0.604의 유사도를 가지고 있음을 알 수 있다. 위와 같은 방법으로 검색된 모든 후보 컴포넌트에 대해서 유사도를 계산한 후 가장 높은 값을 가진 컴포넌트 순서로 출력하도록 하였다.

2. 검색 시스템

본 시스템은 컴포넌트 검색에 대한 사용자의 질

의 형성을 도와주기 위해 그림 2와 같은 클래스 선택 윈도우를 제공한다. 자연어 형식의 검색과 클래스 기능이나 특징에 해당하는 클래스 개념 범주를 선택함으로써 이에 해당하는 클래스 리스트를 제공하며, 클래스 시소러스 정보를 참고함으로써 보다 효과적인 질의 형성이 가능하도록 하였다. 컴포넌트 검색은 질의 입력부분, 검색 결과 리스트 출력 부분, 그리고 컴포넌트 정보 부분으로 세분화된다. 그림 3은 질의로 "ToolBar"와 "SocketFile"을 선택하고 각각의 질의 중요도를 0.7, 0.9로 주었을 때의 검색 결과를 보여준다. 검색 결과 리스트 출력 부분에는 시소러스에 의해 검색된 후보 컴포넌트들이 유사도 순으로 나타나 있고, 선택한 컴포넌트에 대한 자세한 정보가 컴포넌트 정보 부분에 제공되어 진다. 또한 각 컴포넌트를 재사용하는데 있어서 필요한 정보를 얻을 수 있도록 그림 4와 같이 클래스 관계 윈도우, 클래스 다이어그램 윈도우, 그리고 소스코드 윈도우를 제공한다. 시소러스 관리 윈도우는 그림 5에서처럼 시소러스를 갱신하거나 데이터베이스에 접근하고자 할 때 사용된다.

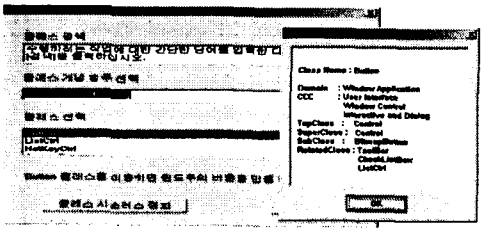

그림 2. 클래스 선택

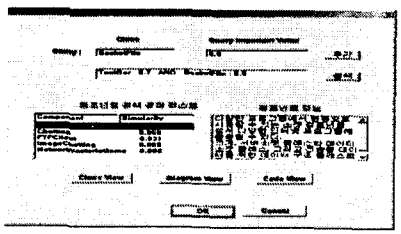

그림 3. 컴포넌트 검색

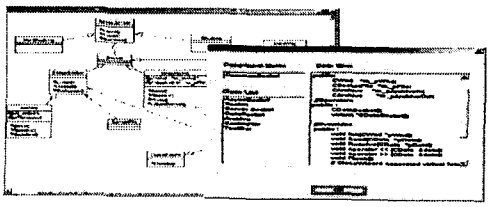

그림 11. 재사용 정보

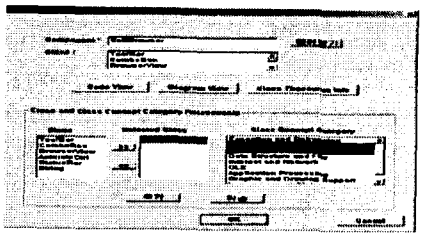

그림 5. 시소러스 관리

V. 결론

본 연구는 컴포넌트의 재사용을 위한 컴포넌트 검색 시스템을 구축하였다. 컴포넌트 검색을 위한 기술은 시소러스를 이용하여 구축하였고, 컴포넌트 검색 인터페이스와 질의 처리기를 구현하였다. 또한 시소러스 정보를 제공하고 검색된 컴포넌트의 재사용을 위하여 소스 코드와 클래스 다이어그램, 시소러스 관리를 지원할 수 있도록 하였다. 그리고 컴포넌트는 유사도 순으로 검색되기 때문에 원하는 컴포넌트를 효율적으로 찾을 수 있다.

참 고 문 헌

[1] ACM, "The Full Computing Reviews Classification System," ACM, New York, 1992.

[2] P. Subtil, N. Mouaddib and O. Foucaut, "A Fuzzy Information Retrieval and Management System and Its Applications," The Proceeding of the ACM Symposium on Applied Computing, pp.537-541. Feb. 1996.

[3] E. Damiani, M. G. Fugini and C. Bellettini, "Aware Approach to Faceted Classification of Object-Oriented Component," ACM Trans. on Software Eng. and Methodology, Vol.8, No.4, pp.425-472. Oct. 1999.

[4] M. Moormann Zaremski and J. M. Wing, "Signature matching : A tool using software librarys," ACM Transaction on Software Engineering and Methodology, Vol.4, No.2, pp.146-170, Apr. 1995.