특성 기반 객체지향 시소러스 구축

정대성* 한정수* 김귀정*
*천안대학교 정보통신학부
*건양대학교 IT학부
e-mail:f15cc@dreamwiz.com jshan@cheonan.ac.kr
gjkim@konyang.ac.kr

Feature Based Object-Oriented Thesaurus Construction

Dae-Sung Jung*, Jung-Soo Han*, Gui-Jung Kim*
*Division of Information & Comunnication, Cheon-An Univ.
*Division of Information Technology, KonYang University

요 약

본 연구는 컴포넌트 검색을 위해서 컴포넌트를 컨텍스트에 의해 패싯 분류하고, 컨텍스트와 특성들간의 관련값에 대한 통계적 분석에 의해 시소러스를 구축하여 다중 패싯 분류된 컴포넌트를 효율적으로 검색할 수 있는 방법을 제안하였다. 소스 코드로부터 추출된 특성은 카이제곱 방법을 통하여 간소화가 이루어지며, E-SARM 방법을 사용하여 컨텍스트의 자동 검색이 이루어질 수 있도록 하였다. 쿼리에 대해 자동 검색된 컨텍스트에 의해 후보 컴포넌트가 선정되고, 쿼리와 컴포넌트 간의 유사도가 계산됨으로써 컴포넌트가 검색될 수 있도록 하였다. 본 연구는 다중 패싯 분류된 컴포넌트의 검색에 효율적이며, 컴포넌트의 재사용성을 높일 수 있도록 하였다.

1. 서론

객체 지향 방법론의 영향으로 다양한 컴포넌트 기반의 개발 방법론이 제안되고 있고 이에 따라 많은 컴포넌트가 개발되고 있지만, 사용자가 원하는 컴포넌트 식별은 대부분 분석자의 경험에 의존하는 경우가 많다. 이러한 검색방법은 주관적이고, 비효율적이며, 시스템의 일관성을 유지하기도 어렵다. 이에 통합환경에서의 인프라를 제공하며 커스터마이징(Customizing)을 위한 정확하고 자동화된 컴포넌트의 검색 방법이 필요한데, 그 중 시소러스 개념이 많이 도입되고 있다. 그러나 기존의 시소러스 방법은 개념적 용어 정의가 어려운 점, 도메인 전문가의 노력이 너무 많이 요구되는 점, 용어 불일치 등의 문제점이 있다.

따라서 본 연구는 재사용이 가능하도록 하는 객체지향 컴포넌트의 검색을 위한 효과적인 시소러스 구축에 목적을 둔다. 컴포넌트는 컨텍스트에 의해 패싯 분류되고, 각 컨텍스트는 소스 코드로부터 추출된 특성(feature)으로 구성된다. 추출된 특성은 카이제곱 방법을 통하여 간소화가 이루어지며, 컨텍스트와 특성들간의 관련값에 대한 통계적 분석에 의해 시소러스가 구축된다. 또한 E-SARM 방법을 사용하여 컨텍스트의 자동 검색이 이루어질 수 있도록 하였으며, 쿼리에 대한 후보 컴포넌트와의 유사도를 측정하여 최적의 컴포넌트를 검색할 수 있도록 하였다.

2. 관련 연구

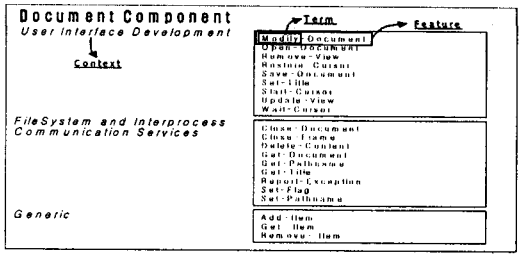
기존의 시소러스 구축 관련 연구에는 계층형 시소러스방법[1], 객체기반 시소러스 구축방법[2] 등이 있고, 유사도를 이용한 검색의 대표적인 방법에는 스프레딩 엑티베이션 방법[3]이 있다. 계층형 시소러스방법[1]은 코드에서 계층적 분류를 위한 범주를 설정하고, 컴포넌트가 행위적 특징에 따라 분류되는 방법을 제안하였다. 그러나 이 방법은 컴포넌트 검색이 아닌 멤버함수와 파라메타를 이용한 클래스 검색이기 때문에 클래스가 증가할수록 노이즈가 많아진다는 단점이 있다. 객체기반 시소러스 구축[2]은 시소러스에 개념 표현 레벨과 인스턴스 표현 레벨로 구성된 객체지향 패라다임을 적용함으로써 객체들 사이에 존재하는 복잡한 관계성들의 표현에 자동 구축 전략을 제공한다. 그러나 이 방법은 효과적인 질의 재형성 과정이 필요하다. 스프레딩 엑티베이션 방법[3]은 용어와 문서 사이의 관계를 이용한 유사 컴포넌트 검색방법으로서 문서의 증가에 따른 계산이 많아 검색에 걸

리는 시간이 늘어나는 단점이 있다. 따라서 본 연구는 컴포넌트를 구성하는 특성을 최적화하기 위해 카이제곱 검정방법[1]을 이용한 특성 간소화 방법을 제안하고, 특성과 컴포넌트 사이의 관련값을 이용한 시소러스를 구축하였다.

3. 시소러스 구축

3.1 특성추출

본 연구에서는 패싯분류의 개념을 사용하여 컴포넌트를 하나 이상의 컨텍스트(context)로 분류하였다[1]. 컨텍스트는 소스 코드로부터 추출된 특성(feature)으로 구성되는데, 특성은 메소드 이름에 해당하는 동사형태와 메소드의 첫 번째 인수에 해당하는 명사형태의 한 쌍으로 이루어져 있다. (그림 1)은 컴포넌트의 구조를 나타낸다.



(그림 1) 컴포넌트 구조

이 컨텍스트와 특성의 값은 카이제곱 통계량 계산에 이용되어 최적의 특성의 수를 구하는데 사용된다. 이는 모든 컨텍스트에 포함된 특성을 이용한 검색은 의미가 없기 때문에 사전에 검색에 불필요한 특성은 제거하여 특성을 간소화함으로써 검색의 효율을 높이고자 하는데 그 목적이 있다. 식(1)은 특성과 컨텍스트간의 컴포넌트 빈도를 이용한 카이제곱 통계량 식을 나타낸 것이다.

$$x^2(f,c)=\frac{N(AD-CB)^2}{(A+C)(B+D)(A+B)(C+D)} \qquad 식(1)$$

f : *feature*
c : *context*
A : f와 c가동시에발생한횟수
B : f는발생했지만 c는발생하지않은횟수
C : f는발생하지않고 c만발생한횟수
D : f와 c모두발생하지않은횟수
N : 전체*context*의수

위 식에 의해 각각의 특성은 특성값(feature value)을 갖게되어, 시뮬레이션을 통한 임계치를 설정하여 검색에 도움을 주지 못하는 특성을 제거하였다. 시뮬레이션 결과 전체 추출된 특성의 약 17% 정도가 제거되었다.

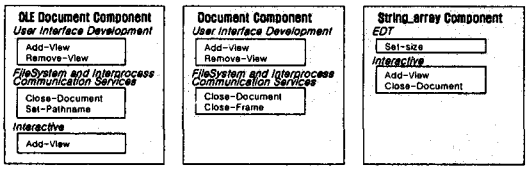
3.2 Feature weight

각 컴포넌트를 구성하는 특성들은 컴포넌트의 행동을 강조하는 역할을 하는데 이러한 컴포넌트와 특성들간의 연관정도를 값으로 표현한 것을 특성 가중치(Feature Weight)라 한다. 특성 가중치에 대한 수식은 식(2)에 나타나 있다.

$$FW_{i,k}=\frac{ff_{i,k}\times\log(\frac{N}{n_k})}{\sqrt{\sum_{z=1}^{F}(ff_{i,z}\times\log(\frac{N}{n_z}))^2}} \qquad 식(2)$$

$FW_{i,k}$: i번째 컴포넌트의 k번째 *feature*의 *weight*
N : 전체 컴포넌트의 수
$ff_{i,z}$: *feature frequency*
n_k : k번째 *feature*가 나타나는 컴포넌트의 수
F : *repository*에 있는 서로 다른 *feature* 수

(그림 2)는 컨텍스트로 분류된 3개의 컴포넌트 구조를 보여주고 있다. 컴포넌트 'OLEDocument'의 특성 'Set-Pathname'의 특성 가중치를 계산하면 다음과 같다.



(그림 2) 컴포넌트의 예

$$FW_{OLE,S-P}=\frac{1\times\log\frac{3}{1}}{\sqrt{(2\times\log\frac{3}{3})^2+(1\times\log\frac{3}{2})^2+(1\times\log\frac{3}{3})^2+(1\times\log\frac{3}{1})^2+0^2+0^2}}=0.941$$

3.3 시소러스 구축

시소러스 구축은 컨텍스트와 특성들간의 관련값에 대한 통계적 분석에 의해 이루어 진다. 먼저 특성과 컨텍스트간의 관련값(FCV)이 얻어진 후에, 이를 이용하여 컨텍스트와 컨텍스트의 관련값(CCV), 특성과 특성의 관련값(FFC)을 계산한다. 다음은 시소러스 구축 과정이다.

① 컨텍스트와 특성으로 이루어진 메트릭스를 구성하여 Feature-Context Value(FCV)를 계산한다. 특성-컨텍스트관계값에 대한 수식은 식(3)에 나타나 있다. 이는 각 컨텍스트에 속한 특성의 수는 컴포넌트와 컨텍스트와의 관련성을 암시해 준다는 의미에 근거한다.

$$FCV_{l,j}=p_j(l)\frac{feature_{l,j}}{feature_l} \qquad 식(3)$$

$FCV_{l,j}$: *Context* j와 *feature* l과의 관계값
$p_j(l)$: *Context* j에서의 l 번째 *feature*의 발생백분율
$feature_{l,j}$: *Context* j에서 l 번째 *feature*의 발생횟수
$feature_l$: 모든*Context*에서 l 번째 *feature*의전체발생횟수

(그림 2)의 컴포넌트에 대한 FCV 매트릭스는 식(3)에 따라 다음과 같이 구성된다.

Feature \ Context	Interface	Service	Interactive	ETD
Add-View	0.25	0	0.333	0
Remove-View	0.5	0	0	0
Close-Doc.	0	0.333	0.111	0
Close-frame	0	0.25	0	0
Set-size	0	0	0	1
Set-Pathname	0	0.25	0	0

<표 1> FCV matrix

② FCV matrix 를이용하여 컨텍스트와 컨텍스트 관련값

(CCV)를 계산한다. 이는 각 특성에 대한 컨텍스트의 매칭정도를 나타내며 계산식은 (4), (5)와 같다.

1)컨텍스트 C1과 C2 사이의 매칭정도를 알아보기 위해 먼저 a1과 a2 사이의 매칭정도 계산

$$ma_{12} = \frac{1}{1+|a1-a2|}, \quad (a1 \neq 0,\ a2 \neq 0)$$
$$ma_{12} = 0, \quad (a1 = 0,\ OR\ a2 = 0) \text{ 또는 } (a1 = a2 = 0) \qquad \text{식(4)}$$

C1과 C2의 모든 특성에 대해서 계산

$$M_{12} = \sum_{j=a}^{N} mj_{12} \qquad \text{식(5)}$$

2)모든 컨텍스트에 대해서 시행한다.

Context \ Context	Interface	Service	Interactive	ETD
Interface	6 (1.000)	0	0.923 (0.155)	0
Service		6 (1.000)	0.818 (0.136)	0
Interactive			6 (1.000)	0
ETD				6 (1.000)

<표 2> CCV matrix

③ 특성과 특성 관련값(FFV)을 계산한다. 이는 각 컨텍스트에 대한 특성의 매칭정도를 나타내는 것이며, 계산방법은 CCV와 유사하다.

1)특성 A와 B 사이의 매칭정도를 계산하기 위해 a1과 b1의 매칭정도 계산

$$ml_{ab} = \frac{1}{1+|a1-b1|}, \quad (a1 \neq 0,\ b1 \neq 0)$$
$$ml_{ab} = 0, \quad (a1 = 0,\ OR\ b1 = 0) \text{ 또는 } (a1 = b1 = 0) \qquad \text{식(6)}$$

A와 B의 모든 컨텍스트에 대해서 계산

$$M_{AB} = \sum_{j=1}^{C} mj_{ab} \qquad \text{식(7)}$$

2)모든 특성에 대해서 시행한다.

Feature \ Feature	Add-View	Remove-View	Close-Doc.	Close-frame	Set-size	Set-Pathname
Add-View	4 (1.000)	0.8 (0.2)	0.818 (0.205)	0	0	0
Remove-View		4 (1.000)	0	0	0	0
Close-Doc.			4 (1.000)	0.923 (0.231)	0	0.923 (0.231)
Close-frame				4 (1.000)	0	1 (0.250)
Set-size					4 (1.000)	0
Set-Pathname						4 (1.000)

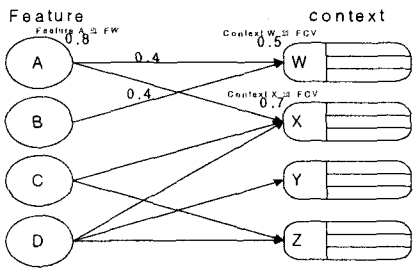
<표 3> FFV matrix

위와 같은 과정에 의해 최종적으로 특성과 특성 사이의 관련값, 즉 특성간 유의어 테이블이 구성되어 특성을 기본으로 한 시소러스가 구축된다.

4. 컴포넌트 검색

4.1 컨텍스트 자동검색

컨텍스트 자동검색에 의해 다중 패싯분류된 컴포넌트의 검색을 위해서는 쿼리가 표현하고자하는 컨텍스트가 어떤 것인가를 찾아야 하고, 또한 그 쿼리가 여러 개의 컨텍스트를 만족할 수 있음을 이해해야 한다. 이에 본 연구에서는 쿼리가 동사-명사의 특성 형태로 주워졌을 때, 이를 만족하는 컨텍스트를 찾기 위하여 스프레딩 엑티베이션방법을 이용하였다. 이를 위해 특성들과 컨텍스트들간의 초기 활성값을 설정하였는데, 각 특성의 초기 활성값은 컴포넌트와 특성들간의 가중치에 의해 결정된다. 스프레딩 엑티베이션 방법을 개선한 E-SARM(Enhanced Spreading Activation Retrieval Method)[4]은 순환과정이 일정 수준 반복된 후 기준에 미치지 못하는 컴포넌트들의 연결정보를 제거하여 연산에서 제외시킴으로써 검색의 확장범위를 줄여 보다 관계가 깊은 후보컴포넌트들만을 검색한다. 이를 컨텍스트의 자동검색에 적용하기 위하여 특성과 컨텍스트 사이에 부여된 연관성을 이용한다. 각 특징의 초기 활성값은 한 특성에 대해서 계산된 특성 가중치(FW)의 평균값이고, 컨텍스트의 초기 활성값은 특성과 컨텍스트 간의 연관성를 이용하여 계산된 특성-컨텍스트관계값(FCV)이다. 이를 바탕으로 동사-명사의 특성 형태로 주어진 쿼리에 대해서 컨텍스트를 검색하는 과정이 (그림 3)에 나타나 있다. 특성 A의 초기 활성값은 0.8이고, 컨텍스트 W의 초기 활성값은 0.5이다. 쿼리로 특성 「A」를 입력하면 3개의 컴포넌트가 검색된다. 여기서 특성 「A」는 컨텍스트 「W」와 「X」에 직접 연결되어 있지만 컨텍스트 「Y」와 「Z」에는 연결되어 있지 않다. 그러나 「A」→「X」→「D」→「Z」를 통하여 연결되고, 「「A」→「X」→「D」→「Y」를 통하여 2개의 컨텍스트(「Z」, 「Y」)가 연결됨을 알 수 있다. 하지만 「Y」는 검색과정에서 적게 참조되므로 연결이 제거된다. 이처럼 각 특성와 컨텍스트는 서로 연결되어 있는 노드를 참조해 가면서 활성값을 계산하게 된다. 순환이 반복될수록 활성값은 안정되며 참조회수가 기준에 미달되는 부분은 자동으로 제거되어 계산과정이 종료된다.



(그림 3) 쿼리에 대한 컨텍스트 검색 과정

4.2 시소러스에 의한 컴포넌트 검색

컨텍스트 검색 결과, 한 쿼리셋에서 공통적으로 나타나는 컨텍스트를 모두 만족하는 후보컴포넌트에 대해서 쿼리셋과 컴포넌트들과의 신뢰도를 계산한다. 최종적인 신뢰도는 동치관계, 포함관계, 유사도를 계산함으로써 얻어진다[5]. 다음은

시소러스에 의한 컴포넌트 검색 과정이다.

1) 쿼리S와 대상 컴포넌트의 각 특성에 대한 동치관계(Equivalence)를 계산한다. 동치관계는 두 특성이 얼마나 유사한가를 나타내주는 값이다. 이 값은 앞서 정의한 FFV에 의해 구해질 수 있다.

$$Eq(S(u), T(v)) = FFV(S(u), T(v)) \quad \text{식(8)}$$

2) 특성가중치와 동치관계를 이용하여 포함관계(Implication)를 계산한다. 포함관계는 두 특성이 교환될 수 있는 정도를 나타낸다.

$$Imp(S(u), T(v)) = \min(1, \max FW(T(v)), FW(S(u)))[Eq(u, v)] \quad \text{식(9)}$$

3) 정규화된 가중치 벡터를 이용하여 유사도(Simmularity)를 계산한다.

$$SIM = (IMP^T * EQ) * normalized\ weight\ vector \quad \text{식(10)}$$

위와 같은 과정에 의해 유사도가 계산된 후보 컴포넌트들은 유사도 순으로 우선순위에 따라 최종적으로 검색된다.

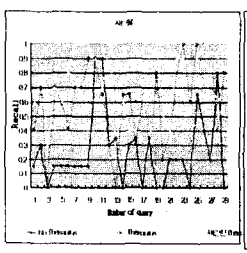
5. 성능 평가

특성 간소화를 위하여 본 연구에서는 카이제곱 방법을 이용하여 30개의 컴포넌트 중 각 컨텍스트별로 제거되는 특성을 시뮬레이션하였다. 각 컴포넌트는 컨텍스트로 분류된 특성으로 구성되어 있으며, 이들 특성은 서로 중복되어 존재한다. <표 4>는 각 컨텍스트가 포함된 컴포넌트 수와 카이제곱 방법에 의해서 변환된 특성의 변화량을 나타낸 것이다. 각 컨텍스트 별로 평균 17% 정도 특성이 간소화되었음을 알 수 있다.

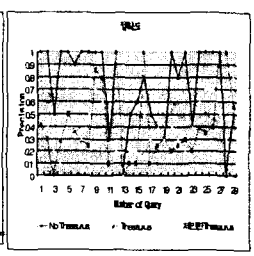
	All	Generic	Graphic	Service	Interacting	Interface	EDT
Context를 포함하는 component의 수	30	2	12	15	4	10	6
모든 Feature의 수 (중복허용)	461	19	106	107	55	119	55
제거된 Feature 수 (중복허용) Distinct Feature기준	78 개 (17%)	2 개 (26.3%)	10 개 (9.4%)	25 개 (23.3%)	13 개 (23.6%)	16 개 (13.5%)	9 개 (16.4%)

<표 4> 특성 간소화

또한 검색의 재현율과 정확성을 측정하기 위해서 쿼리에 대한 컴포넌트 검색 결과를 실험하였다. 쿼리는 임의로 30개를 선정하고 각각의 쿼리 대한 컴포넌트 검색 결과를 측정하였다.



(그림 4) 재현율



(그림 5) 정확도

(그림 4)와 (그림 5)는 이 결과에 대한 정확도와 재현율을 측정한 것이다. 시소러스를 적용하지 않았을 경우, 시소러스만 적용하여 검색하였을 경우, 그리고 E-SARM 방법을 이용한 시소러스를 적용한 제안한 방법을 비교하였다. 정확도 면에서도 뒤지지 않으면서 재현율이 크게 향상되었음을 알 수 있다.

6. 결론

본 연구는 컴포넌트의 효율적인 재사용 위해 특성 기반 시소러스 구축하였다. 컴포넌트는 컨텍스트에 의해 패싯 분류하였고, 각 컨텍스트는 소스 코드로부터 추출된 특성으로 구성된다. 추출된 특성은 카이제곱 방법을 이용하여 간소화하였으며, 컨텍스트와 특성들간의 관련값에 대한 통계적 분석에 의해 시소러스를 구축하였다. 또한 E-SARM 방법을 사용하여 후보 컴포넌트를 검색하였고 유사도를 측정하여 최적의 컴포넌트를 선택할 수 있도록 하였다. 본 연구는 기존의 검색 시스템과 비교하여 다중 패싯 분류된 컴포넌트의 검색에 효율적이며, 검색 시 컨텍스트를 직접 선택해야 하는 사용자 수작업의 부담을 최대한 감소시켜 컴포넌트의 재사용성을 높일 수 있었다. 또한 카이제곱 방법을 이용한 특성 간소화로 데이터 양을 17% 정도 감소시킬 수 있어 시소러스의 복잡함과 방대함을 개선할 수 있었으며, 재현율도 크게 향상시킬 수 있었다.

앞으로의 연구 방향은, 상속 관계를 컨텍스트에 표현하는 방법과 컴포넌트의 효율적인 조립 방법을 연구하는 데 있다.

참고문헌

[1] E. Damini, M.G.Fugini, C. Bellettini, "A Hierarchy-Aware Approach to Faceted Classification of Object-Oriented Components", The ACM Transaction on Software Engineering and Methodology, Vol.8, No.4, Oct. 1999, 425-472.

[2] 최재훈, 한종진, 박종진, 양재동, "구조적인 시소러스 구축을 지원하는 객체 기반 정보 검색 모델", 정보과학회 논문지, 제24권 제11호, 1997.

[3] Scott Henninger, "Information Access Tools for Software Reuse", System Software, pp. 231-247, 1995.

[4] 한정수, 송영재, "개선된 SARM을 이용한 객체지향 부품 재사용 시스템," 정보처리논문지, 제7권 제4호, pp. 1092-1102, 4. 2000.

[5] 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.