제19회 한국정보처리학회 춘계학술발표대회 논문집 제10권 제1호 (2003. 5)

효율적인 컴포넌트 분류와 검색을 위한 질의정보 추출 및 식별자 생성

박제연*, 송영재**
경희대학교 컴퓨터공학과
e-mail:jy_bak@cvs2.khu.ac.kr*
yjsong@khu.ac.kr**

Extraction of Query Information and Generation of Identifier for Effective Component Classification and Retrieval

Jea-Youn Park, Young-Jae Song
Dept of Computer Engineering, Kyung-Hee University

요 약

소프트웨어 생산성과 품질을 개선하기 위한 방안으로 컴포넌트 기반의 소프트웨어 개발이 전개되고 있다. 소프트웨어 컴포넌트 라이브러리를 재사용하기 위해서는 재사용 가능한 컴포넌트를 효율적으로 수집하여 분류, 저장, 검색하여야한다. 기존의 요구사항 정형화 기법들은 요구사항들 간의 의미적 관계를 표현하는 데 초점을 맞추고 있어 컴포넌트 검색에는 적합하지 않으므로 본 연구에서는 개발하려는 유즈케이스 다이어그램을 구문분석을 거쳐 명세하여 질의 정보를 추출하였다.

기존의 자연어를 기반으로 하는 컴포넌트의 비정형적인 명세를 컴포넌트 검색과 조립에 필요한 정보를 효율적으로 얻을 수 있도록 구문분석과 추상화 단계를 거쳐 정형화된 중간형태의 명세로 전환하고 제안한 유사도를 사용하여 컴포넌트를 검색하고자 한다. 또한 개괄명세와 상세명세를 통해 컴포넌트 검색에 필요한 정보를 추출할 뿐만 아니라 컴포넌트의 aspect을 이용하여 컴포넌트 조립에 필요한 정보도 얻을 수 있다. 2차 질의를 통해 컴포넌트 검색의 정확도를 향상시키고 명세를 추상화시켜 검색의 재현율을 향상시킨다.

1. 서론

지난 몇 년 동안 패턴, 프레임워크, CBSE의 출현은 소프트웨어 개발에서 새로운 시대를 열었다[1][2]. CBSE는 소프트웨어 컴포넌트를 plug & play방식으로 어플리케이션에 조립하는 것을 목표로 소프트웨어 개발에 나타난 새로운 패러디움이다[1].

컴포넌트 기반의 소프트웨어를 개발하는 데 있어서 효율성과 생산성을 증가시키기 위해 컴포넌트 식별단계 전인 요구사항 분석단계에서 사용자가 요구하는 역할을 수행하는 적절한 컴포넌트를 검색해야 한다[3].

요구사항을 명세하기 위한 기존의 연구들로는 OCL(Object Constraint Language)[4], Z Notation[5]등이 있다. 하지만 이러한 요구사항 정형화 기법들은 요구사항들 간의 의미적 관계를 표현하는 데 초점을 맞추고 있어서 컴포넌트 검색에는 적합하지 않다. 본 논문에서는 요구사항을 패싯으로 나누어 명세하고 검색시에는 element dictionary를 이용해 상위수준의 질의어를 생성하는 방법을 제안한다.

컴포넌트 명세는 특정 문제에 대한 컴포넌트의 기능을 기술함으로써 컴포넌트의 재사용을 향상시키고 보다 쉽게 접근할 수 있는 방법을 제공한다. 그러나 컴포넌트에 관한 정보를 수집하는데 많은 어려움이 있으며 컴포넌트에 관한 정보를 얻는 방법이 컴포넌트의 종류에 따라 다르며 복잡한 프로그래밍을 통해 가능하다[6].

본 연구에서는 자연어를 기반으로 비정형화 명세를 제안하는 컴포넌트 개괄명세와 상세명세의 추상화를 통해 정형화 시키는 중간형태의 명세방법을 사용한다.

본 논문에서 정의한 컴포넌트 명세방법을 통해 컴포넌트 식별자를 생성하여 컴포넌트를 분류하여 저장하고, 검색시에는 제안한 요구사항 명세방법을 이용하여 다중 질의를 생성하여 검색의 정확도와 재현율을 향상시키고자 한다.

2. 관련 연구

본 장에서는 요구사항 정형화 기법인 Z 표기법과 C2 컴포넌트 명세언어와 Signature matching을 이용한 컴포넌트 검색방법에 대해 기술하고 문제점을 제시하였다.

2.1 Z 표기법(Z notation)

수학공식을 많이 사용하는 정형적 명세서는 이해하기 어렵고 읽기에 지루한 단점이 있다. Z 표기법은 이러한 문제를

해결하기 위해 객체들이 각각의 명세를 만족하도록 정의하는 정확한 규칙들을 제공하고, 구문적 도메인의 구성요소에 관해 작성한다. 다시 말해, 비정형적 기술을 지지함으로써 정형적 명세를 효율적으로 만든다[5].

Z 표기법은 집합론과 1차 논리(first-order logic)에 근거를 두고 있다. Z는 명세의 상태공간과 동작을 기술하기 위해 스키마라고 부르는 구조를 제공해 준다. 스키마는 상태변수를 소개하고 제약조건과 상태에 대한 연산을 정의하는 데 사용된다. 스키마 연산은 스키마 합성, 스키마 재명명과 스키마 은폐를 포함한다. 이 연산들은 스키마를 조작할 수 있게 해 주는 시스템 명세에 대한 강력한 메커니즘이다.

그러나, Z 표기법은 요구사항들 간의 의미적 관계를 표현하는 데 초점을 맞추고 있어서 본 연구에서 제안하는 컴포넌트 검색방안에는 적합하지 않다

2.2 C2

C2는 메시지 기반 스타일을 사용하여 사용자 인터페이스를 기술하는 언어이며 컴포넌트에서 발생한 이벤트를 다른 컴포넌트로 전달하는 커넥터와 컴포넌트로 구성되며, 통지 이벤트와 요구 이벤트의 전달 방향성에 관한 제약조건과 연결규칙을 정의함으로써 올바른 구성을 제어한다.[7]

이런 명세 언어는 나름대로의 장점을 가지고 있지만, 많은 경우 정형 명세를 기반으로 하고, 비기능의 명세 지원이 미흡하고, 재사용을 고려하지 않으므로 명세되는 시스템의 관점에 제한적이다.

2.3 Signature matching

Zaremski and Wing[8]은 오퍼레이션의 시그너쳐 일치를 기반으로 하는 컴포넌트 검색방법을 제안하였다. 이 방법은 정형적인 명세 언어 Larch/ML을 사용하여 오퍼레이션의 행위를 명세한다. 그러나, 이 방법은 오퍼레이션의 시그너쳐에서 나타나는 용어를 기반으로 행위를 기술한다. 그러나 이 방법은 컴포넌트 의미를 완전하게 명세하지 않는다.

사실, 검색방법에 대한 연구는 명세와 질의사이에서 일치를 검증하기위한 이론들에 초점을 맞추고 있다. 그러나 기존의 검색 방법들은 컴포넌트의 의미를 완전하게 명세하는 방법에 대해서는 좀처럼 기록하지 않고 있다. 또한 이 방법은 컴포넌트의 상세설계가 끝난 후에 컴포넌트를 검색해야 한다는 단점이 있다.

2.4 유사도(Similarity)

정형화된 명세에서 술어들을 이용하여 컴포넌트 사이의 유사도를 계산하였다[9].

$$S = \frac{\sum e_{ij}}{\sqrt{n\sqrt{m}}}$$

$e_{ij} = 1$ if $a_i = b_j$
$e_{ij} = 0$ if $a_i \neq b_j$
for i = 1....n and j =1....m

a_i : 컴포넌트 A의 predicate
b_j : 컴포넌트 B의 predicate

그러나 이 계산식은 모든 술어들을 비교하여 유사도를 측정함으로 검색시간의 오버헤드가 발생하는 문제점이 있다.

3. 요구사항 명세와 컴포넌트 명세

3.1 질의정보 추출을 위한 요구사항 명세

기존의 요구사항 정형화 기법들은 요구사항들 간의 의미적 관계를 표현하는 데 초점을 맞추고 있고 구문형태로 명세되어 있어 컴포넌트 검색시 질의어 생성하기 어려웠다. 요구사항을 패싯으로 나누어 명세하고 검색시에는 element dictionary를 이용해 상위수준의 질의어를 생성하는 방법을 제안한다.

요구사항간의 정확한 분석을 위해 UML의 유즈케이스 다이어그램을 사용하고 그 다음으로 구문분석 단계를 거쳐 유즈케이스 다이어그램의 각 항목을 <표 1>과 같이 분류하여 정의하였다.

Special_requirement는 비기능적인 정보를 나타내고 있으며, 유즈케이스 다이어그램에 대한 명세는 구문분석 단계를 거쳐 전체적인 개괄 명세뿐만 아니라 유즈케이스의 각 진행단계들까지 <function>, <object>, <medium>형태로 분류하여 상세 명세하였다.

위에서 정의한 유즈케이스 다이어그램 명세에서 분석한 각 항목들을 이용하여 XML DTD를 정의한다.

항 목	정 의
Usecase_name	유즈케이스 이름
Special_requirement1	운영환경
Special_requirement2	구현언어
Usecase_func_spec	유즈케이스의 기능 명세
Usecase_object_spec	유즈케이스의 객체 명세
Usecase_med_spec	유즈케이스의 매체 명세
Process_name	유즈케이스 진행단계 이름
Process_func_spec	유즈케이스 진행단계 기능 명세
Process_obj_spec	유즈케이스 진행단계 객체 명세
Process_med_spec	유즈케이스 진행단계 매체 명세
Process_information	진행단계 수행을 위한 정보
Process_result	진행단계 수행결과로 생긴 정보

<표 1> 유즈케이스 다이어그램 명세 항목

<Usecase_architecture> 태그를 최상위 태그로 나타내며, 각각의 하위 태그들을 계층적으로 표현함으로서 개발자가 그 의미를 파악하고 데이터를 입력하는데 어려움이 없도록 표현하였다.

3.2 상세명세를 지원하는 컴포넌트 서술자

컴포넌트 서술자는 컴포넌트 재사용과 유지보수하고 업그레이드하는데 필수적인 자료이다. 그러나 컴포넌트에 관한 정보를 수집하는데 많은 어려움이 있으며 컴포넌트에 관한 정보를 얻는 방법이 컴포넌트의 종류에 따라 다르며 복잡한 프로그래밍을 통해 가능하다. 본 연구에서는 자연어를 기반으로 하는 비정형 명세를 제안하는 컴포넌트 개괄명세와 상세명세의 추상화를 통해 정형화 시키는 중간형태

의 명세방법을 사용한다.

비기능적인 정보(General_info)는 컴포넌트의 일반적인 정보를 명세하고, 기능적인 정보(Function_info)는 컴포넌트가 제공하는 서비스에 대한 개괄 명세뿐만 아니라 컴포넌트의 메소드에 대해 <function>, <object>, <medium>형태로 분류하여 상세 명세하였다. 조립 정보<Assemble_info>에는컴포넌트의 aspect을 이용하여 <User interface>, <Distribution>, <Collaborative work>, <Security>등에 관련된 정보를 명세하였다.

<표 2>는 제안하는 컴포넌트 명세에 필요한 항목들의 정의이다.

항 목	정 의
Component_name	컴포넌트 이름
Platform	운영환경
Size	크기
Program_language	구현언어
License	라이센스
Price	가격
Compo_func_spec	컴포넌트의 기능 명세
Compo_obj_spec	컴포넌트의 객체 명세
Compo_med_spec	컴포넌트의 매체 명세
Method_name	메소드 이름
Method_func_spec	메소드의 기능 명세
Method_obj_spec	메소드의 객체 명세
Method_med_spec	메소드의 매체 명세
Method_para_type	메소드의 파라미터 정보
Method_re_type	메소드의 리턴정보
User interface	사용자 인터페이스
Distribution	분산처리서비스
Collaborative work	협동작업 서비스
Security	보안 서비스

<표 2> 컴포넌트 명세 항목

4. 컴포넌트 분류와 검색

본 연구에서는 컴포넌트 명세를 이용하여 컴포넌트 검색과 조립에 필요한 컴포넌트 정보를 추출한다. 이렇게 추출된 정보를 가지고 element dictionary를 사용하여 high level의 컴포넌트 식별자를 생성하여 분류 저장한다.

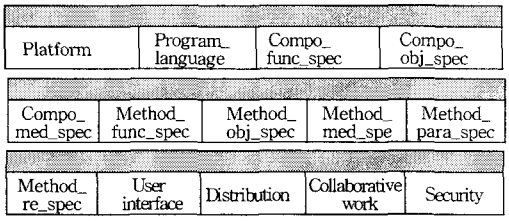
4.1 컴포넌트 분류모델

R.Prieto-Diaz 패싯분류[10]는 컴포넌트의 기본적인 오퍼레이션을 나타내는 <function>패싯과 function의 수행대상을 나타내는 <object>패싯, object이 저장되거나 수행되는 장소를 나타내는 <medium>패싯과 <언어>와 <운영체제>패싯으로만 구성되어 있어 컴포넌트의 특성을 반영하기 어렵고 그 관련성을 표현할 수 없다는 단점이 있다.

본 논문에서는 컴포넌트의 검색의 정확도(precision)를 높이기 위해 컴포넌트의 개괄명세와 상세명세를 자연어를 기반으로 하는 정형화 명세를 통해 다중 패싯으로 분류하고 재현율(recall)을 높이기 위해 구문분석을 거친 특성들을 element dictionary를 이용하여 high level 로 표현하였다.

컴포넌트 명세는 열거형 분류방법 등에서 요구하는 별도의 분류나 조작없이 패싯으로 전환될 수 있다. 즉 XML DTD의 ELEMENT와 ATTRIBUTE에 표현된 내용을 한 개의 패싯으로 만들 수 있다.

저장소에 구축된 컴포넌트의 패싯의 수는 메소드의 수에 따라 유동적이며 (그림 1)과 같이 구성된다.



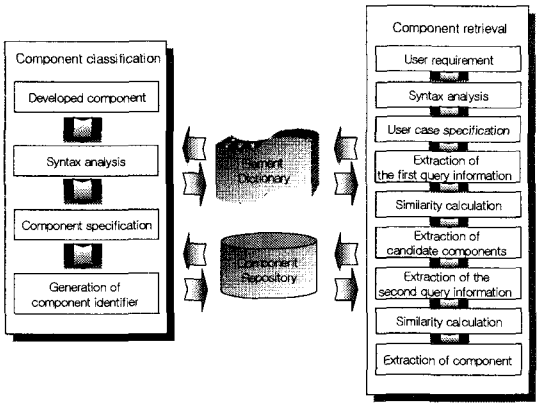
(그림 1) Repository에 구축된 컴포넌트 식별자

4.2 컴포넌트 검색 시스템 구조

본 논문에서 제안하는 검색 모델은 사용자가 구문분석 단계를 거쳐 유즈케이스 다이어그램과 element dictionary를 통해 질의를 생성하여 후보컴포넌트를 추출하게 된다.

입력된 질의는 유사도가 가장 큰 순서대로 컴포넌트를 정렬하여 컴포넌트 정보를 사용자에게 표시한다.

컴포넌트 저장소 구축 및 검색과정은 (그림 2)와 같다.



(그림2) 컴포넌트 저장소 구축 및 검색과정

4.3 유사도

유즈케이스 모델링을 통해 생성된 속성들을 이용하여 생성된 1차 질의와 생성될 후보 컴포넌트의 유사도를 구하기 위해 개선된 식은 아래와 같다.

정의 1) Q_1 : 1차 사용자 질의

q_{2i} : 구문분석 단계를 거친 유즈케이스 모델링을 통해 생성된 속성

(운영환경, 구현언어, 유즈케이스 기능명세, 유즈케이스 객체명세, 유즈케이스 매체명세)

$Q_1 = (q_1, q_2, q_3 \ldots\ldots, q_l)$

정의 2) A : 컴포넌트

a_i : 1차 컴포넌트 개괄명세 속성

(운영환경, 구현언어, 컴포넌트 기능명세, 컴포넌트 객체명세, 컴포넌트 매체명세)

$A = (a_1, a_2, a_3 \ldots\ldots, a_l)$

$$P_1 = \frac{\sum e_i}{l}$$

$e_i = 1$ if $a_i = q_i$
$e_i = 0$ if $a_i \neq q_i$
for $i = 1....l$

정확도를 높이기 위해 컴포넌트 검색에 2차 질의를 실시한다. 2차 질의에서 사용자 질의와 추출된 후보 컴포넌트와의 유사도를 구하는 식은 아래와 같다.

정의 3) Q_2 : 2차 사용자 질의

q_{2i} : 구문분석 단계를 거친 유즈케이스 진행단계 모델링을 통해 생성된 속성
(유즈케이스 진행단계의 기능명세, 유즈케이스 진행단계의 객체명세, 유즈케이스 진행단계의 매체명세, 유즈케이스 진행단계의 파라미터정보, 유즈케이스 진행단계의 리턴정보)

$Q_2 = (q_{l+1}, q_{l+2}, q_{l+3}....., q_m)$

정의 4) B : 1차 질의결과로 추출된 후보컴포넌트

b_i : 2차 컴포넌트 상세명세 속성
(메소드의 기능명세, 메소드의 객체명세, 메소드의 매체명세, 메소드의 파라미터정보, 메소드의 리턴정보)

$B = (b_{l+1}, b_{l+2}, ..., b_n)$

$$P_2 = \frac{\sum e_i}{l}\left(\frac{\sum S_{jk}}{\sqrt{(m-l-1)}\sqrt{(n-l-1)}}\right)$$

$S_{jk} = 1$ if $b_j = q_k$
$S_{jk} = 0$ if $b_j \neq q_k$
for $j = l + 1....n$, $k = l+1...m$

5. 비교평가

본 논문에서 제안하는 방법은 확장된 facet 자연어를 기반으로 컴포넌트를 분류하여 검색의 재현율을 향상시키고 개괄명세와 상세명세를 통해 컴포넌트 검색에 필요한 정보를 추출할 뿐만 아니라 컴포넌트의 aspect을 이용하여 컴포넌트 조립에 필요한 정보도 얻을 수 있다. 또한 2차 질의를 통해 컴포넌트 검색의 정확도를 향상시킨다.

<표 3>은 기존의 시스템과 본 논문에서 제안하는 시스템의 뷰를 비교평가 하였다.

비교항목	RSL	R.Prieto-Diaz	본 논문제시 방법
색인방법	자연어	facet 자연어	확장된 facet 자연어
다중질의	지원하지 않음	지원하지 않음	지원
조립정보	지원하지 않음	지원하지 않음	지원

<표 3> 기존시스템과의 비교

6. 결론

본 연구에서는 개발하려는 시스템의 요구분석 단계에서의 산출물인 유즈케이스 다이어그램에 대한 명세에서 질의 정보를 추출하여 컴포넌트 식별단계전에 후보 컴포넌트를 추출하여 컴포넌트의 재사용성을 향상시킨다.

개괄명세와 상세명세를 통해 컴포넌트 검색에 필요한 정보를 추출할 뿐만 아니라 컴포넌트의 aspect을 이용하여 컴포넌트 조립에 필요한 정보도 얻는다.

자연어를 기반으로 하는 비정형적인 명세를 구문분석과 추상화 단계를 거쳐 정형화 시키는 중간형태의 명세로 전환하여 컴포넌트 검색과 조립에 필요한 정보를 효율적으로 얻을 수 있으며, 유즈케이스 명세를 이용한 다중 컴포넌트 검색은 검색의 정확도와 재현율을 향상시킬 것이다.

참고문헌

[1] Aoyama, M. New Age of Software Development: How component-Based Software Engineering Changes the Way of Software Development, 20th ICSE International Workshop on Component-Based Software Engineering (Kyoto, Japan), 1998

[2]Bergner, K., Rausch, A. and Sihling, M. Componentware-The Big Picture, 20th ICSE International Workshop on Component-Based Software Engineering (Kyoto, Japan), 1998

[3] Joseph Schmuller, "Unified Modeling Language" Second Edition, SAMS, (2002)

[4]Abstract State Machines,
http://www.eecs.umich.edu/gasm

[5]ROGER'S .PRESSMAN, "Software Engineering-A Practitioner's Approach, 5E", McGraw-Hill Korea

[6] Rober C. Seacord et al, "AGORA: A Search Engine for Software Components," IEEE Internet Computing, pp.62-70 Nov. Dec., 1998

[7] P. Oriezy, N. Medvidovic and R. N. Taylor, "Architecture Based Runtime Software Evolution". Proceedings of The International Conference on Software Engineering, 1998

[8]Zaremski, A. and Wing, J. M., Specification Matching of Software Components, Proceedings of 3rd ACM SIGSOFT Symposium on the Foundations of Software Engineering, pp.6-17, October 1995.

[9] Chao-Tsun Chang; Chu, W.c.; Chung-Shyan Liu; Hongji Yang "A Formal Approach to Software Components Classification and Retrieval" Computer Software and Applications Conference, 1997. COMPSAC '97. proceedings.

[10] R.Prieto-Diaz, "Implementing Faceted Classification For Software Reuse" Comm. ACM, vol.34, no.5, pp89-97, May,1991

[11] Grundy, J, "Storage and retrieval of software components using aspects", Computer Science Conference, 2000. ACSC 2000. 23rd Australasian , 2000