요구사항 분류 언어를 통한 반 자동 품질 요구사항 분류

박수용 민성기 최순황
Sooyong Park Sungki Min Soonhwang Choi
시스템체계공학원,
* 서강대학교 창업보육센터 203 호

ABSTRACT

시나리오 형태의 요구사항 분류는 ATAM, SAAM, Software Quality Metrics 과 같은 품질 요구사항 분석 및 평가 방법 등 많은 분야에 응용된다. 이들 기법들은 소프트웨어 시스템의 품질 요구사항을 분석, 평가 하기에 앞서 초기 수집된 요구사항들을 분류하게 된다. 그러나 요구사항을 분류하는 일은 수작업을 통해 이루어지게 되고, 따라서 미 분류, 중복분류, 등의 결함을 가질 수 있다. 결함의 가능성은 요구사항의 수가 많은 대형 프로젝트 일수록 높아지게 된다. 따라서 본 논문에서는 요구사항 분류언어를 통한 품질 요구사항 자동 분류 기법을 제안한다. 제안된 기법은 분류언어와 유사도를 이용한 2 단계 분류기법을 이용하였다. 분류언어는 각 도메인 별로 개발되어 비슷한 도메인일 경우 재사용될 수 있다. 이를 검증하기 위해, 본 논문에서는 15 여개의 프로젝트로부터 수집된 요구사항을 이용해 실험을 수행하고 그 결과를 분석, 평가 하였다.

1. 서 론

요구사항 분류에 관한 응용은 ATAM, SAAM 과 같은 품질 요구사항 분석 및 평가 방법 등 많은 분야에 응용된다. 이들 기법들은 소프트웨어의 각 개발 단계에서 품질 요구사항을 평가하도록 지원하고 있다. 품질 요구사항들이 잘 관리되어 프로젝트 초기에 충돌 요구사항들이 발견된다면 협의를 통하여 이해 당사자들 간에 상호 만족된 해결책을 도출할 수 있을 것이다. 반면 충돌된 요구사항들이 드러나지 않는다면 프로젝트 후반에 이들의 완성도로 인하여 반드시 문제가 생기게 된다.

품질 요구사항간의 충돌을 프로젝트 초기에 발견하기 위해서는 초기 수집된 품질 요구사항들을 각 품질 속성 별로 분류하는 작업이 필요하다. 과거 프로젝트의 크기가 작아 품질 요구사항의 수가 작았을 때에는 이러한 분류를 수작업으로 해 왔으나, 프로젝트의 크기가 점점 커지게 되면서 관리해야 할 요구사항의 수가 증가하여 이를 수작업으로 분류하는 것이 매우 비효율적인 작업이 되었다

이에 따라, 본 논문에서는 품질 요구사항 분류를 반 자동화 하여 수작업의 양을 줄일 수 있는 기법을 제안한다. 제안된 기법은 각 품질 속성별로 그 속성을 대표하는 분류언어를 개발하고, 이를 바탕으로 유사도 기법을 이용하여 2 단계 분류를 한다. 이 기법을 통해 분류된 요구사항은 후에 수작업을 거쳐 완전히 분류될 수 있으며, 기존의 방법보다 수작업의 양을 현저히 줄일 수 있다

그리고, 제시된 기법을 검증하기 위해, 제안된 기법을 구현하는 분류엔진을 개발하였고, 15 여 개의 프로젝트로부터 수집된 요구사항을 이용하여 실험을 수행하고 그 결과를 분석하였다.

2. 관련연구

요구 사항 문서를 자동으로 분류하는 기존의 연구들은 주로 문서에서 추출된 특정한 키워드들과 유의어 사전(thesaurus)을 이용하거나 단어의 반복(reiteration)과 공기 정보(collocation) 같은 언어 현상을 이용한 간단한 유사도 측정 방법을 이용하였다. Palmar 는 2 층 구조의 TTC(Two-tired clustering) 알고리즘을 이용하여 요구 사항 문서를 색인하고 클러스터링 하는 방법을 제안하였다. TTC 알고리즘은 먼저 각 요구사항 문서에 속해 있는 동사들을 키워드로 사용하고 동사 유의어 사전을 사용하여 문서를 기능별로 분류하고, 이렇게 기능별로 분류된 문서들 사이에 코사인 유사도를 측정하여 재 분류하는 방법이다. 그리고, Yaung 은 그래프 모델을 기반으로 하여 결함 개념을 사용해서 높은 결합 상태를 가지는 요구 사항들은 같은 클러스터 안에 위치하고 낮은 결합상태를 가지는 요구사항들은 서로 다른 클러스터에 위치하게 함으로써 요구 사항 클러스터링 분석을 제안하였다. 그러나, 이러한 방법들은 요구 사항 문서들을 클러스터링 하는 기법으로서 요구 사항 문서에 비해 적은 의미 정보를 가지고 있는 문장 단위의 요구 사항들의 주제별 범주화에는 효율적인 방법이 되지 못한다 [1][2][3][4].

3. 접근 방법

제안된 기법은 자연어 형태의 미 분류 요구사항을 분류언어와 유사도 기법을 이용해서 분류한다. 분류언어란 문장의 의미를 가장 잘 나타내는 단어와 그 단어들 간의 조합이고 유사도는 문장 간 유사한 정도를 말한다.

제안된 분류 기법의 전체적인 공정은 그림 1 에 나타나 있다. 분류 시스템은 요구사항을 크게 2 단계로 분류한다.

첫 번째 단계에서는 분류언어를 만족하는 요구사항을 분류한다. 두 번째 단계에서는 첫 번째 단계에서 미 분류된 요구사항과 분류된 요구사항 사이의 유사도를 측정하여 분류한다.

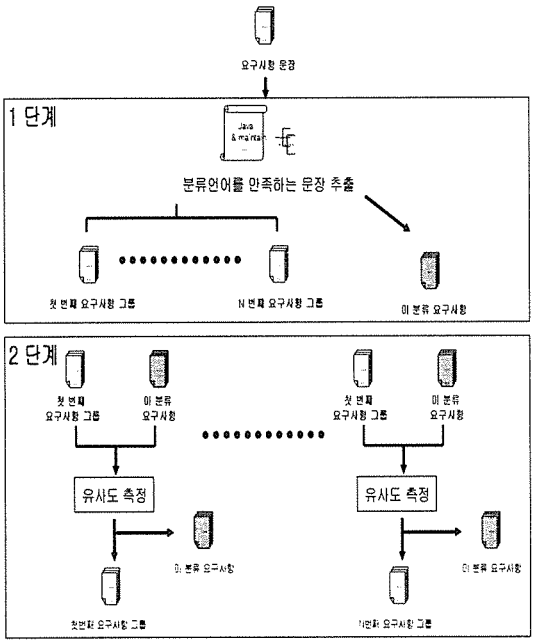

그림 1 분류 시스템의 전체 공정

3.1 첫 번째 단계 : 분류언어 근간의 요구사항 추출

첫 번째 단계에서는 각 품질 속성의 특성을 지닌 요구사항을 추출해야 한다. 본 논문에서는 분류언어를 이용하여 요구사항을 추출하였다. 즉, 본 논문에서는 분류언어를 만족하는 요구사항이 그 품질 속성의 특성을 가장 잘 반영한다고 간주한다. 분류 언어는 예제 요구사항 문장을 분석하고 유의어 사전을 참조하여 작성되었다.

그림 2 는 분류언어를 만족하는 요구사항 추출 공정을 보여준다. 'Step2 : 단어들을 어간 형태로 변형' 단계에서는 요구사항 문장에 포함된 단어들의 접두사나 접미사를 제거하여 그 단어를 어간 형태로 변형한다. 단어들을 어간 형태로 변형하는 이유는 분류언어를 만족하는지 비교하기 전 영어단어의 형태를 통일시키기 위해서 이다. 영어단어의 경우 동사는 시제나 인칭에 따라, 명사는 수에 따라 그 형태가 바뀔 수 있으나 의미는 동일하다.

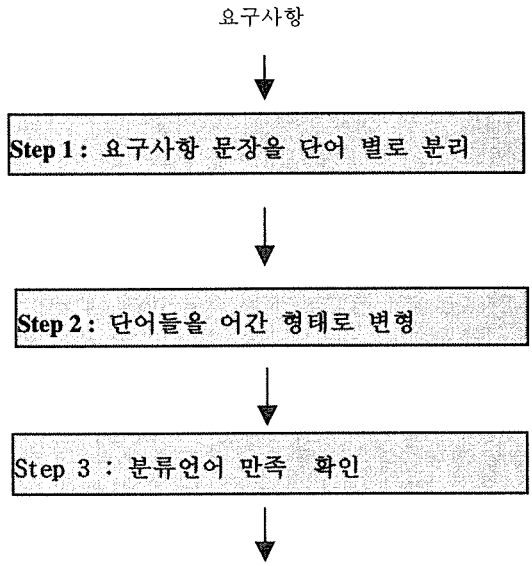

그림 2 분류언어를 만족하는 요구사항 추출 공정

'Step3 : 분류언어 만족 확인' 단계에서는 미리 정의된 분류언어를 근간으로 요구사항 문장이 그 분류언어를 만족하는지를 확인한다. 본 논문에서는 분류언어의 연산자로 검색 분야에서 가장 보편적으로 쓰이는 4 가지 연산자를 지원한다. 그 내용은 다음과 같다.

A. '&' : '그리고'

Ex) "java & maintain" 은 'java' 와 'maintain' 이라는 단어가 모두 포함된 문장을 찾는다.

B. '+' : '또는'

Ex) "java + maintain" 은 'java' 또는 'maintain' 이란 단어를 포함한 문장을 찾는다.

C. '~' : '인접'

Ex) " java ~ maintain" 은 'java' 와 'maintain' 이라는 단어가 모두 포함되고 두 단어가 인접해 있는 문장을 찾는다.

D. '>' : '순서'

Ex) " java > maintain" 은 'java' 와 'maintain' 이라는 단어가 모두 포함되고 'java' 라는 단어가 'maintain' 이라는 단어보다 먼저 나오는 문장을 찾는다.

E. '(' , ')' : 연산 우선순위를 묶는 역할

Ex) " (java > maintain) + (language > maintain)" 은 'java' 라는 단어가

'maintain' 이라는 단어에 선행하거나 'language' 라는 단어가 'maintain' 이라는 단어에 선행하는 문장을 찾는다.

위의 연산자들을 조합하면 다음과 같은 분류언어를 만들 수 있다.

(show + tell) & (use + user + function)

- 'show' 또는 'tell' 이라는 단어가 나오고 ' use' , 'user' 또는 'function' 이라는 단어 가 나오는 문장

(tool + technology + platform) > run > most

- 'too' , 'technology' 또는 'platform' 이라는 단어가 나오고, 그 뒤에 'run' 과 'most' 라는 단어가 순서대로 나오는 문장

3.2 두 번째 단계: 요구사항 간의 유사도 측정

첫 번째 단계에서 미 분류된 요구사항들은 이미 분류된 요구사항들과의 유사도를 측정함으로써 분류될 수 있다. 미 분류된 요구사항과 분류 된 요구사항의 유사도가 일정 값 이상이 되면 미 분류 요구사항은 비교된 분류 요구사항과 같은 품질 속성으로 분류된다.

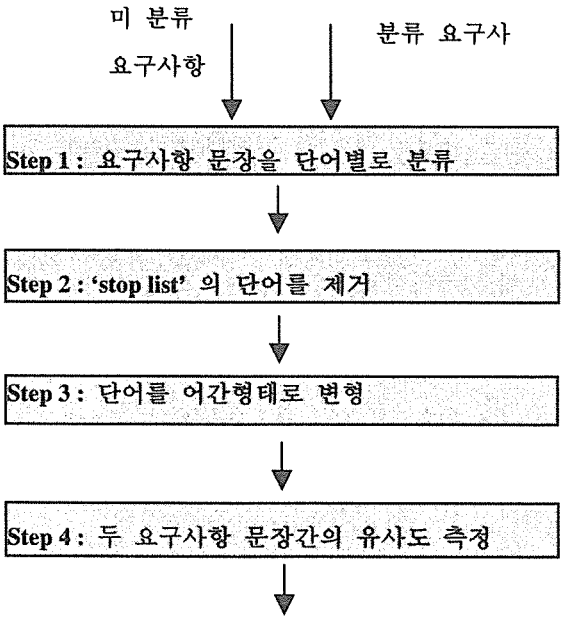

그림 3 유사도를 이용한 분류 공정

그림 3 은 미 분류 요구사항과 분류 요구사항 간의 유사도 측정 공정을 보여준다. 'stop list' 란 빈번하게 출현하지만 문장의 의미에는 영향을 주지 않는 관사, 전치사, 대명사, 부사, 일부 동사 등의 목록을 말한다. 'Stop list' 에 나타난 단어들을 제거함으로써 인덱싱의 공간과 시간을 절약할 수 있다.

두 문장 간의 유사도는 코사인 계수를 통해 구해지는데 구하는 공식은 다음과 같다.[6][5]

$$r_{vw} = \frac{\sum_{i=1}^{t} V_i W_i}{\sqrt{\sum_{i=1}^{t} (V_i)^2 \sum_{i=1}^{t} (W_i)^2}}$$

위 식에서 V 와 W 는 각각 비교할 요구사항을 나타내는 백터를 표시한다. t 는 모든 요구사항에서 추출한 키워드의 총 개수를 나타낸다. 또한, V_i 는 각 벡터에 속해있는 i 번째 요소를 가리키며, 이 요소는 모든 요구사항에서 추출된 키워드 중에서 i 번째 키워드를 가리킨다. i 번째 요소의 가중치가 0 인 경우에는, 속성 i(i 번째 키워드)는 요구사항 V 에 포함되어 있지 않다는 것을 의미한다. 반면에, i 번째 요소의 가중치가 1 이라면, 속성 i(i 번째 키워드)는 요구사항 V 에 포함되어 있다는 것을 나타낸다.

따라서, 코사인 계수의 값이 1 이면, 두 요구사항은 완전히 일치한다는 것을 나타내고, 코사인 계수의 값이 0 과 1 사이이면 두 요구사항이 서로 공유하는 키워드가 존재한다는 것을 의미하며, 코사인 계수의 값이 0 이면 두 요구사항이 서로 공통으로 포함하는 키워드가 전혀 없다는 것을 나타낸다.

4. 실험 및 평가

분류언어 및 유사도를 이용한 요구사항 분류 기법을 검증하기 위해 다음과 같이 실험을 하였다.

총 15 개의 Project 로부터 366 개의 품질 요구사항을 수집하였고, 요구사항의 수와 프로젝트의 성격에 따라 5 개의 그룹으로 나누어 실험을 수행하였다.

먼저, 요구사항의 개수가 가장 많은 Project 1 로부터 94 개의 품질 요구사항을 수집하여 그룹 A 를 구축하였고, 그 다음으로 Project 2, 3 으로부터 47 개의 요구사항을 수집해 그룹 B 를 그리고 Project 4~10 으로부터 105 개의 품질 요구사항을 수집해 그룹 C 를 구축하였다. 또한, Project A~E 으부터는 120 개의 품질 요구사항을 수집하여 그룹 D 를 구축하였다

각 그룹별로 수집된 요구사항들을 먼저 수작업으로

분류해 이를 실험의 정답으로 사용하였다.

본 논문에서는 품질 요구사항의 분류를 Dependability, Interoperability, Usability, Performance, Adaptability, Reusability 의 6 개의 카테고리로 분류 하였다.

다음으로, 각 그룹별로 수작업으로 분류된 요구사항으로부터 실험에 사용될 분류언어를 개발하였다. 그룹 A 는 총 72 개, 그룹 B 는 총 54 개, 그룹 C 는 총 105 개의 분류언어를 개발하였다. 그룹 D 는 분류언어의 범용성 실험을 위한 그룹으로 수작업 분류의 정답까지만 실험 집단을 구성하였다. 각 분류 속성 별 분류언어는 적게는 3 개에서 많게는 24 개를 개발하였다. 본 논문의 실험에서는 수작업으로 분류된 요구사항으로부터 분류언어를 개발 하였기 때문에, 분류 속성별 포함된 요구사항의 수에 따라 개발된 분류언어의 수가 다르다 구성된 실험 집단을 바탕으로 본 논문에서는 다음과 같은 관점에서 실험을 수행하였다.

실험 1 : 제안된 기법을 통한 요구사항 분류가 얼마나 효과적인가? 각 품질 속성 별로 고른 효과를 나타내는가?

실험 2 : 같은 요구사항 분류 언어가 다른 프로젝트에 적용이 가능한가?

제안된 기법의 효용성은 구현된 분류엔진을 이용해 각 실험 집단으로부터 도출된 요구사항을 분류하고 그 결과를 수작업으로 분류한 결과와 비교하는 방법으로 평가되었다.

효용성의 평가방법은 검색 분야에서 일반적으로 사용되는 정확율 (precision)과 재현율(recall)을 사용하였다. 그 식은 다음과 같다[8].

$$precision = \frac{category\ sentences\ found\ and\ correct}{total\ category\ sentences\ found} \quad (식\ 7)$$

$$recall = \frac{category\ sentences\ found\ and\ correct}{total\ category\ sentences\ correct} \quad (식\ 8)$$

여기서, 정확율이 높다는 것은 분류된 요구사항 문장 중 오답이 적다는 것을 의미하고, 재현율이 높다는 것은 분류된 요구사항 문장 중 정답의 수가 많이 포함되어 있다는 것을 의미한다. 예를 들어 100 개의 요구사항 중 Performance 에 해당하는 요구사항이 20 개라고 했을 때, 자동 분류를 통해 Performance 에 해당하는 요구사항이 25 개가 분류되었고, 이 중 15 개가 정답이라고 하면, 정확율은 15/25 이고, 재현율은 15/20 이다. 즉, 재현율이 높으면, 수 작업으로 재 분류할 요구사항의 수가 줄어들고, 정확율이 높으면 자동 분류된 요구사항 속의 정답을 골라내는 작업이 쉬워진다.

실험 1 분류언어의 효용성

제안된 기법을 통한 요구사항 분류가 얼마나 효과적인지를 알아보기 위해 각 그룹 별로 개발된 분류언어 (이를 분류언어 A, 분류언어 B, 분류언어 C 라 하겠다)를 이용해 해당 그룹의 요구사항을 분류엔진으로 자동 분류하였다. 그 다음 이 결과를 수작업으로 분류한 결과와 비교하여 정확율과 재현율을 측정하였다.

표 1 그룹 A 결과 (분류언어 A 이용, base similarity=0.3)

attribute	precision	Recall
dependability	100%(1/1)	100%(1/1)
interoperability	18.18%(4/22)	57.14%(4/7)
Usability	74.19%(23/31)	74.19%(23/31)
performance	60%(9/15)	90%(9/10)
adaptability	52.94%(9/17)	69.23%(9/13)
Reusability	36.36%(4/11)	100%(4/4)

표 1 은 그룹 A 의 결과이다. 그룹 A는 18~74% 의 정확율과 67~100%의 재현율을 보여주고 있다. 이 중 정확율이 40% 이하인 interoperability 와 reusability 의 경우 요구사항의 정답 수(재현율의 분모)가 각각 7 개, 4개로 10개 이하이다. 대체로 정답의 수가 많은 분류 속성일 경우, 높은 정확율과 재현율을 보여주고 있다.

표 2 그룹 B 결과 (분류언어 B 이용, base similarity=0.3)

attribute	precision	Recall
dependability	100%(18/18)	78.26%(18/23)
interoperability	100%(5/5)	83.3%(5/6)
Usability	91.6%(11/12)	91.6%(11/12)
performance	100%(7/7)	100%(7/7)

adaptability	(0/0)	(0/0)
Reusability	(0/0)	(0/0)

표 2 는 그룹 B 의 결과이다. 그룹 B 의 결과는 그룹 A 의 결과보다 높은 수치를 보여주고 있다. 각 분류 속성 별로 91~100%의 정확율을 보여주고 있으며, 78~100%의 재현율을 보여주고 있다. 여기서 Adaptability 와 reusability 의 정확율과 재현율이 0 으로 나온 것은 두 속성이 정답을 가지고 있지 않아서 이다.

표 3 그룹 C 결과 (분류언어 C 이용, base similarity=0.3)

Attribute	Precision	Recall
Dependability	51.51%(17/33)	100%(17/17)
Interoperability	75%(3/4)	60%(3/5)
Usability	76.19%(32/42)	84.21%(32/38)
Performance	55.55%(10/18)	76.92%(10/13)
Adaptability	52.77%(19/36)	61.29%(19/31)
Reusability	0% (0/1)	0%(0/1)

표 3 은 그룹 C 의 결과이다. 그룹 C 의 결과는 51~76% 정확율과 60~100% 의 재현율로 그룹 A 보다 높고, 그룹 B 보다 낮은 수치를 보여주고 있다. 그룹 C 역시 정답의 수가 10 개 이하인 Interoperability 와 Reusability 의 경우 수치가 낮게 측정되었다. 하지만 요구사항 정답 수 가 많은 분류 속성일 경우 70% 이상의 재현율과 50% 이상의 정확율을 보여주고 있다.

각 그룹별 결과를 종합해 본 결과, 대체로 70 이상의 재현율과 50% 이상의 정확율을 보여주고 있으며, 일부 10 개 이하의 정답 수를 가진 분류 속성의 경우 정확율과 재현율의 수치가 낮게 측정되고 있다. 하지만 자동 분류의 효용성은 요구사항 정답 수가 많은 분류속성에 대한 정확율과 재현율이 더욱 중요하다.

실험 2 분류 언어의 범용성

같은 분류언어가 다른 프로젝트에 적용될 수 있다면, 분류언어를 미리 구축해 놓고, 새로운 프로젝트를 시작할 때 재사용 할 수 있다. 본 논문에서는 분류언어의 재사용 가능성을 알아보기 위해 다음과 같이 실험을 하였다.

그룹 C 에서 구축된 분류언어를 비슷한 도메인인 그룹 A,B 그리고 전혀 다른 도메인인 그룹 D 에 그대로 적용해 보고 그 결과를 측정해 보았다.

그룹 C 의 분류언어를 통한 각 그룹의 정확율 과 재현율의 측정 결과는 표 4 ~표 6 과 같다.

표 4 그룹 A 결과 (분류언어 C 이용, base similarity=0.3)

Attribute	Precision	Recall
Dependability	7.6%(1/13)	100%(1/1)
Interoperability	21.42%(3/14)	42.85%(3/7)
Usability	56.25%(27/48)	87.09%(27/31)
Performance	66.6%(8/12)	80%(8/10)
Adaptability	45%(9/20)	69.23(9/13)
Reusability	30%(3/10)	75%(3/4)

표 4 는 그룹 A 의 요구사항을 그룹 C 에서 개발된 분류언어로 분류한 결과이다. 결과를 살펴보면, Dependability 의 정확율과 Interoperability 의 재현율 그리고 Reusablity 의 정확율과 재현율이 떨어진 것을 제외하고는 대부분 비슷한 수치를 보여주고 있다. 위 분류 속성들의 정확율이나 재현율이 낮아진 원인은 두 가지 이유를 찾을 수 있다.

첫째, 분류언어가 다르기 때문이다. 제안된 기법은 분류언어를 기반으로 1 차 분류를 수행 한 후 유사도를 통해 재 분류하는 기법을 사용하고 있다. 그러므로 분류언어를 통한 1 차 분류가 분류 효율에 큰 영향을 주게 된다. 그룹 C 의 분류언어는 그룹 C 의 요구사항들로부터 유도 되었다. 따라서 그룹 A 의 요구사항으로부터 유도된 그룹 A 의 분류언어보다 그룹 C 의 분류언어는 그룹 A 의 요구사항들을 잘 반영하지 못한다.

둘째, 분류 속성별 요구사항 정답 수(재현율의 분모)가 작기 때문이다. 분류속성 별 요구사항의 정답 수가 너무 작을 경우 분류언어를 통한 1 차 선택의 가능성이 낮아 진다. 따라서 분류언어를 통해 해당 정답 요구사항이 추출 될 경우는 아주 높은 정확율이나 재현율이 측정되고, 그렇지 못할 경우는 아주 낮은 정확율이나 재현율이 측정된다. 그러므로 다른 그룹으로부터 유도된 분류언어는 정답 수가 작은 분류속성의 요구사항을 추출하지 못할 가능성이 더욱 커진다.

표 5 그룹 B 결과 (분류언어 C 이용, base

similarity=0.3)

Attribute	Precision	Recall
Dependability	89.4%(17/19)	73.9%(17/23)
Interoperability	71.42%(5/7)	83.3%(5/6)
Usability	91.6%(11/12)	92.6%(11/12)
Performance	87.5%(7/8)	100%(7/7)
Adaptability	(0/3)	(0/0)
Reusability	(0/0)	(0/0)

표 5 는 그룹 B 의 요구사항을 그룹 C 에서 개발된 분류언어를 이용해 분류한 결과이다. Adatability 와 reusability 의 0 으로 측정된 수치는 이 속성의 정답이 없기 때문이다. Interoperability 와 performance 의 정확율이 약 20% 정도 감소한 것을 제외하면 그룹 B 의 부류언어로 분류한 결과와 비슷한 수치를 보여주고 있다. 이 결과는 그룹 A 의 결과와 마찬가지로, 분류언어가 각기 다른 프로젝트의 요구사항으로부터 유도되었기 때문이고,이 속성의 정답 수가 적기 때문이다.

표6 그룹 D 결과 (분류언어 C 이용, base similarity=0.3)

Attribute	Precision	Recall
Dependability	33.3%(3/9)	18.75%(3/16)
Interoperability	60%(3/5)	75%(3/4)
Usability	16.6%(2/12)	33.3%(2/6)
Performance	18.51%(5/27)	50%(5/10)
Adaptability	30.23%(13/43)	61.90%(13/21)
Reusability	0%(0/1)	0%(0/3)

표 6 은 그룹 D 의 요구사항을 그룹 C 에서 개발된 분류언어를 이용해 분류한 결과이다. 그룹 D 의 경우는 그룹 A 와 그룹 B 와 달리 국방 도메인의 프로젝트로부터 추출한 요구사항들로 구성되어 있다. 따라서 그룹 C 에서 개발된 분류언어는 그룹 D 의 요구사항을 제대로 반영하지 못할 가능성이 다른 그룹들에 비해서 높다. 결과를 살펴보면 다른 그룹들의 결과보다 정확율과 재현율의 수치가 더 많이 낮은 것을 알 수 있다. 이는 제안된 분류기법이 분류언어를 기반으로 분류를 수행하고, 이 분류언어는 각 프로젝트의 요구사항들로부터 유도되기 때문에 도메인의 성격이 다를수록 분류언어의 성격도 차이가 나기 때문이다.

위의 결과를 종합해 보면, 본 논문에서 제안하는 분류언어는 비슷한 도메인일 경우 재사용이 가능하고, 특히 요구사항의 정답 수가 많은 분류속성에 대해서 더욱 재사용 효과가 좋은 것을 알 수 있다. 본 실험에서 분류 언어는 미리 수작업으로 분류된 요구사항 들로부터 유도되었다. 그러므로 요구 사항의 정답 수 가 많은 분류 속성의 분류언어는 보다 많은 요구사항들로부터 유도 되었음을 의미한다. 반대로 요구사항의 정답 수가 적은 분류 속성의 분류언어는 적은 요구사항 집단으로부터 유도되었고, 그래서 개발 된 분류언어의 양이 적고, 좋은 분류 효과를 내지 못했다.

따라서, 제안된 기법에 사용되는 분류 언어는 비슷한 도메인일 경우 재사용 될 수 있으며, 프로젝트가 진행됨에 따라서 보다 많은 분류언어를 개발 할 수 있고, 그럴수록 더 좋은 분류 효과를 낼 수 있다.

5. 결론

본 논문에서는 분류언어 와 유사도 측정을 이용한 품질 요구사항 자동 분류 기법에 대해 제안하였다. 제안된 기법은 2 단계 분류 기법을 통해 요구사항을 반 자동 분류하며 요구사항이 많은 프로젝트의 품질 요구사항 분류 시 수작업의 양을 줄여 주고, 미 분류, 중복 분류 등의 오류를 감소시켜 준다.

제안된 기법의 효용성을 검증하기 위해 우리는 본 기법을 구현하는 분류엔진을 개발하였으며, 이를 통해 15 개의 프로젝트로부터 추출한 366 개의 품질 요구사항을 바탕으로 실험을 수행하였다.

실험결과 평균 70~100%의 재현율을 얻었으며, 요구사항의 수가 많은 분류속성일수록 높은 정확율과 재현율을 얻었다. 또한, 같은 분류언어를 유사한 다른 프로젝트에 적용하여 비슷한 결과를 얻었으며 이를 통해 분류언어를 재사용 할 수 있다는 결과를 얻었다.

본 기법은 요구사항의 수가 많은 프로젝트에 효용성이 있다. 요구사항 자동 분류는 요구사항의 수가 많을 때 사용하는 것이므로, 도메인 별 분류언어를 미리 구축해 놓고 본 기법을 적용한다면 효율적인 요구사항 관리를 수행 할 수 있다.

또한, 본 기법은 통합 요구사항 관리 환경과 연계하여 구현 될 때 보다 높은 효과를 얻을 수 있다. 이에 대한 연구는 향후 연구로 남겨 놓는다.

참고문헌

[1] 김학수, 고영중, 박수용, 서정연 "문서간 유사도 측정을 통한 효율적인 사용자 요구 분석", HCI'99 학술대회 논문집, pp.73-79, 1999.

[2] Maarek, Y., Berry, D. and Kaiser, G., "An Information Retrieval Approach For Automatically Construction Software Libraries", IEEE Transaction On Software Engineering, Vol. 17, No. 8, pp.800-813, August 1991.

[3] Gerard Salton, "Automatic Information Organization and Retrieval", McGraw-Hill, 1968.

[4] Palmer, J. and Liang, Y., "Indexing and clustering of software requirements specifications", Information and decision Technologies, Vol 18, pp. 283-299, 1992.

[5] Joseph A. Goguen and Charlotte Lide "Techniques for Requirements Elicitation," Proceedings of the International Symposium on Requirements Engineering, 1993

[6] Salton, G., "Automatic Information Organization and Retrieval", McGraw-Hill, 1968.

[8] Yaung, Y., "An evaluation of statistical approaches to text categorization", Journal of Information Retrieval ,Vol 1, No. 1/2, pp 67--88, 1999.