비정형 요구사항으로부터 원인-결과 그래프 자동 발생을 위한 문장 의미 모델(Sentence Semantic Model) 설계

장우성○, 정세준, 김영철
홍익대학교 소프트웨어공학연구실
{jang, jeong, bob}@selab.hongik.ac.kr

Design of Sentence Semantic Model for Cause-Effect Graph Automatic Generation from Natural Language Oriented Informal Requirement Specifications

Woo Sung Jang○, Se Jun Jung, R.Young Chul Kim
Software Engineering Lab., Hongik University

요 약

현재 한글 언어학 영역에서는 많은 언어 분석 연구가 수행되었다. 또한 소프트웨어공학의 요구공학 영역에서는 명료한 요구사항 정의와 분석이 필요하고, 비정형화된 요구사항 명세서로부터 테스트 케이스 추출이 매우 중요한 이슈이다. 즉, 자연어 기반의 요구사항 명세서로부터 원인-결과 그래프(Cause-Effect Graph)를 통한 의사 결정 테이블(Decision Table) 기반 테스트케이스(Test Case)를 자동 생성하는 방법이 거의 없다. 이런 문제를 해결하기 위해 '한글 언어 의미 분석 기법'을 '요구공학 영역'에 적용하는 방법이 필요하다. 본 논문은 비정형화된 요구사항으로부터 테스트케이스 생성하는 과정의 중간 단계인 요구사항에서 문장 의미 모델(Sentence Semantic Model)을 자동 생성하는 방법을 제안 한다. 이는 요구사항으로부터 생성된 원인-결과 그래프의 정확성을 검증할 수 있다.

주제어: 테스트케이스, 요구사항 의미 분석, 문장 의미 모델, 원인-결과 그래프

1. 서론

최근 소프트웨어 테스트의 중요성은 계속적으로 증가되는 추세이다. 대기업 다양한 방법으로 테스트를 수행하지만, 국내 중소기업의 경우 개발 시간의 부족, 낮은 테스트 기술 등의 이유로 인해 테스트 환경 조성에 한계가 있어 높은 품질의 테스트가 어려운 실정이다[1]. 그렇기 때문에 요구사항으로부터 테스트 케이스 생성 및 실행을 자동화하여 테스트 시간 및 비용을 감소시킨다면, 좀 더 효율적인 테스트를 수행할 수 있을 것이다. 하지만 요구사항 기반 자동 테스트케이스 생성 방법은 소스코드 기반 자동 테스트케이스 생성 방법에 비해 연구 진척도가 부족한 실정이다. 그 이유 중 하나는 자연어 요구사항 명세의 해석에 대한 어려움 때문이다.

우리는 이러한 어려움의 해결을 위해 모델 기반 변환 방법을 사용하여 요구사항으로부터 테스트케이스를 자동 생성하는 방법을 연구 중이다. 본 논문은 연구의 일부로써, 요구사항으로부터 Sentence Semantic Model을 자동 생성하는 방법을 제안한다. Sentence Semantic Model은 요구사항 문장의 구조와 의미를 효과적으로 가시화하는 모델이다. 요구사항 문장을 가장 작은 절로 나누고, 절들의 관계와 의미를 가시화하고, 같은 의미를 가지는 절을 합친다. Sentence Semantic Model은 요구사항으로부터 원인-결과 그래프 생성 과정을 추적하기 위한 방법이 될 수 있다.

본 논문의 구성은 다음과 같다. 2장은 관련 연구를 언급한다. 3장은 Sentence Semantic Model을 설계한다. 4장은 비정형 요구사항으로부터 Sentence Semantic Model 생성 과정을 언급한다. 5장은 실험 결과를 언급한다. 마지막으로 결론 및 향후 연구를 언급한다.

2. 관련 연구

2.1 원인-결과 그래프

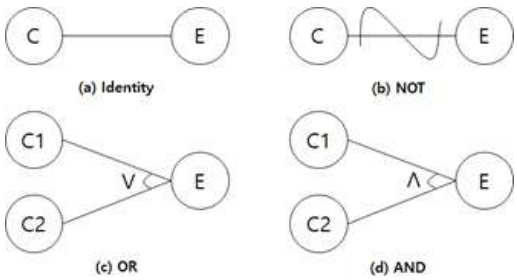
원인-결과 그래프는 요구사항 문장으로부터 원인과 결과를 식별하여 연결한 그래프이다. 원인-결과 그래프 기반의 테스트는 원인이 되는 입력 값이나 서브 함수들의 조합을 통해 체계적으로 테스트 케이스를 추출할 수 있는 방법을 제공해준다. 또한 소프트웨어의 동적 테스트 케이스를 작성하는데 사용된다[11].

원인-결과 그래프의 장점은 테스트 실행 시간과 비용의 감소이다. 원인-결과 그래프는 요구사항에서 소프트웨어의 최대 테스트 영역을 포함하는 최소한의 테스트케이스를 설계할 수 있다.

원인-결과 그래프는 AND, OR, NOT과 같은 논리 연산자를 사용하여 요구사항의 입력 및 출력 조건 간의 논리적 관계를 나타낼 수 있다[12].

원인-결과 그래프의 종류는 아래의 <그림 1>과 같다. C는 "원인(Cause)"을 의미한다. E는 "결과(Effect)"

를 의미한다. "원인"은 시스템 내부 변경 사항과 같은 입력 조건을 의미한다. "결과"는 출력 조건, 시스템의 변환 또는 원인 조합으로 인한 상태를 의미한다[13].

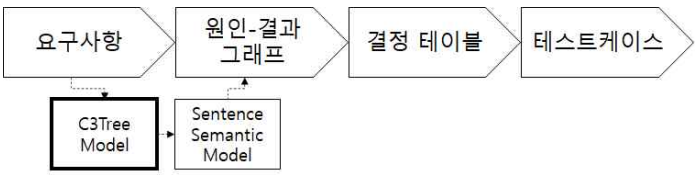


<그림 1> 원인-결과 그래프

2.2 Conditional and Conjunction Clause Tree(C3Tree) Model을 이용한 문장 정보 관리[2]

Gary는 요구사항으로부터 원인-결과 그래프와 결정테이블을 거쳐 테스트케이스 생성하면, 최소한의 테스트케이스로 100% 테스트 커버리지가 가능하다고 설명한다[3]. 하지만 수동적인 방법만을 제안한다.

우리는 <그림2>와 같은 요구사항으로부터 원인-결과 그래프 자동 생성 방법을 연구 중이다. C3Tree Model[2]은 요구사항으로부터 원인-결과 그래프 자동 생성 과정 중의 산출물이다. 요구사항 문장에서 원인절(Condition), 결과절(Result), 접속절(AND, OR)의 정보를 찾아 트리 형태로 가시화한다.

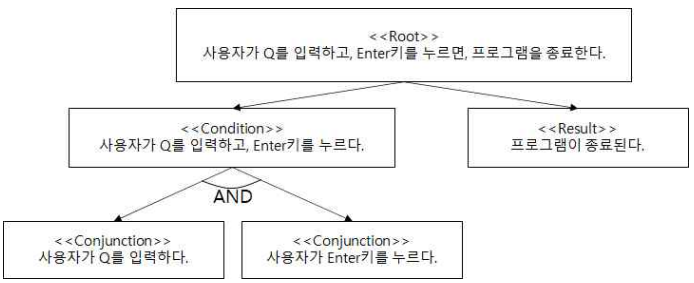


<그림 2> 요구사항으로부터 테스트케이스 자동 생성 방법

<그림 3>은 <표 1>의 요구사항에 대한 C3Tree Model의 예이다. 요구사항 문장(Root)은 원인절(Condition)과 결과절(Result)로 나뉘고, 이 중 원인절(Condition)은 다시 두 개의 접속절(Conjunction)로 나뉜다. 접속절은 AND 관계로 표현된다.

<표 1> 요구사항 예시

사용자가 Q를 입력하고, Enter 키를 누르면, 프로그램을 종료한다.



<그림 3> C3Tree Model의 예

C3Tree Model은 <표 2>와 같이 XMI 코드로 표현된다.

<표 2> C3Tee Model의 XMI code

```
<?xml version = "1.0" encoding = "ISO-8859-1"?>
<ctm:c3tree xmi:version="2.0" xmlns:xmi = "http://www.omg.org/XMI" xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:cem = "http://cem/1.0">
<c3node combinationtype="5" nodetype="1" sentence="사용자가 'Q'키를 입력하고, 'Enter'키를 누르면, 프로그램을 종료한다">
<c3node combinationtype="5" nodetype="2" sentence="사용자가 'Q'키를 입력하고 'Enter'키를 누르다.">
<c3node combinationtype="1" nodetype="7" sentence="사용자가 'Q'키를 입력하다."/>
<c3node combinationtype="3" nodetype="7" sentence="사용자가 'Enter'키를 누르다."/>
</c3node>
<c3node combinationtype="5" nodetype="3" sentence="프로그램을 종료한다."/>
</c3node>
</ctm:c3tree>
```

2.3 한국어의 의미역

요구사항 문장에서 중복되는 의미를 가지는 원인과 결과들을 합치기 위해서는 문장의 의미 분석이 필요하다. Sentence Semantic Model은 의미역을 이용하여 문장에서 중복되는 의미를 가지는 요소들을 파악하고, 합친다.

의미역이란, 술어가 나타내는 의미구조 내에서 참여자(또는 의미 논항)가 가지는 역할을 말하고, 술어가 나타내는 사건과 참여자 사이의 관계를 특정 짓는 개념이다. 의미역 목록의 일부는 <표 3>과 같다.

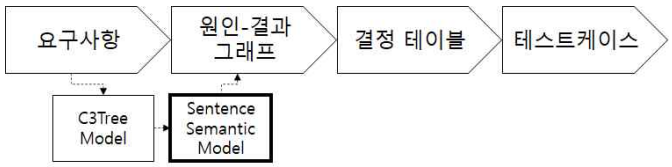
<표 3> 세종전자사전의 의미역의 일부[4]

의미역	설명
행동주역	의도성을 가지고 행위를 일으킬 수 있는 개체에 부여되는 의미역
경험주역	의도를 가지고 어떤 행위를 일으키는 것이 아니라 어떤 행위나 상태를 인식하는 개체 에 부여되는 의미역
동반주역	독립적으로 역할을 하지 못하고 다른 의미역, 즉 행위주나 대상을 보조하여 그것과 같은 역할을 하는 개체에 부여되는 의미역
대상역	의미역 논의에서 가장 중심적인 것으로, 행위나 과정의 영향을 전체적으로 받지만 그 과정을 지배하지는 못하는 개체에 부여되는 의미역

3. Sentence Semantic Model의 설계

본 논문은 모델 기반 변환 방법을 사용하여 요구사항으로부터 테스트케이스를 생성하는 과정 중 하나로써, 요구사항으로부터 Sentence Semantic Model을 자동 생성하는 방법을 제안한다. <그림 4>는 테스트케이스 자동 생성 과정에서 Sentence Semantic Model의 위치이다. Sentence Semantic Model은 C3Tree Model로부터 만들어

지고, 원인-결과 그래프를 그리기 위한 기반이 된다.



<그림 4> 테스트케이스 자동 생성 과정에서 Sentence Semantic Model의 위치

Sentence Semantic Model은 1) C3Tree Model로부터 노드들의 관계 정보를 전달받는다. 2) Exobrain의 인공지능 기반 의미역 분석 방법[5]을 사용하여 말단 노드 내 문장의 의미역을 분석하고, 중복되는 의미를 가지는 말단 노드는 합친다. 3) 분석되고 합쳐진 정보를 가시화한다.

Sentence Semantic Model의 구조는 <표 4>와 같다.

<표 4> Sentence Semantic Model의 구조

이름	속성	설명
clause	scid	인덱스 번호
	value	텍스트 데이터
	arg0	행동주(Agent), 경험주(experiencer)
	arg1	피동주(Patient), 대상(theme)
	arg2	수혜자(Benefactive), instrument, attribute, 결과(end state)
	arg3	기점(Start point), benefactive, instrument, attribute
	arg4	착점(End point)
	argm_adv	부사구
	argm_cau	원인
	argm_cnd	조건
	argm_dir	방향(where to/from)
	argm_dis	담화
	argm_ext	범위, 정도
	argm_loc	장소(Where)
	argm_mnr	방법(How, manner)
	argm_neg	부정
	argm_prd	목적
	argm_tmp	시간(When)
	verb	문장의 핵심 동사
	verb_word	문장의 핵심 동사만 포함한 문장
	lastsyntax	문장의 의미적 순서 상에서 마지막 의미
sentence	value	원본 문장
	connection	문장의 접속절 리스트
	condition	문장의 조건절 리스트
	result	문장의 결과절 리스트
condition	scid	Clause의 고유 인덱스
	value	원본 문장
	connection	조건절 내의 접속절 리스트
result	scid	Clause의 고유 인덱스
	value	원본 문장
	connection	결과절 내의 접속절 리스트
connection	type	AND 또는 OR 또는 LAST 값을 가짐 AND: 다음 접속절과 AND 관계를 가짐 OR: 다음 접속절과 OR 관계를 가짐 LAST: 마지막 접속절
	scid	Clause의 고유 인덱스
	value	원본 문장
	connection	접속절 내의 접속절 리스트
	condition	접속절 내의 조건절
	result	접속절 내의 결과절

clause는 요구사항 문장 내 가장 작은 단위의 절 정보들을 저장한다. condition은 조건절에 해당하는 clause 정보를 저장한다. result는 결과절에 해당하는 clause 정보를 저장한다. connection은 접속절에 해당하는 clause 정보를 저장한다. sentence는 원본 문장을 저장하고, 원본 문장을 구성하는 condition, result, connection 정보를 저장한다.

Sentence Semantic Model은 XMI 코드로 표현된다. XMI 코드의 템플릿은 <표 5>와 같다.

<표 5> Sentence Semantic Model의 XMI 코드 템플릿

```
<?xml version = "1.0" encoding = "ISO-8859-1"?>
<ssm:SentenceSemantic xmi:version="2.0" xmlns:xmi = "http://www.omg.org/XMI" xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:cem = "http://cem/1.0">
<sentence value="">
    <condition value="">
        <connection scid="" type="" value="" />
        <connection scid="" type="" value="" />
    </condition>
    <result value="">
        <connection scid="" type="" value="" />
        <connection scid="" type="" value="" />
    </result>
</sentence>
<sentence value="" >
    <connection scid="" type="" value="" >
        <condition value="" />
        <result value="" />
    </connection>
</sentence>
<clause arg0="" arg1="" arg2="" arg3="" arg4="" argm_adv="" argm_cau="" argm_cnd="" argm_dir="" argm_dis="" argm_ext="" argm_loc="" argm_mnr="" argm_neg="" argm_prd="" argm_prp="" argm_tmp="" scid="" lastsyntax="" value="" verb="" verb_word=""/>
</ssm:SentenceSemantic>
```

4. 비정형 요구사항으로부터 Sentence Semantic Model 생성 과정

비정형 요구사항으로부터 Sentence Semantic Model 생성 과정은 <그림 5>와 같다. 문장 의미 분석기는 문장 분할, 구문 분석, 의미 분석으로 구성된다.

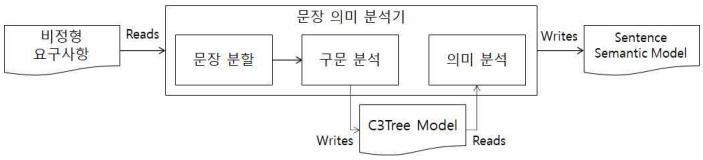
문장 분할은 구문 관계 지식 추출을 위한 코퍼스 정규화 방법의 접속절 분리 방법[6,7], 한국어 조건 연결어미와 분류 방법[8], 대등접속문과 종속 접속문으로 체계화된 접속문의 분류 방법[9]을 사용하여 문장의 접속절과 조건절을 식별한다.

구문 분석은 구문 관계 지식 추출을 위한 코퍼스 정규화 방법[6,7]을 기반으로 문장의 형태소 식별하고, 형태소와 단어 기반의 말뭉치 식별하고, 말뭉치를 재배치 및

변경한다. 결과적으로 문장의 절을 구성하는 격들을 정규화하여 재배치한 후 C3Tree Model을 생성한다.

의미 분석은 AI 기반 문장 구문 의미 순서 분석 방법[10], 한국어 문장에서 의미역 인식 방법[4]을 사용하여 C3Tree Model의 말단 노드들의 의미역을 분석하고, 중복되는 의미의 말단 노드를 합쳐서 Sentence Semantic Model로 구성한다.

각 분석 과정의 이론들을 구현하여 적용할 때, 이론이 잘못된 결과를 도출하는 경우가 존재한다. 이 경우에는 해당 알고리즘을 데이터 사전에 등록하여 예외처리한다. 만약 추가적으로 필요한 분석 방법이 식별되면, 해당 방법 또한 데이터 사전에 등록한다.



<그림 5> Sentence Semantic Model 생성 과정

<표 7>은 <표 6>의 요구사항에 대한 Sentence Semantic Model 생성 예이다. Depth1은 원본 문장이다. Depth2는 첫 번째로 나뉜 문장이다. Depth1의 원인절, 결과절이 Depth2에서 2개의 문장으로 표현되었다. Depth2는 두 번째로 나뉜 문장이다. Depth2의 접속절이 Depth3에서 2개의 문장으로 표현되었다. 나뉘는 절이 많을수록 Depth의 길이가 증가된다. 절을 가장 작은 단위로 나누었다면, 마지막으로 절들의 의미역(행동주, 대상 등)을 분석한다.

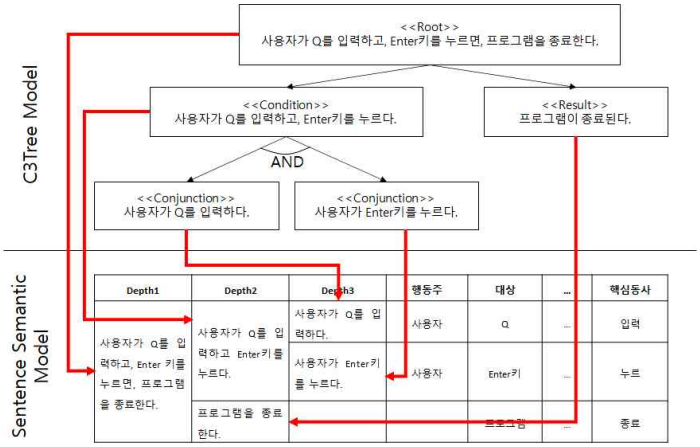
<표 6> 요구사항 예시

사용자가 Q를 입력하고, Enter 키를 누르면, 프로그램을 종료한다.

<표 7> 가시화된 Sentence Semantic Model의 예

Depth1	Depth2	Depth3	행동주	대상	...	핵심동사
사용자가 Q를 입력하고, Enter 키를 누르면, 프로그램을 종료한다.	사용자가 Q를 입력하고 Enter키를 누른다.	사용자가 Q를 입력하다.	사용자	Q	...	입력
		사용자가 Enter키를 누른다.	사용자	Enter 키	...	누르
	프로그램을 종료한다.			프로그램	...	종료

Sentence Semantic Model은 C3Tree Model 내 노드들의 의미역을 분석하고 통합하여 생성된다. C3Tree Model의 노드와 Sentence Semantic Model의 요소의 연관 관계는 <그림 6>과 같다.



<그림 6> C3Tree Model과 Sentence Semantic Model의 연관성

Sentence Semantic Model의 XMI 코드는 <표 8>과 같다.

<표 8> Sentence Semantic Model의 XMI code

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<ssm:SentenceSemantic xmi:version="2.0" xmlns:xmi = "http://www.omg.org/XMI" xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:cem = "http://cem/1.0">
    <sentence ssid="0" value="사용자가 Q를 입력하고, Enter키를 누르면, 프로그램을 종료한다">
        <condition value="사용자가 Q를 입력하고 Enter키를 누르다.">
            <connection scid="0" slid="0" ssid="0" type="and" value="사용자가 Q를 입력하다."/>
            <connection scid="1" slid="1" ssid="0" type="last" value="사용자가 Enter키를 누르다."/>
        </condition>
        <result scid="2" slid="2" ssid="0" value="프로그램을 종료한다."/>
    </sentence>
    <clause arg0="사용자" arg1="Q" arg2="" arg3="" arg4="" argm_adv="" argm_cau="" argm_cnd="" argm_dir="" argm_dis="" argm_ext="" argm_loc="" argm_mnr="" argm_neg="" argm_prd="" argm_prp="" argm_tmp="" lastsyntax="입력하다." scid="0" value="사용자가 Q를 입력하다." verb="입력" verb_word="입력하다."/>
    <clause arg0="사용자" arg1="Enter키" arg2="" arg3="" arg4="" argm_adv="" argm_cau="" argm_cnd="" argm_dir="" argm_dis="" argm_ext="" argm_loc="" argm_mnr="" argm_neg="" argm_prd="" argm_prp="" argm_tmp="" lastsyntax="누르다" scid="1" value="사용자가 Enter키를 누르다." verb="누르" verb_word="누르다."/>
    <clause arg0="" arg1="프로그램" arg2="" arg3="" arg4="" argm_adv="" argm_cau="" argm_cnd="" argm_dir="" argm_dis="" argm_ext="" argm_loc="" argm_mnr="" argm_neg="" argm_prd="" argm_prp="" argm_tmp="" lastsyntax="종료한다." scid="2" value="프로그램을 종료한다." verb="종료" verb_word="종료한다."/>
</ssm:SentenceSemantic>
```

Sentence Semantic Model에서 조건절은 condition 태그이고, 결과절은 result 태그이고, 접속절은 connection 태그이다. 원인-결과 그래프에서 condition 태그는 원인으로, result 태그는 결과로, connection 태그는 원인의 조합으로 표현될 수 있다.

5. 실험 결과

아래의 <표 9>는 문장 의미 분석기에 입력된 기능적 요구사항의 리스트의 일부이다. 총 78개의 요구사항을 입력하였다.

<표 9> 입력된 자연어 요구사항의 일부

1. 사용자는 메모에 영어/한글/한자 텍스트를 직접 입력 또는 복사/붙여넣기 할 수 있다.
2. 메모에 작성할 수 있는 문자는 기본 문자, 숫자, 특수 문자를 모두 포함한다.
3. 사용자는 메모에 테이블을 삽입할 수 있다.
...

문자 의미 분석기를 이용하여 실제 업무에 사용되는 기능적 요구사항을 Sentence Semantic Model로 자동 변환한 결과는 아래의 <표 10>과 같다. 약 96.1%의 정확도로 요구사항이 Sentence Semantic Model로 자동 변환되었다.

<표 10> 요구사항으로부터 Sentence Semantic Model 생성 결과

Category	Count
모든 요구사항	78
올바르게 분석된 요구사항	75
올바르지 않게 분석된 요구사항	3

Sentence Semantic Model에는 절 단위로 쪼개지고, 의미역 분석이 완료된 문장들이 포함된다. 각 문장에 원인과 결과가 포함되었는지를 분석한 결과는 아래의 <표 11>과 같다. 전체 기능적 요구사항 문장의 약 31.9%가 원인과 결과를 포함하였다.

<표 11> 원인과 결과가 포함된 Sentence Semantic Model의 문장들

Category	Count
전체 문장	75
원인과 결과를 포함한 문장	23
원인과 결과를 포함하지 않는 문장	52

5. 결론 및 향후 연구

본 논문은 비정형 요구사항으로부터 테스트케이스를 생성하기 위한 중간과정으로써, 요구사항으로부터 Sentence Semantic Model을 자동 생성하는 방법을 제안한다. 정형적 방법을 사용하여 요구사항 문장을 정제하고, 인공지능 기반 분석 방법을 사용하여 정제된 요구사항 문장에서 의미를 찾는다. Sentence Semantic Model은 원본 문장과 의미 분석된 절들 간의 추적 정보를 저장하며, 원인-결과 그래프를 만들기 위한 기초 정보가 될 수 있다.

향후 연구로써, Sentence Semantic Model로부터 원인-결과 그래프를 자동 생성하기 위한 방법과 원인 결과가 포함되지 않은 문장의 테스트케이스 생성 방법을 연구할 예정이다.

감사의 글

본 논문은 2019년도 산업통상자원부의 '창의산업융합 특성화 인재양성사업'(과제번호 N0000717)과 2017년도정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(NRF-2017R1D1A3B03035421)

참고문헌

[1] 권오승, 홍사능, "로그 기반 효과적 반복 테스트", 한국경영정보학회 학술대회, pp.685-690, 2009.

[2] 장우성, 박보경, 김영철, "비정형 요구사항으로부터 생성된 원인-결과 그래프의 검증을 위한 C3Tree 모델 설계", 한국정보과학회 국방소프트웨어연구회 2019 추계워크샵, 2019.

[3] Gary E.Mogyorodi, B. Math. "Requirements-Based Testing-Cause-Effect Graphing". Software Testing Services 2005.

[4] 박철우, 김종명, "한국어 용언 사전 기술을 위한 의미역 설정의 기본 문제들, 서울대학교 언어교육원, Vol.41, No.3, pp.543-567, 2005.

[5] Exobrain, http://exobrain.kr

[6] 조정미, 조영환, 김길창, "구문 관계 지식 추출을 위한 코퍼스 정규화에 대한 연구", 한국정보과학회 언어공학연구회 학술발표 논문집, pp.207-215, 1996.

[7] 조정미, 김길창, "구문 관계 정보 추출을 위한 말뭉치의 정규화에 대한 연구", 한국인지과학회, vol.7, no.2, pp.39-56, 1996.

[8] 하정문, "한국어 조건 연결어미와 중국어 조건 관련사어의 대조 연구", 경희대학교 대학원 국문학과, 2007.

[9] 김기성, "현대몽골어와 한국어의 접속문 비교연구", 한국몽골학회, Vol.27, No.0, pp.151-186, 2009.

[10] 임수종, 권민정, 김준수, 김현기, "ExoBrain을 위한 한국어 의미역 가이드라인 및 말뭉치 구축", 제27회 한글 및 한국어 정보처리 학술대회, pp.250-254, 2015.

[11] 조선영, 조상훈, "차량 제어기 개발을 위한 요구사항 기반 검증기법 연구", 한국자동차공학회 추계학술대회 및 전시회, Vol.1, pp.1484-1490, 2012.

[12] Mayers, Glenford J. (1979). The Art of Software Testing. John Wiley & Sons. ISBN 0-471-04328-1.

[13] Cause-Effect Graph Technique, http://www.softwaretestingclass.com/what-is-cause-and-effect-graph-testing-technique/