Requirement Diagram 를 자연어로 작성하기 위한 Translation Database Design

이혜련*, 최경희*, 정기현**
*아주대학교 정보통신전문대학원
**아주대학교 전자공학부
e-mail : cocom12@netsgo.com

The translation database design for being written in the Natural Language using the Requirement Diagram

Hye-Ryun Lee*, Kyung-Hee Choi*, Ki-Hyun Jung**
*Graduate School of Information and Communication, Ajou University
**Division of Electronics Enginnering, Ajou Unniversity

요 약

Software testing 은 소프트웨어 개발 과정 중에 1/3 을 차지 할 만큼 중요한 부분 중 하나이다. Software testing 는 Requirement 작성이 제대로 이루어져야만이 제대로 testing 을 할 수 있고, 그에 따라 정확한 결과를 얻을 수 있다. 그 만큼 Requirement 작성이 중요시 되고 있지만, 수동적으로 기술자에 의해서 작성되는 Requirement 에는 많은 문제점을 안고 있다. 본 논문에서는 Requirement 를 Graph 하게 표현한 방법을 소개하고, 표현된 방식을 이용하여 다시 자연어로 표현할 수 있도록 Database 를 설계하는 방식을 제안한다. 그 결과로 Design 된 패턴들을 이용하여 Requirement 자연어로 기술한다. 이를 통하여 Requirment 기술 방식을 통일화 시킬 수 있으며, 기술자간에 의사소통을 원할하게 수행할 수 있으며, Software testing 의 중요 기반으로 제공할 수 있다.

1. 서론

최근 전자기기의 작업들을 보게 되면, 복잡하면서도 지능적인 작업을 점차 많이 수행하고 있다. 이에 따라 전자기기에 존재하는 내장 시스템의 기능적인 Requirement 가 강화되고 있는 상황이다[1,3].

따라서 Requirement 를 하나하나 작성하는데 있어서 System 을 잘 설계할 수 있도록 명확하면서도 충분하게 기술되어야한다[4].

그러나, Requirement 작성법은 자동화가 이루어지지 않아 지금까지 대부분 사람에 의한 수동적으로 기술됨으로써 기술하는 방식이 굉장히 다양하며, 다양한 기술 방식 때문에 Requirement 를 기반으로 하는 software testing 하는데 있어서 많은 어려움을 가지고 있다[2].

본 논문에서는 Requirement 작성에 있어서 Graph 하게 표현하면서 규칙성을 가지게 하고, 이 규칙성에 따라 Requirement 작성을 자동적으로 생성할 수 있도록 하는 Database 설계하는 방식을 제안한다.

본 논문의 구성은 다음과 같다. 3 장은 Requirement 을 Graph 화로 표현한다. 4,5 장에서는 Requirement 자연어 작성을 자동적으로 생성할 수 기반이 되는 Database 를 설계하는 방식에 대하여 제안하고, 6 장에서는 결론을 맺는다.

본 연구는 정보통신부 및 정보통신연구진흥원의 해외교수요원 초빙사업의 연구결과로 수행되었음

2. Requirement 자연어 작성

현재 많은 기업들이 Requirement 를 작성하는 방법으로 자연어로 기술하는 것을 택하고 있다[2].

자연어로 기술하는데 있어서 2 개의 Description 과 4 개의 Attributes 를 구성요소로 가지게 된다.

1) 2 개의 Description
- Short Description : Requirement 를 명확하면서도 간결하게 하나의 완결된 단일 문장으로 기술한다.
- Detail Description : Short Description 에 표현하지 못하는 부분을 서술한다. 즉 수식 및 그래프와 같은 것을 포함하여 Requirement 에 대하여 자세히 설명한다.

2) 4 개의 Attribute
- Key-word : Requirement 에 대한 핵심적인 단어로, Requirement 를 분석하거나, 검색할 때 사용한다.
- Priority : Requirement 의 중요도를 High/Medium/Low 로 구분하여, 중요도가 높으면 높을수록 반드시 수행하거나, 특별한 다른 작업을 수행할 수 있도록 한다.
- Type : Requirement 의 기술 내용에 따라서 종류를 분류 한다. Func/Non-Fun Type 로 2 가지

가 있다.

- Status : Requirement 의 현재 상태를 나타내는 요소이다. 즉 구현이 됐는지 여부를 묻는다. Proposed/Ongoing/Finished - 3 가지로 구분짓는다.

Requirement 7.2.2	
Short Description	난방기동 제어 수행중 냉각 수온 하강시 Blower 및 Mode는 현재값으로 제어할 것. 여기서 Guard okWarmMoving이 난방기동제어를 수행하느냐의 여부를 결정한다. Guard Warm_moving는 난방기동 제어 진입 여부를 결정한다.
Key-word	okWarmMoving
Priority	Medium
Type	Func
Status	Proposed

[표 1. Requirement Example]

3. Requirement Diagram 작성

Requirement Diagram 작성 방법은 Reqirement 를 자연어로 표현할 것을 그래픽하게 표현한 것이다.

Requirement 를 Diagram 으로 기술하는데 있어서 총 2 개의 Block Type 과 Link, port 로만 기술할 수 있다.

1) Block

Entity Block	Requirement Input/Output 을 나타내기 위한 Block 으로, Entity Block 에는 반드시 Name 이 정의되어야 한다. 종류에는 Device, Memory, Timer, Guard, Connector Block 등이 있다.
Operation Block	Reqirement 에서 수행하는 일을 표현하는 Block 으로, Input Entity Block 을 입력으로 받아 Operation Block 에서 어떠한 일을 수행한 후, Output Entity Block 으로 출력한다. 종류에는 Do, AND, OR, Loop, Table, Change Block 등이 있다.

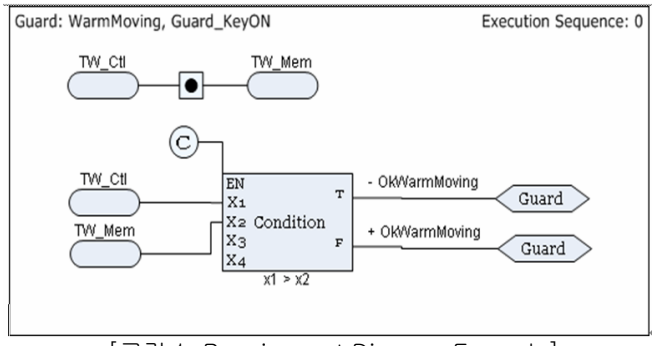
2) Link

Link 은 Entity Block 과 Operation Block 을 연결하거나, Input Entity Block 와 Output Entity Block 간에 연결시 사용 되어지는 직선이다.

Link 에는 Entity Block 이 value 를 가지거나, Operation Block 이 어떠한 특정한 작업을 수행하여 나온 value 가 있을 경우가 존재함으로, 직선 위에 value 를 입력할 수 있다. value 앞에는 value 의 Type 을 나타낼 수 있는 prefix 가 올 수 있다. 그 종류에는 Trap, State, Numeric 가 있다.

3) Port

Block 과 link 을 연결할 수 있는 고리 연결하는 것이 Port 이다. Operation Block 에는 Input Port 와 Output Port 2 가지를 가지고, Entity Block 에서는 Input 일 경우는 Output port 만, Output 일 경우는 Inprt port 만을 가진다.



[그림 1. Requirement Diagram Example]

[그림 1]은 [표 1]에서 예제로 사용하였던 Requirement 를 Diagram 으로 표현한 예로 Entity Block 은 Memory, Constant, Guard Block 이고, Operation Block 은 Condition Block 과 Do Block 이다.

Requirement 를 Diagram 으로 기술함으로써 정형화된 표현 방식으로 이용하여 기술하기 때문에, Requirement 의 내용을 분석하는데 훨씬 수월하고, Embedd system 의 특성상, 입력의 조건 변화에 따른 출력이 어떻게 다른지 한 눈에 파악할 수 있다라는 장점을 가지고 있다.

4. Diagram 구성요소 Database Design

Diagram 을 자연어로 표현하는데 있어서, 2 가지 관점에 초점을 둔다. 첫째로, 어떠한 Block 이 올 경우 어떻게 해석 할 것인지, 두 번째로는 Block 과 Block 사이에 연결된 Link 에 value 가 존재할 경우는 어떻게 해석할 것인지이다.

2 가지 관점에서 초점을 맞추어 Database 를 구축하는데 있어서 규칙을 하나씩 나열하는데, 이 규칙를 BID 의 Number 로 사용한다.

1) Entity Block – input 일 경우

- BID Rule

01 : Link 에 value 가 없을 경우
02 : Link 에 value 가 존재할 경우
03 : Link 에 value 와 prefix 이 존재 할 경우

- Database

Name	Entity block – memory block	
Notation	BID	Database
name	InMem01	{Index}) 변수 {Name} 값
name value	InMem02	{Index}) 변수 {Name} 값이 {value}
name T value	InMem03	{Index}) 변수 {Name} 값으로 {value} (이)가 발생

2) Entity Block – output 일 경우

- BID Rule

01 : Link 에 value 가 없을 경우 02 : Link 에 value 가 존재할 경우 03 : Link 에 value 와 prefix 이 존재 할 경우

- Database

Name	Entity block – memory block	
Notation	BID	Database
name	OutMem01	{Index}) 변수 {Name} 값으로 된다.
value name	OutMem02	{Index}) 변수 {Name} 값은 {value} (이)가 된다.
T value name	OutMem03	{Index} 변수 {Name} 값에 {value} (이)가 발생된다.

3) Operation Block 일 경우

- BID rule

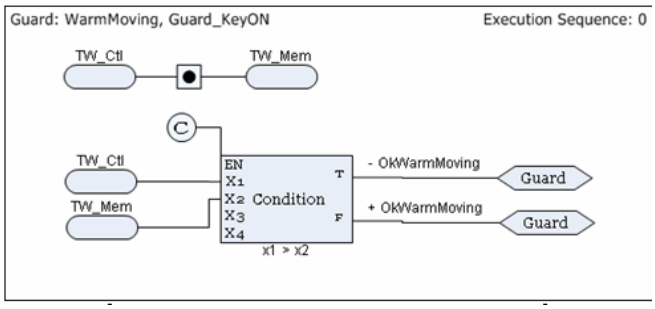
01 : output port 에 연결된 Block 이 Entity Block 일 경우 02 : output port 에 연결된 Block 이 Operation Block 일 경우 03 : output port 에 연결된 Block 이 Operation Block 이면서 Link 에 value 가 존재할 경우 04 : output port 에 연결된 Block 이 Entity Block 이고, input port 로 연결된 Block 이 Entity Block 일 경우

- Database

Name	Do block	
Notation	BID	Database
name value name	ODo01	{Index}) 다음 {Level}.{PortCnt#in#}(이)가 발생하면 {Level}.{PortCnt#out#}(을)를 수행한다.
OP	ODo02	{Index}) {Level}.{PortCnt#in#}(을)를 발생.
value OP	ODo03	{Index}) 다음 {Level}.{PortCnt#in#}(이)가 발생하면 {value}(을)를 출력한다.
name	ODo04	{Index}) 다음 {Level}.{PortCnt#in#} 되면, {Level}.{PortCnt#out#}

5. Requirement Translation 작성

Diagram 에서 그림을 읽어가면서 규칙과 똑같은 그림을 만나게 되면, 규칙에 적합한 BID 를 검색하여 그에 해당 하는 Database 를 그대로 사용하고, {Name} , {value}와 같은 keyword 에 값을 채워서, 자연어로 문장을 완성한다.



[그림 2. Requirement Diagram Example]

- Translation 작성

* 이 Requirement 는 다음의 Guard 가 만족될때 활성화 된다. - WarmMoving, Guard_KeyON 1. 다음 Constant 입력되면, 변수 TW_Ctl 값된 것과 변수 TW_Mem 값된 것에 따라 조건식 : x1 > x2 (을)를 만족하였을 경우, Guards 상태에서 OkWarmMoving 를 해제한다. 반대로, 아닐 경우 Guards 상태에서 OkWarmMoving 를 설정한다. 2. 다음 변수 TW_Ctl 값되면, 변수 TW_Mem 값으로 된다.

[표 2.Requirement Diagram Translation]

6. 결론 및 향후 계획

이 논문에서 Requirement 를 자연어로 자동적으로 생성할 수 있도록 하기 위한 Database 를 설계하는 방식에 대하여 제안하였다. 제안한 Database 를 이용하여 Requirement Diagram 을 Translation 한 결과, Requirement 자연어를 작성할 수 있었다.

향후 연구되어야 할 부분은 이 설계된 Database 를 이용하여 자동으로 생성할 수 있는 software 개발이 이루어져야 한다. 그리고 software 개발시 사용되어진 알고리즘 에 대한 연구도 함께 이루어져야 할 것이다.

참고문헌

[1] 박현상, 최경희, 정기현, “An Implementation of embedded system software requirement management tool” , 2007

[2] Teleogic DOORS – www.telelogic.com/products/doors

[3] S. Faulk, J. Brackett, P. Ward and J. Kirby, “The CoRE method for realtime Requirements”, IEEE software, page 22~33, September 1992.

[4] J.M. Atlee and J. Gannon, “State-based model checking of event-driven system requirements”, IEEE Transaction on software Engineering, page 24~40, January 1993.