UML 모델의 지식화 방안

김일석, 양성봉
연세대학교 컴퓨터과학과

A method for constructing Knowledge of S/W Models in UML

Il-Suk Kim, Sung-Bong Yang
Dept. of Computer Science, Yonsei Univ.

요 약

컴퓨터가 인식할 수 있는 지식의 표현방식에 대한 연구는 오래전부터 인공지능 분야에서 이루어져왔다. 현재는 대규모의 지식베이스를 구축하고 공유하기 위한 Knowledge Sharing Effort(KSE) 프로젝트가 진행중인데, 여기에서는 도메인별로 어휘와 개념을 체계적으로 정의하여 Ontology로 구축하고 있다. 이 논문에서는 객체지향 소프트웨어 개발 방법론에서 모델링언어로서 널리 받아들여지고 있는 UML을, KSE에서 표준으로 제시하고 있는 컴퓨터가 인식할 수 있는 지식의 형태인 Knowledge Interchange Format 형태로 변환하는 방법을 제시하고 이를 통해 추론과 같은 보다 유연한 지식의 처리가 가능함을 보인다.

1. 서론

Unified Modeling Language(UML)는 객체지향 개발 방법론에 있어서 가장 영향력 있었던 BOOCH, OMT, OOSE 방법론을 통합한 모델링 수단으로서 제안된 이후 현재 S/W 공학 전반에 걸쳐 광범위하게 받아들여지고 있다. UML은 S/W 개발 프로세스 전반에 걸쳐 객체지향 S/W의 여러가지 중요한 측면을 모두 표현할 수 있는 개념과 표기법을 제공하고 있다.[6][9][10][11]

한편, AI 분야에서 연구된 지식표현에 관한 연구성과를 바탕으로 컴퓨터에서 인식가능한 지식의 표현법이 연구되었으며 이 성과를 바탕으로한 지식공유노력(Knowledge Sharing Effort, KSE)이 DARPA의 후원아래 이루어지고 있다. 이 노력의 성과로, 여러 분야에 대한 지식단위(온톨로지, Ontology)들이 지식교환포맷(Knowledge Interchange Format, KIF)으로써 표현되어 구축, 공유되고 있다.[7][8][1]

이 논문은 UML에 의해 표기된 S/W 개발지식들이 자동적으로 KIF 형태로 변환될 수 있으며, 그것은 컴퓨터에 의해 인식가능한 지식으로 간주되어 지식관리 시스템에 의해 관리되고, 공유될 수 있음을 보인다. 또한, 이러한 지식은 S/W간에 기본적인 Ontology에 대한 합의가 있다면 컴퓨터에서 인식되어 추론과 같은 보다 고차원적인 질의와 처리가 가능함을 보인다.[5]

2. 관련연구

인공지능 분야의 연구성과를 소프트웨어공학에 응용하려는 노력은 과거로부터 많이 있어왔다. 특히, 요구사항의 분석에 있어 얻어진 지식을 형식화된 언어로써 표현하려는 노력은 1978년 Balzer에 의해 이루어졌으며 그는 프로세스 지향 스펙언어를 제시하고, 비형식적인 설명을 툴을 사용하여 보완하고자 했다. 1985년에는 Borgida가 소프트웨어 요구사항에 대한 기술을 이러한 지식표현에 의해 기술하려고 했다.[7][8]

일반적인 지식표현의 연구는 1990년대에 이르러 컴퓨터에서 지식을 인식하고 처리할 수 있는 수준까지 이루어졌으며, 현재 분야별로 체계화된 지식을 구축하는 작업이 DARPA에 의해 지원되어 진행되고 있다. 이 연구에서는 분야별로 Ontology라는 단위를 두어 지식을 체계적으로 정의하도록 하고 있으며 이를 구현한 지식베이스시스템간의 지식교환을 위해 표준으로 KIF를 제시하고 있다. 현재 KIF는 소프트웨어 에이전트간의 정보교환에 있어서 Standard Generalized Markup language(SGML)와 함께 권고되고 있다.[1]

객체지향 개발방법론의 핵심적인 BOOCH, OMT, OOSE 방법론의 모델 표기법을 통합한 UML은 1997년 1.1버전이 발표된 이후 현재 OMG에서 이의 향상을 위한 재검토작업이 진행중이다. 이 작업의 주요 관심중의 하나는 UML의 표준화된 stream 버전인데, 이는 CASE TOOL간의 자료 호환성을 위해 추진되고 있는 작업으로서 현재 IBM, Unisys등에서 eXtensible Markup Language(XML)에 기반한 제안이 만들어져 있으며 이 외에 일본 게이오대학의 Junichi Suzuki와 Yoshikazu Yamamoto에 의해 XML 기반의 UML 변환포맷인 UML eXchange Format(UXF)이 연구되어있다. 이와같은 작업들은 주로 UML 문서에대한 툴간의 호환성을 제고하는데 초점을 맞추고 작업이 진행되고 있다 [2][4]

이 논문에서는 UML 문서에 표현된 지식이 KIF 표준을 따르는 지식베이스관리시스템에서 관리되고 인식되어 처리될 수 있음을 보이는 것에 관심을 두고 있다. 따라서 뒤에서는 지식기반 시스템에서 어떻게 지식을 처리할 수 있는지를 설명하고, 그렇게 하기위해 UML 문서를 KIF형태로 변환하기위한 아이디어를 제시하고자 한다.

3. KSE와 KIF

3.1 KSE

KSE는 지식베이스관리시스템과 지식베이스의 공유와 재사용을 위한 설비에 대한 협의를 이끌어내기 위한 컨소시엄이다. 이 컨소시엄의 목표는 참가자들의 단독으로 동작하는 시스템보다 더 크고 폭넓은 기능을 지닌 시스템이 가능하도록 해주는 기술과 인프라스트럭처를 정의, 개발, 데스트하는 것이다. 이 노력의 결과물로는 (1)public-domain specification과 지원 기술의 구현, (2) 보고서 논문, 기술논문, (3)재사용가능한 개념증명형태의 공용 예제 라이브러리 등이다. [1]

KIF는 KSE에서 제시하는 Computer-oriented language로서 서로 다른종류의 프로그램간에 지식을 교환하기 위한 언어이다. 이 언어는 선언적인 의미구조를 가지기 때문에 표현식 안에 들어있는 수식의 의미를 그것을 해석하기 위한 인터프리터를 쓰지 않고도 이해할 수 있다[5] KSE에서 지식의 공유와 재사용을 위해 개념과 어휘들을 체계적으로 잘 정리한 덩어리를 Ontology라고 부르는데, Ontology는 도메인별로 하나의 어휘체계를 만들고 있다.[3] Ontology의 맨 밑바닥에 있는 기본적인 지식의 의미에 대한 소프트웨이간의 합의가 이루어진다면, 그 합의에 기반하여 정의된 모든 지식은 추론등의 연산이 가능하다.

3.2. KIF에서 지식의 표현[5]

지식을 표현하는 방법은 KIF외에도 많이 있지만, KIF는 지식을 다른 시스템으로 전달하기 위한 외부포맷으로서 다른 지식베이스관리 시스템의 기능들을 포괄하는 폭넓은 표현능력을 제공한다. 또한, 여러 지식베이스관리 시스템간에 지식을 공유할 수 있는 포맷이므로 직접 KIF로 구현된 지식은, KIF 표준을 따르는 수준에 따라 다르지만, 모든 지식베이스관리시스템에서 관리가 가능하다

KIF에서는 지식을 표현하는 최소 단위로써 form을 두고 있으며, 각각의 form은 object, function, relation을 정의함으로써 하나하나의 지식을 표현하게 된다 예를 들어

(1) (human John)
(2) (human Mary)
(3) (defrelation die (?h) := (=>(human ?h) (die ?h))
(3') (die John)
(3'') (die Mary)
(4) (material moon stilton)

위와 같은 KIF 지식이 있다고 하자. (1)은 "John과 human"이라는 relation이 정의되어있다. 즉, John이 human이라는 뜻이 된다. 또한, (2)에서 human이라는 relation은 Mary에 대해서도 정의되어있는데, 뜻은 역시 Mary도 human이라는 의미가 된다. (3)에서, defrelation 명령을 통해 die라는 릴레이션을 정의하는데, ?h는 변수로서 뒤의 항에서 (human ?h)의 항이 참이 되는 object에 대해 (die ?h)도 참이 된다는 규칙을 정의하고 있다. 이것은 결국 (1) (2)가 정의되어있는 상태에서는 (3')(3'')라는 지식을 하나하나 정의한 것과 같은 기능이 된다 (4)는 세 단어로 된 material이라는 relation을 표현하고 있다.

(5) (believe John '(material moon stilton))
(6) (=> (believe John ?p) (believe Mary ?p))
(6') (believe Mary '(material moon stilton))

(5)에서는 John이 (material moon stilton)을 believe한다는 관계를 표현하고 있다 이와 같이 릴레이션의 각 항으로 단어가 아닌 식이 올 수도 있으며 cons 기호(')를 붙여 식을 평가하지 않고 있는 그대로 관계를 줄 수 있으므로써 지식을 메타레벨에서 다룰 수 있다 그래서, (6)에서와 같은 => 연산자에 의한 식에서, John이 believe하는 것은 Mary도 believe한다는 관계가 정의되면 (5)가 정의되어있는 상태에서 ?p라는 변수가 (material moon stilton)이라는 표현식을 통째로 받아서, (6')과 같은 지식이 자동으로 유추된다

이와같은 지식관리시스템에서는 모든 시스템들이 합의하는 기본적인 어휘의 집합이 필요하게 되며 이런 집합을 도메인별로 분류한 것이 Ontology이다. [3] 현재 제공되고 있는 Ontology로는 가장 기본적인 것으로 수학적인 개념들을 표현하기 위한 Set, Sequence, Numbers and arithmetic, Relations, KIF Meta 등이 있다. 그 외에 현재 구현된 Ontology 들은 [4]에서 참고할 수 있다. 우리는 먼저 UML 표기자체의 의미에 대한 공통적인 지식들을 일종의 UML 메타지식의 입장에서 하나의 Ontology로서 정리할 필요가 있다. 그 기반 위에서 UML로 표기된 각 시스템 개발과정에서의 모델링 자료들을 시스템별로 별도의 Ontology로서 구축하고 정리할 수 있을것으로 보인다.

4. UML로부터의 지식의 추출

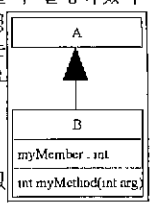
4.1. Unified Modeling Language(UML)[6]

1996년 하반기에 객체지향 개발 방법론을 이끄는 세가지 방법론인 BOOCH, OMT, OOSE 방법론을 통합한 모델링 수단으로서 UML이 제시되었다. 이 모델링기법은 이후 OMG에 제출되어 표준적인 객체 모델링 수단으로 자리잡게 되었다.

UML에서는 객체모델링을 위해 정적, 동적, 구현의 관점에서 Class diagram, Usecase diagram, Sequence diagram등의 총 10개의 다이어그램들을 정의하고 있다

4.2 지식의 추출

UML 다이어그램중 가장 보편적으로 사용되는 것은 class 다이어그램이다. 여기서는 class 다이어그램의 예를 들어 설명하겠다 [그림1]과 같은 UML 다이어그램이 있다고 가정하자. [그림1] 속에서 파악할 수 있는 의미 있는 소프트웨이 지식원소들은 다음과 같이 간단히 요약될 수 있다.



(1) B는 클래스이다.
(2) B는 A의 후손이다
(3) B에서 추가된 속성으로는 myMember가 있다.
(4) B에 추가된 메쏘드로는 myMethod가 있다. [그림 1]

이와같은 정보는 [그림1]로부터 쉽게 추출해낼 수 있다. UML 클래스 다이어그램의 모든 클래스에 대해 같은 추출과정을 거치면 기계적으로 이와 같은 소프트웨어 지식을 뽑아낼 수 있다.

결과적으로, UML 다이어그램으로 표현된 정보들은 이와같이 간단한 몇 개의 정보로서 추출될 수 있음을 알 수 있다 따라서 이렇게 추출된 지식 원소들을 KIF와 같은 지식표현의 한 형태로 변환해주면 기존의 지식베이스관리시스템에서 UML을 유기적으로 관리할 수 있게 되는 것이다.

4.3. UML 모델의 메타널리지(Meta Knowledge)

KIF에서 S/W 지식을 관리하기 위해 다루어야 할 지식은 크게 두 가지로 나뉠 수 있는데, 모든 S/W 시스템에 대한 지식에 공통되는 메타지식과, 개발하고 있는 각각의 S/W시스템을 기술하는 시스템 지식으로 크게 나뉜다 이들은 각각이 Ontology의 형태로서 지식베이스에 저장되고 관리될 수 있다.

여기서 메타지식은 UML다이어그램을 KIF에서 다루기 위해 필요한 UML에 관련한 Ontology라고 볼 수 있다. 여기서는 간단하게 클래스다이어그램상의 하나의 클래스를 기술하기 위해 작성되어야 할 관계들의 목록을 제시하겠다.

(1) IsClass
(2) HasMember
(3) HasMethod

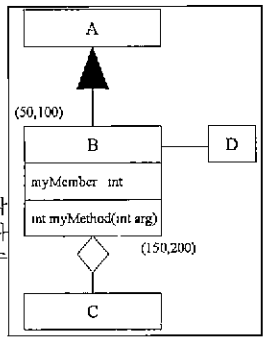
(4) InheritFrom
(5) Associate
(6) Aggregate

위에서, 어떤 object가 클래스라는 것을 정의하기 위해 IsClass 를, 클래스가 가지고 있는 멤버를 기술하기 위해 HasMember를, 클래스에서 정의한 메쏘드를 기술하기 위해 HasMethod 관계가 필요하다.

4.4 시스템 지식(System Knowledge)

5.1에서 제시한 메타지식을 기반으로 어떻게 시스템지식이 구축될 수 있는지를 보이겠다. 먼저, [그림2]와 같은 클래스다이어그램이 있다고 하자 이 다이어그램은 특히 클래스 B에 집중하여 그렸고, 나머지 클래스들은 B와 관련한 만큼만 간략하게 나타냈다. 그림을 보면, 클래스 B는 클래스 A에서 계승받았으며 클래스 D와는 관계(association)가 있다. 또한 클래스 B는 클래스 C를 aggregation으로 포함하고 있으며 정수형 멤버 myMember를 가지고 있고, 정수 arg를 형식인수로 받아 정수를 리턴하는 myMethod라는 멤버함수를 가지고 있다. 이와 같은 정보를 이제 앞의 메타지식을 이용해 정의해 보면 다음과 같이 된다

(1) (IsClass B)
(2) (HasMember B myMember)
(3) (HasMethod B myMethod)
(4) (InheritFrom B A)
(5) (Associate B D)
(6) (Aggregate B C)



[그림 2]

위 표를 보면 알 수 있지만, 각각의 다이어그램 요소들이 각각 하나의 KIF 지식으로 변환됨을 알 수 있다.

5. UML모델에 대한 처리

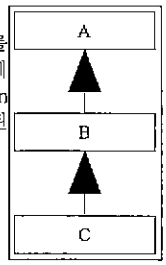
앞절에서 본 바와 같이 UML로 표현된 모델링 정보로부터 지식을 추출하여 이를 KIF로 변환하는 과정은 도형으로 정의되어 있는 UML 모델을 text stream 화하는 과정으로 볼 수 있다. 현재 이루어지고 있는 UML의 stream 표기법 표준화 작업은 UML 정보를 표준화하여 소프트웨어개발툴사이에 정보를 교환하는데 초점을 두고 있다. 이 점이 그 변환된 지식을 처리하는데 보다 관심을 두고 있는 본 연구와는 다르다.

S/W 개발 지식이 지식관리 시스템에서 관리될 때 얻어지는 잇점으로는 대표적으로 추론 등을 이용한 유연한 검색이 가능하다는 것이다 추론은 KSE에서 KIF로써 구현한 재사용가능한 기본 Ontology중 하나인 Frame-Ontology안에서 다음과 같이 정의되어있다.[1]

```
Transitive-Relation
(defrelation Transitive-Relation(?R)
   := (and (binary-relation ?R)
        (=> (and (holds ?R ?x ?y) (holds ?R ?y ?z))
          (holds ?R ?x ?z))))
```

Transitive-Relation R은 2항관계로서, (R x y), (R y z)라는 관계가 존재할때, (R x z)가 참이 되는 관계라는 의미이다. UML 클래스다이어그램에서 가장 대표적인 Transitive-Relation은 계승관계가 된다 [그림3]에서 클래스 A는 클래스 B에서 계승되었고, 다시 클래스 B는 클래스 C에 계승되었다. 따라서 이는 다음과 같이 표현될 수 있다.

(1) (InheritFrom B A)
(2) (InheritFrom C B)



[그림 3]

이것은 [그림3]의 다이어그램의 계승 표시를 그대로 KIF로 표기한 것이다. KIF 메타지식에서 (3) 같이 InheritFrom을 Transitive-Relation으로 정의한다면, 자연스럽게 (4)가 유추되게 된다

(3) (Transitive-Relation InheritFrom)
(4) (InheritFrom C A)

6. 결론 및 향후 연구방향

이 논문에서는 지식베이스관리시스템에서 소프트웨어개발과정의 모델링 자료들을 관리하기 위해 UML 다이어그램 형태로 된 모델링 정보를 지식표현의 외부표준포맷인 KIF로 변환하는 과정을 제시했다. 이를 위해 UML 표현을 위한 메타지식의 형태로서 하나의 Ontology를 구성하고, 이를 이용한 각각의 시스템 지식들을 각각 하나의 Ontology로써 구현하는 방식을 사용했다. 이 연구의 결과로, 지식베이스에 들어간 소프트웨어 개발지식은 기본 Ontology의 의미에 대한 합의가 이루어진 소프트웨어들 사이에서 그 의미가 기계적으로 인식될 수 있으며, 그로 인해 추론과 같은 다양하고 복잡한 질의에 응할 수 있게된다. 또한, 개발작업이 반복됨에 따라 해당 도메인에 대한 도메인지식이 자연스럽게 지식베이스에 생성되게 되며 향후에 패턴지식등이 지식관리시스템에서 관리되게 된다면 각각의 시스템 개발 지식은 보다 쉽게 구축될 수 있으며 이는 S/W 개발 생산성을 향상시키는 효과를 거둘 수 있으리라 판단되어진다. 이들 지식은 KIF포맷으로 표현되어 있으므로 CASE 도구간의 정보교환이나 분산질의를 지원하는데도 유용하게 사용될 수 있다.

참고 문헌

[1] Robert Neches, "The Knowledge Sharing Effort", http://www.ksl stanford edu/pub/knowledge-sharing/papers/kse-overview.html
[2] http://www yy.cs keio.ac.jp/~suzuki/project/uxf/
[3] Tom Gruber, "What is an Ontology?", http://www-ksl.stanford.edu/kst/what-is-an-ontology.html
[4] DSTC, IBM, Oracle, Platinum Technology and Unisys, Joint Initial Submission to the SMIF RFP. OMG document number ad/98-07-01,1998
[5] Michael R.Genesereth, Knowledge Interchange Format - draft proposed American National Standard, 1998
[6] James Rumbaugh et. al., "The Unified Modeling Language Reference Manual", Addison Wesley, 1998
[7] Alexander Borgida et. al., "Knowledge Representation as the Basis for Requirements Specifications", IEEE Computer, 1985
[8] Robert Balzer et. al., "Informality in Program Specifications", IEEE Transactions on Software Engineering Vol. SE-4, No. 2, March 1978
[9] Booch, G. "Object-Oriented Analysis and Design" 2nd Edition The Benjamin/Cummings Publishing, 1994.
[10] Rumbaugh, J et.al. "Object-Oriented Modeling and Design". Prentice Hall, 1991.
[11] Jacobson, I. "Object-Oriented Software Engineering: A Use Case Driven Approach". Addison-Wesley. 1995.