Ontology Languages

1. 온톨로지 개요

1.1 개념

온톨로지란 시멘틱 웹 환경에서 공유하는 데이터의 개념을 표현하는 스펙이다. 간단하게 말해서, 시멘틱 웹의 특정 분야에서 사용되는 표준 어휘의 모임이라 할 수 있다. 시멘틱 웹의 질의어 정보검색에서는 해당 도메인의 개념화를 사용해야하는데, 온톨로지는 이러한 도메인의 개념화에 내포되어 있는 지식을 표현하기 위한 단어를 제공한다.

1.2 시맨틱 웹 과의 관련성

복잡하고 다양한 사용자의 요구사항을 처리 및 통합하는 서비스가 불가능한 이유 중에서 가장 중요한 점은 현재의 웹이 컴퓨터에 의해 처리될 수 있도록 설계되어있지 않다는 점이다. 정보를 어떤 형식으로 표현할 지에 대한 것만 컴퓨터에게 제공할 뿐 그 정보의 의미가 무엇이고 관계된 속성이 무엇인지에 대한 정보는 제공하지 않는다. 따라서 웹에 산재되어있는 자료들을 자동적으로 처리하기 위해서는 컴퓨터가 정보자원의 의미를 이해할 수 있어야한다. 그러나 컴퓨터가 자연어를 이해하고 처리하게 한다는 것은 풀기 힘든 숙제이다.

인터넷 사용자들은 원하는 정보를 찾기 위하여 인터넷 검색엔진을 주로 이용한다. 최근 검색엔진들에서 지식검색 등의 서비스를 실시하고 있지만 웹이 처음 대두되었을 때와 마찬가지로 여전히 웹 검색은 키워드 매칭 방법을 이용하고 있다. 키워드 매칭 방식의 정보검색은 해당 키워드가 들어있는 페이지를 찾아줄 수는 있지만 그 페이지에 들어있는 내용을 파악하지는 못하기 때문에 사용자들은 검색엔진에 의해 찾아진 방대한 자료 중에서 필요한 것을 찾아낸 다음, 수동적으로 이차적인 가공을 할 수 밖에 없다. 예를 들어 Tree를 검색했을 때 식물 Tree를 의미하는지 전산학에서 자료를 저장하는 기법 중의 하나를 말하는 Tree인지 구분이 불가능 하다. 이와 같은 검색방법은 두 개의 문제점이 있는데 첫째는, 서로다른 의미를 가지는 객체를 구분하지 못한다는 것이며, 둘째는 사용자의 요구사항을 정확하게 표현할 수 없다는 것이다.

위와 같은 문제점을 해결하기 위하여 컴퓨터가 이해할 수 있는 데이터를 기반으로 하는 시맨틱 웹이 등장하였다. 시맨틱 웹에서 가장 중요한 문제는 컴퓨터가 이해할 수 있는 데이터의 표현, 즉 지식의 표현이다. 그러나 같은 지식의 표현이라 할 지라도 서로 다른 언어를 사용하는 사람 간에는 지식이 공유될 수 없는 것처럼 시맨틱 웹에서도 표준적으로 사용할 수 있는 지식표현 언어가 있어야 한다. 이러한 관점에서 볼 때 온톨로지는 시맨틱 웹의 요소 중 하나로서 매우 중요하다. 온톨로지는 공유하고자 하는 개념을 형식적이며 명시적으로 표현함으로써, 기계가 이해할 수 있도록 해준다. 또한 각 정보간의 상속(계층)이나 관계를 표현해준다.

2. XML & RDF

2.1 XML

XML(eXtensible Markup Language)은 SGML의 사용하기 어려운 문제를 개선하기 위해 W3C에서 제안된 표준 생성언어이다. 1998년 W3C에서 XML 1.0을 권고안으로 소개하였다. XML은 SGML의 부분집합으로서 SGML과 유사한 형태의 문법체계를 지니고 있으며 SGML과의 호환도 가능하다. XML은 웹상에서 정보를 교환하기 위해 단일화된 구조로서 문서와 정보를 표현하기 위한 포맷으로 엄밀히 XML은 생성언어가 아니라 새 생성언어를 만들기 위한 규칙들의 집합이라고 할 수 있다.

XML이 이전까지의 다른 생성언어와 다른 점은 정보와 표현을 분리시킴으로서 정보의 재사용성을 증가시켰다는 것이다.

XML에 대한 관심이 증가된 이유는 XML 데이터의 생성, 편집 등의 작업을 위하여 특정 응용프로그램에 종속되지 않고 손쉽게 응용프로그램에서 활용할 수 있기 때문이다. 이 XML의 특징은 이기종 응용프로그램간의 상호 작용을 증대시키고, 조직 간에 정보 공유를 향상시킨다.

그 외에도 도메인 이해 증진과 데이터를 생성 언어로 체계화함으로서 응용프로그램들은 이러한 데이터들을 서로 다른 형태로 사용하는 것이 가능하고, 작성된 데이터를 웹에 저장함으로서 적절한 정보를 찾는 능력이 향상된다. 문서에 대한 현재의 검색 방법은 내용에 따라 키워드 일치를 사용하지만 XML 호환 생성언어에 대한 검색은 그 내용뿐 아니라 메타데이터에 대한 검색을 같이 수행하므로 보다 정확하고 지능적이다.

2.2 RDF

RDF는 XML기반으로 만들어진 생성 언어로써 웹상의 분산된 다양한 자원들을 기술함과 동시에 그 의미를 표현하기 위해 개발된 언어이다. 웹 상의 자원을 메타 데이터로써 표현하고 이를 위한 표준화된 방법을 제공하며 웹 자원의 효율적인 관리와 상호운영성을 위해 W3C에서 제안한 것이다. RDF는 웹 자원을 검색하고 참조하며 이를 위한 정보를 효율적으로 교환하고 공유할 수 있고 동시에 기계가 그 자원을 이해함으로써 자원에 대한 연산, 처리가 가능토록 하기 위한 의미기반의 생성 언어이다. 이는 시맨틱 웹의 근간을 이루는 언어로써 추후에 OIL과 DAML과 같은 인공지능 기반의 생성 언어 개발의 기본 언어로써 사용된다. W3C는 RDF 모델 과 문법적 명세서를 1999년에 권고안(Recommendation)으로 발표하였고 이어 RDFS를 2000년에 임시 권고안(Candidate Recommendation)으로 발표하였다.

RDF는 XML의 자유로운 태그사용으로 인한 의미가 불확실한 단점을 보완하기 위해 개발되었다. RDF는 지식표현을 위한 생성 언어로써 시맨틱 네트워크에 기반하여 정보자원의 의미를 표현하고 이를 컴퓨터가 자동적으로 이해 및 처리, 실행할 수 있는 환경

을 제공하고자 설계되었다. 현재의 웹은 HTML로 만들어져 구조화되어 있지 않고 표현과 정보가 뒤섞여 있기 때문에 컴퓨터가 단순 텍스트 정보 추출만이 가능하고 의미를 표현하고 있지 않기 때문에 컴퓨터가 웹 자원의 정보를 인식과 동시에 그를 이해할 수 있도록 자원과 정보의 의미를 표현할 수 있는 환경을 만들고 자하는 목적에서 개발되었다.

RDF 문서에 나오는 자원들은 URI(Universal Resource Identifier)로 나타낸다. 즉 RDF문서에서는 웹 문서, 문서의 일부, 문서를 이루는 요소들을 직접적으로 나타내지 않고 URI를 통해 나타낸다. RDF에서 나타난 데이터 모델은 객체(object)-속성(attribute)-값(value)로써 표현을 한다. 이는 방향성과 라벨이 있는 그래프(directed/labeled graph)로써 모델의 인스턴스(instance)를 표현한다. 이는 객체 위주의 관점에서 프레임 시스템 형태로써 지식을 표현한 것이다. RDF에서 자원들을 표기하는 방법은 한 개의 자원으로 제한할 수도 있고 여러 개를 합쳐서 자원을 나타낼 수 있다. 즉, 자원의 속성들을 문자열을 사용하여 값을 지정할 경우와 여러 개의 자원들이 서로 연관성이 있어 속성의 값으로써 다른 자원이 되는 경우를 나타내는 것이다. URI는 자원의 특성을 규정하는 것으로써 자원의 특성에 지정된 값은 무엇이고, 어떤 종류의 자원을 나타내며, 다른 특성들 간에는 어떠한 관계를 맺고 있는가를 나타내고 있는 것이다. RDF에 나타나는 의미(meaning)는 URI로 정의되고 명명된 용어와 개념으로써 나타낸다. URI가 유일하게 존재하므로 시스템에서는 개념을 정의하고 충돌을 피하기 위해 서로 다른 URI를 사용한다. 즉 URI가 같다는 것은 서로 다른 시스템이라 할지라도 시스템간에 공통된 이해를 가지고 있고 그 의미를 공유하고 있다는 것을 의미한다.

RDF의 문법적 특성은 XML과 유사하다. RDF는 데이터 모델링에 초점을 둔 것으로써 XML을 대신하는 언어가 아니다. 대신 XML의 레이어 위에 존재하는 것으로 XML과의 통합이 가능할 뿐만 아니라 XML문서를 이용할 수 있다는 장점을 지니고 있다. RDF는 원시데이터 타입인 정수형, 실수형 등과 같은 데이터 타입을 지원하지 않고 문자열로써 대신한다.

RDF와 XML은 그 문법적 특성이 유사하고 XML을 기저로 하여 만들어졌지만 RDF는 XML과 다음과 같은 점에서 다르다. XML의 모델은 트리 형태로써 그 스키마는 문법적 해석과 문서의 구조에 초점을 둔 것이다. 그러나 RDF의 모델은 객체지향 방식의 모델로써 RDFS는 문서에 나타난 자원들의 의미해석에 그 초점을 두고 있다. 즉, XML의 데이터 모델은 객체의 배열된 순서가 중요한 트리 형태의 모델을 위한 텍스트 마크업이고 RDF 데이터 모델은 객체와 객체 사이의 관계를 정의하는 데이터 모델이다. XML은 트리 형태의 문서로 고정되게 설계되어 문서를 이루는 자원들이 문서에 포함되어 인덱스되기 때문에 메타 데이터를 표현하기 위한 유연성이 부족하다. 반면 RDF에서의 트리 형태로 문서를 구성하지 않고 URI를 갖는 자원이기 때문에 메타 데이터를 표현하기에 유연성을 지니고 있다.

RDFS는 자원의 특성을 기술하기 위해 사용될 수 있는 자원들의 집합을 나타내고 웹 자원을 기술하기 위해 이용된 메타 데이터의 구조를 해당 어플리케이션에서 사용되는 특정한 어휘로 정의한다. 또한 RDFS는 데이터의 메타 데이터의 무결성을 보증하기 위해

구문에 관계없는 메타 데이터의 자원과 특성에 대한 유효성을 검증하는 역할도 한다.

RDFS는 RDF가 XML과 유사한 것처럼 XML 스키마와 유사하다. RDF 스키마의 range 제약조건과 XML 스키마의 타입 제약 조건, RDFS의 도메인 제약 조건과 XML 스키마의 타입과 요소(element)의 내용(contents) 모델정의, 특정 도메인을 위한 목록 리스트나 통제화된 어휘값의 정의, RDFS의 comment와 XML 스키마의 annotation 등의 유사점을 찾아볼 수 있다. 그러나 그 외의 다른 특성은 매우 상이 하다. XML 스키마는 데이터의 구조, 타입 제약 사항을 규정하지만 메타 데이터의 도메인 사이의 의미를 표현하기 위한 유연성이 부족한 반면 RDFS는 계층, 클래스와 특성간의 관계 등을 제시하기 위한 풍부한 의미를 제시하고 있다. 그러나 데이터 구조의 제약이나, 데이터의 타입 제약은 제공하지 않는다.

RDF Example

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:s="http://iis.korea.ac.kr/schema/">
    <rdf:Description about="http://iis.korea.ac.kr/Home/Sohn">
      <s:Creator>
          <rdf:Description about="http://iis.korea.ac.kr/stdId/2005020626">
              <rdf:type resource="http://iis.korea.ac.kr/schema/Person"/>
              <v:Name>Sohn JongSoo</v:Name>
              <v:Email>mis026@korea.ac.kr</v:Email>
          </rdf:Description>
      </s:Creator>
    </rdf:Description>
</rdf:RDF>
```

2.3 OWL

OWL은 DAML+OIL과 매우 유사한 언어이다. 실제로, OWL을 위한 구문은 DAML+OIL로부터 약간 수정, 보완되었고, OWL의 추상 구문은 DAML+OIL에서의 추상 구문으로써 보여 질 수도 있다.

시맨틱 웹을 구현하기 위한 계층적인 구성은 RDF(S), XML등이 시맨틱 웹 언어의 하부구조를 이루고 있고 추론을 기반으로 하는 높은 시맨틱을 갖는 웹 온톨로지 언어들은 RDF(S), XML등을 하부 구조로 이용하여 확장하여야 한다는 사실에 근거하여, 하부 구조의 표준화가 먼저 선행되어야 한다.

현재 OWL은 W3C에서 표준 온톨로지 언어로 지정한 상태이다. 이에 따라 OWL을 기반으로 수 많은 연구가 진행중이며 OWL을 확장하여 OWL이 표현하지 못하는 지식을 표현하는

프로젝트가 제시되고 있다.

Example 1)

```
<owl:Ontology rdf:about="#soju">
  <rdfs:comment>soju is one of the delicious drink</rdfs:comment>
  <owl:priorVersion rdf:resource="http://iis.korea.ac.kr/goods/soju"/>
</owl:Ontology>

<owl:DeprecatedClass rdf:ID="Lemon soju">
  <rdfs:comment>it will give you a nice smell</rdfs:comment>
  <owl:equivalentClass rdf:resource="#Lemon taste soju"/>
</owl:DeprecatedClass>

<owl:DeprecatedProperty rdf:ID="hasCap">
  <rdfs:comment>lemon soju has a cap</rdfs:comment>
</owl:DeprecatedProperty>
```

Example 2)

```
<owl:Class>
  <owl:intersectionOf rdf:parseType=" collection">
    <owl:Class rdf:about="#Person"/>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasChild"/>
      <owl:toClass>
        <owl:unionOf rdf:parseType=" collection">
          <owl:Class rdf:about="#Doctor"/>
          <owl:Restriction>
            <owl:onProperty rdf:resource="#hasChild"/>
            <owl:hasClass rdf:resource="#Doctor"/>
          </owl:Restriction>
        </owl:unionOf>
      </owl:toClass>
    </owl:Restriction>
  </owl:intersectionOf>
</owl:Class>
```