**시멘틱 웹에서의 메타데이터 검색을 위한 RDF 저장 구조 및 시스템의 개발에 관한연구**

**1. RDF의 데이터 구조**

**1.1 RDF의 개요**

현재의 월드 와이드 웹(World Wide Web, 이하웹)은 HTML( HyperText Markup Language)을 사용해서 정보를 표현하는데 초점이 맞춰져 있다. 이는 단순히 사람에게 보여주기 위한 용도로써，기계가 읽을 수는 있지만 이해할 수는 없다는 것을 의미한다. 시맨틱 웹(Semantic Web)은 이러한 문제를 해결하기 위한 노력으로써, 웹상에 존재하는 데이터를 기계가 이해하고 자동으로 처리하도록 하는 데 그 목적 이 있다. RDF(Resource Description Framework)는 시맨틱웹을 이루는 기반 기술로써 기계가 이해할 수 있는 형태의 메타데이터(Metadata)를 기술하기 위한방법이다. 도서관에서 직접 책을 찾아 본 사람이라면 도서관의 색인, 즉 메타 데이터가 얼마나 중요한 것인지를 알고 있을 것이다. 마찬가지로 웹에 존재하는 자원을 표준화된 메타 데이터 기술 방법인 RDF를 이용해서 기술함으로써 상이한 애플리케이션 간에 정보를 교환하고 이용할 수 있게 한다. 예를 들어 웹 페이지의 표제나 저자，혹은 갱신 일자나 저작권 정보 공유 자원과 같은 웹 자원을 RDF를 이용해서 기술할 수 있다. RDF를 이용함으로써 웹 자원에 대한 메타 데이터를 기술할 URI(Uniform Resource Identifias)를 이용해서 명시하며, 그 자원이 가진 속성과 값으로 자원을 기술한다. 즉 자원과 자원 사이의 관계，혹은 자원과 자원이 가진 특정 값을 이용해서 자원을 표현한다. 이러한 관계는 ER-다이어그램과 유사한 모양을 가진 RDF 그래프(RDF Graph)로 표현할 수 있다. 예를 들어. 다음의 선언문이 있다



이 선언문을 RDF 그래프로 표현하면〈그림 1>과 같다.

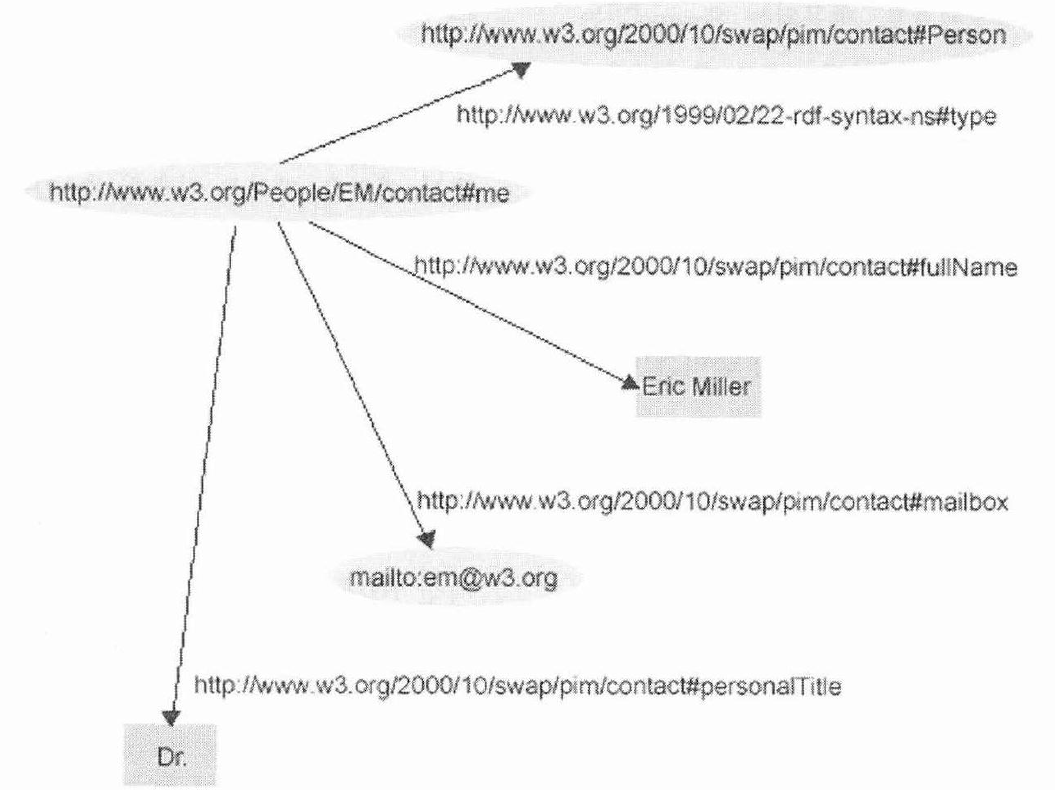
위의 RDF 그래프에서 자원을 식별하기 위해 다음과 같은 URI가 사용되었다.

• Eric Miller 라는 이름을 가진 개체는 http: / / www.w3.org/People/EM/ contact # meURI로 식별된다.

• 해당 개체의 종류는 http://www.w3.org/2000/10/swap/pirn/contact # Person5- 식별되는 사람이다.

• 메일 주소 값과의 관계를 나타내는 속성은http: //www.w3.org/2000/10/swap/pirn/contact #mailbox URI로 식별된다.

• 속성의 값은 mailto:em@w；iorg이다.



[그림 1] 예제 선언문의 RDF 그래프

RDF는 메타 데이터를 표현하기 위한 모델이며，모델의 표현 방법으로써 애플리케이션 사이의 데이터 교환 시 상호 운용성을 지원하기 위해 XML(eXtensible Markup Language) 문법을 따르는 RDF/XML 구문을 사용한다. RDF의 주요 목표 중 하나는 바로 애플리케이션 사이에 표준화된 정보의 교환이며, 이는 XML과 그 제반 기술을 사용함으로써 충족될 수 있다. 위에 기술된 RDF 그래프를 RDF/XML 구문으로 표현하면 다음과 같다.

|  |
| --- |
| <?xml version = "1.0"?)  (rdf:RDF xmlns:rdf=Mht中://www.wliorg/1999/ (E/22~Tdf-syntax-ns# "  xmlns：contact = "http:// www.w3.oi-g/2000/10/swap/piiu/contact#")  (contact：Person rdf：about = "http://www. w3.org/People/EM/contact#me")  <contact：fuliName)Eric MiIlQ-</contact：MName)  (cmtact: mailbox rdf： resource = ' maflto: em @ w3.org"/)  <contact：peisonalTitle)Dr.</contact：peisonalTitle)  </contact-'ftison)  </rff：RDF> |

RDF/XML 구문은 기계에 의해서 처리될 수 있으며, 또한 URI를 사용함으로써 표준화된 방법으로 자원을 유일하게 식별할 수 있다. 이와 같이 RDF는 웹상에 존재하는 자원뿐만 아니라 사람이나 자동차 같은 사물을 기술할 수도 있으며. URI를 사용함으로써 자원과 자원간의 관계를 정확하게 표현하고 식별할 수 있다.

**1.2 RDF 메타데이터의 Triple 구조**

RDF는 웹 자원에 대한 Triple 구조의 선언문을 작성하는 방법을 제공한다. 가령 예를 들어. "http://www.example.org/index.html 웹 페이지롤 John Smith가 작성했다”라는 구문은 영어를 이용하여 다음과 같은 문장으로 표현할 수 있다.

|  |
| --- |
| http://www.example.org/index.html has a creator whose value is John Smith |

위의 구문을 사펴보면 기술하려는 대상이 존재하며, 대상에 대한 속성, 그리고 속성 값으로 이루어져 있다. 이와 같은 선언문에서 기술하려는 대상. 즉, http://www.example.org/index.html을 주어(subject)라고 하며, 대상이 가지고 있는 속성 creator를 술어(predicate), 속성의 값 john Smith를 목적어(object)라고 한다. 이를 정리하면 다음과 같다.

• 주어 (subject)는 URL http ：http://www.example.org/index.html이고

• 술어 (predicate)는 “creator"라는 단어이며

• 목적어(object)는 "John Smith”라는 구이다.

위의 예제에서는 술어와 목적어로써 creator와 John Smith라는 단어를 사용했지만RDF에서는 각각을 웹 자원으로 구분하기 위해 URI로 대체해서 사용한다. 이는 RDF 그래프를 이용해서 다음과 같이 노드(node)와 아크(arc)로 표현할 수 있다.

|  |
| --- |
| http : / / www.example.org/index.html has a creator whose value is John Smith  http : / / www.example.org/index.html has a creation-date whose value is August 16, 1999  http : / / www.example.org/ index.html has a language whose value is English |

RDF 그래프에서 URI를 나타내는 노드는 타원으로 표현되며，리터럴(literal) 값을 갖는 노드는 4

각형으로 표시된다. 또한 아크는 술어를 나타내며. 방향성을 갖는다. RDF는 이와 같이 주어, 술어, 목적어로 구성된 선언문 형태를 가지고 있으며，이는 다음과 같이 Triple 형태로 표현할 수 있다.

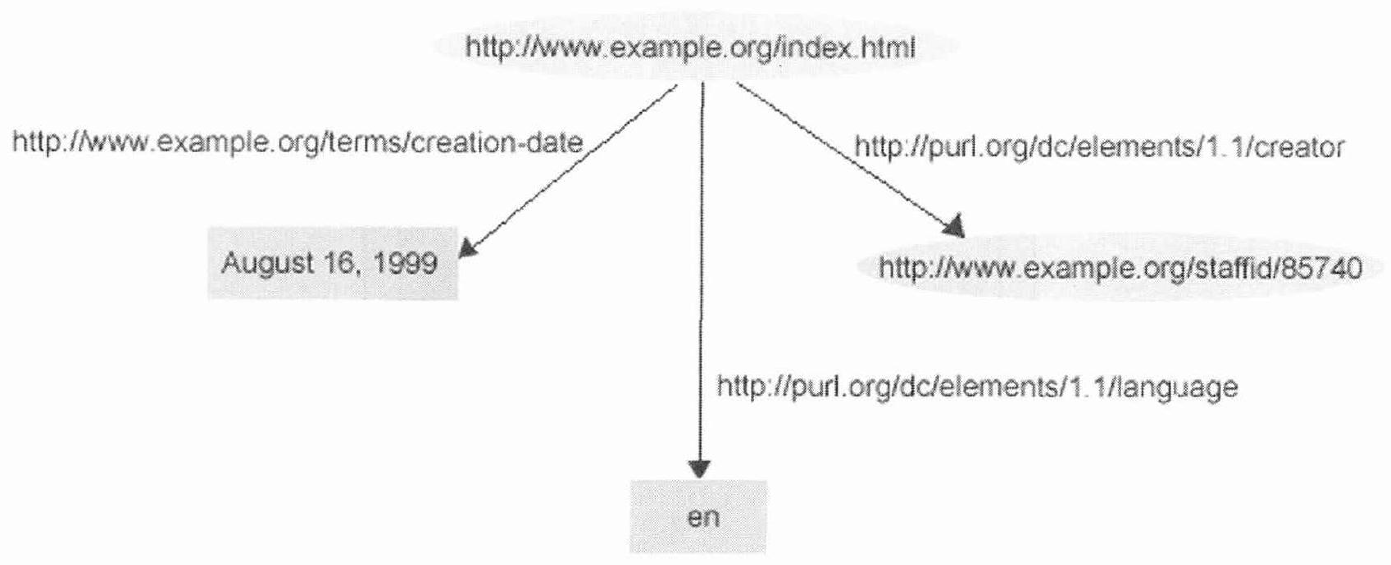
|  |
| --- |
| 〈http : //www.example.org/index.html>  (http: //purLorg/dc/elements/1.1/creator)  <http://www.example.org/staffid/85740)  〈http: / /www.example.org/indexhtmlXhttp: //www  .example.org/terms/creation-date〉  "August 16, 1999" •  〈http: //www.example.org/indexhtmlXhttp: //www  .example.org/terms/language) “en” . |

Triple을 작성하기 위해서 참조되는 전체 URI를 포함하도록 작성해야 하지만 편의상 네임스페이스

를 정의해서 다음과 같은 단축형을 사용할 수도 있다.

|  |
| --- |
| ex：index.html dc : creator ejstaff:85740 .  ex：index.html exterms: creation-date “August 16, 1999” .  ex：index.html dc： language “en” . |

RDF의 모든 선언문은 이와 같이 주어, 술어, 목적어로 구성된 Triple 형태로 이루어져 있다. RDF문서를 작성하기 위해 XML Schema의 데이터 타입과 공백 노드(Blank Node), RDF 스키마(RDF Schema)와 같이 다양한 개념이 사용된다.



[그림 2] 트리플 문장의 RDF 그래프

**2 효율적인 검색을 위한 저장 구조**

**2.1 Triple 저장 구조**

RDF는 정형화된 어휘들(vocabularies)을 기술하기 위한 선언 형식의 표현법이다. RDF를 그래프형태로 표현한 RDF 그래프는 주어, 술어, 목적어로 이루어진 Triple의 집합으로 표현된다.

Triple을 이루고 있는 주어, 술아 목적어 노드는 각각 URI나 리터럴 중 한 가지 형태를 갖는다. 또한 리터럴은 데 이터 타입이 정의되지 않은 문자열인 Plain 리터럴(Plain literal)과 데이터 타입 이 부여된 Typed 리터럴(Typed literal)로 구분되며, Typed리터럴의 경우 XML Schema의 일부 데이터 타입을 따른다.

이를 기반으로 설계하고 구현된 SMARTTriple시스템에서 사용하는 개략적인 데이터 구조는 다음과 계층적 관계를 갖는다.

RDP 문서로부터 추출된 Triple들의 집합은 SMARTModel로 관리되며 즉，이것은 RDF Triple이 모여서 만들어진 하나의 RDF 문서를 나타낸다. 각각의 Triple들은 SMARTIndexMap에 의해 인덱싱된 구조로 메모리에 저장되며. Triple은 SMARTNode 타입의 주어，술어，목적어로 구성된다. SMARTNode는 그 종류에 따라 URI나 Literal 혹은 Blank로 생성되며, Literal의 경우 XML Schema의 데이터 타입 분류에 따라 크게 네 가지 타입으로 구분된다.

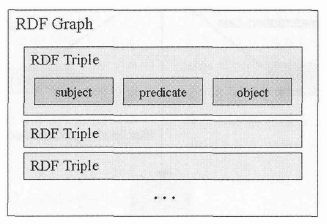
이와 같이 본 SMARTTriple 시스템은 RDF를 기반으로 설계 및 구현함으로써 RDF 시맨틱을 포함한 RDF의 확장 및 OWL과 같은 상위 온톨로지(Ontology) 언어를 반영할 수 있도록 하였으며, 이를 기반으로 시맨틱 웹의 핵심 역할 중 하나인 온톨로지 추론과 규칙 추론 기능이 반영 가능하도록 설계 및 구현하였다.

**2.2 검색을 위한 인덱스 구조**

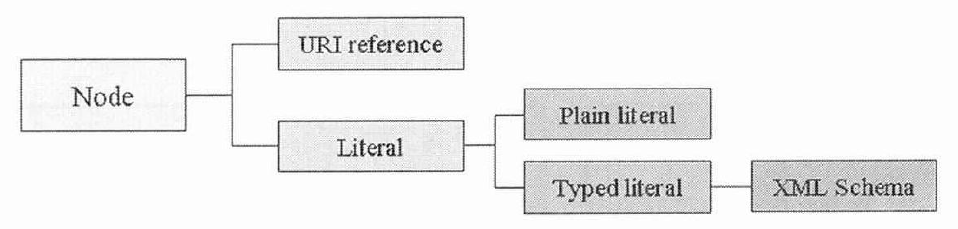
인덱스 구조란 특정 값(value)의 검색 효율을 높이기 위해ㅜ해당 값의 키(key)와 그 값의 위치를 쌍으로 저장하는 데이터 저장 구조를 말하며，키와 값의 연결(Mapping)을 표현한 것이 인덱스 맵이다. 일반적인 인덱스 구조의 검색은 키의 입력을 통해 이루어진다. 키가 입력되면 인덱스 맵 내에 존재하는 키와 인덱스의 쌍을 검색한 후 해당 인덱스와 연결 (mapping) 된 값에 직접 접근하여 원하는 값을 찾게 된다. 본 논문에서 Triple의 신속한 검색을 위해 구성한 인덱스 맵의 구조는 다음과 같다.

특정 Triple에 대해 그 Triple을 구성하는 노드를 키로 설정하고 해당 Triple을 값으로 설정하여，키와 Triple의 위치를 나타내는 인덱스를 저장하는 인덱스 맵을 구성한다. 또한 필요에 따라 여러 개의 인덱스 맵을 구성하여 보다 효율적인 검색이 가능하다. 이러한 인덱스 구조가 없을 경우에는 찾고자 하는 Triple을 저장되어 있는 수많은 Triple과의 비교를 통하여 그 위치 및 내용을 검색하게 되는데, 특정 값을 갖는 노드를 포함한 TMple을 검색하려는 RDF 메타데 이터 검색 시스템에서는 검색해당 노드를 포함한 Triple들의 위치를 인덱스 맵구조를 이용하여 바로 알아낼 수 있기 때문에 Triple의 노드 값을 저장하는 테이블의 모든 값들을 일일이 찾을 필요 없이 찾고자 하는 값의 위치로 바로 접근이 가능하다. Triple의 수가 많아질수록 인덱스 맵으로 인한 검색 속도 및 효율성의 향상은 극대화된다.

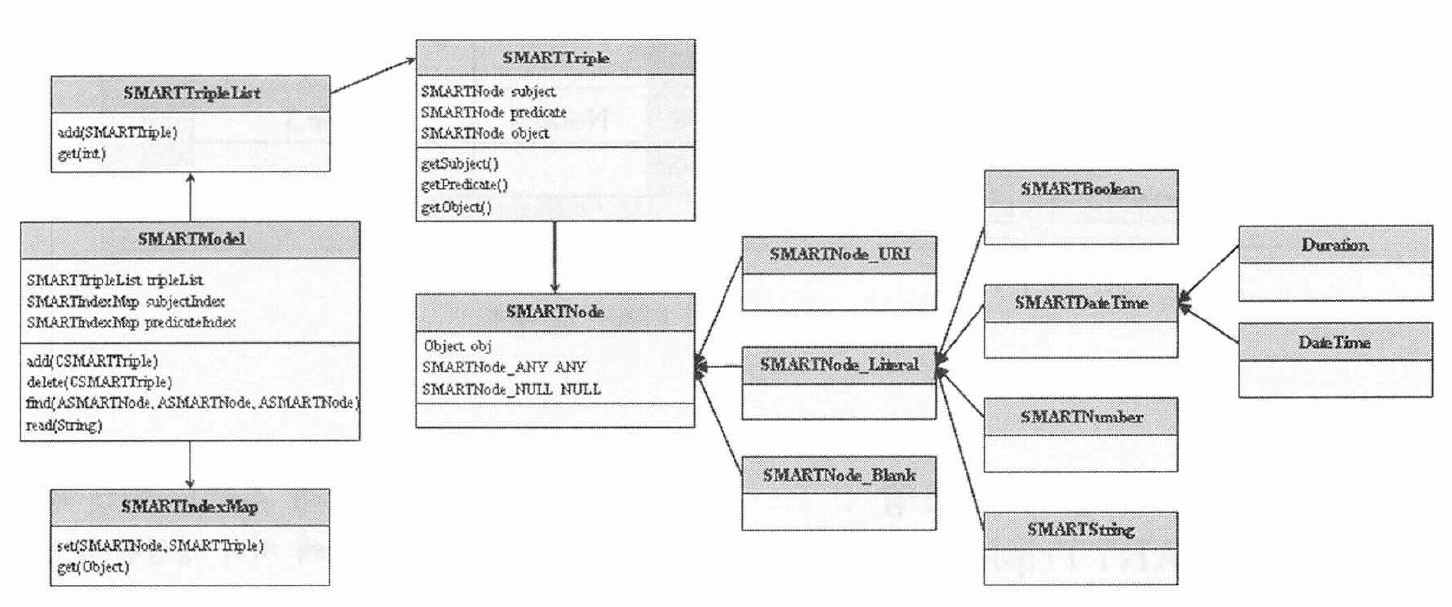
이와 같은 인덱스 맵을 사용함으로써 Triple에 저장하는 노드 데 이터를 문자열 이 아닌 인덱스 맵의 인덱스 값으로 저장할 수 있게 된다. 이러한 특징은 Triple의 검색 뿐만 아니라 Triple의 비교를 위한 노드 비교' 속도의 향상시켜 SMARTTriple 시스템의 전체적인 속도를 향상시키는 추가적인 장점이 된다.



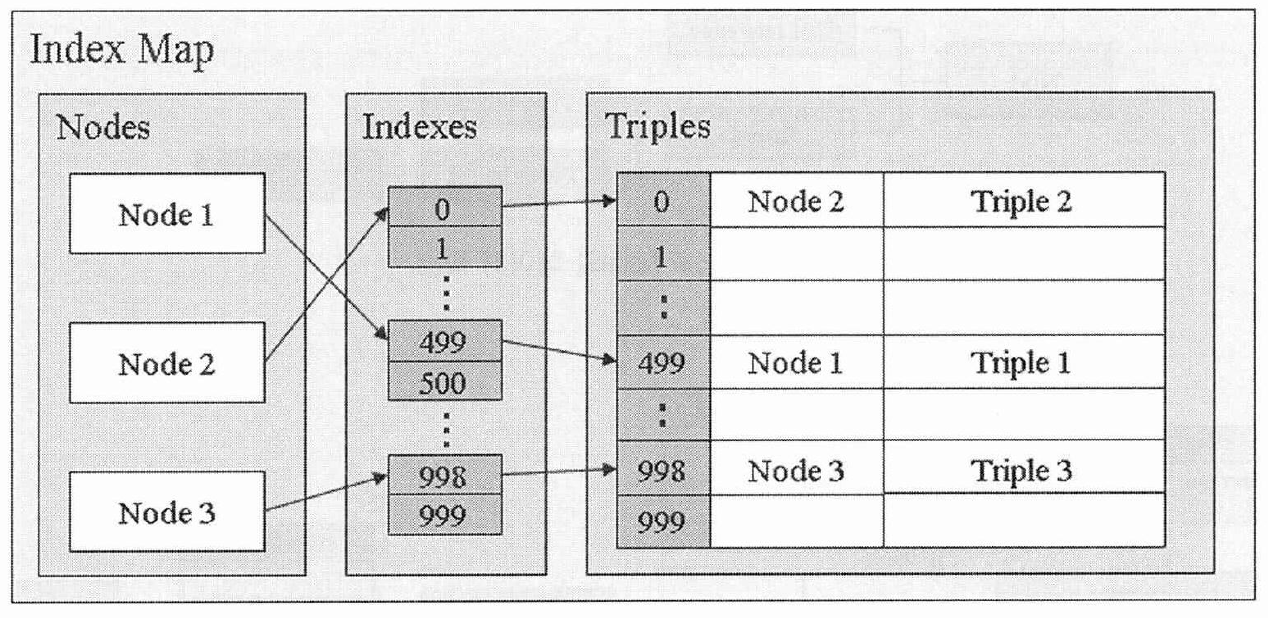
[그림 3] RDF Triple의 저장 구조



[그림 4] RDF Node의 데이터 타입



[그림 5] SMARTTriple의 RDF Triple 데이터 구조



[그림 6] SMARTTriple의 인덱스 캠 구조

**3. 메타데이터 검색 시스템**

**3.1 시스템 구조**

SAMRTTriple 시스템은 크게 두 개의 시스템으로 구성된다. RDF 문서를 읽고 RDF 구조를 만들기 위한 RDF Loader와 노드와 Triple을 생성해서 메모리에 저장하기 위한 Graph Builder, 그리고 사용자의 질의를 처리하기 위한 Query Processor로 구성 된다.

RDF Loader부터 살펴보면，RDF 문서를 처리하기 위해 기본적으로 SAX 파서가 이용된다. SAX파서는 이벤트를 기반으로 한 XML 파서로써 한번 XML 문서를 처리할 경우 메모리의 제약 없이 아주 빠르게 처리할 수 있는 장점을 제공한다. SAX 파서를 거쳐 만들어진 토큰(token)은 RDF Lexer를 통해 RDF의 구조에 따라 생성된다. 생성된 토큰은 Graph Builder를 통해 노드로 만들어지게 되는데 노드의 종류, 즉 URI 노드인지 리터럴 노드인지에 따라 각각의 타입에 맞게 생성된다. 이 때 Duplicator Checker를 통해 노드의 중복성을 해결하고 중복성 이 해결된 노드는 Triple로 생성된다.생성된 Triple은 메모리에 삽입되기 전 빠른 검색을 위한 인덱싱 과정을 거치게 된다. 이와 같은 과정을 통해 비로소 하나의 완전한 RDF 저장 구조를 갖추게 된다.

시맨틱 웹 애플리케이션 개발자는 SMARTTriple 시스템에서 제공하는 질의 API를 이용해서 Triple에 대한 질의를 할 수 있으며, 이때 인덱스를 이용한 저장 구조를 검색함으로써 검색 성능의 향상을 가져올 수 있다.

**3.2 메타데이터 검색**

SMARTTriple 시스템에서는 앞서 소개한 인덱스 구조를 사용함으로써 Triple에 대한 검색 속도를 향상시킬 수 있다. SMARTTriple 시스템에서의 인덱스 구조는 Triple 의 구성 요소를 이루는 각 노드를 키로 하고 Triple을 값을 갖는 인덱스 맵을 사용한다. 키로 사용되는 각 노드는 Triple로의 연결을 위한 인덱스 값을 갖고 있으므로 SMARTTriple시스템에서는 URI 혹은 Literal을 이용해서 필요한 노드를 생성한 후에 Triple을 구성하여 검색을 할 수 있다. 이때 Triple을 각 자리에 변수를 사용함으로써, 여러 개의 결과 Triple을 포함하는 Triple Set을 검색 . 질의하는 것도 가능하다.

사용자로부터 SMARTTriple 시스템의 API를 통하여 Triple에 대한 질의가 올 경우 Indexing Processor는 변수를 제외한 노드를 이용해 인덱스 맵을 검색하게 된다. 이때 인덱스 맵을 이용하는경우는 subject와 predicate이 검색어(상수)인지 변수인지에 따라 모두 네 가지로 나뉘 게 된다. 둘 중에 하나라도 검색어가 있는 경우에는 해당 검색어를 이용하여 인덱스 맵을 검색하게 되고 둘 다 변수인 경우에는 인덱스 맵을 이용할 수 없게 된다. 인덱스 맵을 이용하는 경우에도 subject와 predicate이 모두 검색어(상수)인 경우에는 인덱스 맵을 2번 이용함으로써 찾고자 하는 Triple에 대한 검색 범위를 더욱 줄일 수 있게 된다. 인덱스 맵으로부터 Triple에 대한 인덱스 값을 얻은 후 이렇게 얻은 인덱스 값을 이용해서 사용자가 질의한 Triple을 찾아 Triple Set을 구성해 결과로 반환하게 된다. 이와 같이 인덱스 구조를 사용해서 검색을 함으로써 전체 Triple을 비교하지 않고도 인덱스 값을 이용해서 특정 Triple이 저장된 위치로 바로 접근 가능하기 때문에 타 시스템에 비해 보다 빠른 검색이 가능하다.