시맨틱웹 기반 메타데이터 레지스트리 설계에 관한 연구

오 삼 균
성균관대학교 문헌정보학과 교수

I. 서 론

다양한 기술적인 데이터 처리와 통신 능력이 증가함에 따라 정부 및 사업체에서 정보처리시스템을 통해서 생성되는 정보의 상호운용, 교환 및 공유에 대한 필요성이 증가되었다. 이런 요구에 대응하기 위하여, 국제표준협회 산하의 32 하부위원회(ISO/IEC JTC1 SC32)에서 제정한 11179 문서는 메타데이터 요소의 의미를 명확히 정의함으로써 열린 정보시스템 상황에서 데이터의 공유성을 높이기 위한 노력이다. 데이터가 공유되기 위해서는 데이터의 이용자와 소유자가 그 의미와 표현, 식별에 대한 공통의 이해를 가지고 있어야 한다. 이런 의미에서 ISO/IEC 11179 표준문서는 데이터요소(data element)의 식별, 명명, 정의, 개념, 값, 분류 등에 중점을 두고, 교환될 데이터의 이용자가 의미, 표현형식, 및 식별에 대한 공통된 이해를 갖기 위한 메타데이터 등록(metadata registration)과 유지를 원활히 수행하기 위한 지침을 제공하고 있다. 이 표준은 주로 Metadata Registries(이하, MDR)로 알려져 있고, 이러한 표준에 근거하여 각 기관에서 사용하고 있는 데이터가 정의되고 관리된다면 데이터요소에 관한 참조와 재활용이 용이해지기 때문에 데이터 상호운용성을 크게 증진시킬 수 있을 것이다. 현재는 이 표준에 근거한 메타데이터 등록시스템이 분야별로 개발되어 운영되기 시작하는 단계에 있고 더욱더 활성화될 필요가 있다.

그러나 이런 표준에 근거하여 구축된 MDR에 저장된 정보가 시스템 개발자들에게는 유익을 줄 수 있지만, 기계가독형으로 표현되어 있지 않기 때문에 각종 시스템이나 프로그램들이 이 등록기에 저장된 정보에서 자동으로 의미를 추출하는 데는 한계가 있다. 본 연구에서는 이런 단점을 극복하기 위한 노력의 일환으로 W3C에서 개발한 시멘틱웹 기술을 이런 MDR에 접목시켜 MDR들 간의 의미적 상호운용성을 증진시키려는 것이다. W3C의 시멘틱웹의 활동은 산재한 정보자원을 기계가독형으로 기술할 수 있는 기반을 조성함으로써 웹상의 다양한 정보를 지식화하는 것을 그 목표로 하고 있다. 이 시멘틱웹의 기반은 모든 자원, 클래스, 속성 등에 불변의 고유 식별자를 부과하는 URI 개념, 이런 식별자들을 군집화해서 관리하는 네임스페이스 개념과 상이한 시스템에 저장된 정보의 통합을 용이하게 처리할 수 있는 XML 기술이 그 하부구조를 이루고 있다. 여기에 의미적 상호운용성을 증진시키기 위해서 개발된 Resource Description Framework(이하, RDF), RDF 스키마, 웹 온톨로지언어(Web Ontology Language, 이하 OWL) 등의 새로운 접근 방법들이 제안되었다.

RDF는 자원에 대한 메타데이터를 기술할 때 특정 메타데이터 스키마에 구애 받지 않고 일반적으로 기술할 수 있는 기반을 제공하기 때문에, 각 기관에서 분산적으로 자원에 대한 기술을 할지라도 상호운용성을 증진시킬 수 있는 중요한 방법이다. 메타데이터의 의미적 호환성 구현과 이를 통해 인터넷을 기반으로 하는 정보시스템 간의 상호운용을 용이하게 할 목적으로 RDF가 사용될 것으로 보인다. 여기에 분야별 클래스와 속성을 정의하는데 필요한 어휘를 갖추기 위해서 RDF

스키마가 제정되었고, 이 RDF 스키마에 논리적 표현력과 유추력을 보강하기 위해서 OWL이 제정되었다.

기계가독형 MDR을 구축하기 위해서는 현 MDR 국제표준과 RDF/OWL을 접목할 필요가 있다고 본다. 현 MDR 표준과 OWL의 의미표현력(semantic expressiveness)의 결합은 레지스트리 간의 의미적 상호 운용을 위한 기술적 기반을 제공한다. 이 연구의 목적은 현 MDR의 핵심개념을 소개하고, 다양한 MDR들 간의 의미적 상호운용을 증진시키기 위한 시맨틱웹 기반 MDR 설계의 기본방향과 핵심 하부구조를 제시하는 데 있다.

다음 2장에서는 MDR에 대한 핵심내용을 정리하고, 3장에서는 국제적으로 통용되고 있는 다양한 메타데이터 레지스트리를 소개하고, 4장에서는 기계가독형 MDR의 설계초안을 제시하고자 한다.

Ⅱ. ISO/IEC 11179 (MDR) 개요

MDR은 데이터의 의미(semantics)를 표현, 등록, 관리, 교환, 공유하는 것이 주 목적이다. 정보의 의미 표현과 공유를 가능케 하는 MDR은 레지스트리들에 등록된 정보들 간의 상호운용성의 확보에 중요한 기반을 제공하며, 실제적으로 동일한 정보의 중복과 구문적, 문자적 차이를 식별할 수 있게 함으로써, 의미적으로 동일한 요소들 간의 통합이 이뤄질 수 있게 한다. 다음은 기계가독형 MDR 설계초안을 제시하기 전, 현 MDR 국제표준의 핵심 내용을 정리하여 독자의 이해를 돕기 위한 것이다.

1. 프레임워크(Framework)

MDR 1장은 프레임워크를 제시하고 있으며 ISO/IEC 11179의 5개 하위 표준 간의 유기적 관계에 대해 기술하고 있다. MDR에서 가장 중요한 개념적 모델(conceptual model)의 요소인 데이터요소 개념(data element concepts), 데이터요소(data elements), 값의 영역(value domains), 그리고 개념적 영역(conceptual domains)간의 관계를 설명하고 있다.

A. 데이터요소 개념 (Data element concepts)

데이터요소 개념은 특정 데이터에 관한 개념(또는, 확장하여 해석하면, 의미)을 나타내며, 특정 값 영역의 참조 없이, 즉 표현방식 (representation)에 구애 받지 않고 독립적으로 존재한다. 데이터 모델링의 관점에서 볼 때, 데이터요소 개념은 두 가지 요소, 객체 클래스(object class)와 속성

(property)으로 구성되어 있다. 객체 클래스는 데이터 수집의 대상이 되는 객체의 집합을 나타내며, 객체지향성 모델링의 클래스(class)나 개체-관계 모델(entity-relationship model)의 개체(entity)에 준한다. 그리고 속성(property)은 객체의 특성을 기술한다. 이를 통하여, 데이터요소 개념은 표현하고자 하는 데이터요소의 상황적 의미(contextual semantics)를 표현한다.

B. 값 영역(Value domain)

값 영역은 특정 데이터 요소에 허용된 값(permissible value)의 집합이다. 예를 들어, 데이터 요소가 한 가정의 연간 소득을 표현 하고자 한다면, 0을 포함하지 않는 정수가 이 요소에 적용될 수 있는 값의 영역이다. 데이터요소 개념은 여러 개의 값의 영역과 관계할 수 있으며, 이를 통해 개념적으로 유사한 데이터요소를 구성할 수 있다. 예를 들어, ISO 3166 국가코드는 7가지 국가 표현방식을 제시한다. 각 국가 표현 방식에는 허용된 값의 집합이 정의되어 있으며, 이들 값의 집합들은 데이터요소(여기서는 국가 코드)의 값 영역으로 사용될 수 있다. 선택된 국가 표현 방식에 따라, 허용된 값의 집합, 데이터 타입, 표현 클래스, 또는 치수(unit of measure)가 변한다. 허용값(permissible value)은 값(value)과 의미(meaning)의 쌍으로 이뤄진다. 예를 들어, 〈y, yes〉, 〈n, no〉에서 yes와 no는 값의 의미를 표현한다.

C. 개념 영역(Conceptual domains)

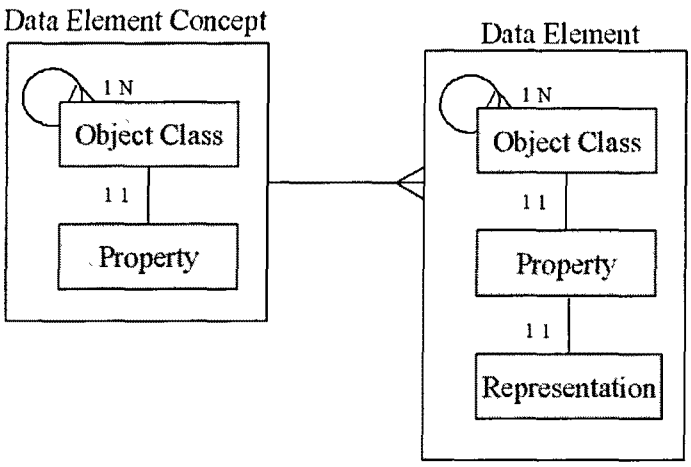
개념적 영역은 값 의미(value meaning)의 집합이다. 모든 값 영역의 의미(semantics)는 개념적 영역을 통해 나타낸다. 개념적 영역에는 두 가지 종류가 있다. 첫째는 열거 가능한 개념적 영역(예: 모든 값의 의미의 나열: yes, no)으로 표현되고, 둘째는 규칙(rule)에 의해서만 정의될 수 있기 때문에 가능한 값을 열거하는 것이 불가능한 개념적 영역으로 표현된다. 개념적 영역은 여러 값 영역과 관계할 수 있다. 그러나 주어진 특정 값 영역은 오직 하나의 개념적 영역과 관계될 수 있다. 특정 개념적 영역은 여러 개의 값 의미와 관계될 수 있고, 각각의 값 의미는 오직 하나의 허용된 값과 관계한다. 값 영역은 다수의 허용된 값을 가질 수 있다.

D. 데이터요소 (Data elements)

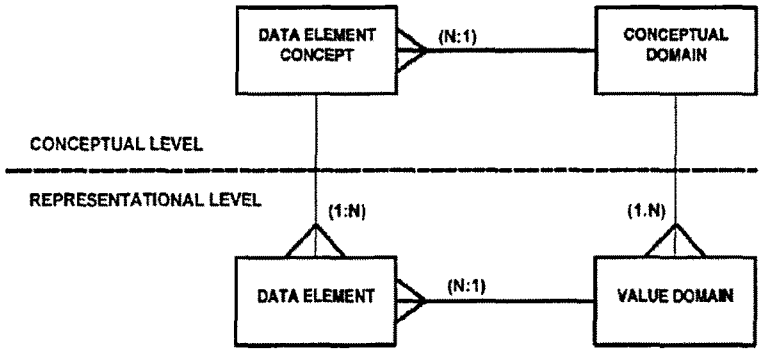
데이터요소는 생산, 관리, 공유의 가장 기본이 되는 데이터 단위이다. 데이터 요소는 데이터 요소 개념이 표현방식(representation), 즉 값의 영역과 연계되었을 때 생성된다. 표현방식은 데이터의 형식 즉, 값의 영역, 데이터 타입, 표현 클래스, 그리고 필요한 경우에는 치수를 포함한다.

- 한 데이터요소 개념은 여러 데이터요소와 관계할 수 있으나, 한 데이터요소는 오직 한 데이터요소 개념에 속해야 한다.

● 데이터요소는 특정 표현방식이 데이터요소 개념에 적용되었을 구체적으로 생성된다.



〈그림 1〉 데이터 요소 모델



〈그림 2〉 ISO 11179 MDR 모델

위 〈그림 2〉은 ISO 11179 MDR 모델로서, 개념영역, 값의 영역, 데이터요소 개념, 데이터요소들 간의 관계를 간명하게 보여준다.

● 특정 데이터 요소는 데이터 요소 개념과 값의 영역으로 이루어진다.
● 많은 데이터 요소가 같은 데이터요소 개념을 공유할 수 있다. 즉, 같은 개념이 여러 다른 방법으로 표현(representation)될 수 있다.

- 많은 데이터요소가 같은 표현을 공유할 수 있다. 즉, 특정 값 영역이 여러 다른 데이터요소에서 재사용될 수 있다.
- 값 영역은 데이터 요소와 반드시 연계될 필요가 없고, 독립적으로 존재 할 수 있다.
- 값 영역들 안의 허용된 값들(permissible values)이 동일한 값의 의미를 공유할 때, 이들은 값 영역들은 개념적으로 연관성을 가진다. 즉, 같은 개념적 영역을 공유한다.
- 많은 값 영역들이 동일한 개념적 영역을 공유할 수 있다.
- 특정 데이터요소 개념은 오직 하나의 개념적 영역에 속한다. 그러므로 동일한 데이터요소 개념을 공유하는 데이터요소들은 개념적으로 유사한 표현방식(representation)을 공유한다.

2. 데이터요소의 분류(Classification for data elements)

데이터요소의 분류는 레지스트리를 관리하는 등록기관(Registry Authority)이 데이터 요소의 등록 시 고려 또는 사용할 수 있는 분류체계에 대해서 서술하기 위한 것이다. 또한, 이러한 분류체계에 공통적으로 적용될 수 있는 분류 속성을 기술하고 있다. 분류 스킴은 데이터 요소 자체로는 쉽게 표현될 수 없는 부가적 정보를 제공하며, 이를 통해 레지스트리의 내용을 조직화하여 정보를 명료하게 체계화는 데 도움을 주기 위한 것이다. 분류 스킴을 적용할 수 있는 대상은 객체 클래스(object classes), 속성(properties), 표현방식(representations), 데이터요소 개념(data element concepts), 데이터요소(data elements) 등이다

3. 레지스트리 메타모델과 기본 요소 (Registry Metamodel and basic attributes)

MDR은 메타 모델의 형태로 기술되어 있다. 메타 모델은 다른 모델을 기술하는 모델이며, 특정 모델의 구조와 구성요소를 이해하는데 도움을 주기 위한 것이다. 이용자나 응용프로그램이 이러한 특정 모델을 공유하는데 반드시 필요한 모델이다. 레지스트리는 데이터를 기술하거나 모델링 하는 데 사용된다. 결국, 레지스트리 메타 모델은 사람이 정보를 이해하는 방식과 가장 근접한 형식의 개념적 데이터 모델이다. MDR을 메타 모델로 표현함으써, 개념(concepts), 용어(terms), 값 영역, 값 의미 등에 대한 통일적 관점이 가능하게 되고, 기술하려고 하는 데이터에 대한 보편적 이해가 가능하게 되며, 구현 내용을 재사용 또는 공유하기가 용이하게 된다. MDR의 메타모델은 기능적으로 6분야로 나뉘어져 있다.

- 레지스트리의 공통적인 기능: 이 기능들은 등록할 데이터의 종류와 무관(다시 말하면, 모든 관리항목(administered items)하게 적용되는 독립적인 기능이며, 다중 레지스트리를 지원한다. 이

기능들은 일반적으로, 식별, 명명, 정의, 연락처, 관리, 참조원, 분류에 관한 정보를 제공한다. 이들 기능은 다음의 3가지로 구별 할 수 있다.

◆ 관리와 식별(Administration and Identification)
◆ 명명과 정의(Naming and definition)
◆ 분류(Classification)

● 특정 관리항목 유형의 서술에 관계하는 기능으로 다음의 3가지로 구분된다.

◆ 데이터 요소 개념(Data element concepts)
◆ 개념적 영역/값의 영역(Conceptual and value domains)
◆ 데이터 요소(Data elements)

A. 관리와 식별(Administration and Identification)

관리와 식별 부분은 다음과 같은 기능을 수행 한다: 1) 레지스트리에 제출된 항목의 식별과 등록, 2) 등록 항목을 제출하거나, 지출된 항목에 책임이 있는 조직의 식별, 3) 조직 연락처 정보, 4) 보조 문서의 등록 항목을 보조하는 참조 문서항목이 유효한 시기, 5) 주석, 이슈 등의 존재 확인, 6) 관리항목들 간의 관계 등이다.

B. 명명과 정의(Naming and definition)

MDR 5장은 데이터 정의에 관한 규칙과 지침을 제공한다. 여기에서는 문맥(context)안에서 관리항목(administered item)을 명명하고 식별하는 원리를 기술한다. 각 관리항목은 하나 이상의 문맥 안에서 명명되고 정의된다. 문맥은 특정 데이터 가지는 의미의 범위를 정의한다. 문맥은 정보 주제의 영역, 데이터베이스, 파일, 데이터 모델 등과 같이, 레지스트리의 소유자가 정할 수 있는 환경이다. 각각의 문맥 또한 관리 항목이며, 이름과 정의를 가질 수 있다. 이 부분에 나오는 다른 항목들, 언어(language), 이름(designation 혹은 name), 정의(definition)간의 관계는 다음과 같다. 특정 문맥 안에 여러 언어가 정의될 수 있고, 특정 언어 안에 여러 이름이 존재 할 수 있으며, 또한 특정 언어 안에 여러 정의가 존재 할 수 있다.

C. 분류(Classification)

분류 부분은 분류 스킴과 관련 분류 스킴 항목을 등록, 관리하는 기능을 제공한다. 관리항목이 반드시 분류 스킴에 의해 분류될 필요는 없으나, 분류 시 하나 이상의 분류 스킴에 의해 분류 될 수 있다. 분류 스킴은 택소노미(taxonomy), 온톨로지(ontology), 또는 키워드 집합일 수도 있다. 각각의 분류 스킴은 관리항목이며, 객체 지향적 속성인 상속(inheritance)에 의해, 일반 관리항목이

갖는 속성과 관계성(relationships)을 갖는다. 다시 말해, 분류 스킴은 식별, 명명, 정의, 또는 분류될 수 있다. 관리항목과 마찬가지로, 분류 스킴은 하나 이상의 문맥(context) 안에서 이름을 가질 수 있다. 특정 관리항목이 주어진 분류 스킴 안에서 이름을 갖기 위해서는, 이 관리항목과 분류 스킴이 같은 문맥을 공유해야 한다.

D. 데이터 요소 개념(Data element concepts)

데이터 요소 개념 부분은 데이터 요소의 기반이 되는 개념에 대한 정보를 관리하고자 하는데 있다. 이 부분의 모든 메타데이터 요소는 이를 위해 의미(semantics)를 표현하는데 초점이 맞추어져 있다. 개념은 어떠한 내/외부적 실제 표현방식 (representation)과는 별개로 독립적으로 존재한다. 데이터 요소 개념은 관련 데이터 요소가 없이도 관리항목으로 등록될 수 있다. 그러나 데이터 요소 개념은 하나의 개념적 영역(conceptual domain)과 연결 되어야만 한다. 개념적 영역은 특정 데이터 요소 개념의 모든 유효 값들의 의미를 나타낸다. 이 부분의 메타데이터 요소는, 개념과 개념 관계를 나타내는 객체 클래스와 데이터 요소 개념과 통합하여 사용될 수 있는 속성(property)으로 이루어져 있다.

E. 개념적 영역/값의 영역(Conceptual and value domains)

개념적 영역은 논리적 코드 집합을 나타내고, 값의 영역은 물리적 코드 집합을 나타낸다. 개념적 영역은 값의 의미의 집합으로 이루어져 있으며, 이 집합은 나열될 수 있는 집합이 있고 (예: ISO 3166: 나라이름), 규칙, 범위 등으로 표현되는 집합이 있다. 특정 값의 의미가 여러 허용 값(permissible values)을 가질 수 있으며 (즉, 다양한 표현방법(representation)을 통해 표현될 수 있으며), 각각의 허용값들은 서로 다른 나열될 수 있는 값의 영역에서 나올 수 있다.

표현(representation)의 주요 구성요소는 값의 영역이다. 즉, 표현은 값의 영역을 통해서 구체화된다. 그러나 값의 영역은 영역 안의 값들과 데이터 요소 개념과의 관계를 나타내지 않으며, 영역 안의 값들이 어떠한 의미를 갖는지도 나타내지 않는다. 특정 값의 영역은 하나의 개념적 영역에 속한다. 즉, 특정 값의 영역은 관련된 개념적 영역의 한 표현인 것이다. 예를 들어, ISO 3166의 경우, '국가 이름 코드'는 개념적 영역이고, ISO 3166안에 정의된 7개의 국가 이름 코드(공식 영어 이름, 공식 프랑스어 이름 등)는 값의 영역들이다. 값의 영역에는 데이터 타입이 정의되며, 필요한 경우, 치수(unit of measure)가 연계될 수 있다.

F. 데이터 요소(Data elements)

데이터 요소는 데이터 요소 개념을 표현한 것이며, 재사용과 공유가 가능하다. 표현 클래스(representation class)는 표현을 위한 분류 스킴이다. 이종의 표현 클래스에 속한 두 데이터 요소

는 이들 요소 간에 행하여 질 수 있는 절차나 기능(function)이 한정 될 수 있다. 예를 들어, "양(amount)"의 표현 클래스에 속한 요소와, "수(number)"라는 표현 클래스에 속한 요소가 있을 때, 이들 두 요소간의 내용(contents) 비교는 무의미 하다. 표현 클래스에는 코드(code), 셈(count), 통화(currency), 날짜(date) 등이 있다. 데이터 요소는 파생 규칙에 의해 생성될 수 있다. 이러한 경우, 이러한 데이터 요소에는 파생 규칙이 기술된다.

4. 데이터 정의 구성(Formulation of data definitions)

MDR 4장의 데이터정의에 관한 규칙과 지침을 기술한다. 데이터 정의 규칙은 다음과 같다.

- 단수 항목을 정의('종이들'에 대한 정의가 아닌 '종이'에 대한 정의)한다.
- 긍정문을 이용한 정의
- 동의어 등을 이용한 정의를 피하고, 개념을 가장 잘 설명할 수 있는 절이나 구를 이용하여 정의한다.
- 약어를 쓸 경우, 보편적으로 아는 약어만을 쓴다.
- 다른 정의를 섞어서 정의하지 않는다.

데이터 정의 지침은 다음과 같다.

- 항상 주어진 개념의 가장 중요한 의미를 표현한다.
- 정확하고, 명확하게 표현한다.
- 간결하게 표현한다.
- 독자적으로 쓸 수 있게 표현한다.
- 이론적 설명이나 기능 사용법, 영역 또는 절차 정보 등을 삽입하지 않는다.
- 순환 논리(circular reasoning)를 피한다. 두 개의 정의가 상대 정의를 이용해 자신을 정의 하지 않도록 한다.
- 서로 관련이 있는 정의에 대해서 일관된 용어와 구조를 사용한다.

5. 명명 및 식별 원리(Naming and identification principles)

MDR 5장은 데이터 요소 개념, 개념적 영역, 데이터 요소, 값의 영역 등의 관리항목에 관한 명명과 식별에 관한 지침을 제공한다. 등록 데이터(관리항목)를 식별을 위해서는, 각각의 등록항목을

식별할 수 있는 식별자가 필요하며, 이를 위해서 IRDI(International registration data identifier)를 사용할 수 있다. IRDI는 RAI(registration authority identifier), DI(data identifier), VI(version identifier)로 구성되어 있다. MDR에서, 각각의 관리항목은 문맥 안에서 최소 하나의 이름을 가져야 한다. 또한, MDR에서 네임스페이스는 문맥으로 정의 될 수 있으며, 특정 네임스페이스안의 이름들은 중복될 수 없다.

명명관례 (naming convention)는 주어진 관리항목이 정의된 문맥과 연계된 참조 문서에 정의된다. 체계적인 명명관례는 다음의 사항들을 고려하여야 한다.

- Scope: 명명관례가 지원하는 영역
- Authority: 이름을 확립한 주체
- Semantic rules: 의미 규칙 의미의 전달을 가능케 하는 규칙.
- Syntactic rules: 이름의 글자 배열에 관한 규칙
- Lexical rules: 동의어, 경멸어, 글자 길이, 철자 등에 관한 규칙
- Uniqueness rules: 동음이의어 등에 관련한 규칙.

6. 등록(Registration)

MDR 6장은 관리항목에 관한 메타데이터의 등록에 관한 표준이다. 등록의 대상이 되는 관리 항목으로는 데이터 요소(data element), 데이터 요소 개념(data element concept), 값의 영역(value domain), 개념적 영역(conceptual domain), 분류 스킴(classification scheme), 문맥(context), 객체 클래스(object class), 특징(property), 그리고 표현 클래스(representation class)가 있다. MDR 6장은 등록 대상이 되는 정보의 종류 외에도, 등록에 필요한 조건과 절차에 관하여 기술하고 있다. 또한 등록에 관여하는 아래의 조직들의 역할과 책임에 관하여 기술한다.

- 등록 요청 기관(Submitting Organization): 등록 대상이 되는 데이터를 기술하고, 보조 문서(참조 문서)등을 등록기관에 제출한다.
- 품질 관리 기관(Responsible Organization): 데이터의 의미, 이름, 영역 속성 등의 적합성을 체크/관리하며, 필요 시, 등록 데이터에 등록상태를 조정을 등록기관에 공시할 수 있다. 또한, 특정 속성 값이 불명확 할 경우, 그것에 대한 값을 결정 할 수 있다.
- 등록 기관(Registration Authority): 제출된 데이터에 대하여, 식별자와 적당한 등록 상태(registration status)를 부여한다. 이 과정에서 품질 관리 기관의 조언을 구할 수 있다. 등록 기관은 특정 데이터에 대해 결정 사항을 관계 등록 요청 기관이나 품질 관리 기관에 공시한다.
- 일반 사용자(User Community)

등록 상태(Registration status)는 동적인 상태와 정정인 상태로 나눌 수 있다. 동적 등록 상태에는 6개의 종류가 있다.

- 최상의 표준(Preferred Standard): 이용자를 위한 최적의 품질.
- 표준(Standard) 이용자를 위한 높은 품질.
- 적정(Qualified) - 필수 메타데이터 요소 충족, 품질 요구 사항에 부합.
- 기록(Recorded) - 필수 메타데이터 요소 충족.
- 후보(Candidate) 등록 상태 승격 제안 중.
- 미완성(Incomplete) 등록 요청자(submitter)가 제출한 관리항목. 가장 아래 단계.

정적 등록 상태에는 6개의 종류가 있다.

- 퇴역(Retired): 더 이상 사용되지 않고, 사용되어서도 안 됨.
- 대치(Superseded): 사용 권장되지 않으며, 새로운 관리항목(successor administered item)의 사용 권장.
- 과거(Historical): 과거에 외부에서 사용.
- 외부 표준(Standardized Elsewhere): 다른 레지스트리 단체에서 표준화됨.
- 레가시(Legacy): 특정 관리항목이 현재 로칼 영역에 존재하며 애플리케이션에 사용되나, 관련 정보가 거의 없음.
- 애플리케이션(Application) - 특정 관리항목이 현재 로칼 영역에 존재하며 애플리케이션에 사용되나, 논리적 수준으로 기술되어 있지 않음.

Ⅲ. 레지스트리의 종류

현재 사용되고 있거나, 개발이 추진되고 있는 레지스트리 종류를 살펴보면 다음과 같다.

■ ISO/IEC 11179 MDR: MDR은 기술, 관리, 공유의 대상이 되는 데이터의 의미적 정보를 기술하는데 중점을 두었다. 관리의 대상이 되는 데이터로는 데이터 요소, 데이터 요소 개념, 값의 영역, 개념적 영역, 문맥, 표현 클래스, 분류 스킴 등이 있다. 대표적 MDR로는 미국 환경보호청의 환경 데이터 레지스트리(http://www.epa.gov/edr)가 있다.

■ OASIS/ebXML XML 레지스트리: XML 스키마나 DTD를 등록한다. 관리 대상 데이터의 구문적 정보에 중점을 둔다. ebXML은 단일 전자 상거래 프레임워크를 제공하기 위한 표준으로서, 이미 여러 산업 분야에서 그 기능과 효용성을 입증 받고 있다. ebXML 표준은 여러 기술 표준으로 이루어져 있고, 이들 표준은 http://www.ebxml.org에서 제공되고 있다.

■ UDDI(Universal Description, Discovery, and Integration) 레지스트리: 기업이 제공하는 웹 서비스에 대한 웹 기반 전화번호부(white page)의 기능을 하며, 프로그램 간의 인터페이스 관리에 중점을 둔다. 현재 버전 3.0의 표준이 제공되고 있다. (http://www.uddi.org)

■ 데이터베이스 시스템 레지스트리: 데이터베이스의 운영에 필요한 스키마, 관계(relations), 무결성 제약조건 등의 정보를 관리하는데 목적이 있다.

■ CASE(Computer-Aided Software Engineering) Tool 레지스트리: 데이터베이스나 프로그램 코드를 생성하는데 필요한 정보를 관리하는데 목적이 있다.

■ 온톨로지 레지스트리: 온톨로지 레지스트리는 분산 환경에서 온톨로지를 등록/검색 할 수 있는 온톨로지의 저장소와 관련 기능을 포함한다. 온톨로지 레지스트리는 온톨로지 안에 포함된 개념적 구조체를 관리하여 개념간의 관계, 시간적, 공간적 추론 등에 대한 가능하게 한다.

■ 소프트웨어 컴포넌트 레지스트리: 재사용 가능한 소프트웨어 컴포넌트를 이용한 표준화된 시스템의 구축에 목적이 있으며, 기본 객체나 객체의 유형 등에 관한 정보를 관리한다.

■ 더블린코어 레지스트리: 더블린코어 레지스트리는 DCMI(Dublin Core Metadata Initiative) 어휘와 그 정의의 검색 및 발견을 용이하게 하게하며, 어휘간의 관계를 나타내기 위해 개발된 애플리케이션이다. 더블린 코어 레지스트리를 통해 어휘의 의미(semantics) 발견, 재사용 및 확장을 촉진하고, 새로운 어휘의 개발을 지원한다. (http://dublincore.org/dcregistry/index.html)

Ⅳ. 시맨틱웹 기반 MDR 설계

ISO 11179 MDR 표준은 스키마 설계 시 이미 등록된 데이터 요소에 대한 정확한 정의를 제공함으로써 데이터의 상호운용성을 높일 수 있는 방향으로 메타데이터 스키마 설계자들은 돕기 위한

것이다. 그러나 현 표준은 기계나 프로그램이 자동적으로 MDR에 정의된 데이터 요소의 의미를 이해하기 어렵다. 이런 단점을 극복하기 위해서, 이 논문은 MDR 메타데이터 모델에 정의된 클래스와 클래스 간의 관계를 RDF/OWL로 정의하여 모델의 기계가독과 의미 추론을 가능하게 하고, MDR 기반으로 구축한 레지스트리들 내에 정의된 데이터 요소의 의미를 보다 명확하게 하고자 한다. 구체적으로, MDR 레지스트리 메타데이터 모델과 속성(Metamodel and attributes)에 정의된 클래스, 클래스 간의 관계, 클래스 내에 정의된 데이터 타입과 변수들을 RDF/OWL을 이용하여 정의함으로써 정의된 메타데이터 요소의 의미에 대한 기계가독성을 높이고자 한다. 다음은 MDR에서 관리대상 아이템에 속하는 것 중에서 핵심적인 내용을 RDF/OWL로 표현한 것이다.

● 네임스페이스 선언

다음 <표 1>은 XML의 URI 개념을 활용하여 기계가독형 MDR 설계에 필요한 네임스페이스(RDF, RDF 스키마, OWL, 더블린코어)에 대한 선언과, 더블린코어를 이용하여 이 네임스페이스에 관한 메타데이터를 기술한 내용이다.

<표 1> ISO 11179 MDR의 RDF/OWL 정의에 필요한 네임스페이스 선언

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
           xmlns:owl="http://www.w3.org/2002/07/owl#"
           xmlns:dc="http://purl.org/dc/elements/1.1/"
           xmlns:dcq="http://purl.org/dc/terms/">
<rdf:Description rdf:about="http://www.dpc.or.kr/mdr/">
   <dc:title xml:lang="en-US">The MDR Expressed in OWL</dc:title>
   <dc:title xml:lang="kr">메타데이터 레지스트리의 핵심 클래스와 속성의 RDF/OWL 표현</dc:title>
   <dc:publisher xml:lang="kr">성균관대학교</dc:publisher>
   <dc:description xml:lang="kr">이 네임스페이스의 목적은 MDR의 클래스와 속성을 RDF/OWL 기반으로
표현하여 상호운용성과 데이터 요소의 재사용을 증진시키려는 것이다.
   </dc:description>
   <dc:language>en-US</dc:language>
   <dc:language>kr</dc:language>
   <dc:date>2003-10-10</dc:date>
</rdf:Description>
```

● **ISO 11179 MDR의 관리 아이템에 관한 OWL 클래스 선언 RDF/OWL 정의**

다음의 <표 2>는 MDR 관리 아이템을 OWL 클래스로 먼저 정의함으로써 다른 파트에서 이 클래스를 재활용할 수 있도록 선언하고 있다.

〈표 2〉 MDR의 관리 아이템의 OWL 클래스 선언

```
〈owl:Class rdf:id="ObjectClass"〉
〈owl:Class rdf:id="ConceptualDomain"〉
〈owl:Class rdf:id="DataElementConcept"〉
〈owl:Class rdf:id="DataElement"〉
〈owl:Class rdf id="Context"〉
〈owl:Class rdf:id="RepresentationClass"〉
〈owl·Class rdf·id="ClassificationScheme"〉
〈owl:Class rdf:id="Property"〉
〈owl:Class rdf:id="AdministeredItem"〉
〈owl·Class rdf id="AdministeredRecord"〉
〈owl:Class rdf:id="Concept"〉
〈owl:Class rdf id="ConceptRelationship"〉
〈owl·Class rdf id="ValueDomain"〉
〈owl:Class rdf:id="Concept"〉
〈owl·Class rdf·id="ConceptRelationship"〉
```

● **데이터요소 개념(Data Element Concept)에 대한 RDF/OWL 정의**

아래의 〈표 3〉은 MDR의 '데이터요소 개념(Data Element Concept)'을 RDF/OWL로 정의한 것이다. 여기에서 클래스 이름 앞에 '#' 기호가 붙은 것은 이미 〈owl:Class〉로 정의된 것을 재사용한다는 의미이다. [데이터요소 개념]은 '표현방식클래스', '파생규칙', '상황', '속성', '데이터요소', '객체클래스', '분류스킴', '개념영역', '값영역' 등과 disjoint 관계를 지니며, '관리 아이템'의 하위요소이고, 이 클래스의 영역(domain)은 오직 하나이어야 하며, 모든 값은 '개념영역'에서만 취해야 한다.

〈표 3〉 데이터 요소의 개념(Data Element Concept)에 대한 RDF/OWL 정의

```
〈owl·Class rdf:about="#DataElementConcept"〉
        〈owl·disjointWith rdf:resource="#RepresentationClass" /〉
        〈owl·disjointWith rdf:resource="#DerivationRule" /〉
        〈owl·disjointWith rdf·resource="#Context" /〉
        〈owl:disjointWith rdf·resource="#Property" /〉
        〈owl:disjointWith rdf:resource="#DataElement" /〉
        〈owl·disjointWith rdf:resource="#ObjectClass" 〉
        〈owl disjointWith rdf:resource="#ClassificationScheme" /〉
        〈owl·disjointWith rdf·resource="#ConceptualDomain" 〉
        〈owl:disjointWith rdf:resource="#ValueDomain" /〉
        〈rdfs:subClassOf〉〈owl:Class rdf·about="#AdministeredItem" /〉〈/rdfs:subClassOf〉
      〈rdfs:subClassOf〉
            〈owl:Restriction〉
                  〈owl:onProperty〉〈owl:ObjectProperty rdf:ID="domain" /〉〈/owl:onProperty〉
```

〈표 3〉 데이터 요소의 개념(Data Element Concept)에 대한 RDF/OWL 정의(계속)

```
                    〈owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"〉1〈/owl:
                    cardinality〉
                </owl:Restriction>
            </rdfs:subClassOf>
            〈rdfs:subClassOf〉
                〈owl:Restriction〉
                    〈owl:onProperty〉〈owl:ObjectProperty rdf:about="#domain" /〉〈/owl:onProperty〉
                    〈owl:allValuesFrom〉〈owl:Class rdf:about="#ConceptualDomain" /〉〈/owl:allValuesF
                    rom〉
                </owl:Restriction〉
            </rdfs:subClassOf〉
</owl:Class〉
```

● **객체 클래스(Object Class)에 대한 RDF/OWL 정의**

아래의 〈표 4〉는 MDR에서 정의된 '객체 클래스(Object Class)'를 기계적으로 그 의미를 파악할 수 있도록 RDF/OWL로 정의한 것이다. [객체 클래스]는 '개념영역', '데이터요소개념', '데이터요소', '상황', '표현방식클래스', '분류스킴', '속성', '값영역' 클래스와 disjoint 관계이고, 동시에 '개념'이나 '개념관계'와 동등 클래스로 정의할 수 있다.

〈표 4〉 객체 클래스(Object Class)에 대한 RDF/OWL 정의

```
〈owl:Class rdf:about="#ObjectClass"〉
        〈rdfs:subClassOf〉〈owl:Class rdf:about="#AdministeredItem" /〉〈/rdfs:subClassOf〉
        〈owl:disjointWith rdf:resource="#ConceptualDomain" /〉
        〈owl:disjointWith rdf:resource="#DataElementConcept" /〉
        〈owl:disjointWith rdf:resource="#DataElement" /〉
        〈owl:disjointWith rdf:resource="#ValueDomain" /〉
        〈owl:disjointWith rdf:resource="#Context(AdministeredItem)" /〉
        〈owl:disjointWith rdf:resource="#RepresentationClass" /〉
        〈owl disjointWith rdf:resource="#ClassificationScheme" /〉
        〈owl:disjointWith rdf:resource="#Property" /〉
        〈owl:equivalentClass〉
            〈owl:Class〉
                〈owl:unionOf rdf:parseType="Collection"〉
                    〈owl:Class rdf about="#Concept" /〉
                    〈owl:Class rdf:about="#ConceptRelationship" /〉
                〈/owl:unionOf〉
            〈/owl:Class〉
        〈/owl:equivalentClass〉
〈/owl:Class〉
```

● **'Property'에 대한 RDF/OWL 정의**

아래의 〈표 5〉는 MDR에서 정의된 '속성(Property)'을 기계적으로 그 의미가 파악될 수 있도록 RDF/OWL로 표현한 것이다. [속성]은 '개념영역', '데이터요소개념', '데이터요소', '상황', '표현방식클래스', '분류스킴', '객체 클래스', '값영역' 클래스 등과 disjoint 관계인 것으로 정의할 수 있다.

〈표 5〉 속성(Property)에대한 RDF/OWL 정의

```
<owl:Class rdf ID="Property">
        <owl:disjointWith rdf:resource="#RepresentationClass" />
        <owl:disjointWith rdf:resource="#DerivationRule" />
        <owl:disjointWith rdf:resource="#ClassificationScheme" />
        <owl:disjointWith rdf:resource="#Context" />
        <owl:disjointWith rdf:resource="#ObjectClass" />
        <owl:disjointWith rdf:resource="#ConceptualDomain" />
        <owl:disjointWith rdf:resource="#ValueDomain" />
        <owl:disjointWith rdf:resource="#DataElementConcept" />
        <owl:disjointWith rdf:resource="#DataElement" />
</owl:Class>
```

● **표현방식 클래스(Representation Class)에 대한 RDF/OWL 정의**

다음의 〈표 6〉은 MDR의 '표현방식 클래스(Representation Class)'를 기계적으로 그 의미를 파악할 수 있도록 RDF/OWL로 정의한 것이다. [표현방식 클래스]도 위에서 언급된 관리아이템에 속하는 다른 클래스와 disjoint 관계에 있고, '관리 아이템'의 하위 클래스로 간단하게 기술될 수 있다.

〈표 6〉 표현방식 클래스(Representation Class)에대한 RDF/OWL 정의

```
<owl:Class rdf:about="#RepresentationClass">
        <owl:disjointWith rdf resource="#DataElement" />
        <owl disjointWith rdf:resource="#DerivationRule" />
        <owl:disjointWith rdf:resource="#ClassificationScheme" />
        <owl disjointWith rdf resource="#Context" />
        <owl:disjointWith rdf:resource="#ObjectClass" />
        <owl:disjointWith rdf:resource="#ConceptualDomain" />
        <owl:disjointWith rdf:resource="#ValueDomain" />
        <owl:disjointWith rdf:resource="#DataElementConcept" />
        <owl:disjointWith rdf:resource="#Property" />
        <rdfs:subClassOf><owl:Class rdf:about="#AdministeredItem" /></rdfs:subClassOf>
</owl:Class>
```

● **개념 영역(Conceptual Domain)에 대한 RDF/OWL 정의**

다음 〈표 7〉은 MDR의 '개념영역(Conceptual Domain)'을 RDF/OWL로 정의한 것이다. 그 내용을 구체적으로 살펴보면, [개념영역]을 '객체클래스', '데이터요소개념', '데이터요소', '상황', '표현방식클래스', '분류스킴', '속성', '값 영역' 클래스와 disjoint 관계에 있는 것으로 정의하였고, 동시에 '열거식-개념영역', '비열거식-개념영역'과는 동등 클래스의 관계, '관리 아이템'의 하위요소로 정의하였다.

〈표 7〉 개념영역(Conceptual Domain)에 대한 RDF/OWL 정의

```
<owl:Class rdf:about="#ConceptualDomain">
        <owl:disjointWith rdf:resource="#Context" />
        <owl:disjointWith rdf:resource="#ObjectClass" />
        <owl:disjointWith rdf:resource="#ClassificationScheme" />
        <owl:disjointWith rdf:resource="#DataElement" />
        <owl:disjointWith rdf:resource="#ValueDomain" />
        <owl:disjointWith rdf:resource="#Property" />
        <owl:disjointWith rdf:resource="#RepresentationClass" />
        <owl:disjointWith rdf:resource="#DataElementConcept" />
      <rdfs:subClassOf rdf:resource="#AdministeredItem" />
        <owl:equivalentClass>
            <owl:Class>
                <owl:unionOf rdf:parseType="Collection">
                    <owl:Class rdf:about="#EnumeratedConceptualDomain" />
                    <owl:Class rdf:about="#NonEnumeratedConceptualDomain" />
                </owl:unionOf>
            </owl:Class>
        </owl:equivalentClass>
</owl:Class>
```

● **열거식 개념영역(Enumerated Conceptual Domain)에 대한 RDF/OWL 정의**

다음 〈표 8〉은 MDR의 '열거식 개념영역(Conceptual Domain)'을 RDF/OWL로 정의한 것이다. 그 내용을 구체적으로 살펴보면, [열거식 개념영역]은 '비열거식 개념영역'과 disjoint 관계에 있고, 동시에 '개념영역'의 하위클래스로 정의할 수 있다. 키 속성인 'representation'의 값은 '열거식 값영역'에서만 취해야 하고, 'memberValue'라는 속성의 값은 '값의미'에서 꼭 하나만 가능하도록 정의하였다.

〈표 8〉 열거식 개념영역(Enumerated Conceptual Domain)에 대한 RDF/OWL 정의

```
<owl:Class rdf about="#EnumeratedConceptualDomain">
        <owl:disjointWith rdf:resource="#NonEnumeratedConceptualDomain" />
        <rdfs subClassOf><owl:Class rdf:about="#ConceptualDomain" /></rdfs:subClassOf>
        <rdfs:subClassOf>
                <owl:Restriction>
                    <owl onProperty><owl:InverseFunctionalProperty rdf:about="#representation" /></owl:onProperty>
                    <owl.allValuesFrom><owl:Class rdf:about="#EnumeratedValueDomain" /></owl:allValuesFrom>
                </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs subClassOf>
                <owl:Restriction>
                     <owl onProperty><owl:ObjectProperty rdf.ID="memberValue" /></owl.onProperty>
                     <owl.allValuesFrom rdf:resource="#ValueMeaning" />
                </owl Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
                <owl:Restriction>
                    <owl:onProperty><owl:ObjectProperty rdf:about="#memberValue" /></owl:onProperty>
                    <owl:minCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int">1</ow
                    l:minCardinality>
                </owl Restriction>
        </rdfs:subClassOf>
</owl Class>
```

● **비열거식 개념영역(Non-Enumerated Conceptual Domain)에 대한 RDF/OWL 정의**

다음 〈표 9〉는 MDR의 '비열거식 개념영역(Conceptual Domain)'을 RDF/OWL로 정의한 것이다. 그 내용을 살펴보면, [비열거식 개념영역]은 '열거식 개념영역'과 disjoint 관계에 있고, 동시에 '개념영역'의 하위클래스로 정의할 수 있다. 키 속성인 'representation'의 값은 '비열거식 값영역'에서만 취해야 하고, 함수적속성인 'description'의 인스턴스는 꼭 한번만 가능하도록 정의하였다.

〈표 9〉 비열거식 개념영역(Non-Enumerated Conceptual Domain)에 대한 RDF/OWL 정의

```
<owl:Class rdf:ID="NonEnumeratedConceptualDomain">
        <owl:disjointWith>
                <owl:Class rdf:about="#EnumeratedConceptualDomain" />
        </owl:disjointWith>
        <rdfs:subClassOf>
            <owl:Class rdf:about="#ConceptualDomain" />
        </rdfs:subClassOf>
        <rdfs:subClassOf>
                <owl:Restriction>
                    <owl:onProperty><owl:InverseFunctionalProperty rdf:about="#representation" /></owl.onPrope
                     rty>
                    <owl:allValuesFrom><owl:Class rdf:ID="NonEnumeratedValueDomain" /></owl:allValuesF
                     rom>
                </owl:Restriction>
        </rdfs:subClassOf>
```

〈표 9〉 비열거식 개념영역(Non-Enumerated Conceptual Domain)에 대한 RDF/OWL 정의(계속)

```
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty><owl:FunctionalProperty rdf:ID="description" /></owl:onProperty>
                <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int">1</owl:card
                inality>
            </owl:Restriction>
        </rdfs:subClassOf>
</owl:Class>
```

- **값 영역(Value Domain)에 대한 RDF/OWL 정의**

아래의 〈표 10〉은 MDR의 '값 영역(Value Domain)'의 의미를 RDF/OWL로 정의한 것이다. [값 영역]은 '관리 아이템'으로 규정된 클래스(표현방식클래스, 파생규칙, 상황, 속성, 데이터요소개념, 데이터요소, 객체클래스, 분류스킴, 개념영역)들과 disjoint 관계에 있고, '열거식 값 영역'과 '비열거식 값 영역'으로 나뉘며, '관리 아이템'의 하위 클래스임과 동시에, 각 값의 영역의 의미는 하나이어야 하고, 한 데이터유형에 속해야 하고, 'application'이라는 키 속성에 의해서 기술되고, 이 속성의 값은 '데이터 요소'에 해당하는 값 중에서만 취해야 한다.

〈표 10〉 값 영역(Value Domain)에 대한 RDF/OWL 정의

```
<owl:Class rdf:about="#ValueDomain">
        <owl:disjointWith rdf:resource="#RepresentationClass" />
        <owl:disjointWith rdf:resource="#DerivationRule" />
        <owl:disjointWith rdf:resource="#ClassificationScheme" />
        <owl:disjointWith rdf:resource="#Context" />
        <owl:disjointWith rdf:resource="#ObjectClass" />
        <owl:disjointWith rdf resource="#ConceptualDomain" />
        <owl:disjointWith rdf:resource="#Property" />
        <owl:disjointWith rdf:resource="#DataElementConcept" />
        <owl:disjointWith rdf:resource="#DataElement" />
        <owl:equivalentClass>
            <owl:Class>
                <owl:unionOf rdf:parseType="Collection">
                    <owl:Class rdf:about="#EnumeratedValueDomain" />
                    <owl:Class rdf about="#NonEnumeratedValueDomain" />
                </owl:unionOf>
            </owl:Class>
        </owl:equivalentClass>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty><owl InverseFunctionalProperty rdf:ID="application" /></owl onProperty>
                <owl:allValuesFrom><owl:Classrdf:about="#DataElement" /></owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
```

〈표 10〉 값 영역(Value Domain)에 대한 RDF/OWL 정의(계속)

```
        〈rdfs:subClassOf〉〈owl:Class rdf:about="#AdministeredItem" /〉〈/rdfs:subClassOf〉
        〈rdfs·subClassOf〉
                〈owl·Restriction〉
                    〈owl:onProperty〉〈owl:FunctionalProperty rdf:about="#meaning" /〉〈/owl:onPropert
                    y〉
                    〈owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"〉1〈/ow
                    l:cardinality〉
                〈/owl:Restriction〉
        〈/rdfs·subClassOf〉
        〈rdfs·subClassOf〉
                〈owl:Restriction〉
                    〈owl:onProperty〉〈owl:FunctionalProperty rdf·ID="datatype" /〉〈/owl:onProperty〉
                    〈owl·cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"〉1〈/ow
                    l:cardinality〉
                〈/owl:Restriction〉
        〈/rdfs:subClassOf〉
〈/owl:Class〉
```

● **열거식 값영역(Enumerated Value Domain)에 대한 RDF/OWL 정의**

다음 〈표 11〉은 MDR의 '열거식 값영역(Value Domain)'을 RDF/OWL로 정의한 것이다. 그 내용을 구체적으로 살펴보면, [열거식 값영역]은 '비열거식 값영역'과 disjoint 관계에 있고, 동시에 '값영역'의 하위클래스로 정의된다. 그리고 함수속성인 'meaning'의 값은 '열거식 개념영역'에서만 취해야 하고, 'memberValue'라는 속성의 인스턴스는 적어도 2개 이상이어야 하고, 'memberValue'의 값은 '허용값'에서, 또 키 속성인 'component'의 값도 '허용값'에서만 취해야 한다.

〈표 11〉 열거식 값영역(Enumerated Value Domain)에 대한 RDF/OWL 정의

```
〈owl:Class rdf:about="#EnumeratedValueDomain"〉
        〈owl·disjointWith〉〈owl·Class rdf:about="#NonEnumeratedValueDomain" /〉〈/owl·disjointWith〉
        〈rdfs:subClassOf rdf:resource="#ValueDomain" /〉
        〈rdfs·subClassOf〉
                〈owl·Restriction〉
                    〈owl onProperty〉〈owl:FunctionalProperty rdf·about="#meaning" /〉〈/owl:onPropert
                    y〉
                    〈owl·allValuesFrom rdf:resource="#EnumeratedConceptualDomain" /〉
                〈/owl:Restriction〉
        〈/rdfs·subClassOf〉
        〈rdfs subClassOf〉
                〈owl:Restriction〉
                    〈owl·onProperty〉〈owl:ObjectProperty rdf:about="#memberValue" /〉〈/owl·onProperty〉
                    〈owl:minCardinality rdf datatype="http://www.w3.org/2001/XMLSchema#int"〉2〈/
                    owl:minCardinality〉
                〈/owl:Restriction〉
        〈/rdfs:subClassOf〉
```

〈표 11〉 열거식 값영역(Enumerated Value Domain)에 대한 RDF/OWL 정의(계속)

```
        〈rdfs·subClassOf〉
                〈owl:Restriction〉
                    〈owl onProperty〉〈owl:ObjectProperty rdf:about="#memberValue" /〉〈/owl onProperty〉
                    〈owl allValuesFrom rdf resource="#PermissibleValue" /〉
                〈/owl·Restriction〉
        〈/rdfs:subClassOf〉
        〈rdfs:subClassOf〉
                〈owl·Restriction〉
                    〈owl·onProperty〉〈owl·InverseFunctionalProperty rdf about="#component" /〉〈/owl:
                     onProperty〉
                   〈owl·allValuesFrom rdf·resource="#PermissibleValue" /〉
                〈/owl:Restriction〉
        〈/rdfs:subClassOf〉
〈/owl·Class〉
```

● **비열거식 값영역(Non-Enumerated Value Domain)에 대한 RDF/OWL 정의**

다음 〈표 12〉는 MDR의 '비열거식 값영역(Value Domain)'을 RDF/OWL로 정의한 것이다. 그 내용을 구체적으로 살펴보면, [비열거식 값영역]은 '열거식 값영역'과 disjoint 관계에 있고, 동시에 '값영역'의 하위클래스로 정의된다. 그리고 함수속성인 'meaning'의 값은 '비열거식 개념영역'에서만 취해야 하고, 또 다른 함수속성인 'description'의 인스턴스는 오직 하나만 허용된다.

〈표 12〉 비열거식 값영역(Non-Enumerated Value Domain)에 대한 RDF/OWL 정의

```
〈owl:Class rdf:about="#NonEnumeratedValueDomain"〉
        〈owl:disjointWith rdf:resource="#EnumeratedValueDomain" /〉
        〈rdfs:subClassOfrdf·resource="#ValueDomain" /〉
        〈rdfs:subClassOf〉
                〈owl:Restriction〉
                    〈owl:onProperty〉〈owl:FunctionalProperty rdf about="#meaning" /〉〈/owl:onProperty〉
                    〈owl:allValuesFrom rdf:resource="#NonEnumeratedConceptualDomain" /〉
                〈/owl:Restriction〉
        〈/rdfs·subClassOf〉
        〈rdfs:subClassOf〉
                〈owl:Restriction〉
                    〈owl:onProperty〉〈owl:FunctionalProperty rdf about="#description" /〉〈/owl:onProperty〉
                    〈owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"〉1〈/owl:cardi
                     nality〉
                〈/owl:Restriction〉
        〈/rdfs:subClassOf〉
〈/owl:Class〉
```

● **값 의미(Value Meaning)에 대한 RDF/OWL 정의**

다음 〈표 13〉은 MDR의 '값 의미(Value Meaning)'을 RDF/OWL로 정의한 것이다. 그 내용을 구체적으로 살펴보면, [값의미]는 'owl:Thing'의 하위클래스로 먼저 정의된다. 그리고 키 속성인 'identifier'의 인스턴스는 하나여야 하고, 그 값은 '값의미식별자'에서만 취해야 하며, 객체속성인 'containingDomain'의 인스턴스도 하나만 허용되고, 그 값은 '열거식개념영역'에서만 선택해야 한다. 데이터 속성인 'beginDate'의 인스턴스도 하나로 제한되며, 키 속성인 'representation'의 값은 '허용값'에서만 취할 수 있다.

〈표 13〉 값 의미(Value Meaning)에 대한 RDF/OWL 정의

```
<owl:Class rdf:about="#ValueMeaning">
        <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing" />
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty><owl:InverseFunctionalProperty rdf:ID="identifier" /></owl:onProperty>
                <owl:allValuesFrom><owl:Class rdf:ID="ValueMeaningIdentifier" /></owl:allValuesFrom>
            </owl:Restriction>
        <rdfs subClassOf>
            <owl:Restriction>
                <owl:onProperty><owl:InverseFunctionalProperty rdf:about="#identifier" /></owl onProp
                  erty>
                <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int">1</owl:cardi
                  nality>
            </owl:Restriction>
        </rdfs subClassOf>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty><owl:ObjectProperty rdf:about="#containingDomain" /></owl:onProper
                  ty>
                <owl:minCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int">1</owl:
                  minCardinality>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty><owl:ObjectProperty rdf:about="#containingDomain" /></owl:onProper
                  ty>
                <owl:allValuesFrom><owl Classrdf:ID="EnumeratedConceptualDomain" /></owl:allValues
                  From>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty><owl:DatatypeProperty rdf:about="#beginDate" /></owl:onProperty>
                <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int">1</owl:card
                  inality>
```

〈표 13〉 값 의미(Value Meaning)에 대한 RDF/OWL 정의(계속)

```
                    </owl Restriction>
            </rdfs:subClassOf>
            <rdfs:subClassOf>
                    <owl:Restriction>
                        <owl:onProperty><owl:InverseFunctionalProperty rdf:ID="representation" /></owl onProp
                          erty>
                        <owl:allValuesFrom rdf.resource="#PermissibleValue" />
                    </owl:Restriction>
            </rdfs.subClassOf>
</owl:Class>
```

● **가능한 값(Possible Value)에 대한 RDF/OWL 정의**

다음 〈표 14〉는 MDR의 '가능한 값(Possible Value)'를 RDF/OWL로 정의한 것이다. 그 내용을 구체적으로 살펴보면, [가능한 값]은 'owl:Thing'의 하위클래스로 먼저 정의된다. 그리고 함수속성인 'meaning'의 인스턴스는 하나여야 하고, 그 값은 '값의미'에서만 취해야 하며, 객체속성인 'containingDomain'의 인스턴스도 하나만 허용되고, 그 값은 '열거식값영역'에서만 선택해야 한다. 또 다른 객체속성인 'aggregate'의 값은 '열거식값영역'에서만 취해야 한다. 그리고 데이터 속성인 'beginDate'와 'value'의 인스턴스도 하나로 제한하였다.

〈표 14〉 가능한 값(Possible Value)에 대한 RDF/OWL 정의

```
<owl.Class rdf.ID="PermissibleValue">
        <rdfs:subClassOf rdf.resource="http://www.w3.org/2002/07/owl#Thing" />
        <rdfs:subClassOf>
                <owl:Restriction>
                        <owl:onProperty><owl:FunctionalProperty rdf:ID="meaning" /></owl:onProperty>
                        <owl'allValuesFrom><owl:Classrdf:ID="ValueMeaning" /></owl'allValuesFrom>
                </owl:Restriction>
        </rdfs'subClassOf>
        <rdfs subClassOf>
                <owl'Restriction>
                      <owl'onProperty><owl FunctionalProperty rdf:about="#meaning" /></owl'onProperty>
                     <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int">1</owl cardinality>
                </owl'Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
                <owl.Restriction>
                      <owl:onProperty><owl:ObjectProperty rdf:ID="containingDomain" /></owl:onProperty>
                      <owl.allValuesFrom><owl:Class rdf:about="#EnumeratedValueDomain" /></owl allValuesFrom>
                </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
                <owl:Restriction>
                     <owl:onProperty><owl'ObjectProperty rdf:about="#containingDomain" /></owl:onProperty>
                     <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int">1</owl:cardinality>
                </owl:Restriction>
```

〈표 14〉 가능한 값(Possible Value)에 대한 RDF/OWL 정의(계속)

```
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty><owl:ObjectProperty rdf:ID="aggregate" /></owl:onProperty>
                <owl:allValuesFrom><owl:Class rdf:ID="EnumeratedValueDomain" /></owl:allValuesFrom>
            </owl.Restriction>
        </rdfs.subClassOf>
        <rdfs.subClassOf>
            <owl:Restriction>
                <owl:onProperty><owl:DatatypeProperty rdf.ID="value" /></owl:onProperty>
                <owl'cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int">1</owl:cardinality>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty><owl:DatatypeProperty rdf'ID="beginDate" /></owl:onProperty>
                <owl'cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int">1</owl cardinality>
            </owl:Restriction>
        </rdfs:subClassOf>
</owl:Class>
```

- **데이터 요소(Data Element)에 대한 RDF/OWL 정의**

〈표 15〉는 MDR의 '데이터요소(Data Element)'를 RDF/OWL로 정의한 것이다. [데이터 요소]도 위에서 정의된 클래스와 같은 형식으로 관리 아이템에 속하는 다른 클래스들과 disjoint 관계에 있고, '관리 아이템'의 하위클래스에 속하고, [데이터 요소]의 의미는 '데이터요소의 개념'에서 다 오는 것이며, [데이터 요소]의 도메인이 하나로 국한되어야 하고, 그 도메인의 인스턴스는 '값의 영역'에서만 취해야 하며, [데이터 요소]는 한 '의미'만을 지녀야 한다고 정의한 것이다.

〈표 15〉 데이터요소(Data Element)에대한 RDF/OWL 정의

```
<owl'Class rdf'about="#DataElement">
        <owl'disjointWith rdf'resource="#RepresentationClass" />
        <owl:disjointWith rdf:resource="#DerivationRule" />
        <owl:disjointWith rdf:resource="#ClassificationScheme" />
        <owl:disjointWith rdf:resource="#Context" />
        <owl.disjointWith rdf.resource="#ObjectClass" />
        <owl:disjointWith rdf'resource="#ConceptualDomain" />
        <owl.disjointWith rdf'resource="#ValueDomain" />
        <owl:disjointWith rdf:resource="#DataElementConcept" />
        <owl:disjointWith rdf:resource="#Property" />
        <rdfs:subClassOf>
            <owl'Class rdf:about="#AdministeredItem" />
        </rdfs.subClassOf>
        <rdfs.subClassOf>
```

〈표 15〉 데이터요소(Data Element)에대한 RDF/OWL 정의(계속)

```
            〈owl:Restriction〉
                〈owl:onProperty〉〈owl:FunctionalProperty rdf:about="#meaning" /〉〈/owl:onProperty〉
                〈owl:allValuesFrom rdf:resource="#DataElementConcept" /〉
            〈/owl:Restriction〉
        〈/rdfs:subClassOf〉
        〈rdfs:subClassOf〉
            〈owl:Restriction〉
                〈owl:onProperty〉〈owl:ObjectProperty rdf:about="#domain" /〉〈/owl:onProperty〉
                〈owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"〉1〈/owl:cardinality〉
            〈/owl:Restriction〉
        〈/rdfs:subClassOf〉
        〈rdfs:subClassOf〉
            〈owl:Restriction〉
                〈owl:onProperty〉〈owl:ObjectProperty rdf:about="#domain" /〉〈/owl:onProperty〉
                〈owl:allValuesFrom rdf:resource="#ValueDomain" /〉
            〈/owl:Restriction〉
        〈/rdfs:subClassOf〉
        〈rdfs:subClassOf〉
            〈owl:Restriction〉
                〈owl:onProperty〉〈owl:FunctionalProperty rdf:about="#meaning" /〉〈/owl:onProperty〉
                〈owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"〉1〈/owl:cardinality〉
            〈/owl:Restriction〉
        〈/rdfs:subClassOf〉
〈/owl:Class〉
```

● **관리 아이템(Administered Item)에 대한 RDF/OWL 정의**

다음의 〈표 16〉은 MDR의 '관리 아이템(Administered Item)'을 RDF/OWL로 정의한 것이다. [관리 아이템]은 '관리 레코드'와 disjoint 관계에 있고, 키 속성인 'identifier'의 인스턴스는 한번으로 제한되어야 하고, 각 관리 아이템을 기술하는 '관리 레코드' 속성의 인스턴스도 하나만 가능하고, 키 속성인 'identifier'의 값은 'itemIdentifier' 에서만 추출되어야 하고, 'terminologicEntry' 속성의 인스턴스도 하나이어야 한다.

〈표 16〉 관리 아이템(Administered Item)에대한 RDF/OWL 정의

```
〈owl:Class rdf:about="#AdministeredItem"〉
        〈owl:disjointWith rdf:resource="#AdministrationRecord" /〉
        〈rdfs:subClassOf〉
            〈owl:Restriction〉
                〈owl:onProperty〉〈owl:InverseFunctionalProperty rdf:about="#identifier" /〉〈/owl:onProperty〉
                〈owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"〉1〈/owl:cardinality〉
            〈/owl:Restriction〉
        〈/rdfs:subClassOf〉
        〈rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing" /〉
        〈rdfs:subClassOf〉
            〈owl:Restriction〉
```

〈표 16〉 관리 아이템(Administered Item)에대한 RDF/OWL 정의

```
                    〈owl:onProperty〉〈owl:FunctionalProperty rdf:ID="administrationRecord" /〉〈/owl:onProperty〉
                    〈owl·cardinality rdf datatype="http://www.w3.org/2001/XMLSchema#int"〉1〈/owl:cardinality〉
                〈/owl·Restriction〉
            〈/rdfs:subClassOf〉
            〈rdfs:subClassOf〉
                〈owl·Restriction〉
                    〈owl onProperty〉〈owl InverseFunctionalProperty rdf·about="#identifier" /〉〈/owl onProperty〉
                    〈owl:allValuesFrom rdf·resource="#ItemIdentifier" /〉
                〈/owl·Restriction〉
            〈/rdfs:subClassOf〉
            〈rdfs:subClassOf〉
                〈owl Restriction〉
                    〈owl:onProperty〉〈owl:ObjectProperty rdf:ID="terminologicalEntry" /〉〈/owl·onProperty〉
                    〈owl·minCardinality rdf datatype="http://www.w3.org/2001/XMLSchema#int"〉1〈/owl minCar
                    dinality〉
                〈/owl:Restriction〉
            〈/rdfs:subClassOf〉
〈/owl Class〉
```

- **개념(Concept)에 대한 RDF/OWL 정의**

다음의 〈표 17〉은 MDR에서 정의된 '개념(Concept)' 클래스를 기계적으로 그 의미를 파악될 수 있도록 RDF/OWL로 정의한 것이다. [개념]은 '객체 클래스'의 하위 클래스로 정의할 수 있으며, '개념 관계'와는 disjoint 관계를 이루고 있다.

〈표 17〉 개념(Concept)에대한 RDF/OWL 정의

```
〈owl:Class rdf:ID="Concept"〉
        〈owl:disjointWith rdf·resource="#ConceptRelationship" /〉
        〈rdfs subClassOf〉〈owl·Class rdf·about="#ObjectClass" /〉〈/rdfs:subClassOf〉
〈/owl:Class〉
```

- **개념 관계(Concept Relationship)에 대한 RDF/OWL 정의**

아래의 〈표 18〉은 MDR의 '개념 관계(Concept Relationship)' 클래스를 RDF/OWL로 정의한 것이다. [개념 관계]는 '개념'과 '객체 클래스'의 하위 클래스이고, 객체 속성인 'item'의 값은 '개념'의 인스턴스에서만 취할 수 있다.

〈표 18〉 개념 관계(Concept Relationship)에대한 RDF/OWL 정의

```
<owl:Class rdf:about="#ConceptRelationship">
        <owl:disjointWith rdf:resource="#Concept" />
        <rdfs:subClassOf><owl:Class rdf:about="#Relationship" /></rdfs:subClassOf>
        <rdfs:subClassOf><owl:Class rdf:about="#ObjectClass" /></rdfs:subClassOf>
        <rdfs:subClassOf>
                <owl:Restriction>
                        <owl:onProperty><owl:ObjectProperty rdf:about="#item" /></owl:onProperty>
                        <owl:allValuesFrom rdf:resource="#Concept" />
                </owl:Restriction>
        </rdfs:subClassOf>
</owl:Class>
```

● **상황(Context)에 대한 RDF/OWL 정의**

아래의 〈표 19〉는 MDR에서 정의된 '상황(Context)'의 의미를 기계가 파악할 수 있도록 RDF/OWL로 표현한 것이다. [상황]도 '관리 아이템'으로 규정된 클래스와 disjoint 관계이고, '관리 아이템'의 하위 클래스로 정의하였다.

〈표 19〉 상황(Context)에대한 RDF/OWL 정의

```
<owl:Class rdf:about="#Context">
        <owl:disjointWith rdf resource="#RepresentationClass" />
        <owl:disjointWith rdf:resource="#DerivationRule" />
        <owl:disjointWith rdf:resource="#ClassificationScheme" />
        <owl:disjointWith rdf:resource="#DataElement" />
        <owl:disjointWith rdf:resource="#ObjectClass" />
        <owl:disjointWith rdf:resource="#ConceptualDomain" />
        <owl:disjointWith rdf resource="#ValueDomain" />
        <owl:disjointWith rdf:resource="#DataElementConcept" />
        <owl:disjointWith rdf:resource="#Property" />
        <rdfs:subClassOf><owl:Class rdf:about="#AdministeredItem" /></rdfs:subClassOf>
</owl:Class>
```

● **분류 스킴(Classification Scheme)에 대한 RDF/OWL 정의**

다음의 〈표 20〉은 MDR의 '분류 스킴(Classification Scheme)'이라는 클래스를 기계적으로 그 의미가 파악될 수 있도록 RDF/OWL로 정의한 것이다. [분류스킴]의 모든 인스턴스는 '분류스킴 아이템'에 속하는 것에서만 가능하고, 'component'를 키 속성으로 정의한 것이다.

〈표 20〉 분류 스킴(Classification Scheme)에대한 RDF/OWL 정의

```
<owl:Class rdf:about="#ClassificationScheme">
        <owl:disjointWith rdf:resource="#RepresentationClass" />
        <owl:disjointWith rdf:resource="#DerivationRule" />
        <owl:disjointWith rdf:resource="#DataElement" />
        <owl:disjointWith rdf:resource="#Context" />
        <owl:disjointWith rdf:resource="#ObjectClass" />
        <owl:disjointWith rdf resource="#ConceptualDomain" />
        <owl:disjointWith rdf:resource="#ValueDomain" />
        <owl:disjointWith rdf:resource="#DataElementConcept" />
        <owl:disjointWith rdf resource="#Property" />
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty><owl:InverseFunctionalProperty rdf:ID="component" /></owl:onProperty>
                <owl:allValuesFrom><owl:Class rdf:about="#ClassificationSchemeItem" /></owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
</owl:Class>
```

V. 결 론

이 논문에서는 데이터를 효율적으로 공유하기 위해서 제정된 국제 메타데이터 레지스트리 표준(ISO/IEC 11179)이 사람이 이해할 수는 있으나 기계나 에이전트가 이해할 수 없기 때문에 이로 인한 단점을 극복하기 위한 첫 단계로 MDR을 RDF/OWL을 활용한 기계가독형 MDR 설계에 대한 골격을 제시하였다. 앞으로 이렇게 시맨틱 기술을 기반으로 한 MDR이 활발하게 구축되어서 널리 활용된다면 다양한 형태의 고급 정보서비스가 가능할 것이다. 메타데이터 요소들에 대한 기계가독형 정의를 넘어서 온톨로지의 개념들과 메타데이터 스키마, 그리고 인스턴스까지 서로 통합된다면 이용자에게 더욱 정교한 맞춤서비스를 제공할 수 있을 것으로 기대된다. 앞으로 이 논문에 제시된 기본 설계를 바탕으로 시맨틱 기술을 활용한 MDR을 구축하여 실험연구가 진행되어야 시맨틱 기반 MDR의 유용성에 대한 객관적 평가가 가능할 것이다.

참 고 문 헌

Metadata Registry (MDR) Part 1: Framework - http://lis.skku.ac.kr/ohs/project/11179-Part-1.pdf

Metadata Registry (MDR) Part 2: Classification for Data Elements - http://lis.skku.ac.kr/ohs/project/11179-Part-2.pdf

Metadata Registry (MDR) Part 3: Registry Metamodel and Basic Attributes- http://lis.skku.ac.kr/ohs/project/11179-Part-3.pdf

Metadata Registry (MDR) Part 4: Formulation of Data Definitions - http://lis.skku.ac.kr/ohs/project/11179-Part-4.pdf

Metadata Registry (MDR) Part 5: Naming and Identification Principles - http://lis.skku.ac.kr/ohs/project/11179-Part-5.pdf

Metadata Registry (MDR) Part 6: Registration - http://lis.skku.ac.kr/ohs/project/11179-Part-6.pdf

OWL Web Ontology Language Overview: W3C Working Draft 31 March 2003 - href="http://www.w3.org/TR/owl-features/"

OWL Web Ontology Language Guide: W3C Working Draft 31 March 2003 - href="http://www.w3.org/TR/owl-guide/"

OWL Web Ontology Language Reference: W3C Working Draft 31 March 2003 - href="http://www.w3.org/TR/owl-ref/"