마스터 데이터를 활용한 메타데이터 통합 프레임워크 구축*

Construction of Framework for Metadata Integration Using Master Data Approach

이 승 민(Seungmin Lee)**

〈목 차〉

초 록

메타데이터 요소들 사이의 매핑에 기반한 현재의 메타데이터 상호운용성 접근방식이 지닌 한계를 극복하기 위해, 본 연구에서는 마스터 데이터를 활용한 메타데이터 상호운용성 접근방식을 제안하고 있다. 이는 메타데이터 요소가 아닌 요소의 값을 기준으로 동일한 값을 지니는 요소를 마스터 요소를 중심으로 통합하는 방식이다. 이를 위해 마스터 요소 프레임워크를 구축하여 메타데이터 요소들을 계층적인 형태로 의미적으로 통합하고, 이를 기반으로 메타데이터 레코드 사이의 상호연결을 확보함으로써 메타데이터 상호운용성을 확보할 수 있다. 이는 메타데이터 상호운용성을 확보하기 위한 대안적인 방법으로 활용될 수 있다.

키워드: 메타데이터 상호운용성, 메타데이터 통합, 마스터 데이터, 마스터 데이터 관리, RDF

ABSTRACT

In order to overcome the problems of current approaches to metadata interoperability based on element mapping, this research proposed Master Element Framework that is an alternative approach to metadata interoperability. It is an approach to integrate metadata elements that have the same value, instead of direct mapping between similar elements. Master Element Framework is constructed to semantically integrate metadata elements in a hierarchical order and to interconnect between heterogeneous metadata standards. This approach is expected to be an alternative approach to metadata interoperability.

Keywords: Metadata interoperability, Metadata integration, Master data, Master data management, Resource description framework

* 이 논문은 2011년도 충남대학교 학술연구비에 의해 지원되었음.
이 논문은 2012년 한국도서관·정보학회 동계학술발표대회에서 발표된 논문을 수정·보완한 것임.

** 충남대학교 사회과학대학 문헌정보학과 조교수(ableman@cnu.ac.kr)

• 접수일: 2013년 2월 19일 • 최초심사일: 2013년 3월 3일 • 최종심사일: 2013년 3월 26일

Ⅰ. 서 론

1. 메타데이터 상호운용성의 필요성

현재의 정보환경에서 메타데이터는 '데이터에 대한 데이터(data about data)'라는 의미를 벗어나 시맨틱웹의 기반으로 인식될 정도로 그 적용범위가 확장되고 있다. 도서관계에서의 메타데이터는 정보자원이 지니고 있는 여러 가지 측면들을 체계화 된 기본단위로 구분하여 표현해 주고 있으며, 이를 통해 정보자원에 대한 구체화 된 정보를 제공해 주고 있다. 메타데이터의 본질적인 목적은 일관된 방식으로 정보자원을 기술하여 이를 여러 기관이 서로 공유하는데 있다.[1] 하지만 여러 커뮤니티들이 자신들의 고유한 목적을 충족시키기 위하여 수많은 이질적인 메타데이터 표준들을 개발하면서, 한 기관에서 생성한 메타데이터 레코드가 다른 기관과 공유되지 못하고 특정 기관 내에서만 사용되는 경향을 보이고 있다. 이러한 문제를 해결하고 메타데이터 레코드의 효과적인 교환을 이루기 위해 메타데이터 상호운용성을 확보하는 방안이 계속해서 제시되어 왔다.

메타데이터 상호운용성은 정보자원의 기술 및 관리에 있어 여러 가지 장점을 지니고 있다. 상이한 메타데이터 표준에 수록된 요소들을 연결시킴으로써 메타데이터 레코드의 상호교환을 이룰 수 있으며, 하나의 메타데이터 표준을 이용해서 복수의 메타데이터 표준을 동시에 활용할 수 있다. 이러한 장점으로 인해, 도서관계에서는 메타데이터 상호운용성을 확보하기 위한 여러 가지 방법들을 개발해 왔다. 현재 사용되고 있는 대표적인 메타데이터 상호운용성 접근 방식으로는 메타데이터 요소 매핑, 크로스워크, 메타데이터 레지스트리, 어플리케이션 프로파일 등을 들 수 있다. 이들 접근방법은 동일한 의미를 갖는 요소들을 직접적으로 서로 연결시키는 1:1 매핑 방식에 기반을 두고 있다.[2] 하지만 이러한 접근방법들은 고유한 장점을 지니고 있는 반면, 신뢰성있는 메타데이터 상호운용성을 확보하는데 있어서 여러 가지 한계 또한 보이고 있다.

2. 메타데이터 상호운용성 접근방식의 한계

메타데이터 표준은 정보자원이 지니고 있는 여러 가지 측면을 표현하기 위해 다양한 요소들을 수록하고 있다. 하지만 정보자원의 동일한 측면을 다루는 요소라 하더라도, 그 의미적 범위는 수록된 메타데이터 표준에 따라 상이하게 나타나고 있다. 예를 들어, 정보자원의 표제를 기술할 경우, Dublin Core에서는 〈title〉 요소 및 한정어 〈alternative〉를 사용하여 표제와 관련된 모든 사항을

1) 남태우, 이승민, "메타데이터의 의미론적 확장에 관한 연구," 한국문헌정보학회지, 제44권, 제4호(2010. 11), p.380.
2) Lee, S.. *Building bridges: An integrated approach to metadata interoperability using Concept Meta-Framework Interoperability Schema (CMF)*. Doctoral Dissertation, School of Library and Information Science, Indiana University at Bloomington. (2010), p.4.

기술하고 있지만, MODS에서는 정보자원의 표제를 주제목, 부제목, 축약표제, 번역표제 등으로 세분하고 있으며, 이들 각각을 기술하기 위해 세분화 된 요소를 계층적으로 배정하고 있다.

이러한 문제를 해결하기 위해, 메타데이터 레지스트리나 어플리케이션 프로파일 등은 동일한 혹은 유사한 의미를 지닌 요소들을 레지스트리의 형식으로 구축하여 복수의 메타데이터 표준 사이의 상호운용성을 확보하는 방식을 취하고 있다. 하지만 이러한 접근방법 역시 근본적으로는 메타데이터 표준이 지닌 고유한 특성을 최소화하여 메타데이터 요소 사이의 직접적인 1:1 매핑에 기반을 두고 있기 때문에[3] 메타데이터 요소 사이에 발생하는 의미적 범위의 차이를 해소하지 못하고 있다. 또한, 요소들 사이의 매핑을 위해 각각의 메타데이터 요소가 지니고 있는 의미를 임의로 절단하여 이들 요소들을 직접적으로 연결하는 수평적 관계를 형성하고 있다. 이러한 의미적 절단은 매핑의 과정에서 많은 메타데이터 요소들이 누락되는 결과를 초래하고 있으며, 이로 인해 의미에 기반한 연결이 아닌 메타데이터 요소의 레이블에 의존하는 피상적인 연결에 그치고 있다.

메타데이터 표준이 계층적 구조를 지니고 있는 경우, 이러한 수평적 매핑의 방식은 메타데이터 표준이 지닌 구조적 특성을 충분히 반영하지 못하게 된다. 이로 인해, 상호연결되는 메타데이터 표준이 동일한 구조와 구문을 사용하지 않을 경우 하나의 메타데이터 요소가 여러 개의 요소와 연결됨으로써 결과적으로 1:1 매핑이 아닌 1:M의 연결이 이루어지게 된다. 또한, 하나의 메타데이터 표준이 복수의 다른 메타데이터 표준과 수평적으로 연결되기 때문에 수많은 상호운용성의 결과가 생성되는 문제를 일으키고 있다 (〈그림 1〉 참조).

〈그림 1〉 현재의 메타데이터 상호운용성 접근 방식

이러한 문제는 메타데이터 표준이 지닌 구조적인 특성에 기인하고 있다. 메타데이터 표준의 구조는 메타데이터 요소의 의미적 범위를 설정하는데 영향을 미치고 있으며, 이로 인해 상이한 구조적 특성을 지닌 메타데이터 표준에 수록된 요소 사이에는 의미적 범위의 차이가 발생하게 된다. 하지

3) Blanchi, C., & Petrone, J. "Distributed interoperable metadata registry," *D-Lib Magazine*, Vol.7, No.12(Dec. 2001). 〈http://www.dlib.org/dlib/december01/blanchi/12blanchi.html〉 [cited 2012. 9. 5].

만 정보자원으로부터 추출한 메타데이터 요소의 값은 변하지 않는 것이며, 상이한 메타데이터 표준을 사용하더라도 그 구문적인 표현에서만 차이가 날 뿐이다.

이에 본 연구에서는 메타데이터 요소가 아닌 요소의 값을 기준으로 상이한 메타데이터 표준에 수록된 메타데이터 요소를 의미적으로 통합하는 방식을 취하고자 한다. 이를 위해서는 각 메타데이터 요소에 반영되는 의미적 범위를 아우를 수 있는 상위단계에서의 요소를 통한 의미적 통합이 보다 효과적일 것이다. 이를 위해 본 연구에서는 마스터 데이터(master data)를 적용하여 메타데이터 요소 사이의 구조적, 의미적 연결을 위한 기준이 되는 체계를 구축하고, 메타데이터 표준 사이의 의미적 상호운용성을 확보하는 기반을 마련하고자 한다.

Ⅱ. 마스터 데이터 접근방식

1. 마스터 데이터의 개념

기업 활동 전반에 걸쳐 정보기술이 도입되면서, 폭발적으로 증가하는 기업 정보를 관리하기 위한 프로세스의 중요성이 점차 커지고 있다. 이에 따라 기업 경영에 필요한 여러 유형의 데이터를 체계적이고 효과적으로 활용하기 위해 다양한 메타데이터가 구축되고 있다. 하지만 데이터 객체의 다양성으로 인해 특정 목적을 충족시키기 위한 이질적인 메타데이터가 계속해서 생성되고 있으며, 동일한 객체에 대한 데이터가 중복되어 생성되거나 호환되지 않는 비효율성이 발생하고 있다. 이러한 문제를 해결하기 위해 기업 내 데이터 관리의 방식이 분산된 형태에서 중앙 집중적인 방식으로 변화하는 경향을 보이고 있다. 이러한 집중화 된 데이터 관리를 위해 마스터 데이터(master data)를 이용하는 방안이 널리 활용되고 있다.

마스터 데이터는 관계형 데이터베이스 관리 시스템(Relational Database Management Systems: RDBMS)의 개발과 함께 등장한 개념이다. 이는 분산된 컴퓨팅 환경에서 데이터의 처리와 관리의 효율성을 도모하기 위해 공통적으로 사용되는 개념들을 기준으로 상호연관된 데이터를 구분하기 위한 목적으로 사용되었다. 데이터베이스 분야에서 사용되던 마스터 데이터의 개념은 현재 기업 경영 분야에 널리 적용되고 있으며, 한 조직의 생산성을 증대하기 위해 데이터와 정보를 효과적으로 운용하는데 사용되고 있다.[4)]

기업 경영 분야에서 사용하는 마스터 데이터는 기업 활동에 필요한 정보를 관리하는 환경이 변화함에 따라 등장한 데이터 관리 어플리케이션으로서의 의미를 지니고 있다. 이는 기존의 기업정보 활용방식에서 발생하는 여러 가지 문제를 해결하고 데이터의 생성과 관리를 효율적으로 처리하기

4) Loshin, David. *Master Data Management*. Morgan Kaufmann OMG Press (2009), p.3.

위한 대안으로 등장하였다.

일반적으로 마스터 데이터는 기업 활동을 유지하는데 근간이 되는 항목들로 이루어진 기준 정보를 의미한다. 이 기준 정보는 기업 전반에 걸쳐있는 여러 어플리케이션에서 공통적으로 사용되는 핵심 객체들로 구성된다.[5] 이는 기업의 운영을 위해 활용하는 참조 데이터(reference data)로서, 고객정보나 제품정보, 자산정보, 판매자정보, 직원정보 등과 같은 기업의 핵심 자산을 표현하는 데이터로 이루어진다. 이들 데이터는 기업 내의 조직 등이 변하더라도 없어지지 않는 중심축과도 같은 데이터이며, 오랜 기간동안 동일하게 유지되는 특징을 지니고 있다.[6] 이를 종합해 보면, 마스터 데이터는 모든 기업 활동에 있어서의 기본이 되는 자료이며, 다른 관련된 정보를 관리하는데 사용될 수 있는 통제정보로서의 기능을 수행한다.

2. 마스터 데이터 관리

마스터 데이터는 기업 활동 전반에 걸쳐 존재하는 핵심적인 데이터이기 때문에, 마스터 데이터를 구성하는 요소들은 해당 데이터의 속성, 정의, 역할, 연결관계, 분류체계에 이르기까지 데이터가 지닌 다양한 측면들을 포괄적으로 표현해 주고 있다.[7] 하지만 기업 정보의 양이 급속하게 증가하고 그 유형이 다양화됨에 따라, 다양한 측면을 지니고 있는 마스터 데이터를 일관적으로 생성·관리하기 위한 표준화 된 체계를 구축해야 할 필요성이 제기되었다. 이러한 요구를 충족시키기 위해 마스터 데이터 관리(Master Data Management: MDM)의 개념이 등장하게 되었다.

마스터 데이터 관리(MDM)는 마스터 데이터를 통합적으로 관리하여, 기업 활동에 필요한 데이터를 일관성있고 정확하게 생성하고 운영하기 위한 프레임워크를 의미한다.[8] 이는 또한 여러 가지의 핵심적인 정보 및 레코드 체계를 기업 내 부서 혹은 기업 외부 기관과 공유하도록 해주는 시스템으로 정의되기도 한다. MDM은 분산된 형태가 아니라 집중화 된 하나의 관점에서 데이터를 관리하기 위한 일련의 프로세스이며, 각 어플리케이션에서 사용하는 데이터 가운데 핵심 데이터들을 중앙집중화하여 관리하고 이를 각 어플리케이션이 활용할 수 있도록 연결시켜 준다.[9] 즉, 분산되어 있는 여러 어플리케이션들은 직접적으로 연결되는 것이 아니라, MDM을 통해 필요한 핵심 데이터를 공동으로 활용할 수 있으며, 이 과정에서 MDM은 관련된 데이터를 일관적으로 관리하는 역할을 수행한다.[10]

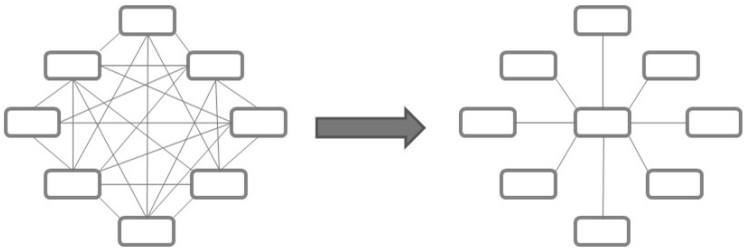
---

5) *Ibid.*, p.8.
6) Kerr, Karolyn. Metadata repositories in health care. In *Proceedings of 7th Annual Conference and Exhibition 2008*, October 15-17, 2008, Rotorua, New Zealand. <http://www.hinz.org.nz/uploads/file/2008conference/P11.pdf>, pp.3-4.
7) Berson, Alex & Dubov, Larry. *Master Data Management and Data Governance*, McGraw-Hill (2007), p.6.
8) Loshin, David, *op. cit.*, p.8.
9) Berson, Alex & Dubov, Larry, *op. cit.*, p.7.
10) Graham, Tyler & Selhorn, Suzanne. *Master Data Services*. McGraw-Hill Osborne Media (2011), p.3.

이러한 기능을 종합해 보면, 마스터 데이터는 기존의 시스템에 존재하고 있는 다양한 데이터 가운데 핵심적인 데이터를 의미하며, 여러 어플리케이션을 서로 연결시켜 줄 수 있는 공통의 특성을 지닌 데이터이다. MDM은 마스터 데이터를 일관성있게 생성·관리할 수 있도록 하는 프레임워크이며, 핵심적인 객체들을 집중화하고 특정 객체에 대한 메타데이터를 통합해 주는 기능을 한다. 이를 통해, 각각의 객체에 대한 핵심 정보가 여러 어플리케이션에 걸쳐 일관성있고 정확하게 교환될 수 있다.

3. 마스터 데이터 관리와 메타데이터 상호운용성

현재 메타데이터 표준은 각 기관의 고유한 목적을 충족시키기 위해 분산된 환경에서 독립적으로 구축되고 있다. 이들 메타데이터 표준 사이에는 요소가 지닌 의미적 범위의 불일치, 상이한 구문과 구조적 차이 등이 나타나고 있으며, 이로 인해 메타데이터 상호운용성을 확보하는데 여러 가지 문제가 발생하고 있다. 기업 경영에서 활용하고 있는 MDM 방식은 메타데이터 상호운용성에서 발생하는 여러 가지 문제를 해결할 수 있는 하나의 대안적인 접근방법으로 활용될 수 있다.



〈그림 2〉 MDM을 이용한 메타데이터 상호운용성 접근방식

〈그림 2〉에 나타난 바와 같이, 분산된 형태의 접근방식은 각각의 메타데이터 표준을 직접적으로 상호연결시켜야 하기 때문에 연결의 방식이 복잡하며 비일관적인 형태로 이루어진다. 반면, MDM의 개념을 메타데이터 상호운용성에 적용하면, 다양한 메타데이터 표준에 공통적으로 존재하는 핵심적인 구성요소들을 마스터 데이터로 추출하고 이를 기반으로 메타데이터 표준의 핵심 데이터를 통한 간접적인 관계를 형성하게 된다. 이러한 접근방식을 통해 독립적으로 분산되어 있는 메타데이터 표준을 집중화시켜서 메타데이터 레코드의 상호교환 및 메타데이터 표준의 통합을 가져올 수 있게 된다.

메타데이터에 적용하는 마스터 데이터는 메타데이터 상호운용성 프로세스의 핵심적인 정보로서, 각 메타데이터 표준에 수록된 요소들을 포괄할 수 있는 항목들로 구성된다. 또한, 각 메타데이터

요소의 의미, 속성, 기능 등 메타데이터 상호운용성에 필요한 개념들을 일관성 있는 방식으로 범주화하는 중재적 프레임워크로서의 기능을 하게 된다.

Ⅲ. 연구내용 및 방법

1. 메타데이터 표준 선정 및 분석

마스터 데이터를 기반으로 메타데이터 상호운용성을 확보하기 위한 첫 번째 단계는 메타데이터 표준을 선정하는 것이다. 본 연구를 위한 메타데이터 표준의 선정은 현재 도서관계에서 널리 사용되고 있는 서지적 메타데이터로 그 범위를 한정하였으며, 특정 목적을 위한 것이 아닌 일반적인 목적을 위해 구축된 메타데이터 표준을 대상으로 하였다. 또한, 연구의 목적을 충족시키기 위해 계층적 및 수평적 구조를 지닌 메타데이터 표준을 선정하였다. 이러한 기준을 충족시키는 메타데이터 표준으로 Machine Readable Cataloging eXtensible Markup Language (MARCXML), Dublin Core, Metadata Object Description Schema (MODS) 등 3개의 메타데이터 표준을 선정하였다.

선정된 메타데이터 표준 가운데 MODS는 계층적 구조를 채용하고 있으며, 이 구조에 따라 요소들 사이에는 의미적 포섭관계가 형성된다. 또한, 〈titleInfo〉 등과 같이 실제로 요소값을 갖지는 않지만 관련된 요소들을 묶어주는 상위요소를 사용하고 있으며, 이 요소들은 'type' 등과 같은 속성을 사용하여 하위요소들의 의미적 범위를 한정해 주고 있다. MARCXML은 XML 구문을 사용하여 MARC의 구조를 표현해 주는 메타데이터 표준이다. 따라서 MARC의 구조를 그대로 사용하고 있으며, 태그, 지시기호, 식별기호 등을 통한 기술 방식을 유지하고 있다. MARCXML에서는 'controlfield'와 'datafield'의 두 가지 요소가 상위요소로서의 기능을 하고 있다. 'datafield'는 'subfield'를 하위요소로 수록하고 있으며, 'subfield' 요소를 통해서 정보자원을 기술해 준다. 반면, Dublin Core는 상위요소를 사용하지 않고 모든 메타데이터 요소들을 열거하는 수평적인 구조를 지니고 있다. 또한, 한정어(qualifier)를 이용하여 Dublin Core 요소들의 의미를 세부적으로 구분하고 있으며, 한정어가 사용되는 경우에도 이들은 각각 독립적인 요소로 기능하고 있다.

2. 메타데이터 요소 사이의 연결

현재 메타데이터 상호운용성 접근방식에서는 메타데이터 표준에 수록된 요소들을 하나의 독립적인 개체로 간주하여 이들 사이의 1:1 매핑을 수행하고 있다. 하지만 메타데이터 표준 사이의 구조적 차이로 인해 상이한 메타데이터 표준에 수록된 요소들은 정보자원의 동일한 측면을 기술한다 하더라도 의미적인 범위의 차이를 보이고 있다.

<표 1> 메타데이터 상호운용성의 결과 (title 요소의 예)[11]

Dublin Core	MODS	MARCXML
–	<titleInfo>	245
<title>	<title>	245a
<title.alternative>	<subTitle>	245b
	<partNumber>	245p
	<partName>	245n
	<nonSort>	–

〈표 1〉에서는 정보자원의 'title'을 표현하는 요소들 사이의 1:1 매핑에 기반한 상호운용성의 결과를 보여주고 있다. 이 결과에서 보면, MODS의 〈titleInfo〉와 MARCXML의 245 필드는 상위요소로서 요소값을 갖지 않기 때문에 Dublin Core와의 연결은 이루어지지 않고 있다. 'main title'을 표현하는 요소들은 모든 메타데이터 표준에 공통적으로 수록되어 있어 1:1 매핑이 이루어지고 있지만, 그 외의 요소들은 의미적인 범위의 차이로 인해 Dublin Core의 〈title.alternative〉 요소와 1:M의 관계로 연결되고 있다.

정보자원은 일반적인 'title'만을 지니고 있는 것이 아니라, 경우에 따라서는 축약표제, 번역표제 등 변형된 형태의 'title'을 지니기도 한다. 이들 모두는 'title'과 밀접하게 관련되어 있으며, 이러한 변형된 형태의 'title'이 포함되는 경우에는 요소 사이의 연결이 더욱 복잡하게 나타난다.

<표 2> 메타데이터 상호운용성의 결과 (축약표제의 예)[12]

Dublin Core	MODS	MARC
–	<titleInfo type=abbreviated>	210
<title.alternative>	<title>	210a
	<subTitle>	–
	<partNumber>	–
	<partName>	–
	<nonSort>	–

〈표 2〉에 나타난 바와 같이, 축약표제를 표현하기 위해 MODS에서는 〈titleInfo〉 요소의 속성으로 'type=abbreviated'를 사용하고 있으며, 이 속성으로 인해 하위의 모든 요소들은 그 의미적 범위가 축약표제로 한정된다. 이 경우, MODS의 〈title〉 요소는 축약표제를 표현해 주는 MARC의

11) Baca, Murtha, ed. *Introduction to Metadata: Pathways to Digital Information*, Getty Information Institute (1998), pp.24-33.

12) *Ibid.*, pp.24-33.

210a와 매핑이 이루어지게 되므로, MODS의 〈title〉은 전체적인 상호운용성의 과정에서 MARC의 245a 및 210a와 동시에 매핑이 이루어지게 된다. 반면, Dublin Core의 〈title.alternative〉 요소는 'main title'을 제외한 모든 'title'을 표현하는데 사용되기 때문에, 모든 MODS의 요소와 매핑이 이루어진다.

이러한 문제는 메타데이터 요소 사이의 의미적 범위의 차이, 메타데이터 표준 사이의 구조적 차이에 기인하고 있다. 하지만 이들 메타데이터 표준은 모두 정보자원의 특정 측면으로부터 요소의 값을 추출한다는 공통점을 지니고 있다. 기술 대상이 되는 정보자원으로부터 추출한 요소의 값은 어떤 메타데이터 표준을 사용하더라도 변하지 않는 것이며, 여러 메타데이터 표준이 동일한 정보자원을 기술할 경우 이 값은 모든 메타데이터 표준에 공통적으로 적용된다. 따라서 이들 값은 상이한 메타데이터 표준에 수록된 메타데이터 요소들을 상호연결시킬 수 있는 근본적인 데이터이며, 메타데이터 상호운용성에서의 기준이 되는 데이터라고 볼 수 있다.

3. 마스터 요소 추출

앞서 언급한 바와 같이, 정보자원으로부터 추출한 메타데이터 요소의 값은 변하지 않는 것이며, 이는 복수의 메타데이터 표준을 동시에 활용할 수 있는 기반이 된다. 또한, 정보자원을 기술하는 측면도 대부분의 메타데이터 표준에 걸쳐 유사하게 나타나고 있다. 이들 각각의 측면들은 그 세부성의 정도에 따라 여러 메타데이터 요소를 통해 기술될 수 있으며, 이 요소들 사이에는 의미적인 관계가 형성된다.

〈그림 3〉 요소값을 기준으로 한 메타데이터 요소 사이의 관계 (title 요소의 예)

〈그림 3〉에서는 정보자원의 'title'과 관련된 요소들 사이의 관계를 보여주고 있다. 이 관계에서 기준이 되는 것은 정보자원이 지니고 있는 특정 측면(예를 들면, title)이며, 추출된 값은 메타데이

터 표준의 구문과 구조에 따라 다양한 메타데이터 요소를 통해 표현된다. 이 요소의 값은 정보자원의 특정 측면(예를 들면, title)을 표현하는 개념이며, 여러 관련된 메타데이터 요소를 의미적으로 통합할 수 있는 기준이 될 수 있다. 따라서 이 개념은 정보자원을 기술할 때 누락되어서는 안되는 핵심적인 항목이며, 메타데이터 상호운용성에 있어서는 다른 관련된 요소들을 관리하고 통제할 수 있는 마스터 요소(master element)라고 볼 수 있다.

마스터 요소를 이용한 메타데이터 상호운용성을 확보하기 위해서는 모든 관계의 기준이 되는 마스터 요소의 선정이 선행되어야 한다. 이를 위해 본 연구에서는 메타데이터 표준에 공통적으로 수록된 상위 단계의 메타데이터 요소들을 추출하였다. 추출된 요소들은 정보자원을 기술함에 있어서 필수적인 항목이며, 관련된 요소들을 포괄할 수 있는 넓은 의미적 범위를 지니고 있다 (〈부록 1〉 참조). 따라서 각 메타데이터 표준에 공통적으로 수록된 상위 단계의 요소들을 마스터 요소로 선정하였다.

이와 함께, 메타데이터 표준에 수록된 요소들 가운데 정보자원의 동일한 측면을 기술하는 요소들 사이의 의미적 관계를 분석하여 마스터 요소의 넓은 의미적 범위를 세분해 주는 속성으로 선정하였다. 이를 통해 총 11개의 마스터 요소를 선정하였으며, 22개의 세부적인 메타데이터 요소를 마스터 요소의 속성으로 선정하였다 (〈표 3〉 참조).

〈표 3〉 마스터 요소 및 속성

마스터 요소	속성	마스터 요소	속성
title	mainTitle	identifier	–
	subtitle	description	abstract
	titlePart		tableOfContent
	remainder	copyright	–
publication	publisherName	format	extent
	publicationPlace		medium
creator	name	type	–
	address	subject	–
	role	relation	version
date	dateIssued		replace
	dateCreated		part
	dateModified		format
	dateCopyright		source

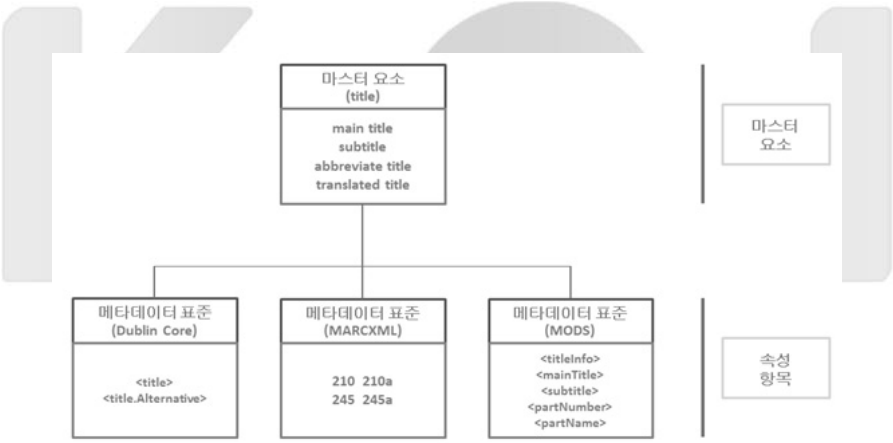
〈표 3〉에 나타난 바와 같이, 마스터 요소는 정보자원의 기술에 필수적으로 사용되는 측면들로 이루어져 있으며, 그 의미를 세분해 주는 관련 속성들을 하위에 수록한다. 각각의 속성들은 기존의

메타데이터 표준에 수록된 메타데이터 요소들로 이루어진다. 마스터 요소는 상호배타적인 독립적인 개념이 되어야 하며, 속성으로 사용되는 관련된 메타데이터 요소들을 의미적으로 포괄해 준다.

4. 마스터 요소 프로파일

마스터 요소에 수록되는 속성들은 여러 메타데이터 표준으로부터 추출된 메타데이터 요소이기 때문에, 각 메타데이터 표준에 따라 다양한 구문으로 표현되고 있다. 따라서 이러한 이질적인 구문의 형태를 조정하여 마스터 요소의 의미를 일관성있게 표현해 줄 수 있는 일반화 된 구조가 필요하다. 이를 위해, 각각의 마스터 요소 및 속성의 구문을 통합해 주는 마스터 요소 프로파일을 구축하였다.

마스터 요소 프로파일은 마스터 요소와 속성 사이의 관계를 설정해 준다. 이는 마스터 요소 프레임워크의 기본 구조를 제공해 주는 과정이며, 각 메타데이터 표준이 지니고 있는 상이한 구문을 조정하여 일관성있는 방식으로 관련된 속성들을 연결시켜 준다.



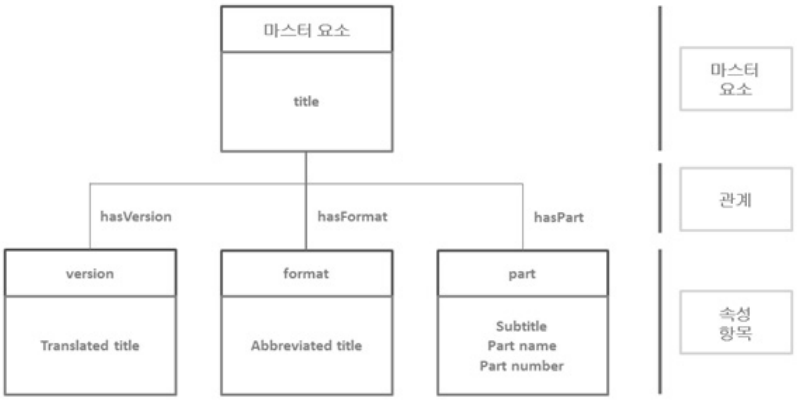
〈그림 4〉 마스터 요소 프로파일의 구조 (title 요소의 예)

〈그림 4〉에 나타난 바와 같이, 마스터 요소는 마스터 요소 프로파일의 상위 구조를 형성한다. 마스터 요소와 속성 사이에는 계층적인 관계가 형성되며, 관련된 속성들은 마스터 요소의 의미적 범주 내에서 열거되는 수평적 관계를 형성한다. 이러한 혼합된 구조를 통해 마스터 요소는 여러 가지의 관련된 속성을 하위에 수록하게 되며, 속성 카테고리에 수록된 메타데이터 요소들은 정보자원으로부터 추출한 값을 마스터 요소와 연결시켜 주는 기능을 한다. 이러한 구조를 통해서, 마스터 요소는 정보자원을 기술하는데 필요한 관련 요소들을 하나의 의미적 범주로 통합해 주며, 메타데이터 표준 사이에서 발생하는 의미적 범위의 차이를 중재하게 된다.

Ⅳ. 마스터 데이터 프레임워크의 구축

1. RDF/RDFS를 이용한 마스터 요소 프로파일 변환

마스터 요소 프로파일을 통해 구조화 된 마스터 요소와 속성들은 메타데이터 레코드를 통합하기 위한 프레임워크를 형성한다. 하지만 마스터 요소 프로파일이 관련된 메타데이터 요소들을 통합하기 위해서는 구체화 된 구문을 통해서 표현되어야 한다. 본 연구에서는 마스터 요소 프로파일의 표현을 위해 XML 기반의 RDF/RDFS 모델을 사용하였다. 이는 개념적으로 연결된 마스터 요소 및 속성들을 구문적으로 설명해 주며, 각 메타데이터 표준에 수록된 요소들을 마스터 요소가 설정하는 의미적 범주로 통합시켜 주는데 사용된다. 이 의미적 통합의 과정에서는 마스터 요소와 속성 사이의 관계가 형성되는데, 이 관계는 마스터 요소의 세부적인 측면(part), 상이한 형식(format) 및 버전(version)의 세 가지 유형으로 구분할 수 있다 (〈그림 5〉 참조).



〈그림 5〉 마스터 요소를 통한 속성의 의미 통합 (title의 예)

〈그림 5〉에서는 마스터 요소인 'title'과 속성 사이에 형성되는 관계를 나타내고 있다. 모든 관계는 마스터 요소를 기준으로 계층적으로 이루어지고 있으며, 그 연결되는 유형에 따라 속성의 의미를 범주화하고 있다. 이 구조는 RDF/RDFS 구문을 이용하여 마스터 요소 프레임워크를 구축하게 되며, 마스터 요소의 각각의 속성들은 실제적으로 값을 가질 수 있는 메타데이터 요소로서의 기능을 한다. 이들 속성들은 의미적인 중복을 지닐 수 있는데, 이를 해소하고 각 속성들 사이의 의미적 상호배타성을 확보하기 위해 마스터 요소 프로파일에서 정의된 관계를 활용한다.

마스터 요소의 속성은 상이한 메타데이터 표준에서 추출한 것이기 때문에, 정보자원의 동일한 측면을 기술한다 하더라도 이들 사이에는 의미적인 범위의 차이가 발생한다. 이 의미적인 범위의 차이를 절단하지 않고 마스터 요소의 의미를 세분해 주는 속성으로 사용하기 위해 Relation 클래스를 정의하였다. Relation 클래스는 마스터 요소 프로파일에서 분석한 메타데이터 요소 사이의 의미적 불일치의 유형에 기반한 것으로, 'version', 'format', 'part'의 세 가지 유형으로 구성된다. 이들 각각의 유형은 Relation 클래스의 하위 클래스로 정의된다 (〈그림 6〉 참조).

```
〈rdfs:Class rdf:about="#Relation"〉
  〈rdfs:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/〉
〈/rdfs:Class〉
〈rdfs:Class rdf:about="#Version"〉
  〈rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/〉
  〈rdfs:subClassOf rdf:about="#Relation"/〉
〈/rdfs:Class〉
〈rdfs:Class rdf:about="#Part"〉
  〈rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/〉
  〈rdfs:subClassOf rdf:about="#Relation"/〉
〈/rdfs:Class〉
〈rdfs:Class rdf:about="#Format"〉
  〈rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/〉
  〈rdfs:subClassOf rdf:about="#Relation"/〉
〈/rdfs:Class〉
```

〈그림 6〉 마스터 요소 프레임워크의 Relation 클래스 정의

하지만 이들 클래스만으로는 상이한 메타데이터 표준이 지닌 구조적 차이 혹은 세분화 정도의 차이를 해소할 수가 없으며, 각각의 메타데이터 표준에 수록된 요소들과 마스터 요소 사이의 관계를 RDF 구문으로 표현한 것에 지나지 않게 된다. 이러한 차이를 해소하고 의미적으로 중복되는 부분을 해소하기 위해서는 마스터 요소 내의 하나의 속성이 다른 속성과 갖는 관련성을 기술해 줄 필요가 있다. 이를 위해, Semantics 클래스를 정의하였다. Semantics 클래스는 마스터 요소가 수록하고 있는 속성들 사이의 의미적 관계를 설정하는데 사용된다 (〈그림 7〉 참조).

```
〈rdfs:Class rdf:about="Semantics"〉
  〈rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/〉
〈/rdfs:Class〉
〈rdfs:Class rdf:about="#Superordinate"〉
  〈rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/〉
  〈rdfs:subClassOf rdf:about="#Relation"/〉
〈/rdfs:Class〉
〈rdfs:Class rdf:about="#Subordinate"〉
  〈rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/〉
  〈rdfs:subClassOf rdf:about="#Relation"/〉
〈/rdfs:Class〉
```

〈그림 7〉 마스터 요소 프레임워크의 Semantics 클래스 정의

이와 함께, 요소의 값을 갖는 속성들은 Property로 정의하였다. 이들 Property는 메타데이터 요소로서의 특성을 지니고 있지만, 마스터 요소 프레임워크에서는 마스터 요소의 하위에 수록되며, 마스터 요소의 의미를 세분해 주기 위해 고유한 구문을 사용하지 않고 RDF/RDFS를 이용하여 표현된다. 〈그림 8〉은 정보자원의 title과 관련된 측면을 기술하기 위한 Property 정의를 보여주고 있다.

```
〈rdf:Property rdf:about="#title"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#mainTitle"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
  〈rdfs:subPropertyOf rdf:about="#title"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#subtitle"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
  〈rdfs:subPropertyOf rdf:about="#title"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#titlePart"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
  〈rdfs:subPropertyOf rdf:about="#title"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#remainder"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
  〈rdfs:subPropertyOf rdf:about="#title"/〉
〈/rdf:Property〉
```

〈그림 8〉 마스터 요소 프레임워크의 Property 정의 (title의 예)

이와 같이, 마스터 요소 프레임워크를 이용해 메타데이터 상호운용성을 확보하기 위해서는 각각의 메타데이터 요소를 마스터 요소 프레임워크의 RDF/RDFS 구문으로 통합하는 과정이 필요하다. 이 과정에서, 속성으로 사용되는 각각의 메타데이터 요소들 사이에 존재하는 관계는 RDF/RDFS를 통해 표준화 된 구문으로 표현되며, 이들 클래스와 Property의 집합은 마스터 요소 프레임워크의 스키마를 구축하게 된다 (〈부록 2〉 참조). 구축된 마스터 요소 프레임워크 스키마를 통해 메타데이터 요소가 지닌 의미적 범위를 임의로 절단하지 않고 이를 마스터 요소의 의미를 세분해 주는 속성으로 변환시키게 된다. 이를 통해, 복수의 메타데이터 표준을 상호연결시킬 수 있는 기반을 마련하게 된다.

2. 마스터 요소 프레임워크의 실행

마스터 요소 프레임워크를 이용한 메타데이터 상호운용성 접근방식은 기존의 메타데이터 상호운용성 접근방법 가운데 어플리케이션 프로파일 및 메타데이터 레지스트리 방식을 혼합한 것으로 볼 수 있다. 하지만 메타데이터 레지스트리 방식이 상이한 메타데이터 표준에 수록된 메타데이터 요소들 가운데 의미적으로 유사한 요소들을 종합하여 레지스트리 형식으로 저장해 놓고 이를 통해 메타데이터 표준 사이의 상호운용성을 확보하는 것인 반면, 마스터 요소 프레임워크는 메타데이터 레코드의 수준에서 레코드의 통합을 수행한다는 점에서 차이가 있다. 또한, 메타데이터 레지스트리와 어플리케이션 프로파일은 메타데이터 표준에 수록된 메타데이터 요소를 중심으로 메타데이터 상호운용성을 확보하는 방식을 취하고 있지만, 마스터 요소 프레임워크는 메타데이터 요소가 아닌 메타데이터 레코드에 수록되는 요소의 값을 기준으로 메타데이터 요소의 통합을 수행한다는 점에서 이들 두 방식과는 다른 접근방법을 취하고 있다. 즉, 메타데이터 레지스트리와 어플리케이션 프로파일의 장점을 유지하면서 요소의 값을 기준으로 통합을 이루기 때문에 메타데이터 요소 사이에서 발생하는 의미적인 차이를 해소할 수 있다.

〈그림 9〉는 MARCXML, MODS, Dublin Core 등의 형식으로 작성된 실제 정보자원(단행본)[13]의 레코드를 마스터 요소 프레임워크를 이용해 통합한 예를 보여주고 있다. 이 예에서 사용된 클래스와 Property는 마스터 요소 프레임워크 스키마에서 정의되어 있으며 (〈부록 2〉 참조), 통합된 레코드는 RDF/RDFS 모델을 사용하여 작성되었다.

13) Library of Congress Online Catalog. 〈http://catalog.loc.gov/cgi-bin/Pwebrecon.cgi?v1=2&ti=1,2&Search_Arg=zeng%20marcia%20lei&Search_Code=NAME%40&CNT=25&PID=Js8qwoFLZwbHGnm5A88zwpB8ZEM50&SEQ=20130311090037&SID=2〉 [cited 2013. 03. 10].

```
〈?xml version="1.0"?〉
〈rdf:RDF xml:lang="en"
        xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
        xmlns:marcxml="http://www.loc.gov/MARC21/slim"
        xmlns:mods="http://www.w3.org/2000/01/rdf-schema#Class"
        xmlns:dc="http://www.w3.org/2000/01/rdf-schema#Class"
        xmlns:mdf="http://ella.slis.indiana.edu/~seungmin/terms#"〉
〈rdf:Description rdf:about="http://lccn.loc.gov/2008015176"〉
〈mdf:title〉
  〈mdf:mainTitle〉 Metadata 〈/mdf:mainTitle〉
〈/mdf:title〉
〈mdf:creator superordinate="mods:name"〉
  〈mdf:name〉 Zeng, Marcia Lei 〈/mdf:name〉
  〈mdf:name relation="part" superordinate="marcxml:100"〉 1956- 〈/mdf:name〉
  〈mdf:role relation="version" subordinate="mods:name"〉 primary 〈/mdf:role〉
〈/mdf:creator〉
〈mdf:creator superordinate="mods:name"〉
  〈mdf:name〉 Qin, Jian 〈/mdf:name〉
  〈mdf:name relation="part" superordinate="marcxml:700"〉 1956- 〈/mdf:name〉
〈/mdf:creator〉
〈mdf:type〉 text 〈/mdf:type〉
〈mdf:publication superordinate="mods:place"〉
  〈mods:placeTerm〉 New York 〈/mods:placeTerm〉
  〈mods:publisher〉 Neal-Schuman Publishers 〈/mods:publisher〉
〈/mdf:publication〉
〈mdf:date〉
  〈mdf:dateIssued〉 c2008 〈/mdf:dateIssued〉
  〈mdf:dateCopyright subordinate="marcxml:260"〉 2008 〈/mdf:dateCopyright〉
〈/mdf:date〉
〈mdf:identifier superordinate="marcxml:020"〉 9781555706357 〈/mdf:identifier〉
〈mdf:identifier superordinate="mods:identifier" schema="lccn"〉 2008015176 〈/mdf:identifier〉
〈mdf:subject scheme="lcsh" superordinate="mods:topic"〉 Metadata 〈/mdf:subject〉
〈mdf:subject scheme="lcsh" superordinate="dc:subject"〉 Metadata 〈/mdf:subject〉
〈mdf:description〉
  〈mdf:tableOfContent superordinate="marcxml:504" subordinate="mods:tableOfContent"〉
  Introduction -- Current standards -- Schemas : structure and semantics -- Schemas : syntax -- Metadata
  records -- Metadata services -- Metadata quality measurement and improvement
  〈/mdf:tableOfContent〉
〈/mdf:description〉
```

〈그림 9〉 마스터 요소 프레임워크를 이용한 메타데이터 레코드 통합의 일부 예

〈그림 9〉에 나타난 바와 같이, 마스터 요소 프레임워크에서는 각 메타데이터 표준의 요소들이 마스터 요소를 기준으로 통합된다. 메타데이터 요소들은 직접적인 1:1 매핑을 통한 연결 대신 관련된 마스터 요소의 하위에 수록되어 마스터 요소의 의미를 세분해 주는 속성으로 사용된다. 또한, 마스터 요소의 속성들은 마스터 요소의 의미를 구성하게 되는데, 정의된 각 클래스의 인스턴스는 속성으로 사용된 메타데이터 요소에 적용되고 있다.

이 가운데, Relation 클래스의 하위 클래스 인스턴스는 메타데이터 요소가 마스터 요소의 의미를 구성하는 방식을 지시해 준다 (e.g., 〈mdf:subtitle relation="part"〉). 마스터 요소가 관련된 메타데이터 요소를 직접 사용할 경우 Semantics 클래스의 하위 클래스를 사용해서 메타데이터 요소가 수록된 메타데이터 표준을 명시해 준다 (e.g., 〈mdf:publication superordinate="mods:place"〉). 이 예에서 mdf:publication의 의미는 mods:place의 상위개념으로 사용되는 것을 나타내며, 이들 사이에는 isSuperordinateOf의 관계가 형성된다. 이때, mods:place는 mdf:publication의 의미를 구성하며, mdf:publication이 지닌 의미적 범위를 세분해 주는 기능을 한다.

또한, 필요한 경우, 각 메타데이터 표준에서 사용하는 메타데이터 요소를 관련된 마스터 요소의 하위에 직접 포함시킴으로써, 각 메타데이터 표준에 수록된 모든 요소들을 상호운용성의 과정에서 활용할 수 있다 (e.g., 〈mods:nonSort semantics="subordinate"〉). 이는 요소들 사이의 1:1 매핑에서 누락되는 요소들을 마스터 요소 프레임워크에 포함시키게 되며, 각각의 메타데이터 요소가 지닌 의미적 범위를 그대로 유지함으로써 의미적 범위의 절단을 방지할 수 있다.

상호운용성의 측면에서 보면, 메타데이터 표준에 수록된 요소들이 마스터 요소의 하위로 집중되어 마스터 요소의 카테고리 내에서는 관련된 모든 메타데이터 요소들이 서로 통합될 수 있다. 이를 통해, 요소의 의미적인 세분성의 정도에 따른 계층적인 형식으로 요소들 사이의 관계가 확인될 수 있다. 이외에도, 기존의 메타데이터 표준의 형식이나 구조를 변경하지 않으면서 메타데이터 요소의 의미를 일반화시키기 때문에, 각 메타데이터 표준의 고유한 특성이나 목적을 그대로 유지하면서 통합적인 방식으로 메타데이터 레코드를 관리할 수 있게 된다.

본 연구에서 제안한 마스터 요소 프레임워크는 기존의 메타데이터 상호운용성 접근방식과는 달리 메타데이터 요소에 중점을 두지 않고 요소의 값을 기준으로 상호운용성을 확보하는 방식이다. 메타데이터 표준 수준에서 상호운용성을 확보하는 메타데이터 레지스트리 방식과는 달리 레코드 생성 이후의 통합이기 때문에 메타데이터 요소 사이의 직접적인 연결방식을 취하지 않고 메타데이터 레코드를 통해 동일한 값을 지니는 요소들을 연결시킴으로써 메타데이터 상호운용성을 확보하는 방식이다. 이는 또한 선정된 메타데이터 표준에 따라 새로운 마스터 요소를 추가하거나 삭제할 수 있는 융통성을 제공할 수 있다. 이를 통해, 다른 유형의 메타데이터 표준으로 작성된 레코드를 손쉽게 추가할 수 있어 기존의 메타데이터 상호운용성 방식에 비해 확장성이 용이하다는 장점이 있다. 하지만 상호운용성의 대상이 되는 메타데이터 표준 자체가 갱신되거나 변경되었을 경우 마스터 요소 프레임워크 구축의

전 과정이 다시 수행되어야 한다는 한계가 있다. 또한, 기존의 메타데이터 레코드가 명확하게 작성되지 않았을 경우, 잘못된 메타데이터 레코드를 통한 상호운용성이 이루어지기 때문에 전체적인 메타데이터 상호운용성의 정확성이 기존의 메타데이터 레코드에 종속된다는 문제가 발생할 수 있다.

Ⅴ. 결 론

메타데이터 요소 사이의 1:1 매핑에 기반한 현재의 메타데이터 상호운용성 접근방식은 메타데이터 요소가 지닌 의미적, 구조적 차이 등을 충분히 반영하지 못하기 때문에 신뢰성있는 상호운용성을 확보하는데 있어 여러 가지 한계를 보이고 있다. 이러한 문제를 해결하기 위해, 마스터 데이터를 이용한 메타데이터 상호운용성 접근방법을 제안하였다.

본 연구에서 제안한 상호운용성 접근방법은 메타데이터 요소가 아닌 요소의 값을 기준으로 메타데이터 레코드의 통합을 수행한다는 점에서 기존의 메타데이터 레지스트리, 어플리케이션 프로파일 등과는 다른 접근방법을 보이고 있다. 메타데이터 레지스트리에서는 데이터 요소를 중심으로 메타데이터 속성을 관리할 수 있는 개념적인 프레임워크를 제공하지만, 마스터 데이터 기반의 접근방법은 메타데이터 요소의 값을 기준으로 삼기 때문에 메타데이터 레코드 수준에서의 상호운용성을 확보한다는 점에서 차이가 있다.

메타데이터 상호운용성에 적용한 마스터 데이터는 다양한 메타데이터 표준에 공통적으로 존재하는 핵심적인 구성요소들을 기준정보로 삼고, 이를 기반으로 관련된 메타데이터 요소를 통합하여 정보자원을 기술하는 방식이다. 마스터 데이터로 사용되는 마스터 요소는 정보자원을 기술하는데 있어 필수적인 항목들을 개념적으로 설정해 주며, 각각의 메타데이터 표준에 수록된 요소들은 마스터 요소의 속성으로서의 기능을 한다.

마스터 요소를 이용하여 메타데이터 상호운용성을 확보하기 위해서는 마스터 요소와 속성들을 구현할 수 있는 프레임워크가 필요하며, 이 프레임워크는 일반화 된 구문을 통해 표현되어야 한다. 본 연구에서는 RDF/RDFS 모델을 사용하여 마스터 요소와 속성을 표현하고 있으며, 이를 통해 관련된 메타데이터 요소를 통합할 수 있는 프레임워크를 구축하였다. 이 마스터 요소 프레임워크를 통해 상호 관련된 메타데이터 요소들이 의미적으로 통합되며, 이를 통해 메타데이터 레코드 사이의 상호 연결이 이루어지게 된다. 또한, 정보자원의 특정 측면을 기술하는데 있어서 여러 가지 메타데이터 표준의 요소들을 통합적으로 활용할 수 있으므로 보다 풍부하게 정보자원을 기술할 수 있으며, 메타데이터 레코드의 통합을 이룰 수 있게 된다. 마스터 요소 프레임워크를 이용한 접근방식은 메타데이터 요소들을 의미적으로 통합함으로써 메타데이터 상호운용성을 확보하기 위한 대안적인 방안으로 활용될 것으로 기대된다.

참고문헌

남태우, 이승민. "메타데이터의 의미론적 확장에 관한 연구." 한국문헌정보학회지, 제44권, 제4호 (2010. 11), pp.373-393.

Baca, Murtha, ed. *Introduction to Metadata: Pathways to Digital Information*, Getty Information Institute (1998).

Berson, Alex & Dubov, Larry. *Master Data Management and Data Governance*, McGraw-Hill (2007).

Blanchi, C., & Petrone, J. "Distributed interoperable metadata registry." *D-Lib Magazine*, Vol.7 No.12(Dec. 2001).
〈http://www.dlib.org/dlib/december01/blanchi/12blanchi.html〉 [cited 2012. 9. 5].

Chan, L. M., & Zeng, M. L. "Metadata interoperability and standardization - A study of methodology, Part I: Achieving interoperability at the schema level." *D-Lib Magazine*, Vol.12 No.6(June. 2006).
〈http://www.dlib.org/dlib/june06/chan/06chan.html〉 [cited 2012. 10. 25].

Graham, Tyler & Selhorn, Suzanne. *Master Data Services*. McGraw-Hill Osborne Media (2011).

Graybeal, J.B., Watson, S., & Bogden, P.S. "Marine metadata interoperability: A community framework." In *Proceedings of International Marine Data and Information Systems Conference*, May 31 - June 3, 2005, Brest, France (2005).
〈http://www.ifremer.fr/imdis/topics.htm〉 [cited 2012. 9. 6.].

Kerr, Karolyn. "Metadata repositories in health care." In *Proceedings of 7th Annual Conference and Exhibition 2008*, October 15-17, 2008, Rotorua, New Zealand (2008).
〈http://www.hinz.org.nz/uploads/file/2008conference/P11.pdf〉 [cited 2012. 9. 15].

Lee, Seungmin. *Building bridges: An integrated approach to metadata interoperability using Concept Meta-Framework Interoperability Schema(CMF)*. Doctoral Dissertation, School of Library and Information Science, Indiana University at Bloomington (2010).

Loshin, David. *Master Data Management*. Morgan Kaufmann OMG Press (2009).

Miller, P. "Interoperability: What is it and why should I want it?." *Ariadne*, 4(June 2000).
〈http://www.ariadne.ac.uk/issue24/interoperability/〉 [cited 2012. 9. 25].

국한문 참고문헌의 영어 표기
(English translation / Romanization of references originally written in Korean)

Nam, Taewoo and Lee, Seungmin. “Study on the Semantic Extension of the Concept of Metadata,” *Journal of the Korean Society for Library and Information Science*, Vol.44, No.4(Nov. 2010), pp.373-393.

부록 1. Dublin Core, MARCXML, MODS의 상위 단계 요소 분석

마스터 요소	Dublin Core	MODS	MARCXML (datafield tag의 값)
title	<title> <alternative>	<titleInfo> <title> <subtitle> <partNumber> <partName> <nonSort>	2XX
publication	<publisher>	<originInfo> <place> <publisher>	260
creator	<creator> <contributor>	<name> <namePart> <affiliation> <role> <description>	1XX, 7XX, 8XX
date	<date> <created> <valid> <available> <issued> <modified> <dateSubmitted>	<originInfo> <dateIssued> <dateCreated> <dateCaptured> <dateValid> <dateModified> <copyrightDate>	008/06, 260
identifier	<identifier>	<identifier>	020, 022, 024, 856
description	<description> <abstract> <tableOfContents>	<physicalDescription>	205, 300, 5XX
copyright	<rights> <accessRights>	<accessCondition>	017, 506, 540
format	<format> <extent> <medium>	<physicalDescription> <form> <extent> <note>	007, 25X, 300
type	<type>	<typeOfResource> <genre>	008, 047, 655
subject	<subject> <coverage> <spatial> <temporal>	<subject> <topic> <geographic> <temporal>	600, 610, 611, 630, 648, 650, 651, 653, 656, 662, 752
relation	<relation> <isVersionOf> <hasVersion> <isReplacedBy> <replaces> <isRequiredBy> <requires> <isPartOf> <hasPart> <isReferencedBy> <references> <isFormatOf> <hasFormat>	<relatedItem>	76X-78X

부록 2. 마스터 요소 프레임워크 스키마

```
〈rdf:Property rdf:about="#title"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#maintitle"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
  〈rdfs:subPropertyOf rdf:about="#title"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#subtitle"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
  〈rdfs:subPropertyOf rdf:about="#title"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#titlePart"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
  〈rdfs:subPropertyOf rdf:about="#title"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#remainder"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
  〈rdfs:subPropertyOf rdf:about="#title"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#publication"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#publisherName"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
  〈rdfs:subPropertyOf rdf:about="#publication"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#publicationPlace"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
  〈rdfs:subPropertyOf rdf:about="#publication"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#creator"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#name"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
  〈rdfs:subPropertyOf rdf:about="#creator"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#address"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
  〈rdfs:subPropertyOf rdf:about="#creator"/〉
〈/rdf:Property〉
```

```
〈rdf:Property rdf:about="#role"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
  〈rdfs:subPropertyOf rdf:about="#creator"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#date"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#dateIssued"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
  〈rdfs:subPropertyOf rdf:about="#date"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#dateCreated"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
  〈rdfs:subPropertyOf rdf:about="#date"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#dateModified"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
  〈rdfs:subPropertyOf rdf:about="#date"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#dateCopyright"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
  〈rdfs:subPropertyOf rdf:about="#date"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#identifier"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#description"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#abstract"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
  〈rdf:subPropertyOf rdf:about="#description"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#tableOfContent"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
  〈rdfs:subPropertyOf rdf:about="#description"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#copyright"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#format"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#extent"〉
```

```
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
  〈rdfs:subPropertyOf rdf:about="#format"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#medium"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
  〈rdfs:subPropertyOf rdf:about="#format"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#type"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#subject"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#relation"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#version"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
  〈rdfs:subPropertyOf rdf:about="#relation"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#replace"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
  〈rdfs:subPropertyOf rdf:about="#relation"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#part"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
  〈rdf:subPropertyOf rdf:about="#relation"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#format"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
  〈rdfs:subPropertyOf rdf:about="#relation"/〉
〈/rdf:Property〉
〈rdf:Property rdf:about="#source"〉
  〈rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/〉
  〈rdfs:subPropertyOf rdf:about="#relation"/〉
〈/rdf:Property〉
〈rdfs:Class rdf:about="#Relation"〉
  〈rdfs:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/〉
〈/rdfs:Class〉
〈rdfs:Class rdf:about="#Version"〉
  〈rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/〉
  〈rdfs:subClassOf rdf:about="#Relation"/〉
〈/rdfs:Class〉
〈rdfs:Class rdf:about="#Part"〉
```

```
  〈rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/〉
  〈rdfs:subClassOf rdf:about="#Relation"/〉
〈/rdfs:Class〉
〈rdfs:Class rdf:about="#Format"〉
  〈rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/〉
  〈rdfs:subClassOf rdf:about="#Relation"/〉
〈/rdfs:Class〉
〈rdfs:Class rdf:about="Semantics"〉
   〈rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/〉
〈/rdfs:Class〉
〈rdfs:Class rdf:about="#Superordinate"〉
  〈rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/〉
  〈rdfs:subClassOf rdf:about="#Relation"/〉
〈/rdfs:Class〉
〈rdfs:Class rdf:about="#Subordinate"〉
  〈rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/〉
  〈rdfs:subClassOf rdf:about="#Relation"/〉
〈/rdfs:Class〉
```