건설공사정보 메타데이터의 XML 스키마 설계에 관한 연구*

A Study on the Design of XML Schema for the Metadata of Construction Information

박 재 원(Jae-Won Park)**
최 재 황(Jae-Hwang Choi)***

목 차

초 록

본 연구의 목적은 국내 건설공사정보 DB들을 대상으로 메타데이터 요소들을 Dublin Core 기반으로 도출하고, 도출된 메타데이터 요소의 특성과 자료형에 맞게 XML 스키마를 설계해 보며, 이를 건설공사정보의 통합 검색에 실제로 적용해 보는데 있다. 건설공사정보 DB로부터 Dublin Core에 기반한 메타데이터를 도출하기 위해, 건설산업분야 실무담당자와 공동으로 작업이 이루어졌으며, 결과적으로 메타데이터 요소 12개와 관리요소 1개, 총 13개의 메타데이터를 도출하였다. 도출된 13개의 메타데이터의 특성들은 검토·분석되었으며, 이를 토대로 요소별 XML 스키마 설계과정이 상세하게 기술되었다. 완성된 XML 스키마를 기반으로 실제 적용된 통합검색 서비스 환경을 살펴보았으며, 앞으로의 연구과제도 제시되었다.

ABSTRACTS

The purpose of this study is to extract metadata based on Dublin Core from national construction information DBs, to design XML schema from the extracted metadata, and to apply the previous works to integrated information retrieval system. To extract metadata, co-work was performed with librarians in the field of construction information, and, as a result, twelve metadata elements from Dublin Core and one administrative metadata could be extracted. Based on the investigation and analysis of those thirteen metadata elements, XML schema design process was described in detail. Finally, the service environments of integrated information retrieval system to which XML schema was applied was introduced. Further study was also provided.

키워드: 건설공사정보, 메타데이터, XML 스키마

---

* 본 연구는 한국건설기술연구원의 지원으로 수행되었음.
** 한국과학기술정보연구원(KISTI) 정보시스템부 연구원(ilonetos@kisti.re.kr)
*** 한국과학기술정보연구원(KISTI) 정보시스템부 선임연구원/팀장(findit@kisti.re.kr)
논문접수일자 2002년 8월 26일
게재확정일자 2002년 9월 14일

1. 서 론

HTML이 1990년대 일반인에게 인터넷과 웹을 확산시킨 기술이라면, 1998년 W3C에 의해서 표준으로 제정된 XML은 HTML이 가진 한계를 극복하는 차세대 인터넷 표준 마크업 언어이다. HTML이 문서 내의 내용을 브라우저를 통하여 어떻게 보여 줄 것인가에 초점이 맞추어진 반면, XML은 문서의 내용을 정보처리를 위한 데이터로 본다는 것이 XML의 가장 큰 특징이다. 차세대 웹 문서 표준으로 잡아가고 있는 XML의 이용은 국내에서도 활발하게 진행되고 있다. 그러나 XML이 기존의 문서 타입에서 벗어나 XML의 응용과 데이터베이스(이하 DB)와 같은 다양한 애플리케이션과 연계되면서 응용 개발자들에게는 또 다른 부담거리가 되었다.

XML의 제약을 극복하고자 2001년 5월에 규정(specification)과 함께 XML 스키마(schema)가 출현하였다. 스키마의 사전적인 의미는 일반화된 계획이나 도해를 뜻하지만 XML의 경우 스키마란, 데이터가 마크업되는 방식으로서 문서 안에서 어떤 요소형과 특성, 값들을 사용할 수 있는 지에 대한 규정이라고 할 수 있다. 즉, 하나의 XML 문서 안에 어떤 것들을 담을 수 있는지에 대한 규칙들의 집합이며 "XML Vocabularies" 또는 "XML Dictionaries"라고도 불린다. 그러나 DTD(Document Type Definition)의 단점을 보완한 제안들[예를 들어, XML-Data Schema, XSD(XML Schema Definition)] 등 용어로서의 스키마가 사용되어짐으로써 DTD와 스키마는 별개의 개념으로 사용되어졌다. XML 스키마는 다양한 데이터 형식과 네임스페이스(namespace) 등을 허용하고 유연한 문서구조를 가지도록 지원하여 데이터 중심의 XML 문서와 애플리케이션에 유효성을 보장시키는 DTD의 대안이라고 할 수 있다(황병연, 김연혜 2001). 현재는 국외를 중심으로 XML 스키마에 대한 연구와 응용분야에 대한 적용이 진행중이며, 특히, 이기종간 DB 환경에서의 통합검색을 위한 메타데이터 표준화 분야, e-business 및 전자상거래와 같은 데이터교환 분야에서 활발하게 이루어지고 있다. 국내에서 XML 스키마에 대한 연구 및 실제 적용사례는 아직 부족한 상태이다.

본 연구는 네 개의 건설공사정보 DB(표 1-1 참조)를 대상으로, 첫째, 이들로부터 메타데이터를 Dublin Core 기반으로 도출하고, 둘째, 도출된 메타데이터 요소의 특성과 자료형에

〈표 1-1〉 건설공사정보 DB 유형 및 개요

DB 명	개 요
시공계획서	국내건설공사 현장에서 주요 공사 수행을 위해 작성된 공사계획서
시공절차서	시공품질 향상 및 표준화를 위해 작성한 시공지도서
원가절감/VE사례	국내 및 해외 건설현장에서 수행된 공정별 우수 시공사례 및 Value Engineering 사례 (건축분야, 토목분야, 플랜트분야 및 관리분야)
건설공사지	건설공사 완료 후 실제 수행된 공사자료를 근간으로 작성된 공사지 (주요시설물-공항, 교량, 댐, 도로, 아파트, 터널, 항만 등)

맞게 XML 스키마를 설계해 보며, 마지막으로, 설계된 XML 스키마를 검색에 실제로 적용해 봄으로써 다중 DB를 위한 메타데이터 통합검색의 틀을 제시해 보는데 있다. 향후 다양한 응용분야에서 표준 기술언어로 XML 스키마를 채택하려고 하는 움직임으로 볼 때, XML 스키마에 대한 연구 및 응용·적용은 중요하다고 할 수 있겠다.

2. XML DTD와 XML 스키마

본 장에서는 인터넷 자원들의 데이터 교환 표준으로 광범위하게 응용되고 있는 XML과 XML DTD에 대해 간단히 조사해보고, 최근에 W3C에서 제안한 XML 스키마가 출현하게 된 배경 및 장점을 살펴봄으로써 건설공사정보 메타데이터 요소들을 XML 스키마로 설계하고자 하는 의미를 살펴보고자 한다.

2. 1 XML과 XML DTD

1998년 2월에 제정된 XML 권고안은 이용자들과 개발자들에게 차세대 웹 문서로 각광을 받았으며, 현재 웹 문서표준으로 활발하게 진행되고 있다. XML 문서는 구조화된 정보를 포함하고 있는 문서들을 위한 마크업 언어라고 할 수 있으며, XML 문서는 문법에 맞는 문서(well-formed document)와 유효한 문서(valid document)로 구분할 수 있다. 문법에 맞는 문서는 XML 문서에 사용된 요소들이 모두 시작 태그와 끝 태그를 가지고 있고, XML 문서에서 요구하는 중첩규칙을 위반하지 않는 문서를 의미하는 것으로 DTD가 없는 XML 문서를 비 검증용 파서(non-validating parser)로 파싱 했을 때 오류가 없는 문서를 말한다. 이에 비해, 유효한 문서는 XML 문서가 DTD의 정의대로 올바로 작성되었는지 검사하는 유효성 검증(validating) 과정을 거치게 되는데 DTD가 있는 XML 문서를 검증용 파서(validating parser)를 사용하여 파싱 했을 때 오류가 없는 문서를 의미한다(채진석 1999). XML의 문서 구조는 마크업만 보고도 해당 태그의 내용을 추측할 수 있고, 마크업 자체에 의미정보를 부여할 수 있으며, 구조검색 및 전문검색이 가능하다.

XML이 기존의 문서 타입에서 벗어나 적용분야가 확대되어 DB를 비롯한 다양한 애플리케이션들과 연계되면서, 해당 응용 환경에 맞도록 데이터들의 타입 지정과 같은 새로운 메커니즘들이 필요하게 되었다. 이처럼 응용처리 시 요구되는 세부사항들을 모두 만족시키는 데는 기존의 XML DTD로는 한계가 있었으며, 이에 대한 대안으로 XML 스키마라고 하는 문서 설계 언어가 나오게 되었다.

2. 2 XML 스키마의 특성

XML 스키마를 XML DTD와 비교하여 그 특성을 살펴보면 크게 다음과 같이 다섯 가지로 요약된다(Birbeck, et al. 2001; Biron, 2001; Wahlin, 2002).

① XML 스키마 자체가 XML 문법

기존의 XML DTD는 XML 문서와 다른 문법(EBNF: Extended Backus-Naur

Form)을 사용함으로써 개발자에게 어렵고, DTD를 위한 별도의 처리 시스템이 필요한데 반해, XML 스키마는 자체가 XML 문법이다. XML 스키마가 XML 문법을 이용하여 작성되므로 XML을 위해 개발된 시스템을 바로 이용할 수 있다. 예를 들어, XML 문서를 query할 수 있는 XPath와 XLink(XML Linking Language) 같은 다른 표준들은 스키마와 함께 사용할 수 있으며, DOM(Document Object Model)을 이용한 검색이나 문서 트리도 처리할 수 있다.

② 다양한 데이터 타입

XML DTD의 모든 정보는 #PCDATA로 정의되어, 문자열, 날짜, 정수 등의 특수한 데이터 타입으로 지정할 수 없고, 지원 가능한 데이터 타입이 극히 제한적인데 반해, XML 스키마는 응용프로그램이나 DB에서 사용하는 데이터형을 이용할 수 있도록 다양한 데이터 타입을 지원하며, 사용자가 원하는 타입을 사용자가 정의하여 무한 확장이 가능하다.

③ 강력하고 융통성 있는 내용모델

XML DTD는 요소의 순서(order)나 출현 횟수(occurrence)의 지정이 제한적이고 빈약한 개념 도입과 내용모델인데 반해, XML 스키마는 출현 횟수를 명시적인 값으로 줄 수 있으며, 사용되는 거의 모든 데이터 타입을 지원하여 보다 강력한 내용 모델을 지원할 수 있고, 요소와 그것의 내용을 표현하기 위한 융통성 있는 내용 모델을 제공한다.

④ 확장의 용이성

XML DTD는 한 문서에 오직 하나의 DTD만이 적용될 수밖에 없고 상속(inheritance) 기능을 지원하지 못해 확장성이 결여되어 있으나, XML 스키마는 상속기능 지원은 물론 강력한 네임스페이스 모델을 지원함으로써 여러 스키마들을 하나의 문서에서 참조할 수 있고, 여러 문서 내의 동일한 용어는 네임스페이스를 통해 서로 다른 스키마 구조를 참조하도록 하여 충돌을 방지한다. 이로 인해, 공유되고 표준화된 XML 어휘집들의 사용을 증가시킬 수 있고, 공유된 XML 기반의 전자상거래와 EDI(Electronic Data Interchange) 같은 분야에 큰 도움을 줄 수 있다.

⑤ 런타임에서의 수정

XML DTD는 일단 작성되면 런타임 상에서 DTD를 변경할 수 없으나, XML 스키마는 런타임 상태에서도 선택되어질 수도 있고 추가, 변경, 삭제 등이 가능하다.

XML DTD가 가지고 있는 단점들로 인하여 XML DTD를 쓰지 말고 XML 스키마를 써야 한다는 것은 아니다. 스키마가 제안된 이유는 현재 XML이 쓰이고 있는 것 보다 다양한 용도의 애플리케이션들이 준비되고 있기 때문이다. XML DTD는 기본적인 문서 교환 용도(예를 들면, 학회지, 보고서, 관공서의 규격 문서 등)에는 적합하지만 현재의 추세처럼 디지털 문서를 다양한 응용 프로그램과 연계할 경우(예를 들면, DB와의 연동을 통한 검색, e-business와 같이 실시간 데이터 처리 또는 특정 데이터를 여러 응용프로그램간에 전송하고 처리하고자 하는 경우)는 데이터 처리

가 용이하지 않은 XML DTD 보다는 다양한 데이터 타입 제공뿐만 아니라 완벽한 데이터 재사용 등에서 장점이 많은 XML 스키마로 처리하는 것이 바람직하다.

국외에서는 XML 스키마를 이용한 응용 XML 처리가 W3C, 마이크로소프트사, 그리고 수많은 단체들을 중심으로 활발하게 진행되고 있다. 국내에서는 W3C에서 2001년 5월에 XML 스키마 표준이 제정된 이후 시작단계로 전자상거래, GIS 및 전자도서관 분야에서 조심스럽게 진행되고 있고, 특히 분산된 이 기종간의 DB를 통합 검색할 수 있도록 하기 위해 메타데이터 요소들을 표현하기 위한 기술로 교육계와 과학기술계를 중심으로 다각적으로 진행되고 있으나 아직은 미흡한 실정이다.

3. 건설공사정보의 XML 스키마 설계

본 장에서는 건설공사정보 DB 통합검색을 위한 요소들의 XML 스키마 설계를 위해 네 개의 건설공사정보 DB(표 1-1 참조)로부터 도출된 공통 메타데이터 요소들을 간략하게 서술하고, 이러한 요소들을 XML 스키마로 기술하기 위한 요소명 및 특성, 가질 수 있는 값의 범위 등을 확정하며, 확정된 정의들을 토대로 XML 스키마를 설계하고 기술하였다.

3. 1 Dublin Core 기반 메타데이터의 도출

본 연구를 위해 네 개의 건설공사정보 DB로부터 Dublin Core에 기반한 메타데이터를 도출하였다. 이를 위해, 건설산업분야 정보자료관리 실무담당자와 공동으로 작업이 이루어졌으며, 결과적으로 메타데이터 요소 12개와 건설산업분야 DB에 전체적으로 필요한 관리요소 1개, 총 13개의 메타데이타를 도출할 수 있었다. 각 DB별 메타데이터 요소들은 건설현장에 종사하는 실무자 그룹을 통해 검증 받았다.

〈표 3-1〉에서는 건설공사정보에 필요한 12개의 Dublin Core 기반 메타데이터와 마지막 13번째 관리요소(공사기관:ProjectOrganization) 메타데이터에 대해 정의하고, 이들로부터 검색 시 필요한 세부(하위) 요소를 마지막 항목에 정의하였다. 비고란에는 세부(하위)요소가 Dublin Core 메타데이터 요소 집합(DCMI 1999)에 포함되는 경우는 "DC정의"로 표기하였고, 건설공사정보의 필요에 의해 구분한 경우는 "자체정의"로 표기하였다.

3. 2 XML 스키마 설계

3. 2. 1 요소별 XML 스키마 설계

〈표 3-2〉는 앞에서 도출된 13개의 건설공사정보 메타데이터 요소들을 XML 스키마에서 기술할 요소명으로 정의하고, 각 요소들이 XML 스키마에 따라 작성될 XML 문서에서 나타나는 특성들을 정리하였다. 정의된 XML 스키마 요소명 선정은 Dublin Core 기반 요소명을 그대로 수용함으로써 범 국가적인 검색이 가능하도록 하였다.

〈표 3-3〉에서는 〈표 3-1〉의 13개 주 요소들의 하위요소들을 XML 스키마로 기술하기 위한 하위 요소명으로 정의하였고, 각 하위요소

〈표 3-1〉 더블린 코아 기반 건설공사정보 메타데이터

번호	도출 요소명	요소에 대한 설명	요소의 세분화 (하의요소)	비고
1	표제요소 (Title)	정보자원에 부여된 제목으로 정보자원을 공식적으로 지칭하는 이름	주표제	자체정의
			대체표제명	DC정의
2	저자요소 (Creator)	정보자원의 내용에 주된 책임을 갖고 있는 개체의 이름으로 사람이나 기관 또는 공공사업이 올 수 있음	개인저자	자체정의
			단체저자	
3	주제요소 (Subject)	정보자원의 내용을 간결하게 기술. 통합건설정보 분류체계를 사용하여 입력	통합건설정보분류	자체정의
			각주제면분류	
4	개요 (Description)	정보자원의 내용에 대한 설명 또는 목적에 대한 기술로 목차, 초록, 요약 또는 자유로운 설명 등을 포함	목차	DC정의
			초록	
5	발행처 (Publisher)	정보자원을 현재의 형태로 이용 가능하도록 책임을 갖고 있는 개체의 이름이며 사람이나 기관이 올 수 있음	개인발행자	자체정의
			단체발행자	
6	날짜 (Date)	정보자원이 존재하는 동안 어떤 변화가 발생한 경우의 해당 날짜로, 정보자원의 작성일과 현재의 형태로 이용 가능하게 된 날짜에 관련됨	작성일	DC정의
			디지털변환일	DC정의
7	자료유형 (Type)	정보자원의 유형으로 한국건설기술연구원에서 정의한 자료유형(시공계획서, 시공절차서, 원가절감사례, VE사례, 건설공사지)을 사용	-	-
8	자료형태 (Format)	정보자원의 물리적 자료형태로, 매체 유형이나 특질을 포함	크기	DC정의
			매체	
9	접근정보 (Identifier)	정보자원을 식별하기 위한 고유한 문자열 또는 지시자이며, 해당 문서의 문서번호 이용 가능	-	-
10	언어 (Language)	정보자원의 내용을 기술하고 있는 언어	-	-
11	관련자료 (Relation)	현재 정보자원에 관련된 다른 정보자원에 대한 확인 및 관계유형	버전관계	DC정의
			버전관계	
			대체관계	
			대체관계	
			의존관계	
			의존관계	
			부분/전체관계	
			부분/전체관계	
			참조관계	
			참조관계	
			포맷변환관계	
			포맷변환관계	
12	저작권 (Rights)	저작권이나 정보이용 권리에 관한 정보나 그러한 정보를 정보를 제공하는 서비스로 연결 하기 위한 식별자	-	
13	공사기관 (ProjectOrg-anization)	해당 정보자원이 발생된 공사와 관련된 기관	공사기관	자체정의
			발주처	
			시공사	
			설계자	
			감리자/CM	

〈표 3-2〉 XML 스키마 요소명 및 특성

번호	요소명	XML 스키마 요소명	필수여부	반복유무	하위요소 유무	비고
1	표제	title	Y	N	Y	DC 기본요소
2	저자	creator	Y	Y	Y	
3	주제	subject	Y	Y	Y	
4	개요	description	N	Y	Y	
5	발행처	publisher	Y	Y	Y	
6	날짜	date	Y	Y	Y	
7	자료유형	type	Y	Y	N	
8	자료형태	format	Y	N	Y	
9	접근정보	identifier	Y	N	N	
10	언어	language	Y	N	N	
11	관련자료	relation	N	N	Y	
12	저작권	rights	N	N	N	
13	공사기관	ProjectOrganization	N	Y	Y	자체관리 요소

들과 해당 요소들의 값들이 가지는 특성 및 표현 범위를 나타내었으며, 이를 토대로 각 하위요소(주 요소 포함)들의 데이터 타입을 정의하였다. 특히, 해당 하위요소가 DCQ(Dublin Core Qualifiers)에서 정의한 요소 특성(DCMI 2000)과 동일할 경우는 주 요소명과 마찬가지로 범 국가적인 검색이 가능하도록 DCQ 정의 요소 이름을 그대로 수용하였다. 데이터 타입 컬럼은 13개의 주 요소뿐만 아니라 하위요소들이 가질 수 있는 데이터의 특성에 따라 XML 스키마에서 선언할 데이터 타입을 확정하였다. 비고 컬럼에서는 해당 하위요소명이 DCQ 정의 요소 기반일 경우는 "DCQ"라 하고, DCQ 정의가 아닌 자체 정의 하위요소명인 경우는 "자체" 라 표기하였다.

〈표 3-4〉는 각 주 요소 혹은 하위요소들이 가질 속성들을 정의하고 각 속성들의 필수 여부를 선언하였다. 마지막 컬럼은 해당 속성들이 갖는 값의 범위를 표현하였다.

번호 3 "subject"의 하위 요소인 "specific"은 스킴(scheme)으로 "KCICf", "KCICs", "KCICe", "KCICw", "KCICr" 중 하나가 되는데, 이들은 해당 자료에 대한 성격을 나타내는 주제를 건설정보에 대한 통합 분류체계인 통합건설정보분류체계(KCIC, Korea Construction Information Classification, 건설교통부 공고 제 2001-230호)의 "시설물분류(KCICf)", "공간분류(KCICs)", "부위분류(KCICe)", "공종분류(KCICw)", "자원분류(KCICr)" 코드 값이다.

번호 6 "date"의 하위요소인 "created"와 "issued"는 각각 스킴을 갖는데, 그 값은 ISO 8601의 프로파일에 따라 기술된다. 따라서 날짜는 YYYY-MM-DD 형식을 갖는다. 만일 년월일을 완전하게 알 수 없고 년과 월만을 아는 경우 YYYY-MM으로, 연도만을 알 수 있는 경우 YYYY 형식으로 기술한다. XML 스키마의 자료형 중 "date" 타입은 ISO 8601

〈표 3-3〉 요소별 하위요소 정의 및 선언

번호	요소명	하위요소의 스키마 명	필수 여부	반복 유무	요소 값의 표현 범위	데이터 타입	비고
1	표제	mainTitle	Y	N	-	string	자체
		alternative	N	Y	-	string	DCQ
2	저자	personalName	N	N	-	string	자체
		corporateName	N	N	-	string	자체
3	주제	combined	N	N	건설교통부의 통합건설정보분류체계 (KCIC) 코드값 참조	string	자체
		specific	N	Y	〃	string	자체
4	개요	tableOfContents	N	N	-	string	DCQ
		abstract	N	N	-	string	DCQ
5	발행처	personalName	N	N	-	string	자체
		corporateName	N	N	-	string	자체
6	날짜	created	Y	N	ISO 8601 형식(YYYY-MM-DD)	date	DCQ
		issued	Y	N	〃	〃	DCQ
7	자료유형	-	-	-	시공계획서, 시공절차서, 원가절감 /VE사례, 건설공사지 중 하나	string	-
8	자료형태	extent	Y	N	자료의 페이지수 혹은 크기	string	DCQ
		medium	Y	N	한국건설기술연구원 정의 자료형식	string	DCQ
9	접근정보	-	-	-	인터넷 주소 형식(URI)	anyURI	-
10	언어	-	-	-	kor, eng와 같은 3자리(ISO 639-2)	language	-
11	관련자료	isVersionOf	N	Y	인터넷 주소 형식(URI)	anyURI	DCQ
		hasVersion	N	Y	〃	〃	〃
		isReplacedBy	N	Y	〃	〃	〃
		replaces	N	Y	〃	〃	〃
		isRequiredBy	N	Y	〃	〃	〃
		requires	N	Y	〃	〃	〃
		isPartOf	N	Y	〃	〃	〃
		hasPart	N	Y	〃	〃	〃
		isReferencedBy	N	Y	〃	〃	〃
		references	N	Y	〃	〃	〃
		isFormatOf	N	Y	〃	〃	〃
		hasFormat	N	Y	〃	〃	〃
12	저작권	-	-	-	-	string	-
13	공사기관	owner	N	N	-	string	자체
		contractor	N	N	-	string	자체
		designer	N	N	-	string	자체
		constructionManager	N	N	-	string	자체

형식에 따른다.

번호 7 "type" 요소가 갖는 스킴은 그 값이 "KICT-Type"인데, 이는 "type"이 갖게되는 값은 KICT-Type 범위 내에서 입력하라는 의미이며, KICT-Type의 범위는 "Execution-Planing(시공계획서)", "ConstructionGuide(시공절차서)", "CostDown(원가절감사례)", "ValueEngineering(VE사례)", "Execution-Report(건설공사지)"중 하나가 된다.

번호 8 "format"의 하위요소인 "extent"는 "Unit"라는 속성을 갖고, "Unit" 속성이 가질 수 있는 값은 "Page(해당자료의 페이지 수)", "Size(해당자료의 크기)", "Duration(해당자료의 보존기간)"중 하나가 된다. 이는 "format" 요소의 입력되는 값은 위의 세 가지 속성 값에 따라 정해짐을 의미한다. 또한, 하위요소 "medium"의 스킴은 "KICT-IMT"를 값으로 가지며, "KICT-IMT"는 확장 IMT (IANA 2001)로서 건설산업에 필요한 미디어 타입이 추가된 IMT 이다. 예를 들어, 해당 자료가 XML 문서일 경우, "format"의 값은 "KICT-IMT"에 정의된 값인 text/xml을 입력하고, tiff image 문서일 경우 image/tiff를 입력한다.

번호 9 "identifier"의 스킴 값은 "URI" 이다. 이는 "identifier" 요소의 값은 URI 형식으로 입력되어야 함을 의미하며, 〈표 3-3〉의 9번 접근정보의 자료형인 "anyURI" 정의에 따라 URI 형식의 데이터만을 값으로 입력할 수 있다. 이는 번호 11의 "relation" 요소의 하위요소들이 갖는 속성 값과 동일하다.

번호 10 "language" 요소의 스킴 값은 "ISO639-2"이다. 이는 3바이트 언어코드 표준 형식인 ISO 639-2를 따른다. 예를 들어, 한국어는 ISO 639-2에 따라 "kor"이 되고 영어의 경우 "eng"가 된다.

번호 12 "rights" 요소는 속성값으로 "xml:lang"을 가지며, "xml:lang"의 저작권요소는 한글로 저작권요소를 표현할 수도 있고 영문 혹은 그밖에 기타 언어로 표현하는 경우도 있어, XML 1.0 규정에 정의된 "xml:lang"을 이용한다.

3. 2. 2 XML 스키마 선언 및 정의

앞에서 설계한 13개 메타데이터 요소들의 XML 스키마 요소 및 하위요소, 속성들에 대한 정의, 값의 범위, 그리고 데이터 형에 대한 XML 스키마는 크게 헤더부분, XML 스키마 선언부분, 그리고 각 요소들에 대한 정의부분으로 나눌 수 있다.

(1) 헤더부분

〈표 3-5〉는 전체 스키마 부분 중 헤더부분을 기술한 것이다. 모든 XML 스키마는 〈xs:schema〉 라는 태그로 시작한다. 〈xs:schema〉 태그는 기본 네임스페이스와 앞으로 기술되어지는 13개 요소들이 참조하고 적용되는 모든 네임스페이스들을 속성으로 갖는다. [targetNamespace = "http://www.kict.re.kr/~metadata/terms/"]를 기술함으로써, "http://www.kict.re.kr/~metadata/terms/"를 기본 네임스페이스로 설정하였으며, 이는 식별자(xs, dc, dcterms 등)가 없는 요소들이 참조하는 네임스페이스 임을 의미한다. 그 외 주 요소 및 하위요소들 앞에 붙는 식별자들이 참조하는 네임스페이스들을 아래에 연속하여 정의하였고, 이는 "식별자:" 다음에 나타나는

〈표 3-4〉 각 요소별 속성(attribute) 정의 및 선언

번호	스키마 요소명	하위요소의 스키마 명	사용 속성명	필수여부	지정 속성값
1	title	mainTitle	-	-	-
		alternative	-	-	-
2	creator	personalName	-	-	-
		corporateName	-	-	-
3	subject	combined	-	-	-
		specific	scheme	required	KCICf, KCICs, KCICe, KCICw, KCICr 중 하나
4	description	tableOfContents	-	-	-
		abstract	-	-	-
5	publisher	personalName	-	-	-
		corporateName	-	-	-
6	date	created	scheme	required	ISO8601
		issued	〃	〃	〃
7	type	-	〃	〃	KICT-Type
8	format	extent	Unit	〃	Page, Size, Duration
		medium	scheme	〃	KICT-IMT
9	identifier	-	〃	〃	URI
10	language	-	〃	〃	ISO639-2
11	relation	isVersionOf	〃	〃	URI
		hasVersion	〃	〃	〃
		isReplacedBy	〃	〃	〃
		replaces	〃	〃	〃
		isRequiredBy	〃	〃	〃
		requires	〃	〃	〃
		isPartOf	〃	〃	〃
		hasPart	〃	〃	〃
		isReferencedBy	〃	〃	〃
		references	〃	〃	〃
		isFormatOf	〃	〃	〃
		hasFormat	〃	〃	〃
12	rights	-	-	-	xml:lang
13	ProjectOrganization	owner	-	-	-
		contractor	-	-	-
		designer	-	-	-
		constructionManager	-	-	-

〈표 3-5〉 XML 스키마의 헤더부분

```
〈xs:schema
targetNamespace = "http://www.kict.re.kr/~metadata/terms/"
xmlns:xs = "http://www.w3.org/2001/XMLSchema"
xmlns:dcxml = "http://purl.org/dc/xml/"
xmlns:dctype = "http://purl.org/dc/dcmitype/"
xmlns:dcterms = "http://purl.org/dc/terms/"
xmlns = "http://www.kict.re.kr/metadata/terms/"
xmlns:dc = "http://purl.org/dc/elements/1.1/"elementFormDefault = "qualified" attributeFormDefault
        = "qualified"〉

    〈xs:annotation〉
        〈xs:documentation〉
건설공사정보 메타데이터 요소들에 대한 XML 스키마 정의 및 선언
(파일명 : kict-metadata.xsd)
        〈/xs:documentation〉
    〈/xs:annotation〉
```

요소명이 식별자가 참조하는 네임스페이스로부터 왔다는 것을 의미한다. 예를 들어, "dc:subject"의 경우, "subject" 요소는 식별자 "dc"의 네임스페이스(Dublin Core 기반 15개 메타데이터 요소들의 집합이 정의되고 설명되어 있는 주소 즉, http://purl.org/dc/elements/1.1/)영역에서 온 것이고, "xs:schema" 앞의 "xs" 식별자는 "xs"의 네임스페이스(XML 스키마 요소들이 정의되고 설명되어 있는 주소 즉, http://www.w3.org/2001/XMLSchema)를 "schema"가 참조하고 있음을 의미한다. [elementFormDefault = "qualified", attributeFormDefault = "qualified"]는 스키마에서 정의된 하위 요소들과 속성들이 별도의 식별자가 붙지 않으면 해당 하위요소의 상위요소인 주 요소의 네임스페이스를 따르게 됨을 표현한 것이다(Thompson 2001).

(2) 메타데이터 그룹과 13개 요소들에 대한 XML 스키마 선언

〈표 3-6〉은 건설공사정보 메타데이터 13개 요소를 하나의 그룹으로 갖는 "KICTMetadata"를 선언하였다. "KICTMetadata" 요소는 최상위 요소로 〈표 3-7〉에 정의된 13개 주 요소들을 하위요소로 갖는다. 또한 "id" 속성을 가지며, 중복되지 않는 유일한 값을 의미하는 "ID" 타입의 값을 갖는다. 임의의 자료를 기술된 XML 스키마 형식에 따라 XML 스키마 인스턴스로 변환할 때, 이들을 구분하기 위한 식별 값으로 이용된다.

〈표 3-7〉은 앞서 〈표 3-6〉의 "KICT-elementsGroup"에 대한 정의이며, 13개 각각의

〈표 3-6〉 XML 스키마 선언부분

```
〈xs:element name="KICTMetadata"〉
    〈xs:complexType〉
        〈xs:group ref="KICT-elementsGroup"/〉
        〈xs:attribute name="id" type="xs:ID"/〉
    〈/xs:complexType〉
〈/xs:element〉
```

주 요소마다 XML 문서에서 이용될 스키마 요소명을 선언하고, 해당 요소가 갖는 타입을 선언한다. 이중 "ProjectOrganization" 요소를 제외하고 나머지 12개 요소는 식별자로 "dc"를 갖는데, 이는 12개의 요소는 "dc" 식별자가 참조하는 네임스페이스 범주에 있음을 의미한다. "ProjectOrganization"은 기본 네임스페이스 범주에 있으므로 식별자가 필요 없다.

(3) 메타데이터 요소들에 대한 XML 스키마 정의

① Title 요소에 대한 Type 정의

〈표 3-7〉 "KICT-elementsGroup"에 대한 정의

```
〈xs:group name="KICT-elementsGroup"〉
    〈xs:sequence〉
        〈xs:element name="dc:title" type="titleType"/〉
        〈xs:element name="dc:creator" type="nameType" maxOccurs="unbounded"/〉
        〈xs:element name="dc:subject" type="subjectType"/〉
        〈xs:element name="dc:description" type="descriptionType" maxOccurs="unbounded"/〉
        〈xs:element name="dc:publisher" type="nameType" maxOccurs="ur.bounded"/〉
        〈xs:element name="dc:date" type="dateType"/〉
        〈xs:element name="dc:type" type="resourceType"/〉
        〈xs:element name="dc:format" type="formatType"/〉
        〈xs:element name="dc:identifier" type="uriType"/〉
        〈xs:element name="dc:language" type="languageType"/〉
        〈xs:element name="dc:relation" type="relationType"/〉
        〈xs:element name="dc:rights" type="rightsType"/〉
        〈xs:element name="ProjectOrganization" type="ProjectOrganizationType"/〉
    〈/xs:sequence〉
〈/xs:group〉
```

〈표 3-8〉는 〈표 3-7〉의 "dc:title" 요소가 갖는 타입인 "titleType"에 대한 정의이다. "titleType"은 두 개의 하위요소인 "main Title"과 "dcterms:alternative"를 가지며, "mainTitle"은 기본 네임스페이스를 참조하고 "alternative"는 식별자 "dcterms"의 네임스페이스를 갖는다. "mainTitle"의 발생횟수는 minOccurs="1", maxOccurs="1"이 생략되어, 반드시 한번은 나타나야 함을 의미하고 있고, "dcterms:alternative"는 나타나지 않을 수도 있고, 원하는 만큼 여러 번 쓸 수 있음을 의미한다.

② Creator와 Publisher 요소에 대한 Type 정의

〈표 3-9〉는 〈표 3-7〉의 "dc:creator"와 "dc:publisher" 요소가 갖는 타입인 "nameType"에 대한 정의이다. "nameType"은 두 개의 하위요소인 "personalName"과 "corporateName"을 가지며, 기본 네임스페이스 범주에 있는 요소이다. 두 요소 모두, "string" 타입이며, 한번도 안 나타날 수 있고 필요하면 하위요소로서 한번만 나타날 수도 있다. "maxOccurs" 표현범위가 생략되면 디폴트로 "1"이 설정되며, 이는 maxOccurs="1"과 동일하다.

③ Subject 요소에 대한 Type 정의

〈표 3-10〉은 〈표 3-7〉의 "dc:subject" 요소가 갖는 타입인 "subjectType"에 대한 정의이

〈표 3-8〉 Title 요소에 대한 Type 정의

```
〈xs:complexType name="titleType"〉
    〈xs:sequence〉
        〈xs:element name="mainTitle" type="xs:string"/〉
        〈xs:element name="dcterms:alternative" type="xs:string" minOccurs="0" maxO
         ccurs="unbounded"/〉
    〈/xs:sequence〉
〈/xs:complexType〉
```

〈표 3-9〉 Creator와 Publisher 요소에 대한 Type 정의

```
〈xs:complexType name="nameType"〉
    〈xs:sequence〉
        〈xs:element name="personalName" type="xs:string" minOccurs="0"/〉
        〈xs:element name="corporateName" type="xs:string" minOccurs="0"/〉
    〈/xs:sequence〉
〈/xs:complexType〉
```

다. "subjectType"은 두 개의 하위요소인 "combined" 요소와 "specific" 요소를 가지며, 기본 네임스페이스 범주에 있는 요소이다. "specific" 타입은 사용자 정의 타입인 "specificSchemeType"으로 선언되고, 정의된 "specificSchemeType"은 "dc:subject"가 "KCICf", "KCICs", "KCICe", "KCICw", "KCICr" 값 중 하나가 됨을 표현하고 있다.

④ Description 요소에 대한 Type 정의

〈표 3-11〉는 〈표 3-7〉의 "dc:description" 요소가 갖는 타입인 "descriptionType"에 대한 정의이다. "descriptionType"은 두 개의 하위요소인 "dcterms:abstract" 요소와 "dcterms:tableOfContents" 요소를 가지며, "dcterms" 식별자가 참조하는 네임스페이스 범주에 있는 요소이다. "dcterms:abstract" 요소는 반드시 한 번은 나오도록 하고, "dcterms:

〈표 3-10〉 Subject 요소에 대한 Type 정의

```
〈xs:complexType name = "subjectType"〉
    〈xs:sequence〉
        〈xs:element name = "combined" type = "xs:string" minOccurs = "0"/〉
        〈xs:element name = "specific" minOccurs = "0" maxOccurs = "unbounded"〉
            〈xs:complexType〉
                〈xs:simpleContent〉
                    〈xs:extension base = "xs:string"〉
                     〈xs:attribute name = "scheme" type = "specificSchemeType" use = "required"/〉
                    〈/xs:extension〉
                〈xs:simpleContent〉
            〈/xs:complexType〉
        〈/xs:element〉
    〈/xs:sequence〉
〈/xs:complexType〉

〈xs:simpleType name = "specificSchemeType"〉
    〈xs:restriction base = "xs:string"
        〈xs:enumeration value = "KCICf"/〉
        〈xs:enumeration value = "KCICs"/〉
        〈xs:enumeration value = "KCICe"/〉
        〈xs:enumeration value = "KCICw"/〉
        〈xs:enumeration value = "KCICr"/〉
    〈xs:restriction〉
〈/xs:simplexType〉
```

〈표 3-11〉 Description 요소에 대한 Type 정의

```
〈xs:complexType name="descriptionType"〉
    〈xs:sequence〉
        〈xs:element name="dcterms:abstract" type="xs:string"/〉
        〈xs:element name="dcterms:tableOfContents" type="xs:string" minOccurs="0"/〉
    〈/xs:sequence〉
〈/xs:complexType〉
```

tableOfContents" 요소는 없을 경우 쓰지 않아도 되도록 정의하였다.

⑤ Date 요소에 대한 Type 정의

〈표 3-12〉는 〈표 3-7〉의 "dc:date" 요소가 갖는 타입인 "dateType"에 대한 정의이다. "dateType"은 두 개의 하위요소인 "dcterms:created" 요소와 "dcterms:issued" 요소를 가지며, "dcterms" 식별자가 참조하는 네임스페이스 범주에 있는 요소이다. 두 요소 모두 반드시 한 번은 나오도록 하고, "dateFormatType" 타입을 갖도록 선언하였으며 구체적인 정의를 아래에 기술하였다. 이는 "dcterms:created" 요소와 "dcterms:issued" 요소 모두 XML 스키마에서 정의된 "date" 타입의 값을 갖고 두 개의 요소는 "scheme" 이라는 속성을 가지며, 속성 값은 아래 "dateSchemeType"에서 정의된 "ISO8601"임을 표현한 것이다.

⑥ Type 요소에 대한 Type 정의

〈표 3-13〉은 〈표 3-7〉의 "dc:type" 요소가 갖는 타입인 "resourceType"에 대한 정의이다. "resourceType"의 값은 아래에 정의된 "DocType"이 가질 수 있는 값의 범주("ExecutionPlanning", "ConstructionGuide", "CostDown", "ValueEngineering", "ExecutionReport") 내에서 선택할 수 있도록 하였다. "dc:type" 요소가 갖는 속성은 "scheme" 이며, 속성 값은 항상 "KICT-Type" 임을 선언하여 "dc:type" 요소의 값이 "DocType" 범주의 값임을 의미하도록 하였다.

⑦ Format 요소에 대한 Type 정의

〈표 3-14〉는 〈표 3-7〉의 "dc:format" 요소가 갖는 타입인 "formatType"에 대한 정의이다. "formatType" 두 개의 하위요소인 "dcterms:extent" 요소와 "dcterms:medium" 요소를 가지며, "dcterms" 식별자가 참조하는 네임스페이스 범주에 있는 요소이다. "dcterms:extent" 요소는 "UNIT"라는 속성을 갖도록 하였으며, "UNIT" 값의 범위는 아래 "extentScheme"에서 정의한 값("Page", "Size", "Duration") 범주 내에서 선택되도록 하였다. "dcterms:medium" 요소는 "scheme" 속성을 가지며, "scheme"값은 아래 "mediumScheme"에 정의된 "KICT-IMT"가 된다.

〈표 3-12〉 Date 요소에 대한 Type 정의

```
〈xs:complexType name="dateType"〉
    〈xs:sequence〉
        〈xs:element name="dcterms:created" type="dateFormatType"/〉
        〈xs:element name="dcterms:issued" type="dateFormatType"/〉
    〈/xs:sequence〉
〈/xs:complexType〉

〈xs:complexType name="dateFormatType"〉
    〈xs:simpleContent〉
        〈xs:extension base = "xs:date"〉
            〈xs:attribute name="scheme" type="dateSchemeType" use="required"/〉
        〈/xs:extension〉
    〈xs:simpleContent〉
〈/xs:complexType〉

〈xs:simpleType name="dateSchemeType"〉
    〈xs:restriction base="xs:string"〉
        〈xs:enumeration value="ISO8601"/〉
    〈/xs:restriction〉
〈/xs:simpleType
```

〈표 3-13〉 Type 요소에 대한 Type 정의

```
〈xs:complexType name="resourceType"〉
    〈xs:simpleContent〉
        〈xs:extension base="DocType"〉
            〈xs:attribute name="scheme" type="xs:string" default="KICT-Type"/〉
        〈/xs:extension〉
    〈/xs:simpleContent〉
〈/xs:complexType〉

〈xs:simpleType name="DocType"〉
    〈xs:restriction base="xs:string"〉
        〈xs:enumeration value="ExecutionPlanning"/〉
        〈xs:enumeration value="ConstructionGuide"/〉
        〈xs:enumeration value="CostDown"/〉
        〈xs:enumeration value="ValueEngineering"/〉
        〈xs:enumeration value="ExecutionReport"/〉
    〈/xs:restriction〉
〈/simpleType〉
```

〈표 3-14〉 Format 요소에 대한 Type 정의

```
〈xs:complexType name="formatType"〉
    〈xs:sequence〉
        〈xs:element name="dcterms:extent"〉
            〈xs:complexType〉
                〈xs:simpleContent〉
                    〈xs:extension base = "xs:string"〉
                        〈xs:attribute name="UNIT" type="extentScheme" use="required"/〉
                    〈/xs:extension〉
                〈xs:simpleContent〉
            〈/xs:complexType〉
        〈/xs:element〉
        〈xs:element name="dcterms:medium"〉
            〈xs:complexType〉
                〈xs:simpleContent〉
                    〈xs:extension base = "xs:string"〉
                      〈xs:attribute name="scheme" type="mediumScheme" use="required"/〉
                    〈/xs:extension〉
                〈xs:simpleContent〉
            〈/xs:complexType〉
        〈/xs:element〉
    〈/xs:sequence〉
〈/xs:complexType〉

〈xs:simpleType name="extentScheme"〉
    〈xs:restriction base="xs:string"
        〈xs:enumeration value="Page"/〉
        〈xs:enumeration value="Size"/〉
        〈xs:enumeration value="Duration"/〉
    〈xs:restriction〉
〈/xs:simplexType〉

〈xs:simpleType name="mediumScheme"〉
    〈xs:restriction base="xs:string"〉
        〈xs:enumeration value="KICT-IMT"/〉
    〈/xs:restriction〉
〈/xs:simpleType〉
```

⑧ Identifier 요소에 대한 Type 정의

〈표 3-15〉는 〈표 3-7〉의 "dc:identifier" 요소가 갖는 타입인 "uriType"에 대한 정의이다. "uriType"의 타입이 "xs:anyURI"로 선언된 것은, "dc:identifier"가 XML 스키마의 데이터 형인 "anyURI" 타입이고, "anyURI" 타입의 값을 갖도록 표현한 것이다. 또한, "dc:identifier" 요소는 "scheme" 속성을 갖도록 하였고, "scheme"의 값은 아래에 정의된 "URISchemeType"의 값 곧, "URI"가 된다.

⑨ Language 요소에 대한 Type 정의

〈표 3-16〉은 〈표 3-7〉의 "dc:language" 요소가 갖는 타입인 "languageType"에 대한 정의이다. "languageType"의 타입은 "rfc3066__LanguageCoeds"로 선언되고, 이는 XML 규정 1.0에서 정의된 "language" 타입임을 정의하였다. "language" 타입은 XML 규정 1.0에서 정의된 대로 아직까지는 두 자리의 국가코드(kr, en 등)형식이지만, 향후 ISO 639-2의 세 자리 형식의 국가코드 사용이 확산되고 있는 만큼 "dc:language"의 값이 다양한 국가코드 집합을 수용하도록 정의하였다. "dc:language"는 속성으로 "scheme"을 갖도록 하였고, 그 값은 "ISO639-2"로 표현하여, "dc:language"의 값이 ISO 639-2 표준 국가코드에 따르도록 표현하였다.

⑩ Relation 요소에 대한 Type 정의

〈표 3-17〉은 〈표 3-7〉의 "dc:relation" 요소가 갖는 타입인 "relationType"에 대한 정의이다. "relationType"은 "dcterms:isPartOf" 등 총 12개의 하위요소를 가지며, 모두 "dcterms" 식별자가 참조하는 네임스페이스 범주에 있는 요소이다. 12개 하위요소들의 타입은 "uriType"으로 정의하였고, 이는

〈표 3-15〉 Identifier 요소에 대한 Type 정의

```
〈xs:complexType name="uriType"〉
    〈xs:simpleContent〉
        〈xs:extension base="xs:anyURI"〉
            〈xs:attribute name="scheme" type="URISchemeType" use="required"/〉
        〈/xs:extension〉
    〈/xs:simpleContent〉
〈/xs:complexType〉

〈xs:simpleType name="URISchemeType"〉
    〈xs:restriction base="xs:string"〉
        〈xs:enumeration value="URI"/〉
    〈/xs:restriction〉
〈/xs:simpleType〉
```

〈표 3-16〉 Language 요소에 대한 Type 정의

```
〈xs:complexType name="languageType"〉
    〈xs:simpleContent〉
        〈xs:extension base="rfc3066_LanguageCodes"〉
            〈xs:attribute name="scheme" type="languageSchemeType" use="required"/〉
        〈/xs:extension〉
    〈/xs:simpleContent〉
〈/xs:complexType〉

〈xs:simplexType name="rfc3066_LanguageCodes"〉
    〈xs:list itemType="xs:language"/〉
〈/xs:simplexType〉

〈xs:simpleType name="languageSchemeType"〉
    〈xs:restriction base="xs:string"〉
        〈xs:enumeration value="ISO639-2"/〉
    〈/xs:restriction〉
〈/xs:simpleType〉
```

〈표 3-17〉 Relation 요소에 대한 Type 정의

```
〈xs:complexType name="relationType"〉
  〈xs:sequence〉
      〈xs:element name="dcterms:isPartOf" type="uriType" minOccurs="0" maxOccurs="unbounded"/〉
    〈xs:element name="dcterms:hasPart" type="uriType" minOccurs="0" maxOccurs="unbounded"〉
    〈xs:element name="dcterms:isVersionOf" type="uriType" minOccurs="0" maxOccurs="unbounded"/〉
    〈xs:element name="dcterms:hasVersion" type="uriType" minOccurs="0" maxOccurs="unbounded"/〉
    〈xs:element name="dcterms:isFormatOf" type="uriType" minOccurs="0" maxOccurs="unbounded"/〉
    〈xs:element name="dcterms:hasFormat" type="uriType" minOccurs="0" maxOccurs="unbounded"/〉
    〈xs:element name="dcterms:references" type="uriType" minOccurs="0" maxOccurs="unbounded"/〉
    〈xs:element name="dcterms:isReferencedBy" type="uriType" minOccurs="0" maxOccurs="unbounded"/〉
    〈xs:element name="dcterms:replaces" type="uriType" minOccurs="0" maxOccurs="unbounded"/〉
    〈xs:element name="dcterms:isReplacedBy" type="uriType" minOccurs="0" maxOccurs="unbounded"/〉
    〈xs:element name="dcterms:requires" type="uriType" minOccurs="0" maxOccurs="unbounded"/〉
    〈xs:element name="dcterms:isRequiredBy" type="uriType" minOccurs="0" maxOccurs="unbounded"/〉
  〈/xs:sequence〉
〈/xs:complexType〉
```

"dc:identifier"의 "uriType"과 동일하다.

⑪ Rights 요소에 대한 Type 정의

〈표 3-18〉은 〈표 3-7〉의 "dc:rights" 요소가 갖는 타입인 "rightsType"에 대한 정의이다. "rightsType"은 "dc:rights"가 "string" 타입의 값을 갖도록 정의하고 속성으로 "xml:lang"을 가지며, 해당 속성의 값은 "dc:language"의 값과 마찬가지로 세 자리 국가코드로 표현할 수 있도록 하였다. "dc:rights"의 내용이 영문인 경우, XML 인스턴스 표현은 [〈dc:rights xml:lang="eng"〉 영문내용 〈/dc:rights〉]가 된다.

⑫ Project Organization 요소에 대한 Type 정의

〈표 3-19〉는 〈표 3-7〉의 "ProjectOrganization" 요소가 갖는 타입인 "ProjectOrga-

〈표 3-18〉 Rights 요소에 대한 Type 정의

```
〈xs:complexType name="rightsType"〉
    〈xs:simpleContent〉
        〈xs:extension base="xs:string"〉
            〈xs:attribute ref="xml:lang"/〉
        〈/xs:extension〉
    〈/xs:simpleContent〉
〈/xs:complexType〉

〈xs:attribute name="xml:lang" type="rfc3066__LanguageCodes"/〉
```

〈표 3-19〉 Project Organization 요소에 대한 Type 정의

```
〈xs:complexType name="ProjectOrganizationType"〉
  〈xs:sequence〉
    〈xs:element name="owner" type="xs:string" minOccurs="0" maxOccurs="unbounded"/〉
    〈xs:element name="contractor" type="xs:string" minOccurs="0" maxOccurs="unbounded"/〉
    〈xs:element name="designer" type="xs:string" minOccurs="0" maxOccurs="unbounded"/〉
    〈xs:element name="constructionManager" type="xs:string" minOccurs="0" maxOccurs
      ="unbounded"/〉
  〈/xs:sequence〉
〈/xs:complexType〉

〈/schema〉
```

nizationType"에 대한 정의이다. "Project-OrganizationType"은 "ProjectOrganization" 요소가 네 개의 하위요소("owner", "contractor", "designer", "constructionManager")를 가지고 있음을 표현하고 있다. 네 개의 하위요소의 타입은 모두 "string" 형으로 선언되었으며, 필요할 경우에 하위요소로서 사용될 수 있음을 표현하였다. 주 요소뿐만 아니라 하위 네 개의 요소 모두 기본 네임스페이스 범주에 있으므로 식별자가 필요 없다. 마지막은 〈schema〉의 종결자인 마침 태그 〈/schema〉를 주었다.

4. XML 스키마 설계를 통한 건설공사정보의 통합검색

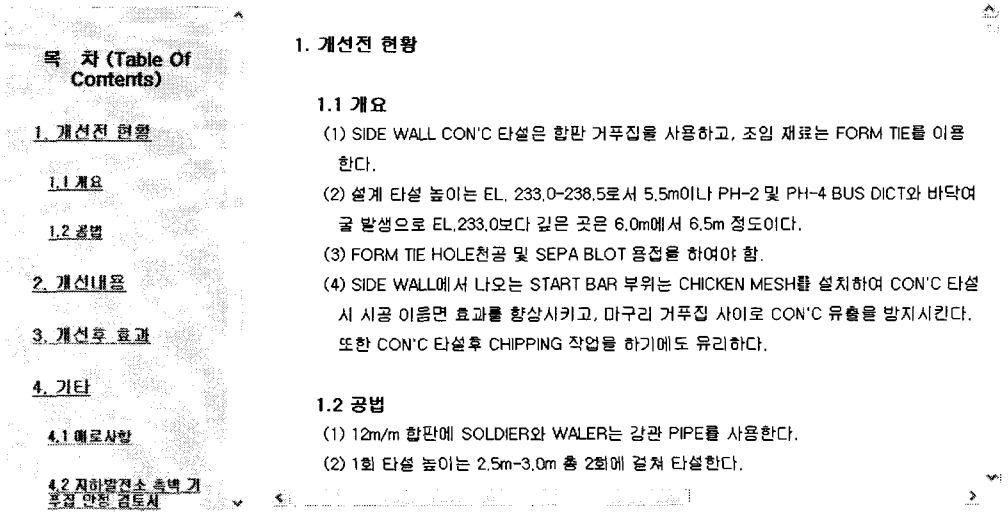
본 장에서는 앞의 3장에서 설계한 건설공사정보의 메타데이터 요소와 기술된 XML 스키마에 기반하여 XML 스키마 인스턴스를 작성하였다. 아래 〈그림 4-1〉은 각 메타데이터 요소별로 해당하는 데이터를 요소의 특성과 속성정의에 따라 입력된 예이다. 이러한 XML 스키마 인스턴스들이 모여 내부에 설계된 RDB 테이블의 해당 요소로 저장되며, 검색을 위한 메타데이터 DB를 구성하게 된다. 통합 메타데이터 검색을 통해 〈그림 4-1〉의 데이터를 검색하게 되면, 메타데이터 요소중 "identifier"에 정의된 "URI"를 통해 〈그림 4-2〉의 XML 원문을 볼 수 있게 된다.

건설공사정보 XML 스키마를 토대로 XML 메타데이터 인스턴스들과 XML 문서들을 연계하여 건설공사정보 DB를 검색할 수 있도록 "간략검색", "상세검색", "분류검색"으로 이들을 적용해 보면 아래와 같다. 〈그림 4-3a〉에서 〈그림 4-5b〉까지는 현재 서비스되고 있는 한국건

```
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
<?xml version="1.0" encoding="EUC-KR"?>
<KICTMetadata xmlns="http://www.kict.re.kr/metadata/terms/"
xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:dcterms="http://purl.org/dc/terms/"
xmlns:dcxml="http://purl.org/dc/xml/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:SchemaLocation="kict-metadata.xsd">

<dc:title>
        <mainTitle>보, 슬라브 거푸집 공법개선</mainTitle>
        <dcterms:alternative>
        보, 슬라브 거푸집 공법개선-U.E 활동계획서
        </dcterms:alternative>
</dc:title>
<dc:creator>
        <personalName>이한곤</personalName>
        <corporateName> 동아건설산업(주)  </corporateName>
</dc:creator>
<dc:creator>
        <personalName>이명찬</personalName>
        <corporateName>동아건설산업(주)</corporateName>
</dc:creator>
<dc:subject>
        <combined>
                F4213&gt;W354&gt;R215
        </combined>
</dc:subject>
<dc:description>
        <dcterms:abstract>
본 현장의 골조는 동일한 형태의 구조가 반복, 연속되므로 기존의 재래식 거푸집 처럼 사용할 때 마다 작은 부재을 조립, 분해하
        </dcterms:abstract>
        <dcterms:tableOfContents>
1. 개선전 현황
1.1 개요
1.2 공법
2. 개선내용
3. 개선후 효과
4. 기타
4.1 애로사항
4.2 지하발전소 측벽 거푸집 안정 검토서
```

〈그림 4-1〉 XML 스키마 인스턴스 예



〈그림 4-2〉 XML 문서 예

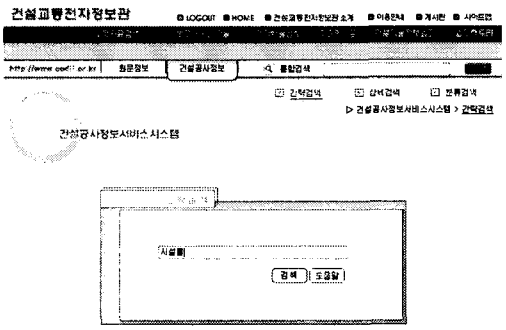
설기술연구원(http://www.kict.re.kr)의 건설교통전자정보관(http://www.codil.or.kr)에서 인용된 화면들이다.

〈그림 4-3a〉은 XML 스키마를 기반으로 구현한 인스턴스 들(그림 4-1 참조) 중, 필수로 지정된 메타데이터 요소들(title, creator, subject 등)을 통해 검색하는 "간략검색"이며, "간략검색" 결과는 〈그림 4-3b〉에서 보여주고 있다. "간략검색" 결과들 중 원하는 리스트를 클릭하면, 〈그림 4-2〉와 같이 해당 리스트의 XML 원문을 볼 수 있다.

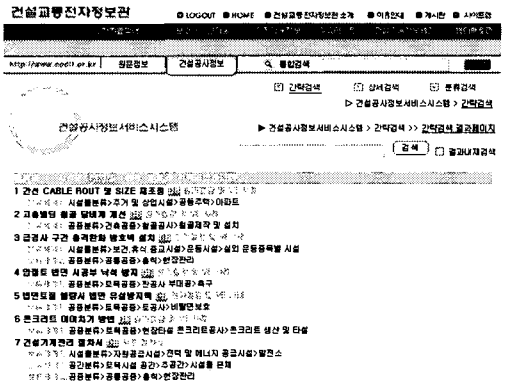
〈그림 4-4a〉는 "상세검색" 화면이며, XML 스키마 요소인 "type(자료유형)" 속성 의 "KICT-Type" 값으로 사용되는 "시공계획서(ExecutionPlanning)", "시공절차서(ConstructionGuide)", "원가절감(CostDown)/VE사례(ValueEngineering)", "건설공사지(ExecutionReport)"를 표현하는 선택 표시 단추와 "Title(Main Title: 표제)", XML 문서의 "Subject(주제어)", "Body(본문)" 세 요소의 "or" 조합으로 검색한다. 결과화면은 "간략보기" 형식으로, 〈그림 4-4b〉에서 보여주고 있다. "상세검색" 또한 원하는 리스트를 클릭하면 해당 리스트의 XML 원문을 볼 수 있다.

〈그림 4-5a〉는 "분류검색" 화면이며, "상세검색"과 마찬가지로 XML 스키마 요소 중 "type(자료유형)" 요소가 갖는 값들과, "subject(주제)" 요소의 하위요소인 "combined"와 "specific"의 값들(KCICf: 시설물분류, KCICs: 공간분류, KCICe: 부위분류, KCICw: 공종분류, KCICr: 자원분류)의 조합으로 검색한다. 검색 결과화면은 "간략보기" 형식으로 〈그림 4-5b〉에서 보여주고 있다. "분류검색"은 원하는 리스트를 클릭하면 해당 리스트의 XML 원문을 볼 수 있다.

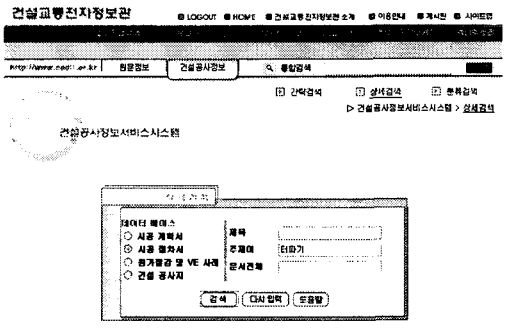
상기의 통합검색 인터페이스를 위한 메타데이터 XML 스키마 요소들의 RDB 테이블과 테이블 필드 설계는 요소들과 해당 요소들이 갖는 속성들의 명확성으로 인해, 쉽게 정의될 수 있었고, 이들의 특정 데이터 타입의 명확성



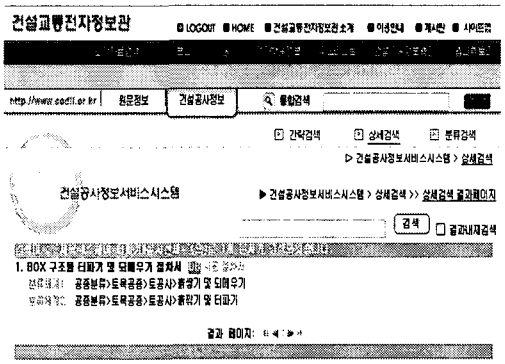
〈그림 4-3a〉 간략검색



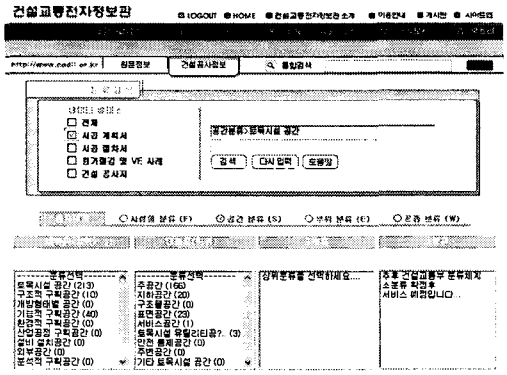
〈그림 4-3b〉 간략검색 결과



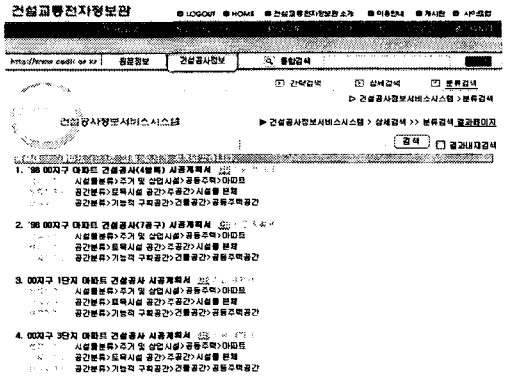
〈그림 4-4a〉 상세검색



〈그림 4-4b〉 상세검색 결과



〈그림 4-5a〉 분류검색



〈그림 4-5b〉 분류검색 결과

으로 인해 DB 테이블로 로딩하기 위한 특정 프로그램 없이도 쉽게 적용할 수 있었다. 그 외에 날짜별, 자료형태별, 관련자료 등과 같은 검색은 초기 XML DB구축으로 인하여 아직 지원이 부족한 상태이나, 향후 DB가 상당량 축적되면 보완될 수 있을 것이다.

5. 결 론

본 연구에서는 국내 건설공사정보 DB의 통합검색을 위해 건설공사정보 DB로부터 Dublin Core 기반의 메타데이터 요소들을 도출하고, 이 요소들을 최근 메타데이터의 기술언어로 부각되고 있는 XML 스키마로 기술하였다. 기술된 XML 스키마를 통해, 기존의 XML DTD로는 표현할 수 없었던 날짜(date) 및 접근정보(identifier) 요소들을 쉽게 처리 할 수가 있었으며, 사용자 정의에 따른 자료형 정의에 있어서도 자료형태(format)와 주제(subject)의 인코딩 스킴에서 처럼 얼마든지 확장 가능함을 볼 수 있었다. 특히, 앞서 기술된 건설공사정보의 메타데이터들에 대한 XML 스키마의 구조와 자료형 정의들은 DB에 저장하기 위한 스키마와 형 선언을 정확하게 하여, XML DTD에서처럼 자료형 처리를 위한 별다른 응용프로그램 없이도 처리 할 수가 있었다.

본 연구의 의의는 건설공사정보의 통합 검색을 위하여 건설산업분야에서 처음으로 메타데이터 요소들을 XML 스키마로 기술하여 이를 실제로 적용하였다는 것이다. 또한 국내 건설산업분야 DB 구축에서 검색을 위한 메타데이터 요소의 표준화를 제시하여 건설공사정보의 DB 확충은 물론 향후 타 기관별 DB의 단계적 연계를 통한 다중 DB 메타데이터 통합검색 및 공동활용의 기반을 다졌다는데 본 연구의 의의가 있다고 할 수 있다.

앞으로, 국내 건설산업분야 뿐만 아니라 국내 모든 연구분야에 까지도 확장하여 메타데이터 표준화를 확장하고, 이들을 XML 스키마로 기술하여 서로 다른 분야들의 XML 스키마 정보들을 축적해놓은 애플리케이션 프로파일(application profile)을 개발함으로써, 이를 통해 대규모 정보자원 공유 및 상호 정보교환이 원활하게 이루어지도록 하는 연구가 계속되어야 할 것이다.

참 고 문 헌

채진석. 2000. XML문서와 DB의 효율적인 연동을 위한 XML 스키마 기술연구. 『데이터베이스 연구회지』, 16(2): 23-28.

황병연, 김연혜. 2001. XML 스키마의 발전 동향. 『한국정보처리학회지』, 8(3): 3-9.

Birbeck, Mark, et al. 2001. 『Professional XML』. 2nd Edition. Birmingham, Wrox Press Ltd., pp.237-280.

Biron, Paul V., Ashok Malhotra. 2001.

XML Schema Part 2: Datatypes. (http://www.w3.org/TR/xmlschema-2/).

DCMI. 1999. Dublin Core Metadata Element Set, Version 1.1: ReferenceDescription. (http://www.dublincore.org/documents/dces/).

DCMI. 2000. Dublin Core Qualifiers. (http://dublincore.org/documents/2000/07/ 11/dcmes-qualifiers/).

Duckett, Jon, Oliver Griffin, Stephen Mohr, Francis Norton, Ian Stokes-Rees, Kevin Williams, Kurt Cagle, Nikola Ozu, and Jeni Tennison. 2001. 『Professional XML Schemas』. Birmingham, Wrox Press Ltd.

IANA. 2001. Media Types. (http://www.isi.edu/in-notes/iana/assignments/media-types/media-types).

Thompson, Henry S., David Beech, Murray Maloney, Noah Mendelsohn. 2001. XML Schema Part 1: Structures. (http://www.w3.org/TR/xmlschema-1/).

Wahlin, Dan 2002. "XML Schemas vs. DTDs with the release of XML schemas, you have a more power-ful mechanism for validating XML documents". (http://www.fawcette.com/xmlmag/2002__04/online/online__eprods/xml__dwahlin04__22/default__pf.asp#).