XML Schema 기반 이질 정보 통합의 충돌 분류와 해결 방안

권 석 훈* · 이 경 하** · 이 규 철***

Classification and Resolution of Conflicts for Integration of Heterogeneous Information Based on XML Schema

Seok-Hun Kwon* · Kyong-Ha Lee** · Kyu-Chul Lee***

Abstract

Due to the evolution of computer systems and the proliferation of Internet, numerous information resources have been constructed. The deluge of information makes the need to integrate information, which are distributed on the Internet and are handled in heterogeneous systems. Recently, most of the XML-based information integration systems use XML DTD(Document Type Definition) for describing integrated global schema. However, DTD has some limitations in modeling local information resources such as datatypes.

Although W3C's XML Schema is more flexible and powerful than XML DTD in specifying integrated global schema, it has more complex problems in resolving conflicts than using DTD. In this paper, we provide a taxonomy of conflict problems in integration information resources using XML Schema, and propose conflict resolution mechanism using XQuery.

Keywords : Database, XML, 정보 통합

* (주)우린정보 기술연구소 연구원
** 충남대학교 컴퓨터공학과 박사과정
*** 충남대학교 컴퓨터공학과 교수

1. 서 론

컴퓨터 시스템이 발달하고 그에 따라 인터넷의 사용도 보편화되어 가는 현대 정보 사회에서는 다양한 정보들이 인터넷상에 분산되어 존재하고 있다. 이러한 정보 자원들은 파일, 데이터베이스, 웹 응용(ASP, JSP, CGI, PHP), 또는 웹 서비스(web service) 등의 이질적인 형태로 분산되어 존재하므로 이들 정보를 통합하여 접근하고자 하는 응용(application)이나 사용자가 이질적이고 분산되어 있는 정보 자원들에 대한 접근을 단일화된 형태로 할 수 있는 방법이 요구되어 왔다. 이와 같은 요구에 따라 분산 이질 정보 자원에 대한 단일화된 접근 방식을 제공하기 위하여 정보 통합(information integration)에 대한 많은 연구가 진행되어 왔다[1]. 이러한 정보 통합에서는 각 정보 자원들이 저장하고 있는 데이터의 데이터 모델과 스키마의 **이질성**(heterogeneity), 데이터의 의미적인 차이 등에 의해서 데이터 간의 **충돌**(conflict)이 발생할 수 있으며, 이러한 충돌은 정보 통합을 어렵게 하는 주요 원인으로 작용한다.

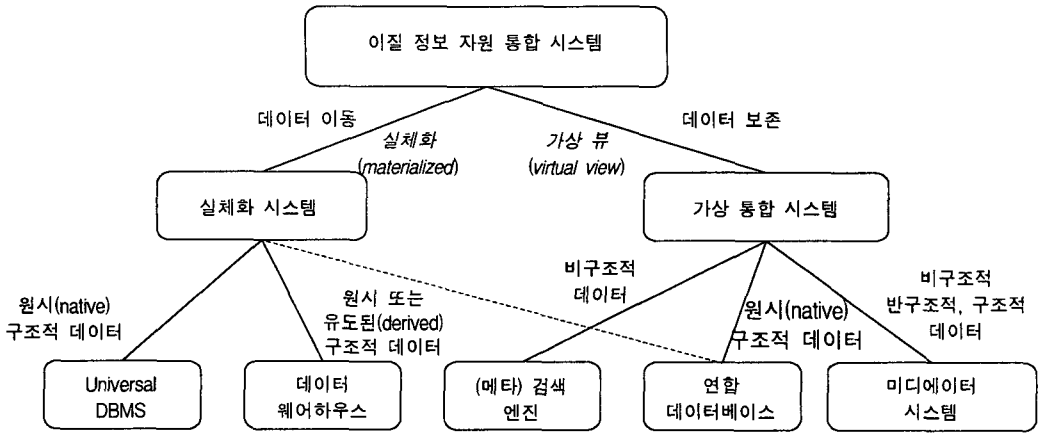
최근에는 통합 과정 시 사용하는 **공통 데이터 모델**(common data model)로 XML[2]을 이용하여 단일화함으로써 데이터 모델에서의 이질성을 해결하며, XML 데이터에 대한 스키마로 DTD 또는 XML Schema[3]를 이용함으로써 스키마 표현에 대한 이질성을 해결하기 위한 연구가 진행되어 왔다[8, 9, 10]. 하지만 이러한 경우에도 유사한 정보 자원들에 대해 다른 스키마의 정의에서 발생하는 이질성과 데이터의 의미적인 이질성은 여전히 남게 되므로, 이에 대한 분류와 해결방안이 모색되어야 한다. 본 논문에서는 이러한 이질성을 통합하고자 하는 데이터의 스키마로부터 각 정보 자원의 이질적인 차이를 확인할 수 있는 **스키마 충돌**(schematic conflict)과 스키마에서는 확인할 수 없는 의미적인 차이에 대한 **의미적 충돌**(semantic conflict)로 크게 분류하였다. 스키마 충돌이란, 각 데이터의 데이터 타입, 구조, 정보의 이름과 같이 선언적으로 명시되어 있는 스키마 정의상에서 발생하는 충돌이며, 의미적 충돌이란 정보의 값 자체에 다른 단위나 표현 방법을 사용하는 데에서 발생하는 충돌이다.

이와 같은 스키마 충돌과 의미적 충돌은 정보 통합 시 나타나는 일반적인 현상으로서, 어떤 데이터 모델을 공통 데이터 모델로 사용하느냐에 따라 충돌이 나타나는 방식이 다양해지기 때문에, 공통 데이터 모델의 충돌 대상에 대한 분류와 이에 따른 각 스키마 충돌과 의미적 충돌의 분류 및 각 분류에 대한 해결 방안의 제시가 필요하다. 이를 위해 기존에 김 원[4], Lakshmanan[6], Sattler[7] 등은 객체 지향 데이터 모델 또는 관계형 모델을 기반으로 한 충돌의 분류 및 해결책을 제시한 바가 있으나 XML Schema에 대한 충돌의 분류와 해결 방법, 절차 등에 대해서는 아직까지 충분한 연구가 진행되지 않아 왔다. 이에 따라 본 논문에서는 XML Schema를 이용한 스키마 통합 시에 발생하는 여러 가지 충돌에 대해 체계적으로 분류하고, 각 충돌에 대해 XQuery[5]를 이용한 통합 스키마 정의 방법을 통한 해결 방안을 제시한다. 또한 본 논문에서 제안한 충돌 분류와 해결 방안에 따른 XQuery 정의를 생성할 수 있는 통합 스키마 저작 도구에 대해 설명한다.

2. 관련 연구

2.1 정보 통합 시스템

이질적인 정보 자원들에 대한 기존의 통합 방법에 대한 연구는 (그림 1)의 분류에서 보는 바와 같이 크게 두 가지로 나눌 수 있다. 첫째로



(그림 1) 이질 정보 통합의 접근 방식 분류

는 정보 자원들의 데이터를 가져와서 이를 통합하여 저장한 후 이에 대한 질의를 지원하는 실체화 시스템(materialized system)을 구축하는 방법이 있으며, 둘째로는 통합하고자 하는 정보 자원들의 데이터를 실제로 실체화하는 것이 아니라 각 정보 자원들을 네트워크, 또는 인터페이스로 연결하고 이에 대해 질의를 분해, 전달 후 질의 결과를 통합하는 가상 통합 시스템(virtual integrated system)을 구축하는 방법이 존재한다[1].

실체화 시스템의 경우에는 통합 데이터베이스에 질의하기 때문에 질의 시에 질의 변환이나 결과 통합 등의 과정이 요구되지 않는다. 가상 통합 시스템의 경우는 통합 대상이 되는 데이터를 이동하지 않고 통합을 요구할 때에 지역 시스템으로 질의를 하여 그 결과만을 전역 시스템으로 전달하므로 지역 시스템의 자치성을 보장해 줄 수 있지만, 실체화 시스템보다는 질의 변환과 결과 통합의 과정이 복잡하다. 가상 통합 시스템 중에서 특히 미디에이터 시스템은 데이터 모델을 지역 시스템과는 독립적으로 사용할 수 있기 때문에 다양한 이질성의 문제를 해결할 수 있고, 통합하고자 하는 정보 자원의 형태가 정형 또는 비정형 데이터에 상관없이 모두 통합할 수 있는 장점을 가진다. 따라서 인터넷 정보 자원들과 같이 그 구조가 다양하고 시간에 따라 변화량이 많은 정보들을 통합하기 위한 시스템에서는 현재까지 미디에이터 구조가 가장 적합하다고 할 수 있다.

2.2 충돌 문제 해결

2.2.1 데이터베이스의 통합 시 충돌 문제 해결

이질 데이터베이스 통합에 관한 연구는 최근까지 많은 활동이 있어 왔으며, 특히 통합 시에 발생하는 충돌을 해결하고자 다음과 같은 연구들이 진행되어 왔다.

Lakshmanan[6]은 관계 데이터베이스에 대한 통합을 데이터베이스 질의어인 SQL을 확장한 Schema SQL라는 데이터베이스에 대한 스키마 정의 언어를 개발하여 수행하고자 하였다. 사용자는 기존에 사용해 왔던 SQL 문법과 크게 다르지 않은 SchemaSQL로 통합 대상이 되는 지역 데이터베이스들의 스키마에 대한 대응 관계를 기술하며, SchemaSQL은 이러한 기술을 대해 재구성된(restructured) 통합 뷰를 생성하고 그 스키마도 동적으로 생성하는 기능을 제공하며, 이러한 기능을 통하여 관계 데이터베이스간의 통합 시에 발생하는 충돌 문제를 해결한다.

Sattler[7] 또한 관계 데이터베이스에 대한 통합을 위해 FraQL이라는 스키마 정의 언어를 개발하였는데, FraQL은 SchemaSQL과 유사한 형태의 데이터베이스 통합을 지원하지만 객체관계 데이터 모델(Object-Relational Data Model)에 기반을 두어 새로운 정보 자원의 동적인 통합을 위해 사용자 데이터 타입 정의 및 함수 정의의 기능을 제공하였다. 이 연구에서는 통합 시의 충돌 해결에 대한 순서를 제시하였는데, (1) 광역 스키마(타입)를 정의하고, (2) 통합하고자 하는 릴레이션들에 대해 애트리뷰트들 간의 대응 관계를 명시한 후, (3) 해당하는 릴레이션들에 대한 통합을 수행하기 위해 구조적인 충돌을 먼저 해결하고 의미적인 충돌을 해결하는 순서를 제안하였다.

앞서 설명한 관계 데이터베이스 통합에 대한 연구와는 달리, [4]는 통합의 대상이 관계 데이터베이스만으로 제한되지 않고 다양한 종류의 데이터베이스를 통합할 수 있는 환경을 제시하였다. 또한 이 연구에서는 멀티 데이터베이스 시스템을 이용하여 이질적인 데이터베이스 시스템들을 통합할 때 발생하는 충돌에 대해서 크게 엔터티(entity)와 애트리뷰트(attribute) 간에 발생하는 구조적인 충돌과 동일한 데이터의 이질적인 표현에서 발생하는 의미적인 충돌로 나누었다. 그리고 이러한 대상에 따른 분류에 대하여 여러 가지 경우에 따른 충돌로 세분화하였고 이에 대한 해결방안도 제시하였다. 이 연구는 이질 정보 자원의 통합에 있어서의 충돌과 해결 방안을 체계적으로 분류하였고 이를 기반으로 충돌을 해결하기 위한 통합 뷰를 정의하지만 그 대상이 여전히 데이터베이스 시스템에 국한되어 있다.

앞에 언급한 데이터베이스 통합에 관한 연구들은 통합 시에 발생하는 충돌에 대한 해결 방안을 광역 스키마 정의를 통해 제시하였지만, 그 통합의 대상이 데이터베이스 시스템에 한정되어 있기 때문에, 인터넷 환경에 분산되어 있는 다양한 형태의 정보에 대한 통합에 적용할 수 없다는 단점이 존재한다.

2.2.2 XML 기반 정보 통합 시 충돌 문제 해결

XML 기반의 정보 통합에 대한 연구로는 다음과 같은 연구가 진행되어 왔다.

TSIMMIS(The Standard-IBM Manager of Multiple Information Sources)[8]는 일반적인 파일이나 DBMS에 대한 통합을 수행하는 미디에이터 시스템으로써, 통합하고자 하는 정보 자원의 광역 데이터 모델로 반구조적 데이터 모델인 OEM을 사용하였다. TSIMMIS에서는 datalog, Hilog, F-Logic 등의 논리 기반의 언어에서 영향을 받은 MSL(Mediator Specification Language)이라는 규칙 정의 언어를 사용하여 OEM으로 기술된 데이터에 대한 뷰를 정의하여 정보 자원 통합 시에 발생하는 충돌 문제를 해결하였다.

YAT[9]는 XML 기반 정보 통합 시스템으로서, XML 기반 통합 시스템에서 효과적인 질의 처리를 위한 대수식 기반의 접근 방식을 제안하였다. YAT는 YATL이라는 언어로 통합에 관련된 여러 가지 관계를 정의하고 통합 정보에 대한 질의를 수행하는 과정에서 YATL로 정의한 규칙 및 관계 기술은 YATL 대수식(algebraic expression)으로 변환되어 XML의 데이터 타입에 대한 연산과 질의문의 최적화를 위한 연산을 수행하여 효과적인 XML 기반 정보 통합의 수행을 목표로 한다.

XML 기반의 통합 시스템인 MIX(Mediation of Information using XML)[10]는 XML 자원들에 대한 통합을 수행하는 미디에이터 방식의 통합 시스템이다. 이 시스템은 미디에이터 뷰 정의에 XMAS(XML Matching And Structuring Query Language)를 이용하여 XML 자원이 제

공하는 데이터에 대한 뷰를 정의하고, 이 통합된 뷰에 대하여 질의를 수행한다. 결과 데이터 통합 시에 DTD Inference 모듈을 통하여 결과에 대한 DTD를 생성해낸다. MIX는 XMAS로 정의된 미디에이터 뷰 정의에 검색 조건을 추가시키는 방법으로, 즉 질의 처리 과정으로 데이터 통합을 수행한다. 따라서 글로벌 스키마의 정의 및 질의 정의를 일관되게 하고 있으며, 공통 데이터 모델로서 XML을 이용함으로써 반구조적인 데이터의 통합 또한 가능하게 하고 있다.

그러나 앞에서 살펴 본 XML 기반 정보 통합 방법은 통합의 대상이 특정 데이터 모델에 한정되어 있고 통합 시에 발생하는 충돌에 대한 분류가 명확히 정의되지 않았으며, 이에 따라 그 해결 방안도 체계적으로 제시되지 않아 왔다. 또 다른 XML 기반의 통합 시스템으로는 미디에이터 방식의 XMF(XML-based Mediation Framework) 시스템이 있으며 MIX와는 달리 XML에 대한 스키마 정의로서 DTD를 사용하였으며[11, 12], 그 통합 대상도 데이터베이스나 XML 자원뿐만 아니라 인터넷상에 존재하는 다양한 자원들을 포함하고 있으며, 통합 시 발생하는 충돌을 체계적으로 분류하였고 그에 따른 해결 방안도 제시하였다.

XMF 시스템에서는 정보 자원들을 통합할 때 발생하는 충돌을 크게 스키마 충돌과 데이터 충돌로 분류하였다. 스키마 충돌은 XML 데이터 모델을 따르는 정보 자원의 구조, 즉, DTD의 이질성에 의하여 발생하는 충돌 문제이며, 데이터 충돌은 정보 자원에 대한 표현 방법 등의 이질성에 의해 발생하는 충돌 문제이다. XMF에서는 이와 같이 분류한 충돌 문제에 대하여 스키마간의 연관 관계를 정의한 중재 규칙을 이용하여 해결 방안을 제시하였다. 하지만 XMF는 정보 자원에 대한 스키마 정의에 DTD를 사용하였는데, DTD는 3.2.1절에서 설명하고 있듯이 여러 가지 단점을 나타내어 DTD의 단점을 보완할 수 있는 새로운 스키마 정의 언어가 요구되었다.

3. 이질 정보 통합 환경

3.1 XML Schema

본 연구에서는 이질 자료 모델의 통합을 위해 각 정보 자원들의 자료들을 공통적으로 표현하기 위한 공통 데이터 모델로서 XML을 이용하였다. XML 문서는 엘리먼트들의 의미와 순서, 그리고 엘리먼트들이 가질 수 있는 특성들을 정확하게 명시하는 스키마를 가지며 보통 DTD나 XML Schema 등이 주로 스키마로써 사용된다.

3.1.1 DTD의 단점

DTD를 스키마 정보로 활용하는 정보 통합 방법은 다양한 이질 정보들을 인터넷상에서의 구조화된 문서를 표현하기 위한 언어인 XML이라는 형태로 통합함으로써 사용자들이 데이터가 저장된 시스템의 환경이나 저장형태에 무관하게 동일한 질의를 통해 데이터들을 통합 검색할 수 있게 하였다. 하지만 DTD는 여러 가지 면에서 많은 단점이 지적되었고 이에 따라 점차 그 한계를 드러내었다.

여러 연구에 의해 지적된 DTD의 단점을 살펴보면 다음과 같다.

- 데이터 타입을 명시할 수 있는 기능이 극히 제한적이어서 데이터로서의 XML을 제대로 기술하지 못한다.
- 한 DTD에서 다른 DTD로의 상속의 개념이 없고, 확장할 수 없다.
- Namespace를 지원하지 못한다.
- DTD는 XML 문법(Syntax)을 사용하지 않기

때문에, DTD에 관련된 소프트웨어가 별도로 있어야 하며, 파서 또한 별도로 존재해야 한다.

3.1.2 XML Schema의 특징

W3C에서는 DTD의 이러한 단점을 보완한 XML Schema[3, 14, 15]라는 새로운 스키마 정의 언어를 표준화하였으며 그 특징은 다음과 같다.

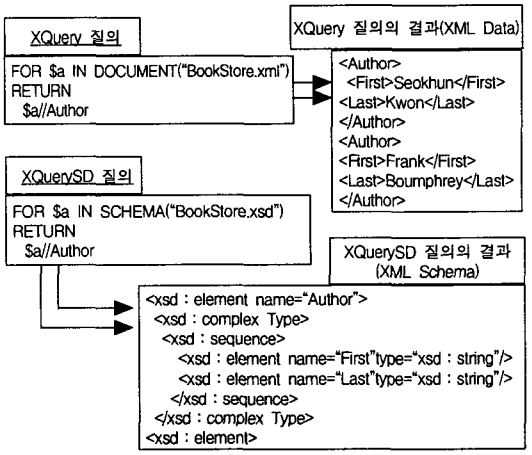
- 44개 이상의 데이터 타입을 지원하며, 스키마 작성자가 데이터 타입을 만들어낼 수 있다. (DTD는 10개 정도의 데이터 타입 지원)
- 객체지향(Object-Oriented)적이기 때문에, 데이터 타입에 대한 확장, 제한이 가능하다.
- Namespace를 지원하여 같은 이름을 가지면서 다른 내용을 포함하는 다중 엘리먼트를 정의할 수 있다.
- XML 문서와 동일한 문법을 사용하여 스키마 기술을 위한 별도의 기술을 요구하지 않으며, 스키마의 처리를 위한 별도의 파서도 필요치 않는다.

그러나 XML Schema의 이런 장점은 스키마 통합 시 DTD를 사용할 때의 여러 단점들을 보완할 수 있지만, DTD에 비해 훨씬 복잡하고 방대해진 데이터 타입과 제약 조건(Constraint) 등은 데이터 통합 시 많은 문제점을 야기하게 되었다.

3.2 XQuerySD

정보 통합 시스템에서는 Lakshmanan[6], Sattler [7] 등의 연구에서 제안한 것처럼, 광역 스키마와 지역 스키마 간의 대응 관계를 표현할 수 있는 스키마 정의 언어가 필요하며 본 연구에서도 이와 같은 요구에 의해 스키마 정의 언어인 XQuerySD(XQuery for Schema Definition)를 정의하였다. XQuerySD는 XQuery에 기반을 둔 통합 스키마 정의 언어로서, 정의된 XQuerySD는 정보통합 시스템 내부에서 XML Schema 규격으로 추론되어 저장된다[13].

XQuerySD는 XQuery의 거의 모든 기능과 특성들을 가지고 있지만 XQuery가 실제 XML 데이터에 대한 질의문이라면 XQuerySD는 XML 데이터의 스키마 정보를 기술한 XML Schema에 대한 정의로써 이용된다. (그림 2)에서 보이고 있듯이, XQuery 질의는 BookStore.xml이라는 XML 인스턴스를 대상으로 하고 있으며, XML 인스턴스의 부분적인 엘리먼트 집합이 질의의 결과로써 제공된다. 반면에, XQuery SD는 BookStore XML 인스턴스에 대한 스키마 정의인 BookStore.xsd를 대상으로 하고 있으며, 그 결과로서 XML Schema 형태의 스키마 정의를 제공하고 있다.



(그림 2) XQuery와 XQuerySD의 비교

본 논문에서는 XQuerySD를 이용하여 이질적인 스키마에 대한 통합을 수행한다. XQuerySD가 가지는 XQuery와의 차이점은 다음과 같다.

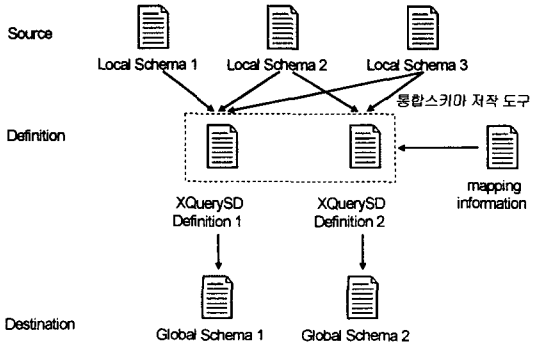
- 표현식이 엘리먼트 생성자와 FLWR 표현식으로 제한
- XQuerySD의 처리 결과는 XML Schema의 유효성을 만족하는 결과 스키마를 생성

- 지역 스키마를 지정할 수 있는 FLWR 표현식
- 조건에 의한 동적 스키마 생성을 지원하지 않음

이와 같이 XQuerySD에서 XQuery의 일부 기능들을 제한한 것은 XQuerySD의 그 결과 값이 XML Schema 형태로 나타나기 때문에 실제 XML 데이터를 결과 값으로 하는 XQuery에서 사용되는 모든 연산을 제공할 필요가 없기 때문이다.

3.3 통합 스키마 저작 도구

본 논문에서 분류한 충돌 유형과 해결 방안은 통합 스키마 저작 도구[16]에서 이용되며, 이 시스템은 미디에이터 기반 정보 통합 시스템인 Data blender[13]의 일부로써 동작한다. 통합 스키마 저작 도구는 XMLSchema로 정의된 각 지역 스키마들, 그리고 각 스키마 내 요소들에 대한 매핑 정보들을 입력으로 받아 중재 규칙 및 광역 스키마 정의를 XQuerySD 형태로 제공한다. 각 지역 스키마들을 통합하는데 있어서 여러 가지 통합 연산이 적용될 수 있으며, 이러한 정보 통합 시의 통합 연산은 Shanmugasundaram [17]이 제안한 연산들을 그 대상으로 한다. (그림 3)은 통합 스키마 저작 도구의 역할을 도식화한 것이다.



(그림 3) 지역 스키마 중재와 통합 스키마의 작성

4. 이질 자료 모델의 통합 시의 충돌 분류 및 해결 방안

충돌 문제를 분류하고 이에 대한 해결방안을 제시하기 위해서는, 먼저 충돌을 발생시키는 충돌 대상을 명확히 찾아내야 한다. XML Schema를 통한 데이터 통합 시 충돌 문제를 야기하는 대상으로는 엘리먼트와 애트리뷰트가 존재한다. XML Schema에서의 구성 요소는 엘리먼트 또는 애트리뷰트만으로 한정되어 있으며, DTD에서의 엔터티나 노테이션(Notation) 선언은 지원하지 않는다. 또한 처리 지시부(Processing Instruction)와 주석(Comment)과 같은 구성 요소들은 XML Schema에서는 모두 <annotation> 태그로 대체되는데 이러한 <annotation>은 실제 스키마의 구성 요소로서 작용하지 않는 것이기 때문에, <표 1>에서와 같이 XML Schema에서의 충돌 대상을 분류할 수 있다.

XML Schema에서는 엘리먼트나 애트리뷰트의 선언에 대해서 각각의 컨텐트 모델을 타입으로서 지원하고 있는데, 이러한 타입에는 크게 Simple Type과 Complex Type이 존재한다. Complex Type은 한 엘리먼트가 내부에 하위 엘리먼트를 포함하고, 애트리뷰트를 포함할 수 있는 경우를 표현하기 위한 컨텐트 모델 정의이며, Simple Type은 하위 엘리먼트나 애트리뷰트를 포함하지 않는 컨텐트 모델에 대한 정의이다. Complex Type이나 Simple Type과 같은 타입 선언은 각 엘리먼트 또는 애트리뷰트가 가지는 컨텐트 모델에 대한 정의이다. 따라서 타입 선언은 엘리먼트나 애트리뷰트가 가지는 컨텐트 모델에 대한 구조의 정의에 이용된다. 그러나 Simple Type의 경우에는 컨텐트 모델이 사용하는 데이터 타입과 제약 조건에 대한 명세로서 이용되므로 이는 구조적 충돌보다는 데이터 타입 충돌 또는 제약 조건에 대한 충돌에 해당된

〈표 1〉 충돌 대상의 분류

충돌 대상 분류	세부 분류
1. 엘리먼트 대 엘리먼트	A. 엘리먼트의 컨텐트 모델이 모두 Simple Type인 경우 B. 엘리먼트의 컨텐트 모델이 모두 Complex Type인 경우 C. 엘리먼트의 컨텐트 모델이 한쪽은 Simple Type이고 다른 한쪽은 Complex Type인 경우
2. 엘리먼트 대 애트리뷰트	A. 엘리먼트의 컨텐트 모델이 Simple Type이고 단일 애트리뷰트인 경우 B. 엘리먼트의 컨텐트 모델이 Complex Type이고 단일 애트리뷰트인 경우 C. 엘리먼트의 컨텐트 모델이 Simple Type이고, 애트리뷰트 그룹 또는 복수 개의 애트리뷰트인 경우 D. 엘리먼트의 컨텐트 모델이 Complex Type이고, 애트리뷰트 그룹 또는 복수 개의 애트리뷰트인 경우
3. 애트리뷰트 대 애트리뷰트	A. 양쪽 모두 단일 애트리뷰트인 경우 B. 한쪽은 복수 개의 애트리뷰트 또는 애트리뷰트 그룹이고 다른 한쪽은 단일 애트리 뷰트인 경우 C. 양쪽 모두 복수 개의 애트리뷰트 또는 애트리뷰트 그룹인 경우

다. 따라서 <표 1>에서 1-A, 3-A에서는 구조적 충돌은 발생하지 않으며, 데이터 타입 충돌만이 발생하게 된다. 이와 같은 분류에 따라 본 논문에서는 XML Schema에서의 충돌 유형을 각 충돌 대상에 따라 <표 2>에서 보이는 바와 같이 분류하였다.

〈표 2〉 충돌 대상에 따른 충돌 유형의 분류

1. 엘리먼트 vs. 엘리먼트
   (a) 1 : 1 엘리먼트
      i. 엘리먼트 이름
         - 이음동의어 : 동일 엘리먼트에 대한 서로 다른 이름
         - 동음이의어 : 서로 다른 엘리먼트에 대한 같은 이름
      ii. 엘리먼트의 제약 조건
         - 엘리먼트의 발생 빈도
         - 컨텐트 모델(타입)
         - 기본 값
         - Null 유무
         - Facet
      iii. 엘리먼트 구조
         - 누락된 애트리뷰트
         - 모호한 애트리뷰트
      iv. 데이터 타입
         - 기본형 vs. 기본형
         - 기본형 vs. 유도형
         - 유도형 vs. 유도형
   (b) M : N 엘리먼트
      i. 엘리먼트 이름
      ii. 엘리먼트 제약 조건
      iii. 엘리먼트 구조
         - 자식 엘리먼트의 발생 빈도
         - 자식 엘리먼트의 순서
         - 자식 엘리먼트의 데이터 타입
         - 자식 엘리먼트의 이름
         - 자식 엘리먼트의 노드 구조

2. 애트리뷰트 vs. 애트리뷰트
   (a) 1 : 1 애트리뷰트
      i. 애트리뷰트 이름
      ii. 애트리뷰트 제약 조건
         - 발생 빈도
         - 기본값
         - 데이터 타입
         - Facet

3. 엘리먼트 vs. 애트리뷰트
   - 1. 2와 동일하나 엘리먼트 구조에 대한 비교는 수행할 수 없음.

4. 동일한 데이터에 대한 서로 다른 표현
   (a) 동일한 정보에 대한 서로 다른 표현
   (b) 서로 다른 단위
   (c) 서로 다른 측정 레벨

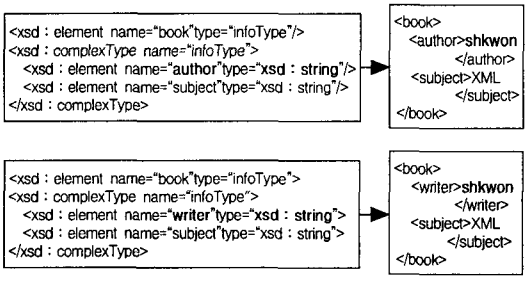
4.1 스키마 충돌

스키마 충돌(Schematic Conflict)은 데이터의 이름, 구조, 데이터 타입과 같이 스키마 상에 명시되어 있는 데이터의 설명을 통하여 확인할 수 있는 충돌로 이름 충돌, 구조적 충돌, 데이터 타입 충돌 그리고 제약 조건에 따른 충돌 등으로 분류할 수 있다.

4.1.1 이름 충돌

이름 충돌(naming conflict)은 같은 의미의 엘리먼트나 애트리뷰트가 다른 이름으로 정의(이음동의)되거나, 다른 의미의 엘리먼트나 애트리뷰트가 같은 이름으로 정의(동음이의)되었기 때문에 발생하는 충돌이다.

예를 들어, (그림 4)의 XML Schema 정의는 모두 책의 저자와 제목을 나타내는 데에 사용되지만, 한 쪽은 저자를 나타내는 엘리먼트를 <author>로, 다른 한 쪽은 <writer>로 이름을 다르게 나타내어 이들의 통합 시에 이름 충돌(이음동의)을 유발한다.

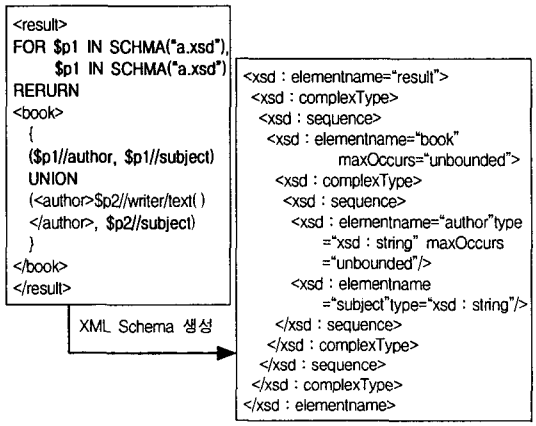


(그림 4) 이름 충돌의 예

[해결 방안 1] 이름 충돌 해결

이름 충돌은 충돌을 발생하는 엘리먼트나 애트리뷰트의 이름을 변경시키는 방법(renaming)으로 해결한다.

(그림 5)는 이름의 변경을 통하여 이름 충돌을 해결하는 XQuerySD의 예이다. 이 예제는 동일한 의미를 가지지만 다른 엘리먼트 이름으로 정의된 <author>와 <writer>가 이름 충돌을 발생하기 때문에 두 엘리먼트의 이름을 일치시켜야 한다. 그래서 b.xsd의 <writer>로 명명된 엘리먼트를 <author>로 변경시켜 b.xsd의 <author> 엘리먼트와 union 연산을 수행시키게 된다.



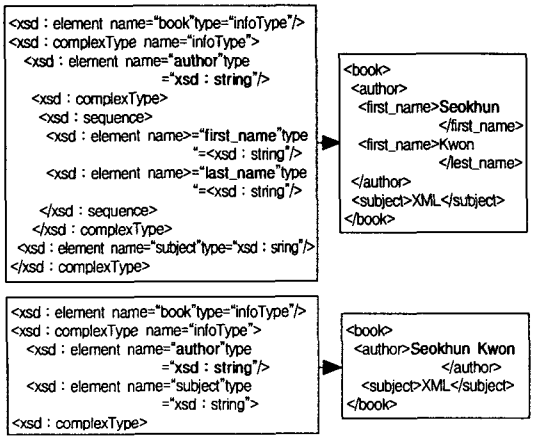
(그림 5) 이름 충돌의 해결

4.1.2 구조적 충돌

구조적 충돌(structural conflict)은 엘리먼트와 애트리뷰트가 가지고 있는 구조적인 이질성으로 인해 발생하는 충돌로서, 통합의 대상이 되는 데이터들이 같은 의미를 지니고 있지만 그들을 나타내는 컨텐트 모델(content model)이 서로 상이한 경우 발생한다. 또한 구조적 충돌은 엘리먼트 대 엘리먼트, 엘리먼트 대 애트리뷰트, 애트리뷰트 대 애트리뷰트 사이의 충돌로 분류할 수 있다.

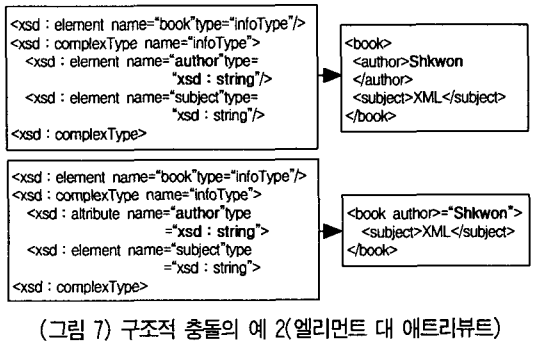
첫째로 엘리먼트와 엘리먼트 사이의 구조적 충돌의 예로서, (그림 6)에서 한 쪽의 스키마는 책의 저자 이름(<author>)을 이름(<first_name>)과 성(<last_ name>)의 두 엘리먼트로 나타내었지만, 다른 한 쪽에서는 이를 성과 이름을 합친 형식으로 나타내어 이들의 통합 시에 구조적 충돌이 일어나게 된다. (그림 6)의 경우와 다른

예를 들면, 엘리먼트 A를 나타내는 컨텐트 모델로써 한 쪽의 스키마에서는 B, C를 가지고 다른 한 쪽의 스키마에서는 B, D를 가지는 경우에 있어서도 통합 시 구조적 충돌을 일으키게 된다. 또한, 한 쪽은 B, C, D를 다른 한 쪽은 B, C를 A를 나타내는 컨텐트 모델로 가질 경우도 구조적 충돌의 예가 될 수 있다.



(그림 6) 구조적 충돌의 예 1(엘리먼트 대 엘리먼트)

둘째, 엘리먼트와 애트리뷰트 사이의 구조적 충돌은 (그림 7)과 같이, 한 쪽에서는 엘리먼트로 나타낸 것을 다른 한 쪽에서는 애트리뷰트로 나타낸 경우 발생한다.



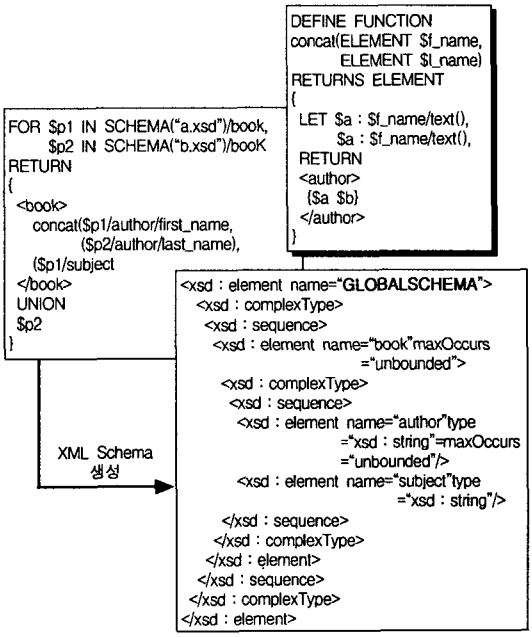
(그림 7) 구조적 충돌의 예 2(엘리먼트 대 애트리뷰트)

마지막으로, 애트리뷰트들 간의 구조적 충돌은 한 쪽 스키마에서는 하나의 애트리뷰트로 나타낸 정보를 다른 쪽에서는 애트리뷰트 그룹(attribute group)으로 나타낸 경우 등을 포함하여 애트리뷰트 사이의 구조적인 이질성에 의해 발생한다.

[해결 방안 2] 구조적 충돌 해결

엘리먼트나 애트리뷰트가 가지는 구조적 충돌은 해당 구조를 변경하여 두 스키마 사이의 구조를 동일하게 하는 방법을 통하여 통합을 수행한다. 이를 위해서 각 요소들의 연결 또는 애트리뷰트의 추가, 통합 연산, XQuery SD의 FUNCTION을 이용한다.

이러한 스키마 사이의 구조는 두 스키마의 논리적 관계성 측면에서 살펴볼 때, 구조가 서로 상이한 경우, 포함 관계를 가지는 경우, 그리고 부분적으로 일치하는 경우가 있을 수 있다. 논리적으로 동등한 경우엔 구조적 충돌을 발생하지 않으며 단지 이름 충돌만을 발생할 수 있다.



(그림 8) 구조적 충돌의 해결 예

(그림 8)은 한쪽의 스키마에선 저자명을 <author> 엘리먼트의 자식 엘리먼트인 <first_name>

과 <last_name>으로 성과 이름으로 나누어서 나타냈지만, 다른 한 쪽에서는 성과 이름을 붙여서 <author> 엘리먼트 자체에 나타내어 발생하는 구조적 충돌을 해결하는 예제이다. 이를 해결하기 위해서는 사용자 정의 함수인 concat 함수를 사용하여 두 엘리먼트를 입력받아 그 두 엘리먼트의 내용을 합친 결과에 <author>라는 태그를 붙여 반환하여 성과 이름으로 나뉘어서 표현되었던 내용을 합쳐서 충돌을 해결한다.

4.1.3 데이터 타입 충돌

데이터 타입 충돌(data type conflict)은 엘리먼트나 애트리뷰트의 데이터 타입이 일치하지 않는 경우에 발생하게 되며, DTD를 이용한 데이터 통합에서는 쉽게 해결되는 문제였지만 XML Schema의 경우에는 XML Schema가 기존 데이터 타입의 상속과 확장을 지원함으로써 부각되는 충돌이다. XML Schema에서는 내장하고 있는 기본형(primitive) 데이터 타입과, 이것에 제약 또는 확장을 통하여 새로운 타입을 정의하는 유도형(derived) 데이터 타입을 지원한다. 그러므로 데이터 타입 충돌은 기본형 대 기본형, 기본형 대 유도형, 유도형 대 유도형 사이의 충돌로 분류될 수 있다.

(그림 9)는 기본형 데이터 타입간의 데이터 타입 충돌을 나타내는 예제로, 두 스키마 모두 책의 정보를 나타내지만, 책의 가격을 나타내는 <price> 엘리먼트의 데이터 타입은 정수형(int)과 문자열(string) 타입으로 각각 달라 이런 경우 두 정보를 통합할 때 충돌이 유발된다.

```
<xsd : complexType name="bookInfo">
 <xsd : element name>="price"type
                    ="xsd : int"/>
 <xsd : element name>="title"type
                    ="xsd : string"/>
</xsd : complexType>
```

```
<xsd : complexType name="bookInfo">
 <xsd : element name>="price"type
                    ="xsd : string"/>
 <xsd : element name>="title"type
                    ="xsd : string"/>
</xsd : complexType>
```

(그림 9) 데이터 타입 충돌의 예 1(기본형 대 기본형)

(그림 10)의 경우에는 가격을 나타내는 <price> 엘리먼트의 데이터 타입을 한쪽 스키마에선 정수형 타입에 제약 조건을 가하여 생성한 유도형 데이터 타입으로 정의하고 있지만, 다른 한 쪽의 스키마에선 기본형 데이터 타입으로 정의하고 있다.

```
<xsd : element name>="price"
 <xsd : simple Type>
  <xsd : restriction base="xsd : int">
   <xsd : minInclusive value="0"/>
   <xsd : maxInclusive value=
                    "100000"/>
  </xsd : restriction>
 </xsd : simple Type>
</xsd : element>
```

```
<xsd : element name="price" type
                    ="xsd : positiveInteger">
```

(그림 10) 데이터 타입 충돌의 예 2(기본형 대 유도형)

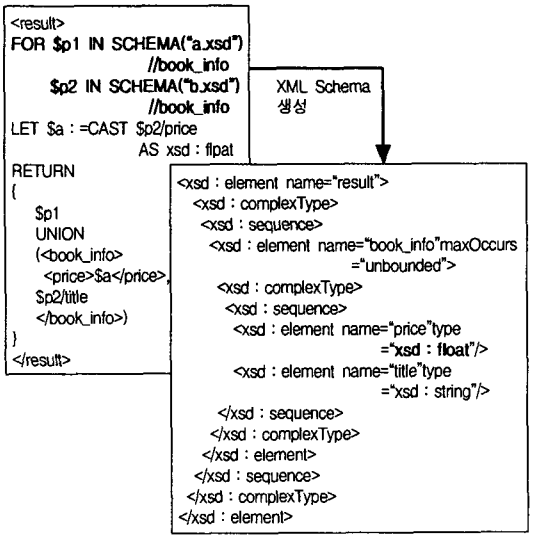
유도형 데이터 타입의 정의는 사용된 패싯(facet)의 종류와 그 값, 그리고 유도형 데이터 타입의 기본 타입의 종류에 따라서 다양한 형태의 타입이 파생될 수 있기 때문에, 유도형 데이터 타입 사이의 충돌은 다양하고 복잡하게 발생하게 된다. 이러한 충돌의 경우엔 유도된 기본 타입이 동일하거나 그렇지 않은 경우로 분류할 수 있으며, 기본 타입이 동일하지 않은 경우에는 기본 타입에서의 이질성뿐만 아니라 패싯의 종류와 그 값에 대한 이질성까지도 고려해야 하므로 더욱 복잡하다.

[해결 방안 3] 데이터 타입 충돌 해결

데이터 타입 충돌은 충돌을 발생하고 있는 데이터 타입들 중에서 한 쪽 데이터 타입을 다른 한 쪽과 동일한 타입으로 타입 변환(type casting)함으로써 해결되거나, 양 쪽 데이터 타입 모두를 동일한 상위 데이터 타입으로 변환시켜 해결된다.

다른 한 쪽의 데이터 타입으로 변환시키는 방법은 XQuerySD가 제공하는 타입 변환 연산인 CAST AS를 이용하여 수행한다. (그림 11)은 충돌을 일으키는 엘리먼트의 데이터 타입을 정

수형으로 타입 변환하여 원래 정수형이었던 엘리먼트와 union 연산을 취하는 예제이다.



(그림 11) 데이터 타입 충돌의 해결 예

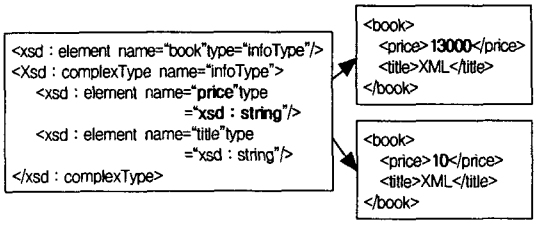
4.2 의미적 충돌

XML Schema는 다양한 데이터 타입과 제약조건을 지원하기 때문에 기존의 DTD를 이용한 통합 방법보다 XML Schema를 이용할 경우에는 보다 많은 이질성들을 명시적으로 확인할 수 있는 방법을 제공한다. 그러나 이러한 스키마에는 앞서 살펴 본 스키마 충돌을 파악할 수 있는 정보만이 제공되어 데이터의 의미적인 차이에 의해 발생되는 충돌은 파악되기 어렵다.

의미적 충돌(semantic conflict)은 이러한 데이터의 의미적인 이질성에 의해 발생하는 충돌이며, 이는 스키마를 통해서 파악될 수 없는 특성을 가진다. 의미적 충돌은 단위 충돌, 측정 레벨 충돌, 표현 충돌 등으로 분류된다.

4.2.1 단위 충돌

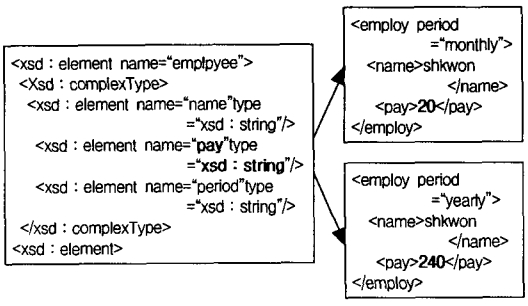
단위 충돌(unit conflict)은 데이터의 단위의 차이에 의해 발생하는 충돌로써 그 단위들 간에는 수학적인 관계성이 존재한다. 예를 들면, (그림 12)에서 <price> 엘리먼트는 책의 가격에 대한 정보를 담고 있는데, 같은 책에 대해서 그 가격 정보는 한 쪽은 13,000, 다른 쪽은 10이라는 다른 값을 나타내고 있다.여기서 각각 13,000은 '원(Won)'으로, 10은 '달러(Dollar)'로 책의 가격을 표현하고 하고 있는 것으로써, 단위 충돌이 발생하고 있다.



(그림 12) 단위 충돌의 예

4.2.2 측정 레벨 충돌

측정 레벨 충돌(granularity conflict)은 측정 단위 간의 관계가 포함 관계를 가지고 있거나, 정확도(precision) 레벨의 차이가 있는 경우에 발생하는 의미적 충돌의 한 형태로써, 단위 레벨 충돌처럼 각 단위 간에 명시적인 수학적 관계성을 가진다. 예를 들면, (그림 13)에서 의미와 타입 등의 스키마 정보가 동일한 <pay> 엘리먼트에 대해서 한쪽은 20이라는 월급의 의미를, 다른 한쪽은 240이라는 연봉의 의미를 나타내고 있을 경우 측정 레벨 충돌이 발생한다.

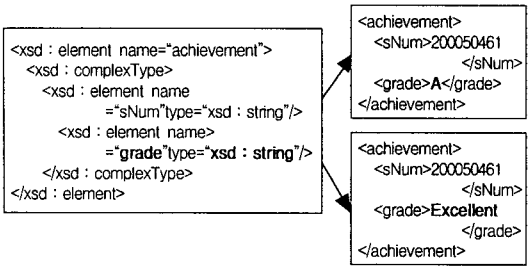


(그림 13) 측정 레벨 충돌의 예

4.2.3 표현 충돌

표현 충돌(representation conflict)은 동일한 데이터에 대하여 서로 다른 방법으로 표현한 경우에 발생하는 의미적 충돌의 한 형태로써, 대부분의 의미적 충돌이 이에 해당된다. 표현 충돌은 단위 충돌이나 측정 레벨 충돌과는 달리 어떤 수학적인 관계성이 존재하지 않아서 통합 시에 시스템이 발견하고 해결할 수 없기 때문에 통합의 중재 규칙을 명세하는 사용자가 충돌을 파악하고 해결해야 하는 충돌이다.

예를 들어 (그림 14)에서 학점을 나타내는 <grade> 엘리먼트의 데이터가 각각 'A'와 'Excellent'로 상이하게 표현되고 있지만 이들은 모두 최고 학점을 나타내는 동일한 의미를 가진다.



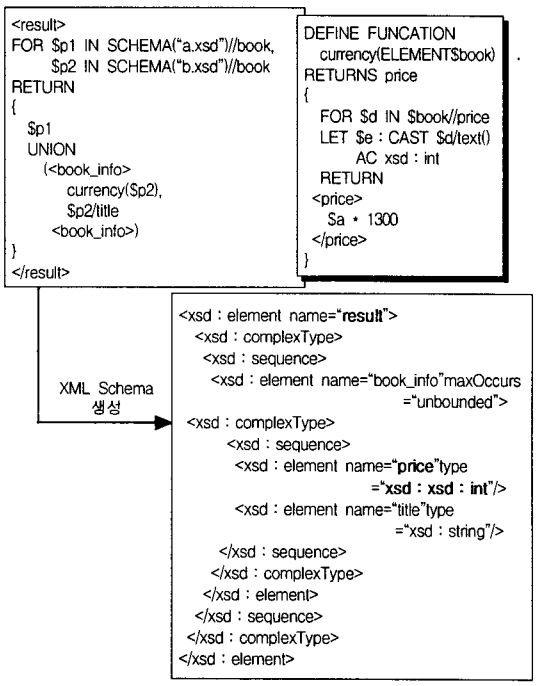
(그림 14) 표현 충돌의 예

[해결 방안 4] 의미적 충돌 해결

의미적 충돌은 XQuerySD에서의 함수 정의를 통한 의미적인 일치나 대응 관계 기술을 통하여 해결한다.

일반적으로 의미적 충돌은 스키마를 통해서는 파악할 수 없는 데이터 간의 의미적인 차이에 의해 발생하기 때문에, 통합 스키마 선언에서는 이러한 의미적 차이에 대한 중재 규칙을 기술하기 어렵다. 따라서 이러한 의미적 충돌 중에 표현 충돌과 같은 수학적 관계성이 없는 충돌에 대해서는 동일한 의미를 가지는 데이터들에 대한 대응 관계를 인위적으로 지정해야 하고, 수학적 관계성을 가지는 충돌들에 대해서는 그것을 해결하기 위한 수식 연산 등을 지원하기 위해 함수를 정의한다.

(그림 15)에서는 함수 정의를 통하여 의미적 충돌의 한 종류인 단위 충돌을 해결하는 예를 보이고 있다. 두 스키마 사이의 통화 단위가 다르기 때문에 이를 통합하기 위해서 currency라는 함수를 통해서 인자로 넘어온 엘리먼트에서 가격 정보(달러)를 정수형으로 변환하고 이것에 1,300을 곱하여 원(Won)으로 환원하였다.

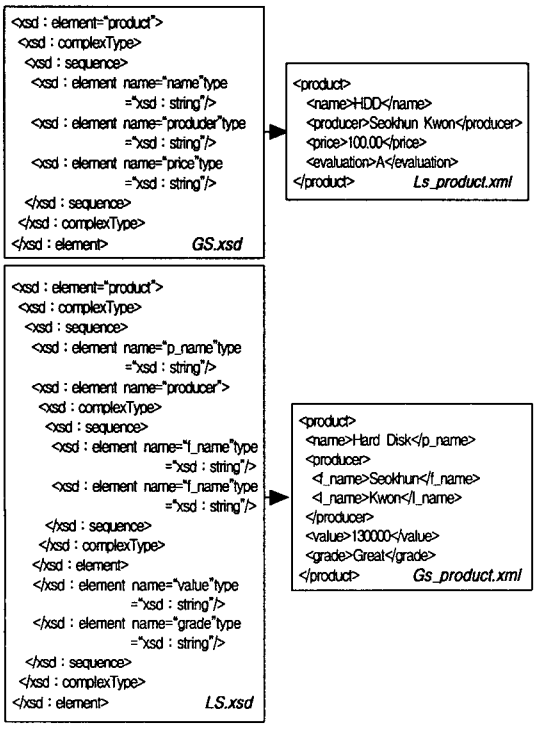


(그림 15) 의미적 충돌(단위 충돌)의 해결 예

4.3 복합 충돌에 대한 해결

앞서 살펴본 충돌 해결 방안은 모두 단일한 충돌에 대한 해결 방안이다. 그러나 (그림 16)와 같이 충돌이 발생할 땐 두 개 이상의 충돌이 해당 엘리먼트에 동시에 발생하는 경우가 있을 수 있다. 이런 경우를 **복합 충돌**(complex conflicts)

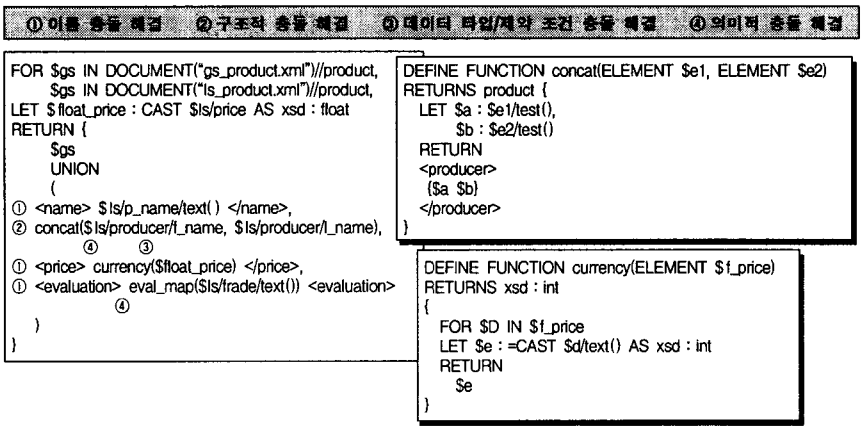
라 정의하였으며, 이 경우엔 XQuery를 이용한 통합 스키마를 정의할 때 일련의 순서를 적용하여야 정확하고 효율적인 뷰 정의가 가능하게 된다.



(그림 16) 복합 충돌의 예

(그림 17)은 (그림 16)에서 발생하는 복합 충돌을 해결하기 위해 XQuerySD로 정의한 통합 스키마이다. FOR 절에서는 충돌의 대상에 대하여 설정하고, LET 절에서는 데이터 타입 충돌을 해결하기 위한 타입 변환이나 함수 대신 간단한 산술 연산(평균, 최소값, 최대값 등)에 관련된 내용을 기술해 줄 수 있다. 그리고 위 예제에는 나타나 있지 않지만 조건을 기술할 필요가 있을 경우엔 WHERE 절에 이를 기술할 수 있다. 마지막으로 RETURN 절에서 충돌을 일으키는 요소들에 대한 해결을 위해 통합 스키마를 정의한다.

이러한 저작 순서로 이루어질 수 있는 통합 스키마 정의에서 가장 중요하다 할 수 있는 RETURN 절에서의 정의는 아래 제시할 복합 충돌의 의미적 해결 순서 정의를 참조하여 만들 수 있다. 예를 들어, 이름 충돌과 구조적 충돌을 발생시키는 두 엘리먼트가 있을 때, 이름 충돌에 대한 해결이 선행되어 두 엘리먼트의 이름이 동일하게 맞춰진 후에 구조적 충돌을 해결해야 한다. 구조적 충돌은 엘리먼트의 이름이 동일한 것들에 대해 정의된 충돌이기 때문에 이름이 다른 엘리먼트들에 대해선 그 충돌 여부를 파악할 수 없기 때문이다. 위와 같은 이유로 XQuery SD의 RETURN 절에서의 충돌 해결은 통합 스키마 작성자가 일관적으로 참조할 수 있는 의미



(그림 17) 복합 충돌을 해결하기 위한 XQuery 뷰 정의

적인 순서가 요구되며 이러한 순서를 감안하여 XQuerySD의 RETURN 절을 정의해야 더욱 정확한 뷰를 보다 쉽게 정의할 수 있다. 본 논문에서 정의한 복합 충돌에 대한 의미적 해결 순서는 다음과 같으며, 각 충돌 유형의 해결 방안은 <표 3>과 동일하다.

1. 스키마 충돌 해결
   1) 이름 충돌의 해결
   2) 구조적 충돌의 해결
   3) 데이터 타입/제약 조건의 충돌 해결
2. 의미적 충돌 해결
   1) 단위/측정 레벨 충돌의 해결
   2) 표현 충돌의 해결

〈표 3〉 충돌 해결 방안의 분류

충돌 유형		해결 방안
스키마 충돌	이름 충돌	엘리먼트 또는 애트리뷰트 이름의 변경
	구조적 충돌	엘리먼트/애트리뷰트의 조합
	데이터 타입 충돌	한 쪽 데이터 타입의 변환, 또는 상위 데이터 타입으로의 변환
	제약 조건 충돌	제약 조건의 제거 또는 한쪽 제약 조건으로의 일치
의미적 충돌	단위 충돌	함수를 이용한 표현 방법의 일치
	측정 레벨 충돌	
	표현 충돌	

5. 통합 스키마 저작 도구

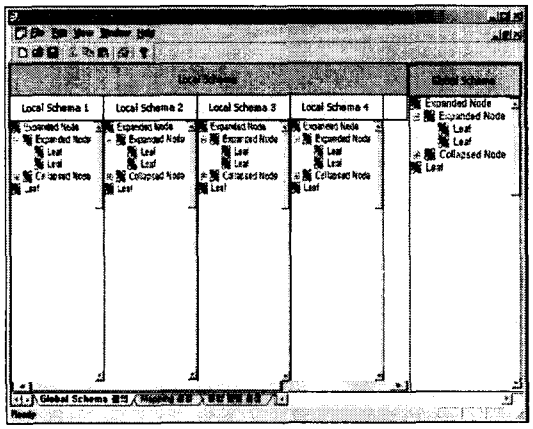
앞에서 살펴 본 충돌에 대한 해결 방안을 참조하여 이질 정보 자원 통합을 수행하게 되는데, 이 때 충돌에 대하여 통합 스키마를 생성해 내기 위해서 사용자는 XQuerySD의 문법을 이해하고 있어야 하며 XQuerySD의 기술(description)에 있어서도 전문성이 요구된다. 또한 정보 자원들 간의 대응 관계를 텍스트 기반으로 사용자가 직접 기술하는 경우에 효율성이 떨어지게 되며 정확성 측면에서도 확실한 보장이 이루어질 수 없다. 이러한 문제점 때문에 통합 스키마를 생성하기 위해 저작자가 용이하게 사용할 수 있는 저작 도구(authoring tool)의 개발이 요구된다. 이러한 저작 도구는 사용자의 편의를 위해 GUI(graphical user interface) 환경으로 개발되어야 하며, 사용자에게 정보자원간의 대응 관계 기술에 대한 기반 언어인 XQuerySD에 대한 많은 이해를 요구해선 안 된다. 또한 자주 요구되는 함수라 볼 수 있는 값 변환 함수나 데이터 타입 변환 함수 등은 컴포넌트 화하여 미리 제공하여야 하며, 표현 충돌에 대해서 대응 관계를 정의할 수 있는 기능도 제공되어야 한다.

5.1 통합 스키마 저작 도구

본 논문에서 개발한 통합 스키마 저작 도구의 주요 기능은 다음과 같다.

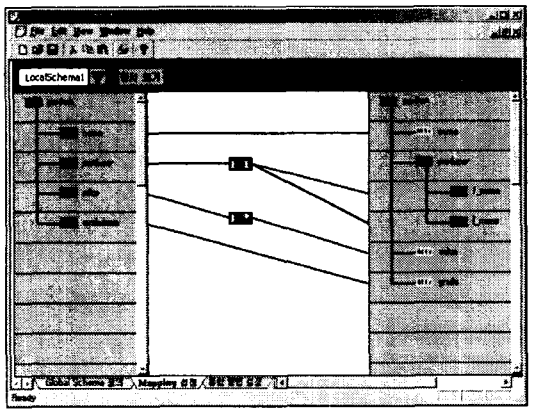
- 지역 스키마(local schema)의 적재 : XML Schema 파일로 정의된 각 지역 스키마를 적재하여 트리 형태로 사용자에게 보인다.
- 통합 스키마(global schema)의 정의 : 사용자는 트리 형태로 작성된 지역 스키마들을 보고 통합 스키마를 트리 형태로 직접 정의한다.
- 지역 스키마와 통합 스키마의 노드 간 중재 규칙 정의 : 통합 시 연관되는 지역 스키마와 통합 스키마의 노드간 매핑을 각 노드간 연결선을 통해 설정한다.
- 다양한 통합 연산의 설정 : 각 스키마들 간의 다양한 통합 방법을 제공한다.
- 사용자 정의 함수의 정의 : 의미적 충돌의 해결을 위한 기본 함수 라이브러리의 제공과 함께 기본 함수를 이용한 사용자 정의 함수의 작성을 제공한다.

이 개발 도구의 사용자 인터페이스는 3개의 작업화면(통합 스키마 정의, 매핑 설정, 통합 방법 설정)으로 구성된다. 통합 스키마 정의는 (그림 18)에서 보이는 바와 같이 각 지역 스키마들을 적재하여 트리 형태로 보여주고 적재된 지역 스키마들을 참조하여 사용자가 직접 통합 스키마를 구성할 수 있도록 한다. 트리 형태로 구성된 통합 스키마는 내부적으로 XQuerySD의 RETURN 절로 변환된다.



(그림 18) 통합 스키마 정의 화면

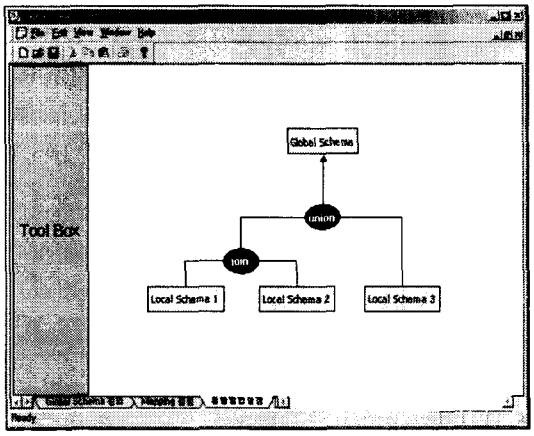
통합 스키마를 정의한 후, 매핑 설정화면에서 각 지역 스키마와 통합 스키마 간의 노드들을 (그림 19)와 같이 매핑시킨다. 지역 스키마와 통합



(그림 19) 노드간 매핑 설정 화면

스키마간의 매핑 설정은 양단의 노드들에 대한 연결선을 그려줌으로써 이루어지고 사용자 정의 함수가 필요한 경우 사용자 정의 함수를 연결선상에 끌어 놓음으로써 설정해 줄 수 있다.

위의 두 작업을 마치면, 통합 방법 설정 화면에서 각 지역 스키마들을 어떠한 통합 연산[17]으로 통합할 것인지를 결정해야 한다. (그림 20)에서 보는 바와 같이 통합 방법 설정 화면의 왼쪽에 도구 상자가 존재하는데, 이 도구 상자는 통합 방법 설정 시 필요한 도구들을 제공한다. 예를 들어, 지역 스키마들을 JOIN으로 통합하기 위한 연결선 같은 도구들을 제공한다. 통합 방법 설정 화면을 분리시킴으로써, 좀 더 복잡한 통합 방법의 설정을 가능하게 한다.



(그림 20) 통합 방법 설정 화면

6. 결 론

정보 자원에 대한 통합은 인터넷상에 분산되어 있는 정보 자원들을 통합함으로써 새로운 정보의 추출 가능, 인터넷 정보 자원의 통합에 대한 요구 사항 증대, 그리고 이미 구축되어 있는 정보 자원들의 재사용 등에 있어서 매우 중요하다. 하지만 각 분산되어 있는 정보 자원들이 가

지는 특성들에 의해서 여러 가지 이질성이 발생하기 때문에, 이러한 정보 자원에 대한 효과적이고 정확한 통합 작업을 수행하기는 매우 어렵다. 본 논문에서는 정보 자원들의 특성들을 고려하여 이질 정보 통합의 환경을 웹 환경에서의 XML 기반 미디에이터 방식을 택하였고, XML로 표현되는 정보 자원들의 스키마로써 XML Schema를 사용하였다.

XML Schema는 기존의 DTD에 비해 많은 데이터 타입과 제약 사항, 다양한 구조의 엘리먼트와 애트리뷰트 선언을 지원함으로써 보다 풍부한 XML 인스턴스에 대한 스키마 정의를 가능하게 하였다. 그러나 XML Schema에서의 이러한 기능들은 XML Schema를 스키마 정보로 활용하는 정보 통합에서 기존의 DTD를 스키마 정보로 활용할 때 보다 더욱 많은 충돌 문제를 발생시키게 되었다. 본 논문에서는 이러한 XML Schema에서의 충돌 문제를 스키마 정의를 통하여 명시적으로 파악할 수 있는 스키마 충돌과 의미적인 이질성에 의해 발생하여 스키마 정의에서는 파악할 수 없는 의미적 충돌로 분류하였고, 충돌 대상에 따라 각 충돌 문제를 보다 세분화하여 구분하였다. 그리고 이에 대한 해결 방안으로써, 분류된 각 충돌 문제에 대해 XML 질의 언어인 XQuery를 스키마 정의를 위한 언어로 변형한 XQuerySD를 이용한 통합 스키마 정의를 통한 중재 방법을 제시하였다. 또한, 이질 정보 자원들 사이의 충돌 문제는 여러 개의 충돌이 동시에 발생하는 복합 충돌의 경우가 존재하기 때문에 이를 해결하기 위해 의미적인 순서를 정의하여 통합 스키마를 정의하는 방법을 제안하였다.

현재까지 XML Schema를 기반으로 한 정보 통합 시스템에서 발생할 수 있는 충돌의 분류 및 해결 방안에 대한 연구는 없음에 따라 본 논문에서는 이를 해결하기 위한 방안을 고려하였으며, 개발한 통합 스키마 저작 도구에서 이용할 수 있는 절차적 방법론을 제시하였다. 향후 연구로는 정보 자원 사이의 충돌을 파악하여 스키마 요소 간의 가능한 매핑 관계를 자동으로 찾기 위한 방법과 작성된 통합 스키마 및 중재 규칙에 대한 정확성을 보장할 수 있는 방안의 연구를 필요로 한다.

참 고 문 헌

[1] Ruxandra Domenig, Klaus R. Dittrich, "An Overview and Classification of Mediated Query System", SIGMOD Record, Vol. 8, No. 3, 1999, pp. 63-72.

[2] T. Bray, J. Paoli, and C.Sperberg-McQueen, "XML(eXtensible Markup Language) 1.0", W3C Recommendation, W3C, http://www.w3.org/TR/REC-xml, February 1998.

[3] David C. Fallside, "XML Schema Part 0 : Primer", W3C Recommendation, W3C, http ://www.w3.org/TR/xmlschema-0, 2001.

[4] Won Kim, Injun Choi, Sunit Gala, and Mark Scheevel, "On Resolving Schematic Heterogeneity in Multidatabase Systems", *Modern Database Systems* ; The Object Model, Interoperability, and Beyond, Addison-Wesley Publishing Company, 1995, pp. 521-550.

[5] Don Chamberlin., et al., "XQuery 1.0 : An XML Query Language", W3C Working Draft, W3C, http://www.w3.org/TR/XQuery, 2001.

[6] Laks V. S. Lakshmanan. Fereidoon Sadri

and Iyer N. Subramanian, "SchemaSQL : A Language for Interoperability in Relational Multi-database Systems", *Proceedings of the 22nd VLDB Conference*, 1996.

[7] Kai-Uwe Sattler, Stefan Conrad and Gunter Saake, "Adding Conflict Resolution Features to a Query Language for Database Federations", *Proc. 3nd Int.* Workshop on Engineering Federated Information Systems, EFIS'00, 2000.

[8] Yannis Papakonstantinou, Serge Abiteboul and Hector Garcia-Molina, "Object Fusion in Mediator Systems", *Proceedings of International Conference on VLDB*, 1996, pp. 413-424.

[9] Vassilis Christophides, Sophie Cluet and Jerome Simeon, "On Wrapping Query Languages and Efficient XML Integration", http://www-db.research. bell-labs. com/user/simeon/yamg_full.ps, 1999.

[10] Bertram Ludscher, Yannis Papakonstantinou and Pavel Velikhov, "A Framework for Navigation-Driven Lazy Mediators", *Proceeding of International Workshop on the Web and Databases*, 1999, pp. 85-90.

[11] 이강찬, 이경하, 이규철, "XML 기반의 인터넷 정보 자원 통합", *데이터베이스 연구*, 한국데이타베이스연구회, 제16권 2호, 2000년, 12월, pp. 5-21.

[12] 이경하, 조정수, 이강찬, 이규철, "XMF : XML 기반 분산 이질 정보 자원의 통합 프레임워크", 한국데이타베이스연구회 2000 학술 발표 논문집, 2000년, 5월, pp. 262-270.

[13] Mi-Young Lee, Myung-Joon Kim, Kyu-Chul Lee, "Design of an XML-base Database Integration Middleware : Data Blender", *Proceedings of ICIS 2002*, 2002년, 8월, pp. 863-867.

[14] Tompson, H., et al., "XML Schema Part 1 : Structure", W3C Recommendation, W3C, http://www.w3.org/TR/xmlschema-1, 2001.

[15] Paul V. Biron., et al., "XML Schema Part 2 : Datatypes", W3C Recommendation, W3C, http://www.w3.org/TR/XMLSchema-2, 2001.

[16] 김미혜, 이승원, 이경하, 이규철, 김병섭, 이미영, "이질 자료 모델 통합을 위한 통합 스키마 저작 도구 설계", *한국정보과학회 봄 학술발표논문집(B)*, 2002년, 4월, pp. 19-21.

[17] J. Shanmugasundaram, J. Kiernan, E. Shekita, C. Fan, J. Funderburk, "Querying XML views of relational data", Proceedings of the VLDB Conference, 2001.

■ 저자소개

권 석 훈

충남대학교 컴퓨터공학교육과 학사, 충남대학교 대학원 컴퓨터공학 석사. 현재 (주)우린정보 기술 연구소에서 연구원으로 재직 중

이 경 하

충남대학교 정보통신공학과 학사, 충남대학교 대학원 정보통신공학과 석사, 충남대학교 대학원 컴퓨터공학과 박사 수료, 현재 동 대학원에서 전문연구요원으로 활동 중이다. 주요 연구분야는 데이터베이스, XML, e-Business 등 이다.

이 규 철

서울대학교 컴퓨터공학과에서 학사, 석사 및 박사. 현재 충남대학교 컴퓨터공학과 교수로 재직 중이며 한국 ebXML 전문위원회 위원장, 전자상거래 표준화 통합 포럼 전자거래 기반 기술위원회 부위원장, 한국정보과학회 논문 편집위원으로 활동 중이다. 주요 연구분야로는 데이터베이스, XML, 정보 통합, 멀티미디어 시스템 등 이다.