XBRL 택사노미의 효과적인 재활용을 위한 컴포넌트 기반 전자문서 개발에 관한 연구

김 형 도* · 박 찬 권**

Research on a Component-Based Method for Developing Electronic Documents for the Effective Reuse of XBRL Taxonomies

Hyoung Do Kim* · Chankwon Park**

Abstract

As XBRL applications for sharing the contents and semantics of business reports get activated, it is increasingly necessary to reuse components systematically by the standardization in the national level. The current standardization of Korean electronic documents follows the "Guideline for XML Documents Development", which aims to maximize the reusability of components by providing a business library as well as the UN/CEFACT core components methodology. XBRL takes the same approach with the standardization of electronic documents in that their components (concepts and their structures) are defined using XSD. In XBRL, however, concept are restricted to specific types and diverse relationship types can be defined for associating the concepts to themselves and other additional information. As a result, there are some issues that may not be solved by just applying the guideline to XBRL. This paper presents a basic method for applying the methodology to XBRL applications in order to realize the systematic reusing and standardization of XBRL documents in the national level. For case analysis, FSS (Financial Supervisory Service) income statement is employed to demonstrate the possibility of the method.

Keywords : XBRL, Core Component, XML, Electronic Document, B2B e-Business

논문접수일 : 2008년 06월 18일 논문게재확정일 : 2008년 12월 19일

* 한양사이버대학교 경영학부 교수

** 교신저자, 한양사이버대학교 경영학부 교수, Tel : 02-2290-2813, e-mail : chankwon@hycu.ac.kr

1. 서 론

비즈니스 보고의 내용과 의미를 공유할 수 있는 XBRL(XML Business Reporting Language) 응용이 활성화되면서, 국가적인 수준에서의 표준화와 함께 체계적으로 구성요소들을 재활용해야 할 필요성이 증대되고 있다. XBRL은 기업 보고 문서의 구성요소들을 XSD(XML Schema Definition)로 정의한다는 점에서 기존의 전자문서 표준과 기본적으로 동일하다. 그러나 개념들의 유형이 특정하게 한정되어 있고, 이들 개념들을 연결하고 부가 정보를 제공하기 위한 다양한 관계 유형을 제공한다는 점에서, 기존의 전자문서 표준과는 근본적인 차이가 있다. XSD로 정의된 하나의 복잡한 수형 구조체로 전자문서를 규정하는 것이 기존의 접근 방법이었다고 한다면, XBRL에서는 개념들을 평면적으로 나열한 뒤, 이들을 다차원적으로 연결하는 방법을 채택하고 있다.

XBRL에서의 재활용은 기본적으로 XML 이름공간[16, 17]을 이용한 점증적 확장 메커니즘을 통해서 이루어지고 있다[18, 19]. 이름공간에 의한 재사용은 파일 기반의 재사용보다는 발전된 것이지만, 전자문서의 구성요소를 개념적이고 논리적인 수준에서 등록하고 체계적으로 이용하는 것보다는, 재활용도가 낮고, 유지관리하기에 어려움이 많다.

국가적 표준으로 전자문서를 제정하고 관리하는 작업도 활성화 단계에 있는 XBRL에 있어서 매우 중요하다. 현재 우리나라의 표준 전자문서 개발은 "XML 전자문서 개발 지침"[7]을 준수하여 이루어지고 있다. 이 지침에서는 전자문서를 구성하는 데이터 항목을 컴포넌트로 만들어 등록/관리하고, 이를 전자문서 개발 시 재활용할 수 있는 핵심 컴포넌트 방법론과 함께 라이브러리(http://www.remko.or.kr)를 제공하고 있다[5, 7].

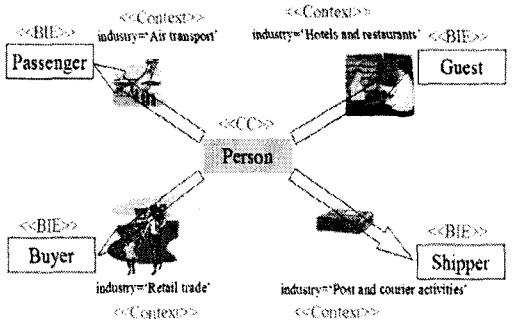
핵심 컴포넌트 방법론은 비즈니스와 관련된 의미를 가지는 요소와 그렇지 않은 요소로 구분해서 자원을 체계적으로 재활용할 수 있도록 지원한다. 여기서 후자는 전자문서를 정의하기 위한 온톨로지 역할을 하는 것으로 특정한 문맥(Context) 하에서 전자와 연결된다. 또한 문법 중립적인 측면을 강조해서, 개념적이고 논리적인 수준에서 표준 문서에 대한 정의가 이루어지도록 하며, 이를 바탕으로 XML 등의 다양한 문법을 적용할 수 있도록 지원한다. 이러한 방법론과 라이브러리를 적용하여 전자문서를 개발하고 활용[6]하기 위해서는, 각 산업별 또는 업종별로 진행되는 표준 전자문서 개발 과정에서부터 기존 자원에 대한 재활용 여부를 충분히 검토해야 한다. 이러한 기존 자원의 재활용은 중복된 자원의 개발을 예방하고, 표준 전자문서 개발 시간을 크게 단축시키게 된다.

본 연구에서는 XBRL 문서의 체계적인 재활용과 국가 표준화가 가능하도록, UN/CEFACT의 핵심 컴포넌트 방법론을 적용하기 위한 기본적인 방법을 제시하고, 그 가능성을 사례를 통해서 분석해보고자 한다. 이 논문은 다음과 같이 구성되어 있다. 제 2장에서는 UN/CEFACT의 핵심 컴포넌트 방법론에서 사용되는 주요 개념과 재활용 방법에 대하여 정리한다. 제 3장에서는 핵심 컴포넌트 방법론 기반의 XBRL 전자문서 개발 방법을 제시한다. 이를 실현하기 위하여 XBRL을 분석하여 핵심 컴포넌트와 비즈니스 정보 개체를 찾아내고, 이들 중에서 비즈니스 라이브러리에 신규로 등록해야 할 요소들을 결정하였다. 이러한 역공학적 방법으로 XBRL을 재구성하여 핵심 컴포넌트 방법론을 적용할 수 있도록 하되, 최종적으로는 XBRL 문법을 준수하는 스키마를 생성할 수 있도록 변환규칙을 제공한다. 제 4장에서는 손익계산서 사례에 대하여

이 방법을 적용해 보고, 그 가능성을 검토한다. 마지막으로 제 5장에서는 논문을 정리하고, 앞으로의 연구방향을 제시한다.

2. 핵심 컴포넌트 방법론

비즈니스 데이터 교환에 있어서 보다 유연한 상호 작용을 지원하기 위해서는 비즈니스 어의(Business Semantics)에 관한 표준화가 선행되어야 한다. UN/CEFACT의 핵심 컴포넌트(CC : Core Component) 방법론은 문서 구조에 대한 정의를 쉽게 할 수 있고, 과거의 정의를 여러 가지 방법으로 재활용할 수 있도록 핵심 컴포넌트(CC), 비즈니스 정보 개체(BIE : Business Information Entity), 비즈니스 문맥(Business Context) 등의 개념을 제공한다[12]. <그림 1>의 사례에서 보면, 항공업종에서는 Passenger, 소매업종에서는 Buyer, 운송업종에서는 Shipper, 호텔업종에서는 Guest라는 것이, 실제로는 동일하게 Person을 지칭하는 것을 표현하고 있다. 재활용을 극대화하기 위해서 모든 업계에서 Person을 사용하는 것으로 강제할 수도 있으나, 핵심 컴포넌트 방법론에서는 업계의 관행을 존중하면서도 재사용을 증대시키기 위한 방안을 제시하고 있다. 업종에 무관한 것을 핵심 컴포넌트로 부르고, 이것을 업종에 따라서 Passenger와 같이 비즈니스 정보 개체로서 재활용하게 되는 것이다. 어떤 업종이냐 하는 것은 비즈니스 문맥에 해당되는데, 이러한 문맥이라는 개념을 단순하게 업종에만 한정하는 것이 아니라, 비즈니스 프로세스, 제품, 지역이나 국가 등으로 확대하면 핵심 컴포넌트의 재활용을 좀 더 높일 수 있게 된다. 업종, 비즈니스 프로세스, 제품, 지역, 국가 등을 문맥 카테고리(Context Category)라고 부르며, 구체적인 값이 설정된 것을 문맥이라고 한다.



〈그림 1〉 문서 구성요소의 재활용을 위한 주요 개념

핵심 컴포넌트는 UML[9, 10] 클래스 다이어그램에 기반을 두고 있기 때문에, 사실상 UML 객체 클래스에 대한 정보의 일반 표현에 해당된다. 하나의 클래스 다이어그램은 객체 클래스들, 특정 클래스를 수식하는 프로퍼티들, 그리고 클래스들 간의 관계를 보여준다[13]. 이와 마찬가지로, 핵심 컴포넌트도 객체 클래스를 나타내는 집합 핵심 컴포넌트(ACC : Aggregate CC), 객체 클래스의 단순 프로퍼티를 나타내는 기본 핵심 컴포넌트(BCC : Basic CC), 하나의 객체 클래스가 또 다른 객체 클래스의(복잡한) 프로퍼티가 되는 경우 두 객체 클래스 사이의 관계를 나타내는 연관 핵심 컴포넌트(ASCC : Association CC)가 있다. 여기에 추가적으로 텍스트(Text), 숫자(Number), 날짜(Date)처럼 기본 핵심 컴포넌트가 가질 수 있는 정보의 유형을 정의하는 핵심 컴포넌트 타입(CCT : CC Type)이 있다. 이렇게 핵심 컴포넌트는 네 가지 종류로 분류될 수 있다. 각각의 핵심 컴포넌트에는 유일한 명칭과 함께 그 의미가 정의되며, UID(Universal Identifier)나 다수의 비즈니스 용어 혹은 동의어들도 규정된다.

비즈니스 협업에서 실제로 교환되는 정보는 핵심 컴포넌트로 정의되는 것이 아니고, 비즈니스 문맥을 반영한 비즈니스 정보 개체(BIE)로 정의된다. 일단 비즈니스 문맥이 파악되면, 해당

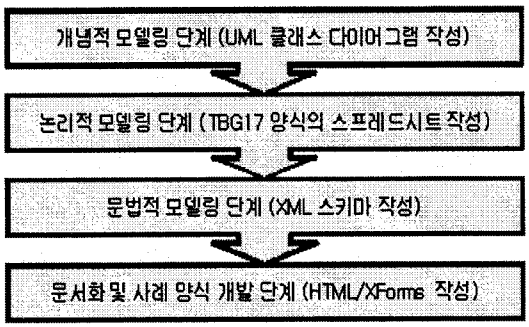
비즈니스 문맥에서의 핵심 컴포넌트 사용에 필요한 한정과 정제를 고려하기 위하여 핵심 컴포넌트를 구별할 수 있다. 비즈니스 정보개체는 핵심 컴포넌트를 특정 비즈니스 문맥에서 사용한 결과이다. 따라서 비즈니스 정보 개체는 각각의 핵심 컴포넌트 분류에 대해서 상응하는 방식으로 분류될 수 있다. 그 결과 ACC에 상응하는 집합 비즈니스 정보개체(ABIE : Aggregate BIE), ASCC에 상응하는 연관 비즈니스 정보 개체(ASBIE : Association BIE), 그리고 BCC에 상응하는 기본 비즈니스 정보 개체(BBIE : Basic BIE)로 분류된다. 하나의 비즈니스 정보 개체는 핵심 컴포넌트 이름에 '한정어(Qualifier)'를 추가함으로써 상응하는 핵심 컴포넌트와 구별될 수 있다.

3. 핵심 컴포넌트 방법론 기반의 XBRL 문서 개발 방법

3.1 XBRL 개발 프레임워크

핵심 컴포넌트 방법론은 주어진 비즈니스 상황이나 환경 즉, 비즈니스 문맥과 독립적인 표준화된 데이터 요소를 핵심 컴포넌트로 정의하고, 이를 핵심 컴포넌트 라이브러리에 저장한 다음 특정 비즈니스 문맥에 맞게 재사용함으로써 XML 전자문서를 만들어낼 수 있도록 하는 방법이다[12]. 이를 위해서는 핵심 컴포넌트와 비즈니스 정보 개체를 CC 라이브러리에 등록하고, 필요에 따라 이를 재사용함으로써 전자문서를 개발하는 체계적인 절차가 요구된다[6, 12].

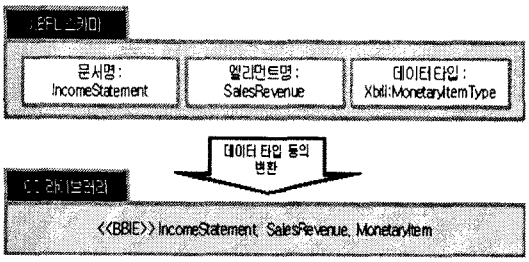
XML 전자문서 개발 지침[7]에서 채택하고 있는 OASIS UBL(Universal Business Language)[8] 기술위원회의 전자문서 개발 절차에서는 특정 전자문서에 포함될 비즈니스 정보 개체들에 대한 UML 클래스 다이어그램을 먼저 작성하고, 이를 기반으로 UN/CEFACT TBG17의 제출 절차 및 포맷[13, 14]을 준용한 스프레드시트를 작성하여, XML 명명 및 설계규칙[15]을 준수하는 XML Schema로 자동 변환함으로써 전자문서를 문법적으로 정의하도록 하고 있다. 이후에 새로 작성된 XML Schema의 내용을 구체적인 UML 클래스 다이어그램으로 문서화하여 사용자들이 XML Schema의 내용을 쉽게 이해할 수 있도록 참고자료로 제공한다. 이상과 같은 4단계는 <그림 2>와 같이 개념적 모델링 단계, 논리적 모델링 단계, 문법적 모델링 단계, 문서화 및 사례 양식 개발 단계로 체계화할 수 있다.



<그림 2> 전자문서 개발 절차

한편 XBRL에서는 <그림 2>의 문법적 모델링 단계에 해당하는 이후 과정에 집중하고 있기 때문에 핵심 컴포넌트 방법론을 통해 XBRL 문서 개발을 지원하기 위해서는 개념적이고 논리적인 단계에서 핵심 컴포넌트 방법론을 적용할 수 있도록 해야 한다. 즉, <그림 3>과 같이 XBRL의 택사노미 스키마로부터 CC/BIE에 해당하는 요소를 추출해서 이를 CC 라이브러리에 등록한다. 이를 위해서는 먼저 문법적 모델링 단계에서 작성되는 내용을 XBRL로 작성되는 내용과 비교하는 작업이 필요하다. 이는 XBRL이 XSD를 사용한 문법적 모델링 방법만을 제공하고 있어서 이 두 가지를 쉽게 비교할 수 있

기 때문이다.



〈그림 3〉 XBRL 스키마로부터 CC/BIE 정의

```
1.  <schema targetNamespace="http://www.xbrl.org/2003/instance"
2.          xmlns="http://www.w3.org/2001/XMLSchema"
3.          xmlns:xbrli="http://www.xbrl.org/2003/instance"
4.          xmlns:link="http://www.xbrl.org/2003/linkbase"
5.          elementFormDefault="qualified">
6.    <element name="xbrl">
7.      <annotation>
8.        <documentation>XBRL instance root element.</documentation>
9.      </annotation>
10.     <complexType>
11.       <sequence>
12.         <element ref="link:schemaRef" minOccurs="1" maxOccurs="unbounded" />
13.         <element ref="link:linkbaseRef" minOccurs="0" maxOccurs="unbounded" />
14.         <element ref="link:roleRef" minOccurs="0" maxOccurs="unbounded" />
15.         <element ref="link:arcroleRef" minOccurs="0" maxOccurs="unbounded" />
16.         <choice minOccurs="0" maxOccurs="unbounded">
17.           <element ref="xbrli:item"/>
18.           <element ref="xbrli:tuple"/>
19.           <element ref="xbrli:context"/>
20.           <element ref="xbrli:unit"/>
21.           <element ref="link:footnoteLink"/>
22.         </choice>
23.       </sequence>
24.       <attribute name="id" type="ID" use="optional" />
25.       <anyAttribute namespace="http://www.w3.org/XML/1998/namespace" processContents="lax" />
26.     </complexType>
27.   </element>
28. </schema>
```

〈그림 4〉 XBRL 전자문서의 구조

3.2 XBRL 구성요소와 CC/BIE의 매핑

XBRL 전자문서는 <그림 4>와 같이 xbrl이라고 하는 최상위 엘리먼트 내에 item, tuple, context, unit, footnoteLink가 다수 포함되는 구조를 가지고 있다. 기업보고를 위한 구체적인 값을 포함할 수 있는 item과 다른 개념들을 묶는 역할을 하는 tuple의 경우에는 전자문서를 정의할 때

〈표 1〉 XBRL에서 정의된 item 타입

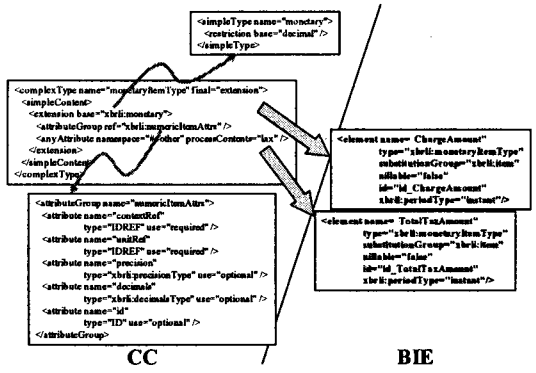
XBRL item 타입	XSD 내장 타입	XBRL item 타입	XSD 내장 타입
decimalItemType	decimal	stringItemType	string
floatItemType	float	booleanItemType	Boolean
doubleItemType	double	hexBinaryItemType	hexBinary
integerItemType	integer	base64BinaryItemType	base64Binary
nonPositiveIntegerItemType	nonPositiveInteger	anyURIItemType	anyURI
negativeIntegerItemType	negativeInteger	QNameItemType	QName
longItemType	long	durationItemType	duration
intItemType	int	dateTimeItemType	dateTime
shortItemType	short	timeItemType	time
byteItemType	byte	dateItemType	date
nonNegativeIntegerItemType	nonNegativeInteger	gYearMonthItemType	gYearMonth
unsignedLongItemType	unsignedLong	gYearItemType	gYear
unsignedIntItemType	unsignedInt	gMonthDayItemType	gMonthDay
unsignedShortItemType	unsignedShort	gDayItemType	gDay
unsignedByteItemType	unsignedByte	gMonthItemType	gMonth
positiveIntegerItemType	positiveInteger	normalizedStringItemType	normalizedString
monetaryItemType	xbrli : monetary	tokenItemType	token
sharesItemType	xbrli : shares	languageItemType	language
pureItemType	xbrli : pure	NameItemType	Name
fractionItemType	정수인 numerator와 0이 아닌 정수인 denominator의 복합체	NCNameItemType	NCName

substitutionGroup을 이용하여 사용자 정의 타입으로 대체하게 되며, 이들 사용자 정의 타입 간에는 관계를 추가적으로 정의할 수 있다. 나머지 3개의 엘리먼트는 사례문서에서 직접 사용되게 된다.

(1) item 타입의 매핑

XBRL의 모든 item 타입은 <표 1>과 같이 사전에 설정된 것들과 이들 중 하나로부터 파생된 것이어야 한다. 대부분이 XSD의 기본 타입을 이용한 것이지만, monetaryItemType, sharesItemType, pureItemType, fractionItemType 등 4가지는 XBRL 자체적으로 정의하고 있다.

예컨대, <그림 5>와 같이 XBRL에서는 monetary라고 하는 단순 타입을 확장한 monetaryItemType을 이용해서 화폐와 관련된 개념들을 표현할 수 있도록 하고 있는데, 이것은 핵심 컴포넌트 방법론에서의 데이터 타입 역할을 하게 된다. 다만 유의할 점은 핵심 컴포넌트 방법론에서는 엘리먼트나 타입을 명명할 때 대문자 카멜 케이스(Upper-camel-case)를 사용하여야 한다는 점이다. 이러한 요구사항은 "국내 XML 전자문서 개발 지침[8]"에 규정된 것으로, 정확하게 이를 준수하여 국내 표준문서로 등록하고자 하는 경우에는 엘리먼트나 타입의 이름을 부



<그림 5> monetaryItemType의 재구성

분적으로 수정해야 함을 의미한다. 그러나 이러한 이름의 수정은 기존의 XBRL 표준을 위반하게 되므로 적절한 중재 방안이 마련되어야 한다. tuple의 경우에는 사전에 정의된 타입이 존재하지 않으며, 따라서 각 문서를 정의할 때 필요한 tuple은 ACC 및 ABIE를 이용해서 정의될 수 있다.

실제로 각 item 타입을 분석하여 핵심 컴포넌트 또는 데이터 타입으로 설정하는 과정에서 핵심 컴포넌트 방법론의 규정으로 인한 규칙 위반 문제를 해결해야 했다. 가장 대표적인 것이 대문자 카멜 케이스에 의한 엘리먼트나 타입 명명 규칙이다. QNameItemType, NameItemType, NCNameItemType을 제외한 거의 모든 item 타입을 데이터 타입으로 변경하는 데 있어서 이 규칙을 위반하게 된다. 따라서 이를 방지할 수 있도록 모든 item 타입 이름의 첫 번째 알파벳을 대문자로 변경하여 데이터 타입을 설정하도록 하였다.

전자문서 개발지침[7]에 따르면 비한정 데이터 타입(Unqualified Data Type)은 XSD 내장 데이터 타입을 restriction으로 제약한 simpleType으로 정의되거나(규칙 121, 122), 하나의 simpleContent를 포함하는 complexType으로 규정되어야 한다(규칙 123, 124). XSD의 simpleContent에서는 extension에 의한 속성 확장만이 가능하다. fractionItemType의 경우 두 개의 엘리먼트를 포함하고 있으므로, 데이터 타입으로 변경하는 데 있어서 이러한 요구사항을 만족시키지 못한다. 이러한 규칙 위반을 회피하기 위하여, <그림 6>과 같이 XSD의 내장 데이터 타입인 double을 확장하여 numerator와 denominator이라는 두 개의 속성을 추가하도록 구성하였다. 이 데이터 타입을 이용하여 정의되는 엘리먼트의 값은 numerator 속성값을 denominator 속성값으로 나눈 결과값으로 설정됨을 원칙으로 하

되, 실제 처리시에는 무시해도 된다.

비한정 데이터 타입(UDT)에 관한 규칙 124에 의하면, complexType으로 정의될 경우 반드시 simpleContent를 포함해야 한다[8]. complexType으로 규정되어야 할 대부분의 item 타입들은 XSD 내장 데이터 타입을 기반으로 하므로

〈표 2〉 비한정 데이터 타입 설정

XBRL item 타입	XSD 베이스 타입	CC 데이터 타입
stringItemType	string	udt : StringItemType
booleanItemType	Boolean	udt : BooleanItemType
hexBinaryItemType	hexBinary	udt : HexBinaryItemType
base64BinaryItemType	base64Binary	udt : Base64BinaryItemType
anyURIItemType	anyURI	udt : AnyURIItemType
QNameItemType	QName	udt : QNameItemType(동일)
durationItemType	duration	udt : DurationItemType
dateTimeItemType	date, dateTime	udt : DateTimeItemType(xbrli : dateUnion을 내장 타입의 하나로 처리)
timeItemType	time	udt : TimeItemType
dateItemType	date	udt : DateItemType
gYearMonthItemType	gYearMonth	udt : GYearMonthItemType
gYearItemType	gYear	udt : GYearItemType
gMonthDayItemType	gMonthDay	udt : GMonthDayItemType
gDayItemType	gDay	udt : GDayItemType
gMonthItemType	gMonth	udt : GMonthItemType
normalizedStringItemType	normalizedString	udt : NormalizedStringItemType
tokenItemType	token	udt : TokenItemType
languageItemType	language	udt : LanguageItemType
NameItemType	Name	udt : NameItemType(동일)
NCNameItemType	NCName	udt : NCNameItemType(동일)
stringItemType	string	udt : StringItemType
booleanItemType	Boolean	udt : BooleanItemType
hexBinaryItemType	hexBinary	udt : HexBinaryItemType
base64BinaryItemType	base64Binary	udt : Base64BinaryItemType
anyURIItemType	anyURI	udt : AnyURIItemType
QNameItemType	QName	udt : QNameItemType(동일)
durationItemType	duration	udt : DurationItemType
dateTimeItemType	date, dateTime	udt : DateTimeItemType(xbrli : dateUnion을 내장 타입의 하나로 처리)
timeItemType	time	udt : TimeItemType
dateItemType	date	udt : DateItemType
gYearMonthItemType	gYearMonth	udt : GYearMonthItemType
gYearItemType	gYear	udt : GYearItemType
gMonthDayItemType	gMonthDay	udt : GMonthDayItemType
gDayItemType	gDay	udt : GDayItemType
gMonthItemType	gMonth	udt : GMonthItemType
normalizedStringItemType	normalizedString	udt : NormalizedStringItemType
tokenItemType	token	udt : TokenItemType
languageItemType	language	udt : LanguageItemType
NameItemType	Name	udt : NameItemType(동일)
NCNameItemType	NCName	udt : NCNameItemType(동일)

문제없이 처리될 수 있다. dateTimeImteType의 경우 XBRL에서 정의된 dateUnion을 기반으로 하는데, 이것은 XSD 내장 데이터 타입이 아니므로 문제가 될 수 있다. 그럼에도 불구하고 이 연구에서는 <그림 7>과 같이 이를 예외사항으로 인정하는 것으로 결정하였다.

```
1. <complexType name="FractionItemType" final="extension">
2.   <simpleContent>
3.    <extension base="xsd:double">
4.     <attribute name="numerator" type="xsd:decimal" />
5.     <attribute name="denominator" type="xbrli:nonZeroDecimal" />
6.     <attributeGroup ref="xbrli:essentialNumericItemAttrs" />
7.    </extension>
8.   </simpleContent>
9. </complexType>
```

<그림 6> FractionItemType 데이터 타입의 정의

```
1. <complexType name="DateTimeItemType" final="extension">
2.   <simpleContent>
3.    <extension base="xbrli:dateUnion">
4.     <attributeGroup ref="xbrli:nonNumericItemAttrs" />
5.    </extension>
6.   </simpleContent>
7. </complexType>
```

<그림 7> DateTimeItemType의 정의

이상과 같은 사항들을 총 정리하면, 다음 <표 2>와 같이 정리할 수 있다.

(2) XBRL 사례문서에서 사용되는 요소의 매핑

모든 XBRL 사례 문서에서 사용될 수 있는 요소로는 context, unit, 그리고 footnoteLink가 있다. 이들 중에서 footnoteLink는 사례문서에 포함된 요소들을 연결하는 역할을 하는 것으로, 전자문서의 구조와는 직접적으로 무관하므로, 이 연구에서는 제외하기로 한다.

먼저 context라고 하는 엘리먼트는 entity, period, scenario라고 하는 엘리먼트로 구성되어 있는데, 각각 contextEntityType, contextPeriodType, contextScenarioType으로 정의된다. 이들 세 가지는 모두 데이터 타입으로 정의하는 것이 불가능하며, 특별한 CC/BIE로 정의해야 한다. contextEntityType은 identifier와 segment로 구성되며, contextPeriodType은 startDate와 endDate, instant, forever중 하나를 선택하는 것으로 구성된다. contextEntityType은 ContextEntityType이라는 이름으로 <그림 8>과 같이 identifier와 segment로 이루어진 특별한 CC/BIE로 규정하고, 각각 URIIdentifierType과 SegmentType을 이용하여 정의하였다. <그림 9>와 같이 URIIdentifierType은 비한정 데이터 타입으로 정의하였으며, <그림 10>과 같이 SegmentType의 경우 CC/BIE가 any를 이용하고 있어서 핵심컴포넌트 방법론의 규칙을 위반하고 있으나, XBRL에 대한 지원이 가능하도록 하기 위해서는 이를 예외로서 인정하도록 하였다. ContextPeriodType의 경우 비한정 데이터 타입으로 이미 규정된 DateUnionType을 이용하여 정의된다. contextScenarioType의 경우 ContextScenarioType이라는 이름으로 <그림 11>과 같이 특별한 CC/BIE로 규정하였다. 이 CC/BIE도 SegmentType과 마찬가지로 any를 이용하고 있어서, 이를 예외로 인정하기로 결정하였다. 마지막으로 ContextType은 <그림 12>와 같이 특별한 CC/BIE로 정의된다.

```
1. <xsd:complexType name="ContextEntityType">
2.   <xsd:sequence>
3.     <xsd:element name="Identifier"
                    type="udt:URIIdentifierType" />
4.     <xsd:element name="Segment"
                    type="ram:SegmentType" minOccurs="0" />
5.   </xsd:sequence>
6. </xsd:complexType>
```

<그림 8> ContextEntityType의 정의

```
1.  <complexType name="URIIdentifierType">
2.    <simpleContent>
3.      <extension base="token">
4.        <attribute name="scheme" use="required">
5.          <simpleType>
6.            <restriction base="anyURI">
7.              <minLength value="1" />
8.            </restriction>
9.          </simpleType>
10.       </attribute>
11.     </extension>
12.   </simpleContent>
13. </complexType>
```

<그림 9> URIIdentifierType의 정의

```
1.  <xsd:complexType name="SegmentType">
2.    <xsd:sequence>
3.      <any namespace="##other" processContents="lax"
4.        minOccurs="1" maxOccurs="unbounded" />
5.    </xsd:sequence>
6.  </xsd:complexType>
```

〈그림 10〉 SegmentType의 정의

```
1.  <xsd:complexType name="ContextScenarioType">
2.    <xsd:sequence>
3.      <xsd:any namespace="##other" processContents="lax"
4.        minOccurs="1" maxOccurs="unbounded" />
5.    </xsd:sequence>
6.  </xsd:complexType>
```

〈그림 11〉 ContextScenarioType의 정의

```
1.  <xsd:complexType name="ContextType">
2.    <xsd:sequence>
3.      <xsd:element name="ContextEntity" type="ram:ContextEntityType" />
4.      <xsd:element name="ContextPeriod" type="ram:ContextPeriodType" />
5.      <xsd:element name="ContextScenario" type="ram:ContextScenarioType"
                  minOccurs="0" />
6.    </xsd:sequence>
7.    <xsd:attribute name="id" type="ID" use="required" />
8.  </xsd:complexType>
```

〈그림 12〉 ContextType의 정의

XBRL에서 unit은 measure와 divide 중 하나를 선택하는 것으로 정의되어 있는데, divide의 경우 measuresType을 이용한 unitNumerator와 unitDenominator로 구성됨을 알 수 있다. 이를 핵심 컴포넌트 방법론에 포함시키기 위하여 <그림 13>과 같은 UnitType이라고 하는 특별한 CC/BIE를 생성하였다. unitNumerator와 unitDenominator를 정의하기 위해서 MeasuresType을 <그림 14>와 같이 특별한 CC/ABIE로 정의하고, 다시 <그림 15>와 같이 비한정 데이터

```
1.  <xsd:complexType name="UnitType">
2.    <xsd:choice>
3.      <xsd:element name="Measures" type="udt:MeasuresType" />
4.      <xsd:element name="Divide">
5.        <xsd:complexType>
6.          <xsd:sequence>
7.            <xsd:element name="UnitNumerator" type="ram:MeasuresType" />
8.            <xsd:element name="UnitDenominator" type="ram:MeasuresType" />
9.          </xsd:sequence>
10.       </xsd:complexType>
11.     </xsd:element>
12.   </xsd:choice>
13.   <xsd:attribute name="id" type="ID" use="required" />
14. </xsd:complexType>
```

〈그림 13〉 UnitType의 정의

타입인 MeasureType을 정의하였다. MeasuresType의 경우 복수형을 사용하고 있는데, MeasureType을 반복적으로 이용하여 정의되는 형식으로 그 필요성이 인정되므로, 여기서는 예외로서 인정한다.

```
1.  <xsd:complexType name="MeasuresType">
2.    <xsd:sequence>
3.      <xsd:element name="Measure" type="udt:MeasureType"
4.        minOccurs="1" maxOccurs="unbounded" />
5.    </xsd:sequence>
6.  </xsd:complexType>
```

〈그림 14〉 MeasuresType의 정의

```
1.  <xsd:simpleType name="MeasureType">
2.    <xsd:restriction base="QName"/>
3.  </xsd:simpleType>
```

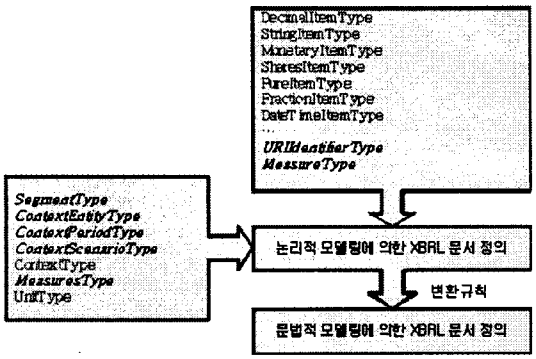
〈그림 15〉 MeasureType의 정의

(3) CC/BIE에서 XBRL로의 변환

앞서 정의된 데이터 타입과 특별한 CC/BIE를 이용해서 XBRL 문서를 정의하는 과정은 <그림 16>과 같이 정의할 수 있다. 이 그림에서 이탤릭으로 표현된 이름은 CC/BIE를 규정하는 과정에서 간접적으로 이용됨을 의미한다. 따라서 XBRL 문서정의에서는 이탤릭이 아닌 이름들을 이용하게 되는데, 비한정 데이터 타입을 이용하여 ACC/ABIE에 속하는 BCC/BBIE를 논리적인 방식으로 모델링 하게 된다.

이렇게 모델링 된 내용은 변환 규칙을 통해서 XBRL 문법을 준수하는 전자문서로 정의하게 된다. 비록 모델링 된 내용을 핵심 컴포넌트 방법론의 문법적 규칙에 따라서 전자문서로 정의할 수도 있지만, 이것은 XBRL 표준과는 무관한 것이므로 생성할 필요가 없다. 향후 XBRL의 발전에 일부 참조될 수는 있을 것으로 보이며, XBRL 표준화 과정에 반영하는 노력이 필요할 수 있다.

논리적인 모델링은 UN/CEFACT TBG17 제



〈그림 16〉 핵심 컴포넌트 방법론에 의한 XBRL 문서 정의 절차

출 양식을 이용하여 이루어진다. 하지만 이러한 양식은 일반적인 형태로서 XBRL의 자동화된 생성을 보장해주지는 못한다. 이를 위해서는 추가적인 특징들이 정의되어야 한다. 추가적인 특징으로는 id, abstract, nillable, balance, periodType 등 다섯 가지가 필요하다. 이러한 특징을 TBG17에 직접 반영하는 것은 많은 문제를 일으킬 것으로 예견되므로(예를 들어 이러한 특징들은 향후 더 확장될 수도 있음), 이들을 정리한 XBRL만의 양식으로 정의하였는데, 이러한 양식의 형태는 <표 3>과 같다.

이러한 XBRL의 논리적 모델링에서는 핵심 컴포넌트 방법론에서 활발히 사용되고 있는 ApplicationArea와 DataArea라고 하는 두 가지 BIE가 요구되지 않는다. 대신에 ContextType과 UnitType이 반드시 포함되어야 한다. 또 한 가지 유의할 점은 id의 경우 택사노미의 이름공간 접두어(Namespace Prefix)와 사전 기재 이름(Dictionary Entry Name)을 '_'로 연결하여 표현한다는 점이다. 이러한 id의 명명규칙은 XBRL의 택사노미 재사용 메커니즘을 지원하기 위한 밑바탕이 된다.

논리적으로 모델링 된 내용으로부터 문법적인 내용을 생성하는 메커니즘은 XBRL에서CC/BIE를 규정하는 과정과는 정반대로 진행되면 된다. 무엇보다도 전자문서에 해당되는 ABIE에 대하여 하나의 tuple을 설정하고, 하위 요소들을 item 또는 또 다른 tuple로서 생성하면 된다. ABIE의 경우 sbustitutionGroup을 xbrli : tuple로 설정하여 엘리먼트를 생성하고, 그 하위요소들을 complexType으로 구성하여 내용을 만들면 된다.

BBIE의 경우 비한정 데이터 타입을 item의 type 속성값으로, xbrli : item을 substitutionGroup 속성값으로 설정하여 생성한다. 그리고 id, abstract, nillable, balance, periodType 등의 속성값은 <표 3>의 양식으로부터 얻어서 설정하면 된다. 이러한 엘리먼트의 이름공간은 id 속성으로부터 구하게 된다. StringItemType으로 정의된 BBIE 요소의 경우를 예로 들면, type 속성값을 stringItemType으로, substitutionGroup을 xbrli : item으로 각각 설정하여 item을 생성하면 된다.

QNameItemType, NameItemType, NCNameItemType로 정의된 요소를 생성할 경우에는 type 속성값으로 이름을 그대로 사용하도록 하며, 나머지의 경우에는 소문자 카멜 케이스가 되도록 첫 번째 영문 알파벳을 소문자로 바꾼다.

〈표 3〉 XBRL 모델링 양식

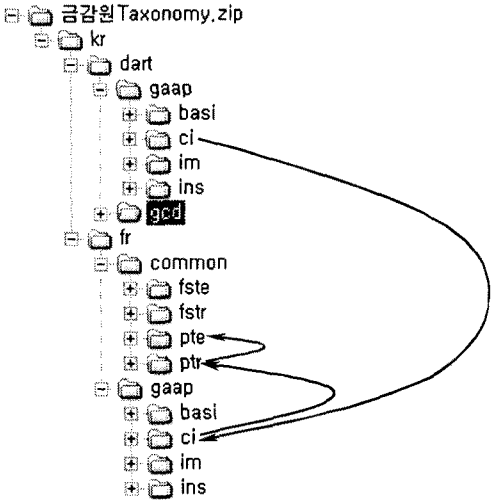
Entry Name (한글)	Entry Name (영문)	XBRL				
		id	abstract	balance	nillable	periodType

4. XBRL 문서 사례

현재 금융감독원에서 작성 중인 택사노미 가운데 손익계산서를 대상으로 핵심 컴포넌트 기반의 전자문서를 정의하고, 여기에서 필요한 샘플 CC/BIE를 정리한다. 이렇게 함으로써 이 연구에서 제안한 방법의 현실적인 가능성을 점검하고, 향후 체계적인 확장에 대비하고자 한다.

금융감독원의 택사노미는 <그림 17>과 같이 크게 dart와 fr로 이루어져 있다[20]. 여기서 dart란 금융감독원 전자공시시스템 DART(Data Analysis, Retrieval and Transfer System)를 의미하며, fr은 Financial Reporting의 약어이다. dart에는 gaap와 gcd가 있으며, fr에는 common과 gaap가 있다. 어느 쪽이나 gaap(Generally Accepted Accounting Principles)에는 basi, ci, im, ins가 포함되어 있는데, 여기서 basi는 Banking and Savings Institutions를, ci는 Commercial and Industrial을, ins는 Insurance Entities를, im은 Investment Management를 각각 나타낸다. Common에 속해있는 fste는 Financial Service Terms Elements를, fstr은 Financial Service Terms Relationships를, pte는 Primary Terms Elements를, ptr은 Primary Terms Relationships를 각각 의미한다.

이러한 구성요소들은 미국의 재무 보고 택사노미 프레임워크를 준용하고 있다. usft-pte는 가장 기본이 되는 요소이며, usfr-ptr와 usfr-ime로 구성되어 있다. usfr-ptr을 이용하여 us-gaap-ci가 구성되며, 이것이 가장 기초적인 택사노미 요소로서 기능하게 된다. 은행예금과 보험을 위한 us-gaap-basi와 us-gaap-ins가 제공되는데, 이들은 usfr-fstr을 이용하여 규정된다. usfr-fstr은 usfr-ptr과 usfr-fste를 포함하고 있다. 투자관리를 위한 us-gaap-im은 usfr-ime를 기반으로 구성된다.



〈그림 17〉 금융감독원 택사노미의 구성요소

여기서는 핵심 컴포넌트 방법론 기반의 XBRL 문서 표현을 시험하기 위해서 가장 기본이 되는 재무 문서 중의 하나인 대차대조표를 사용하고자 한다. 이러한 대차대조표는 ci 요소로서 존재한다. dart의 ci를 살펴보면 xbrl_dart-gaap-ci-2006-07-31.xsd라고 하는 파일이 있는데, 그 내용은 <그림 18>과 같다. 여기서 보면 "kr-gaap-ci"라고 하는 이름공간을 통해서 fr의 kr-gaap-ci-2006-05-31.xsd를 인용함을 알 수 있다.

<그림 19>와 같이 fr의 ci에서는 먼저 annotation의 appinfo를 이용해서 링크베이스 참조와 역할 타입을 정의하고 있다. 여기에서는 BalanceSheet라고 하는 역할 유형이 제시되어 있으며, 기타 역할들은 생략되어 있다. 그리고 annotation 이후에는 import를 사용해서 이름공간의 스키마 위치를 3개 지정하고 있는데, 마지막 import를 보면 fr의 krfr-ptr-2006-05-31.xsd를 참조함을 알 수 있다.

<그림 20>과 같이 fr의 ptr에서는 링크베이스를 참조하고, NetIncomeLoss BalanceSheet, NetIncomeLossIncomeStatement 등의 항목들을 모두 정의하고 있다. 방대한 량의 엘리먼트

들이 정의되어 있고, 이들을 링크베이스에서 관계로서 연결하고 있으므로 사례로서 사용하기가 쉽지 않다. 이 연구에서는 NetIncomeLoss IncomeStatement에 포함되는 내용만을 가지고 분석한다.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema elementFormDefault="qualified" xmlns:link="http://www.xbrl.org/2003/linkbase"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://xbrl_dart.fss.or.kr/kr/dart/gaap/ci/2006-07-31"
xmlns:xbrli="http://www.xbrl.org/2003/instance"
xmlns:dart-gaap-ci="http://xbrl_dart.fss.or.kr/kr/dart/gaap/ci/2006-07-31"
xmlns:kr-gaap-ci="http://www.xbrl.or.kr/kr/fr/gaap/ci/2006-05-31">
 <xsd:import namespace=http://www.xbrl.org/2003/instance"
schemaLocation="http://www.xbrl.org/2003/xbrl-instance-2003-12-31.xsd"/>
<xsd:import namespace="http://www.xbrl.or.kr/kr/fr/gaap/ci/2006-05-31"
schemaLocation="http://www.xbrl.or.kr/kr/fr/gaap/ci/2006-05-31/kr-gaap-ci-2006-05-31.xsd"/>
</xsd:schema>
```

〈그림 18〉 dart의 ci

```
<?xml version="1.0" encoding="UTF-8"?>
<schema elementFormDefault="qualified" xmlns:link="http://www.xbrl.org/2003/linkbase"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.xbrl.or.kr/kr/fr/gaap/ci/2006-05-31"
attributeFormDefault="unqualified" xmlns:xbrli="http://www.xbrl.org/2003/instance"
xmlns:kr-gaap-ci="http://www.xbrl.or.kr/kr/fr/gaap/ci/2006-05-31">
 <annotation>
  <appinfo>
   <link:linkbaseRef xlink:type="simple"
xlink:arcrole="http://www.w3.org/1999/xlink/properties/linkbase"
xlink:href="kr-gaap-ci-2006-05-31-presentation.xml" xlink:title="Presentation Links, all"
xlink:role="http://www.xbrl.org/2003/role/presentationLinkbaseRef"/>
   <link:linkbaseRef xlink:type="simple"
xlink:arcrole="http://www.w3.org/1999/xlink/properties/linkbase"
xlink:href="kr-gaap-ci-2006-05-31-calculation.xml" xlink:title="Calculation Links, all"
xlink:role="http://www.xbrl.org/2003/role/calculationLinkbaseRef"/>
   <link:roleType roleURI="http://www.xbrl.or.kr/role/BalanceSheet" id="BalanceSheet">
    <link:definition>Balance Sheet</link:definition>
    <link:usedOn>link:presentationLink</link:usedOn>
    <link:usedOn>link:calculationLink</link:usedOn>
   </link:roleType>
...
</appinfo>
 </annotation>
<import namespace="http://www.xbrl.org/2003/instance"
schemaLocation="http://www.xbrl.org/2003/xbrl-instance-2003-12-31.xsd"/>
<import namespace="http://www.xbrl.org/2003/linkbase"
schemaLocation="http://www.xbrl.org/2003/xbrl-linkbase-2003-12-31.xsd"/>
<import namespace="http://www.xbrl.or.kr/kr/fr/common/pte/2006-05-31"
schemaLocation="http://www.xbrl.or.kr/kr/fr/common/pte/2006-05-31/krfr-pte-2006-05-31.xsd"/>
</schema>
```

〈그림 19〉 fr의 ci

손익계산서를 나타내는 IncomeStatement에 포함된 요소들 간의 계산관계는 수형구조로 표현할 수 있다. 이들에 대한 내용은 fr의 ci에 있는 kr-gaap-ci-2006-05-31-calculation.xml에서 규정하고 있다. 최종적으로 계산되는 항목은 당기순이익(손실), 즉 NetIncomeLossIncomeStatement이다. 이러한 계산관계는 fr의 ci에 정의된 kr-gaap-ci-2006-05-31-calculation.xml을 가지고 확인할 수 있다.

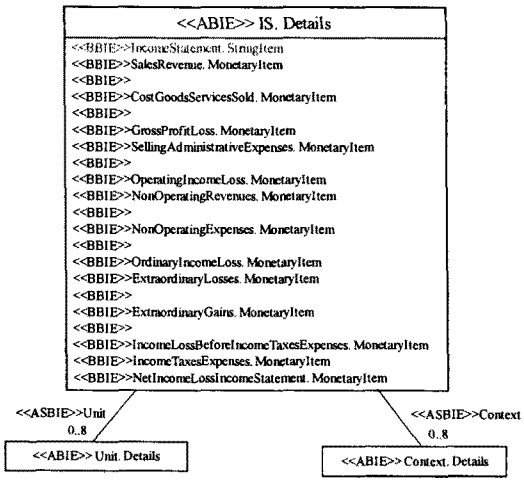
```
<?xml version="1.0" encoding="UTF-8"?>
<schema elementFormDefault="qualified" xmlns:link="http://www.xbrl.org/2003/linkbase"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.xbrl.or.kr/kr/fr/common/pte/2006-05-31"
attributeFormDefault="unqualified" xmlns:xbrli="http://www.xbrl.org/2003/instance"
xmlns:krfr-pte="http://www.xbrl.or.kr/kr/fr/common/pte/2006-05-31"
xmlns:p0="http://www.xbrl.or.kr/2006/role/subitem-notes">
 <annotation>
  <appinfo>
   <link:linkbaseRef xlink:type="simple"
xlink:arcrole="http://www.w3.org/1999/xlink/properties/linkbase"
xlink:href="krfr-pte-2006-05-31-label.xml" xlink:title="Label Links, all"
xlink:role="http://www.xbrl.org/2003/role/labelLinkbaseRef"/>
   <link:linkbaseRef xlink:type="simple"
       xlink:arcrole="http://www.w3.org/1999/xlink/properties/linkbase"
       xlink:href="krfr-pte-2006-05-31-reference.xml" xlink:title="Reference Links, all"
       xlink:role="http://www.xbrl.org/2003/role/referenceLinkbaseRef"/>
  </appinfo>
 </annotation>
<import namespace="http://www.xbrl.org/2003/instance"
schemaLocation="http://www.xbrl.org/2003/xbrl-instance-2003-12-31.xsd"/>
<import namespace="http://www.xbrl.org/2003/linkbase"
schemaLocation="http://www.xbrl.org/2003/xbrl-linkbase-2003-12-31.xsd"/>
   <element name="NetIncomeLossBalanceSheet" id="krfr-pte_NetIncomeLossBalanceSheet"
   type="xbrli:monetaryItemType" p0:subitem-notes="yes" substitutionGroup="xbrli:item"
   nillable="true" xbrli:periodType="duration"/>
   <element name="NetIncomeLossIncomeStatement"
   id="krfr-pte_NetIncomeLossIncomeStatement" type="xbrli:monetaryItemType"
   substitutionGroup="xbrli:item" nillable="true" xbrli:balance="credit"
   xbrli:periodType="duration"/>
...
</schema>
```

〈그림 20〉 fr의 ptr

IncomeStatement를 작성하는 데 필요한 프리젠테이션 관계는 kr-gaap-ci-2006-05-31-presentation.xml에서 규정하고 있다. 프리젠테이션은 여러 가지 방법으로 가능하지만, 논리적으로 함께 이용되는 데이터 요소들의 집합이 전자문서의 실체로서 인식될 수 있다. <그림 21>은 이 문서의 구조를 이해하기 쉽도록 정리한 것이다.

1.	IncomeStatement
2.	SalesRevenue
3.	...
4.	CostGoodsServicesSold
5.	...
6.	GrossProfitLoss
7.	SellingAdministrativeExpenses
8.	...
9.	OperatingIncomeLoss
10.	NonOperatingRevenues
11.	...
12.	NonOperatingExpenses
13.	...
14.	OrdinaryIncomeLoss
15.	ExtraordinaryLosses
16.	...
17.	ExtraordinaryGains
18.	...
19.	IncomeLossBeforeIncomeTaxesExpenses
20.	IncomeTaxesExpenses
21.	*NetIncomeLossIncomeStatement*

〈그림 21〉 손익계산서의 프리젠테이션



〈그림 22〉 손익계산서의 개념적 정의

손익계산서는 모두 200여 개의 항목으로 이루어져 있는데, xbrli : stringItemType인 IncomeStatement를 제외하고는 xbrli : monetaryItemType의 일종이다. 이러한 내용은 <그림 22>와 같은 하나의 UML 클래스로 표현될 수 있다. 여기서 IS는 손익계산서를 표현하는 ABIE가 되며, IS를 구성하는 개개의 속성들은 BBIE가 된다. 이러한 내용은 UN/CEFACT TBG17 양식과 XBRL 보조양식을 이용하여 논리적으로 모델링된다. XBRL 보조양식을 이용한 모델의 전체적 구조는 <표 4>와 같다.

5. 결 론

본 연구에서는 XBRL 문서의 표준화 및 재활용이 가능하도록 UN/CEFACT의 핵심 컴포넌트 방법론을 적용하는 방법을 제시하였다. 문법적인 수준에서 이루어지고 있는 XBRL 비즈니스 보고에 대해 개념적이고 논리적인 개발체계를 적용할 수 있도록 핵심 컴포넌트 기반의 XML 전자문서 개발 방법을 도입하는 방안을 제시하였고, 아울러 이러한 접근방법의 유효성을 점검하기 위하여 금융감독원에서 작성 중인 손익계산서 택사노미를 대상으로 핵심 컴포넌트 기반 전자문서를 정의하는 데 필요한 핵심 컴포넌트와 비즈니스 정보 개체도 도출하였다. 특히, XBRL 문서를 모델링하는 데 핵심 컴포넌트 방법론을 적용하기 위해 두 방법 사이의 구성요소들 사이의 매핑 규칙을 제안하였고, 제안된 방법을 통해 XBRL 문서를 개발하는 데 필요한 모델링 양식도 제시하였다.

핵심 컴포넌트 기반의 전자문서 개발 방법론의 유용성은 전자문서를 개발하고 사용하는 기관들 사이에서 이들 핵심 컴포넌트와 비즈니스 정보 개체들을 제출받아 등록하고 관리하는 CC/BIE 라이브러리의 유지에 있다고 하겠다. 국내 XML 전자문서의 경우 한국전자거래진흥원을 통해 여기에 필요한 절차가 진행되고, 그 결과로서 관련 라이브러리(http://www.remko.or.kr)가 운영되고 있다. XBRL의 경우도 동일한 라이브러리를 활용하거나 혹은 유사한 체계를 갖춤으로써 XBRL 문서의 표준화와 재활용 가능성을 높이기 위해서는 본 연구를 통해 제시된 방법에 따라 XBRL 문서를 개발할 필요가 있다고 하겠다.

XBRL 문서를 개발하는 데 핵심 컴포넌트를 기반으로 하는 방법론을 적용하거나 혹은 상호 연동할 수 있는 국제 표준이나 국내 표준화 활동은 아직 이루어지지 않고 있다. 그럼에도 불구하고 본 논문에서 제시하는 방안은 이에 대한 가능성을 제시함으로써 보다 활발한 표준화 논의가 진행될 수 있는 길을 열었다고 할 수 있다. 이와 관련하여 이후 진행되어야 할 연구방향을 정리하면 다음과 같다. 먼저, XBRL 문서 요소의 개발과 등록, 승인 그리고 재사용을 위한 검색에 필요한 일련의 절차에 대해 기존 XML 전자문서 개발 체계를 적용할 수 있는 방안을 수립하는 작업이 필요하다. 다음으로는 이

〈표 4〉 XBRL 모델링 양식을 이용한 손익계산서 모델링

Entry Name(한글)	Entry Name (영문)	XBRL				
		id	abstract	balance	nillable	period Type
손익계산서	IncomeStatement	krfr-pte_ IncomeStatement	TRUE		TRUE	duration
매출액	SalesRevenue	krfr-pte_ SalesRevenue	FALSE	credit	TRUE	duration
…						
매출원가	CostGoodsServicesSold	krfr-pte_CostGoodsServicesSold	FALSE	debit	TRUE	duration
…						
매출총이익(매출총손실)	GrossProfitLoss	krfr-pte_GrossProfitLoss	FALSE	credit	TRUE	duration
판매비와관리비	SellingAdministrativeExpenses	krfr-pte_SellingAdministrativeExpenses	FALSE	debit	TRUE	duration
…						
영업이익(영업손실)	OperatingIncomeLoss	krfr-pte_OperatingIncomeLoss	FALSE	credit	TRUE	duration
영업외수익	NonOperatingRevenues	krfr-pte_NonOperatingRevenues	FALSE	credit	TRUE	duration
…						
영업외비용	NonOperatingExpenses	krfr-pte_NonOperatingExpenses	FALSE	debit	TRUE	duration
…						
경상이익(경상손실)	OrdinaryIncomeLoss	krfr-pte_OrdinaryIncomeLoss	FALSE	credit	TRUE	duration
특별이익	ExtraordinaryGains	krfr-pte_ExtraordinaryGains	FALSE	credit	TRUE	duration
…						
특별손실	ExtraordinaryLosses	krfr-pte_ExtraordinaryLosses	FALSE	debit	TRUE	duration
…						
법인세비용차감전순이익(법인세비용차감전순손실)	IncomeLossBeforeIncomeTaxesExpenses	krfr-pte_IncomeLossBeforeIncomeTaxesExpenses	FALSE	credit	TRUE	duration
법인세비용	IncomeTaxesExpenses	krfr-pte_IncomeTaxesExpenses	FALSE	debit	TRUE	duration
당기순이익(당기순손실)	NetIncomeLossIncomeStatement	krfr-pte_NetIncomeLossIncomeStatement	FALSE	credit	TRUE	duration

같은 토대 위에서 다양한 비즈니스 보고에 적용될 수 있는 CC/BIE의 라이브러리를 구축할 필요가 있다. 현재 한국전자거래진흥원을 통하여 국가적으로 운영되고 있는 CC/BIE 라이브러리

에는 비즈니스 보고 관련 요소들은 포함되어 있지 않은 반면, 금융, 환경 분야에서의 비즈니스 보고를 위한 수요가 급증하고[1, 2, 3, 4, 11, 20] 있는 만큼 이러한 라이브러리의 필요성은 크다고 하겠다. 마지막으로 본 논문에서 제시하는 방안에 따라 단계별 진행과정을 자동화된 방법으로 지원할 수 있는 도구가 개발된다면 비즈니스 보고 개발 과정의 효율성과 라이브러리 재사용성을 보다 효과적으로 제고시킬 수 있을 것으로 기대한다.

참 고 문 헌

[1] 구정옥, 김강정, "XBRL과 회계정보투명성", *회계정보연구*, 제22권 제2호, 2004년 6월, pp. 179-202.

[2] 김형도, "비즈니스 프로세스 및 정보모델 기반의 전자문서 구조 개발지침(ECIF 104 : 2004)", *2004 ECIF 정기총회 및 기념심포지엄 발표논문집*, 2004년 3월.

[3] 김형도, "중소기업 생산성 향상을 위한 XBRL기반의 지속가능성 보고 방안 연구", *생산성논집*, 제19권 제4호, 2005년 12월, pp. 147-169.

[4] 노미현, "인터넷 재무보고와 XBRL에 관한 연구", *인터넷비즈니스연구*, 2005, pp. 139-159.

[5] 박찬권, "XML 라이브러리의 효율적 재사용을 위한 비즈니스 정보 개발 방안", *한국정보기술응용학회 2008 추계 공동 국제학술대회 논문집*, 2008년 10월, pp. 656-663.

[6] 안경림, 김동희, 박찬권, 박정천, "철도물류정보 표준화 방안 및 정보시스템 개선에 대한 연구," *한국전자거래학회지*, 제13권 제3호, 2008년 8월, pp. 121-135.

[7] 한국전자거래진흥원, *XML 전자문서 개발지침 v3.0*, 2005년 5월.

[8] OASIS, Universal Business Language 1.0, September 2004.

[9] OMG, Unified Modeling Language Specification(Version 1.3), http://www.omg.org/, 1999.

[10] Rumbaugh, J., Jacobson, I., and Booch, G., The Unified Modeling Language Reference Manual, Wesley, 1999.

[11] Seetharman A., Subramaniam R., and Shyong, S. Y., "Internet Financial Reporting : Problems and Prospects", *Corporate Finance Review*, Sep/Oct 2005, pp. 29-35.

[12] UN/CEFACT, Core Component Technical Specification v2.01, November 2003.

[13] UN/CEFACT, Submission Guidelines and Procedures v1.0, September 2004.

[14] UN/CEFACT, TBG17 Library Administration Procedures v1.0, March 2006a.

[15] UN/CEFACT, XML Naming and Design Rules v2.0, February 2006b.

[16] W3C, Extensible Markup Language(XML) 1.0(FourthEdition), http://www.w3.org/TR/xml, August 2006a.

[17] W3C, XML Schema, http://www.w3.org/XML/Schema, 2006b.

[18] XBRL International, XBRL 2.1, http:// www.xbrl.org/SpecRecommendations/, Nov. 2005.

[19] XBRL International, XBRL 2.1 Conformance Suite, http://www.xbrl.org/. SpecRecommendations/, 2005.

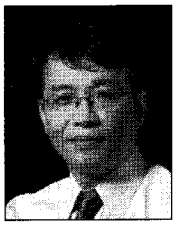
[20] XBRL Korea, Korean GAAP Taxonomy, http://www.xbrl.org/Taxonomy/kore-

an/fr/gaap/ci/pfs/2004-01-19/korean-gaap-ci-pfs-2004-01-19.doc, 2004.

저자소개

김 형 도

서울대학교 산업공학과에서 학사, 그리고 한국과학기술원 경영과학과에서 석사와 박사 학위를 취득하였다. 현재 한양사이버대학교 부교수로 재직하고 있으며, ECIF 전자문서기술위원회 부위원장, 전자거래진흥원 비즈니스 프로세스 워킹그룹 위원장 등을 맡고 있다. ebXML 전문위원회 위원장을 역임한 바 있다. 새로운 전자거래 기반을 중심으로 연구개발, 교육 및 표준화 활동을 활발히 전개하고 있으며, Decision Support Systems, ACM SIGMOD Record, IEICE Transactions on Communications, Expert Systems with Applications, Int'l Journal of Management Science 등에 다수의 논문을 게재하였다. 저서로는 "B2B 전자상거래@XML", "전자상거래 원론", "전자문서 용어사전", "데이터마이닝의 이해" 등이 있다. 주요 관심 분야는 XML, 전자상거래, 비즈니스 프로세스, 디지털 워터마킹, IT 아키텍쳐, 데이터마이닝 응용 등이다.



박 찬 권

서울대학교에서 산업공학을 전공하였고, 동 대학원에서 석사 및 박사학위를 취득하였다. 서울대학교 자동화시스템 공동연구소에서 Post Doc을 거친 후, 영산대학교 정보경영학부에 재직하였으며, 현재는 한양사이버대학교 경영학부에 재직 중이다. 현재 한국전자거래진흥원의 전자문서워킹그룹위원장을 맡고 있으며, 주요 관심분야는 ERP/SCM, 정보시스템 개발 방법론, XML, 전자문서 개발방법론 및 표준화 활동 등이다.

◈ 이 논문은 2008년 06월 18일 접수하여 2008년 12월 19일 게재확정되었습니다.