KIEC 2003 - 042
2003. 3

국내 XML 전자문서 개발 가이드라인

Development Guidelines for XML Electronic Messages in Korea

한국전자거래진흥원

머 리 말

20세기 후반 인터넷 정보기술의 발달이 전 세계를 네트워크로 연결하여 거대한 단일 시장을 형성하고, 경제활동뿐만 아니라 사회, 문화, 정치활동 등 모든 부분에 걸쳐서 우리 일상 생활에 없어서는 안 될 새로운 패러다임으로 자리잡아가고 있습니다. 이와 함께 상거래에 있어서도 가상의 인터넷 공간에서 상품과 서비스를 구매하는 전자상거래 방식으로 급속히 증가하고 있습니다.

전자상거래가 국경 없는 경제환경을 이룩함에 따라 세계 각국은 전자상거래를 국가의 전략과제로 인식하여 체계적인 경제 질서를 창조하려는 노력을 아끼고 있지 않습니다. 국내에서도 전자상거래의 활성화를 위하여 지속적인 연구 및 보급에 최선을 다 하고 있으며, 그 결과로 2002년에는 전자상거래 시장규모가 177조원으로 상거래 시장의 12.7%를 차지하고 있습니다.

이와 같이 전자상거래의 규모가 대폭적으로 늘어남에 따라 유통되는 문서의 양도 크게 증가하고 있으며, 늘어나는 문서량에 비례해서 업종에 맞는 합리적인 문서의 표준화와 효율적인 문서의 보관 및 관리의 중요성이 부각되었습니다.

전자문서의 표준은 과거의 EDI에서 현재는 XML 기반의 국제표준인 ebXML이 표준 프레임워크로 자리를 잡고 있습니다. 한국전자거래진흥원에서는 이와 같은 세계 표준화의 변화 흐름에 맞추어서 ebXML의 국내 보급과 활성화에 많은 노력을 기울여 왔으며, 그 노력의 일환으로 2001년에는 “전자문서 개발 가이드라인” 초판을 마련하여 발표하였습니다. 2002년에는 그 동안 UN/CEFACT에서 작업한 결과를 토대로 해서 초판에서 좀 더 발전시킨 가이드라인을 발간하게 되었습니다.

본 가이드라인은 국내 현업에서 XML 전자문서 개발을 더욱 쉽게 할 수 있고, 시스템간 전자문서의 상호연동을 효율적으로 지원할 수 있도록 XML 전자문서를 포함한 XML 라이브러리와 공통 비즈니스 참조 라이브러리를 포함하였습니다. 그러므로 B2B 표준화를 추진하는 업체와 XML 기반의 전자상거래 관련 업체 및 학과에서 참고하여 전자문서 개발에 조금이라도 도움이 되는 계기가 되었으면 하는 바램입니다.

아직은 미진한 부분과 부족한 내용이 많다는 것을 인정하지 않을 수 없습니다. 하지만, 저희 진흥원에서 앞으로 보완해 나갈 예정이며 독자 여러분들의 끊임없는 관심과 성원을 부탁드립니다. 감사합니다.

2003. 3.

한국전자거래진흥원장

정 득 진

목 차

1. 소 개

1.1 가이드라인 개요

1.1.1 본 가이드라인은 한국 전자문서 교환위원회에 제출하기 위한 초안상태이다.

1.1.2 많은 기업이나 조직들이 업무상에서 발생되는 행정, 상업 그리고 운송 정보를 서로간의 컴퓨터 어플리케이션을 통해 교환하기 위해 가장 기본적으로 요구되는 사항은 전송되는 정보의 내용과 구조에 대한 합의이다.

1.1.3 기업이나 조직들간에 서로 정보를 교환하기 위해서는 각 기업이나 조직들에서 운영하는 어플리케이션이 서로 공유/인식할 수 있는 공통의 비즈니스 언어가 있어야 한다.

1.1.4 이 지침과 규칙은 XML 국제 표준이 부재한 상태에서 국내의 XML 전자문서 개발과 관련된 혼란을 최소화하기 위해 전자거래진흥원에서 국내를 대상으로 XML 전자문서 개발시 거래 상대방간의 상호연동을 위해서 준용하여야 하는 가이드라인이다.

1.1.5 국내에서 EDI 혹은 기업간 전자상거래를 하기 위하여 필요로하는 메시지를 개발하기 위해서는 전 세계적으로 많이 활용하는 단체 표준이나, 업종의 대표적인 표준을 활용할 것을 권고하며, 마땅한 표준이나 참고 스펙이 없거나, 국내용으로만 한정하여 사용할 경우, 본 XML 전자문서 표준화 가이드라인에서 제공하는 XML 라이브러리와 지침에 맞춰서 개발할 것을 권고한다.

1.1.6 표준 메시지로 개발된 전자문서의 경우, 이미 설계된 메시지 기본 구조를 재설계할 필요는 없다. 시스템간 전자문서의 상호 연동이 이루어지기 위해서는 통합된 전자문서 개발은 통합되고 혁신적이고 구조화된 방법으로 수행되어져야 한다.

1.1.7 국내 전자문서 개발자들은 신규 전자문서 설계전에 전자거래진흥원에 고시된 표준 전자문서를 먼저 검색을 하거나, 표준개발팀과 협의할 것을 권고한다. 왜냐하면 어떤 경우 필요한 전자문서가 이미 존재하거나 또는 동일하거나 비슷한 전자문서에 대한 작업이 이미 시작되었을 수도 있기 때문이다.

1.1.8 전자문서교환(EDI)을 위해 설계된 전자문서는 명확히 정의된 비즈니스 기능을 가지고 있어야 한다. 전자문서에서 자료가 나타나는 순서와 자료의 내용은 반드시 서류에서 나오는 자료의 순서와 내용을 따를 필요는 없다.

1.1.9 전자문서에는 EDIFACT의 경우에는 각각의 전송항목과 전자문서내 전송항목의 순서가 규정되어 있으며, XML 라이브러리의 경우 컴포넌트와 전자문서내 컴포넌트와 기본 정보 개체가 있으며, 전자문서내 이들에 대한 순서가 규정되어 있다. 이러한 전자문서내 순서와 반복횟수를 준용하여야 한다. 단, Conditional하게 정의되어 있을 경우에는 생략이 가능하며, 반복횟수는 축약될 수 있다.

1.1.10 전자문서의 기능이 비즈니스 서류와 동일하더라도, 전자문서 개발자는 전자문서 자료항목 내용이 반드시 동일한 서류안에 포함된 자료와 같다고 추측하여서는 안된다. 전자문서를 개발할 때는 기존 업무 절차를 분석할 기회를 가져야만 한다. 왜냐하면, 단순히 원본을 모방하는 것보다 새로운 접근 방법을 채택하는 것이 더 효과적일 수 있기 때문이다. 어떤 경우, 전자문서의 기능은 몇 개의 전통적인 서류나 사업과정과 동등할 수도 있으며, 다수의 서류에 포함하는 기능을 하나의 전자문서가 대신할 수도 있다. 또한 서류나 비즈니스 과정의 일부분 내지 전체를 직접적으로 대체할 수도 있다. 비즈니스 및 정보 모델링기번은 업무를 지원하는 구조적인 접근방법을 제공한다.

1.1.11 전자문서는 다음과 같은 Context 정보를 이용하여 구분짓는다.

1) BusinessProcess : 해당 전자문서가 사용되는 비즈니스 프로세스 식별

(예:Order Process)

2) Region : 해당 전자문서가 사용되는 지역 범위 설정

(예:All in Contexts, KR, ASIA, JP 등)

3) Industry : 해당 전자문서가 사용되는 업종/부문 식별

(예:All in Contexts, Electronics, Textile 등)

1.1.12 이 지침과 규칙은 아래와 같은 사람들이 참고하기 위해 작성되었다.

a) 신규 XML 표준 전자문서로 등록하기 위하여 전자문서 초안을 개발하고자 하는 사람

b) 변경요청으로 기존 XML 표준 전자문서를 개정하고자 하는 사람

c) 심사평가를 하는 심사평가위원회

1.1.13 전자문서 개발은 두가지 유형의 전문가를 필요로 한다. : 전자문서를 실질적으로 사용하는 상업, 금융, 보험, 운송 등 특정한 사업 영역내 전문가와 전자문서 설계 원칙 및 비즈니스 및 정보모델링과 컴퓨팅 시스템을 이해하는 전문가가 필요하다.

1.2 용어와 정의

1.2.1 **EDI** : 국제간 또는 국내 기업간의 컴퓨터 통신을 통하여 표준화된 거래문서를 전자적으로 상호 교환하는 방식을 EDI라 하고 전자문서교환 또는 전자자료 교환으로 해석되며, 서로 다른 기업 또는 조직간에 표준화된 상거래 서식 또는 공공서식을 서로 합의한 통신표준에 따라 컴퓨터간에 교환하는 새로운 정보전달방식으로 정의할 수 있다. 다시 말해 EDI는 구조화된 형태의 데이터(Structured Format Data), 즉 표준전자문서를 컴퓨터와 컴퓨터간에 교환하여 재입력 과정없이 즉시 업무에 활용할 수 있도록 하는 새로운 정보전달방식이다.

1.2.2 **EDI 표준** : EDI표준은 서로 다른 거래당사자들간 또는 이들의 내부 업무시스템간에 전자문서교환이 이루어지게 하는 가장 핵심적인 역할을 수행한다. EDI표준은 용도에 따라 전자문서 표준(Formatting Standard) 또는 메시지 표준(Message Standard)과 통신표준(Communication Standard)으로 구분된다. 전자문서 표준은 전자적으로 전달될 수 있는 문서, 전자문서에 포함되는 정보, 정보의 순서와 형태, 정보의 각 부분의 의미 등에 대한 지침 등을 포함한다. 이에 비해 통신표준은 컴퓨터를 통한 전자문서의 송수신 규약으로 어떤 정보를 어떤 방식으로 전송할 것인가에 관한 사용자간의 합의를 말한다.

1.2.3 **전자문서** : 전자거래기본법에서는 전자거래를 재화나 용역(서비스)의 거래에 있어서 그 전부 또는 일부가 전자문서에 의하여 처리되는 거래로 정의한다. 전자상거래'라는 표현 대신에 '전자거래'라는 표현을 사용하는데, 그건 상행위가 아닌 공공의 거래도 포함하기 위한 개념이다. 이러한 전자거래에서 거래의 대상을 표현하는 문서를 전자문서라 하는데 전자문서는 "정보처리 시스템에 의하여 전자적 형태로 작성, 송수신 또는 저장된 정보"로 정의하고 있으며, UN/CEFACT에서는 "실제 비즈니스 거래와 관련된 식별자나 정보를 구조화되고 기능화된 방법으로 구성한 비즈니스 정보 개체들의 집합"이라고 정의한다.

1.2.4 **표준전자문서** : 표준 전자문서는 전자문서 표준 스펙안에 이미 정해놓은 전자문서로 전자문서 표준을 이루고 있는 디렉토리나 데이터 사전 등을 이용하여 메시지 설계 규칙에 맞게 비즈니스 정보를 체계적으로 구성해

놓은 메시지라고 할 수 있다. 이 표준 전자문서는 해당 비즈니스 거래에 필요한 비즈니스 정보를 망라하여 많은 사람들이 서브세트 형태나 확장하여 사용할 수 있도록 개발되어져 있다. 현재 표준 전자문서로는 국제표준으로는 유엔 표준전자문서(UNSM:UN Standard Message)가 있으며, ANSI X.12에서는 거래 세트(Transaction Set)가 있다. 아울러 국내에서도 국내에서만 표준적으로 사용할 수 있도록 국내 표준전자문서(KRSM)가 있다.

1.2.5 **EDI 소프트웨어** : 일반적으로 EDI 변환처리기로 알려져있는 EDI 소프트웨어의 기본적인 기능은 EDIFACT/EANCOM과 같은 전자문서표준에 의해 전송된 전자문서를 기업 내부의 파일형태로 변환하며, 반대로 외부로 전송하는 전자문서를 표준문서로 변환하는 일을 한다. 이와 더불어 EDI 소프트웨어 패키지는 다양한 전자문서표준과 전자문서 개정판을 변환할 수 있는 기능, 거래업체 프로필 관리, 타 응용프로그램간의 인터페이스, 직접 통신 또는 VAN사업자를 통한 통신 등을 지원하느 통신 모듈, 송신 또는 수신된 전자문서의 관리, 메뉴방식의 데이터 입력모듈 그리고 패스워드에 의한 보안이나 접근통제(Access Control)기능 등을 갖추고 있다. EDI 소프트웨어에 의해 기업내부 파일형태에서 표준 전자문서 형태로 데이터가 변환되면, 이러한 데이터는 의도하는 전자문서 수신자에게 실제 전송되어야 한다. 통신망을 이용한 데이터 전송 뿐만이 아니라, 비록 데이터를 테이프나 디스켓 같은 자기매체를 이용하여 전달하는 것도 이론적으로는 EDI의 일부분이다.

1.2.6 **B2B(Business To Business)** : 특정기업간의 CALS 및 EDI를 통한 수주, 구매, 조달 및 납품 등과 관련된 기업간 전자상거래로 각종 문서·양식·교환·처리 비용의 절감, 내부 업무처리방법·내용에 대한 표준화와 운영비용 절감 및 마케팅 및 영업 채널의 확대에 따른 이윤의 증대 등의 효과가 있다.

1.2.7 **XML(eXtended Markup Language)** : XML (eXtensible Markup Language)은 글자 뜻 그대로 해석하면 확장 가능한 마크업 언어라는 뜻이다. 마크업은 그 사용용도에 따라서 크게 3가지로 분류할 수 있는데 사용된 마크업이 콘텐츠의 화면표현과 관계된 정보의 마크업을 스타일 마크업이라고 한다. 또 HTML이나 일반적으로 문서작성기에서 흔히 사

용하는 테이블과 같이 사용자 데이터에 구조적인 정보를 표현하기 위한 마크업이면 구조적 마크업이라고 하며, 엘리먼트 컨텐츠의 의미를 기술하는 마크업의 경우는 의미론적인 마크업이라고 한다.

1.2.8 **코어 컴포넌트(Core Component)** : 코어 컴포넌트는 비즈니스 메시지를 구성할 때 이용되는 하나의 의미적 빌딩 블록이다. 코어 컴포넌트는 비즈니스 컨텍스트가 결합되어 다양한 곳에서 재사용될 수 있다. 코어 컴포넌트는 코어 컴포넌트 타입(Core Component Type), 기본 코어 컴포넌트(Basic Core Component), 복합 코어 컴포넌트(Aggregate Core Component) 등으로 구분된다.

1.2.9 **코어 컴포넌트 타입(Core Component Type)** : 그 자체로는 비즈니스적 의미를 가지고 있지 않은 핵심 컴포넌트들이다. 이것이 비즈니스 컨텍스트로서 재사용될 때, 이것은 기본 정보 엔터티가 된다. 예를들어 날짜(date) 자체는 아무런 비즈니스적인 의미가 없지만, 배송일자(delivery date), 계약일자(contract date)등은 비즈니스적 의미를 가진다.

1.2.10 **기본 코어 컴포넌트(Basic Core Component)** : 기본 코어 컴포넌트는 한 특정 비즈니스적인 의미를 정의(semantic definition)한 것의 일부인 어떤 단일 비즈니스 개념을 대표한다. 기본 코어 컴포넌트는 한 Core Component Type을 사용하여 구조되어 진다. 기본 코어 컴포넌트는 Aggregate Core Component를 개발하는데 사용된다.

1.2.11 **복합 코어 컴포넌트(Aggregate Core Component)** : 여러종류의 비즈니스 정보의 집합으로 하나의 단일 비즈니스 개념을 형성한다(예:Postal address). 각 ACC는 고유의 특정한 비즈니스 의미를 가진 정의(semantic definition)를 가지고 있으며, 또한 다음과 같은 것들을 포함한다.

(1) 두개 또는 그 이상의 Basic Core Component, 또는

(2) 최소 한 개의 Basic Core Component와 한 개 또는 그 이상의 Aggregate Core Component

1.2.12 **비즈니스 정보 개체(Business Information Entity)** : 코어 컴포넌트가 실 비즈니스 상황에 적용된다는 말은 코어 컴포넌트로 비즈니스 정보 개

체를 정의한다는 말이다. 즉 코어 컴포넌트가 특정 비즈니스 컨텍스트에 사용되면 비즈니스 정보 개체가 되기 때문에 코어 컴포넌트로부터 비즈니스 정보 개체가 정의된다고 볼 수 있다. 비즈니스 정보 개체란 하나의 유일한 비즈니스 의미적 정의를 갖는 비즈니스 데이터를 말한다. 이는 크게 기본 비즈니스 정보 개체와 집합 비즈니스 정보 개체로 나눌 수 있다

1.3 전자문서 표준의 활용

1.3.1 전자문서 정의와 관련된 개념은 주로 EDI(Electronic Data Interchange)에서 출발하였으며, EDI 자체를 전자문서 혹은 전자문서교환방식으로 인식하고 있다. 전자문서에 대한 정의로는 여러 가지로 나누어 볼 수 있는데, 전자거래기본법에서는 "정보처리 시스템에 의하여 전자적 형태로 작성, 송수신 또는 저장된 정보"로 정의하고 있으며, UN/CEFACT에서는 "실제 비즈니스 거래와 관련된 식별자나 정보를 구조화되고 기능화된 방법으로 구성한 비즈니스 정보 개체들의 집합"이라고 정의하고 있다. 이러한 전자문서는 표준의 범위에 따라 아래와 같이 구분될 수 있다.

국제표준 UN/EDIFACT (UNTDI+ANSI X12)
지역표준 ANSI X12 (ANSI표준, 미국)
국가표준 KEDIFACT (한국EDI 표준)
업종표준 TDCC (미국의 자동차업계 표준)
사설표준 K-MART (미국 수퍼체인회사내 표준)

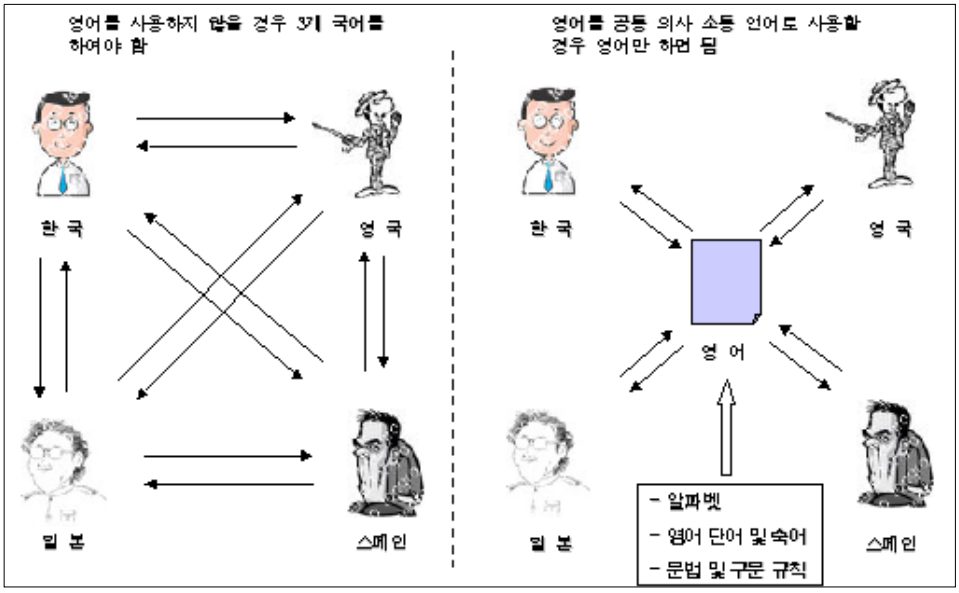
<그림> 전자문서 표준 범위에 따른 구분도

1.3.2 전자문서의 활용은 주로 기업간 거래(B2B)에서 많이 사용된다. 물론 기업 내부에서도 순차 파일이나 백업 파일로 많이 활용이 되지만, EDI에서 다루고 있는 전송, 저장단위로서의 전자문서는 기업 내부 시스템에서 거의 활용되지 않는다. 기업 내부에서만 사용되는 전자문서의 경우라면 전자문서 표준을 사용할 필요는 전혀 없다. 기업 내부에서 총괄적으로 정의된 포맷에 맞춰서 사용하면 되는 것이다. 그러나 기업간 거래에서는 전자문서 표준의 필요성이 요구된다. 각 기업에서 자체적으로 정의한 데이터 포맷이나 식별 코드, 전자문서의 구조 등이 서로 상이하기 때문에 각자의 시스템에서 이러한 상이한 전자문서를 처리하기가 힘들기 때문이다. 그래서 전자문서 표준화에 대한 요구가 각 기업별로 높아지면서 선도적인 업체에서 자신이 관리하는 공급망에 대해서 표준포맷을 정의하고, 운송업종에서는 업종내의 모든 회사에서 공통적으로 송수신할 수 있는 전자문서 표준을 제정하기에 이르렀다. 이러한 자발적인 표준화 노

력이 모아지면서 먼저 북미를 중심으로 ANSI X.12라는 EDI 표준을 개발하였으며, 이후 유럽에서도 이와 비슷한 EDI 표준을 만들고, 1987년 UN 산하의 WP.4위원회(나중에 UN/CEFACT로 개칭)에서 UN/EDIFACT라는 국제 표준을 제정하였다.

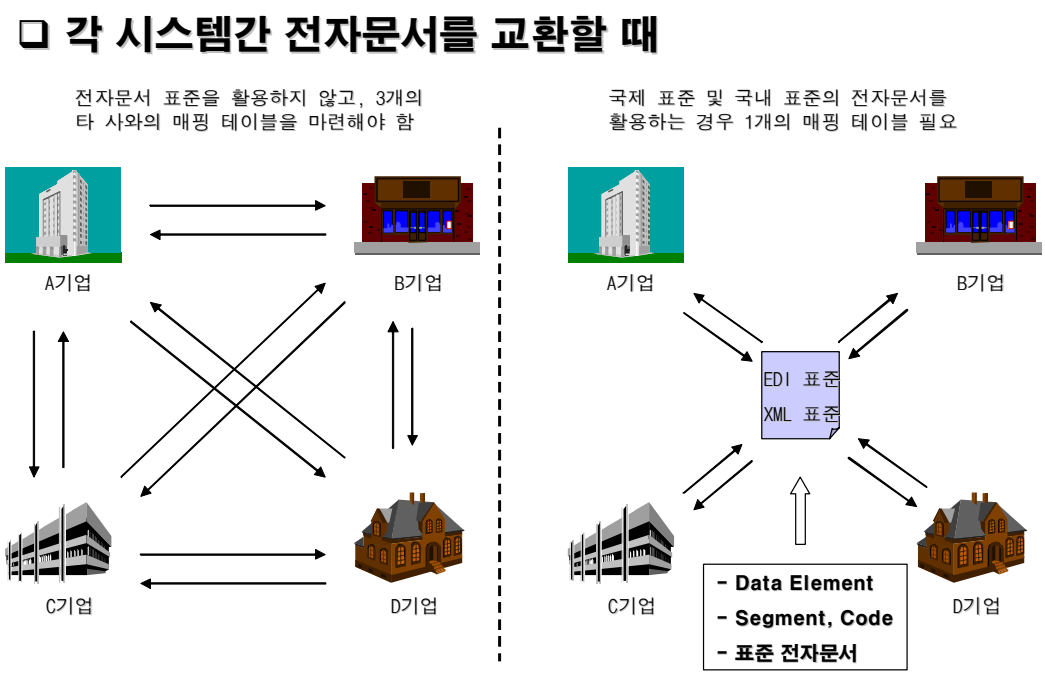
1.3.3 전자문서 표준에 대한 개념을 보다 더 쉽게 이해하기 위해서 아래와 같은 그림을 보면 쉽게 알수 있다. 만약 한국사람이 국내에서 한국 사람들끼리 회의를 할 경우에는 한국어로만 사용하여도 회의 진행에 아무런 지장이 없지만, 국제회의장에서 다른 국가들의 대표와 서로 회의를 하고자 할 경우, 공통으로 사용하는 언어가 없다면 한국 대표는 각 국가의 언어를 알아야 할 것이다. 그러나 만약에 영어와 같이 공통으로 활용할 수 있는 언어가 있다면, 영어만 알더라도 여러 국가의 대표들과 충분히 의사소통을 할 수 있을 것이다.

1.3.4 여러 언어로 의사소통할 때 : 여러 언어로 의사소통할 때 아래 그림에서 보는바와 같이 각 시스템 내부에서는 시스템 자체 정의된 포맷으로도 충분히 시스템 운영에 문제가 없지만, 외부의 시스템이나 기업간의 거래를 위해서는 공통의 전자문서 표준을 사용하여야 효율성이 극대화 될 것이다.



<그림> 언어를 비유한 전자문서 표준 개념도

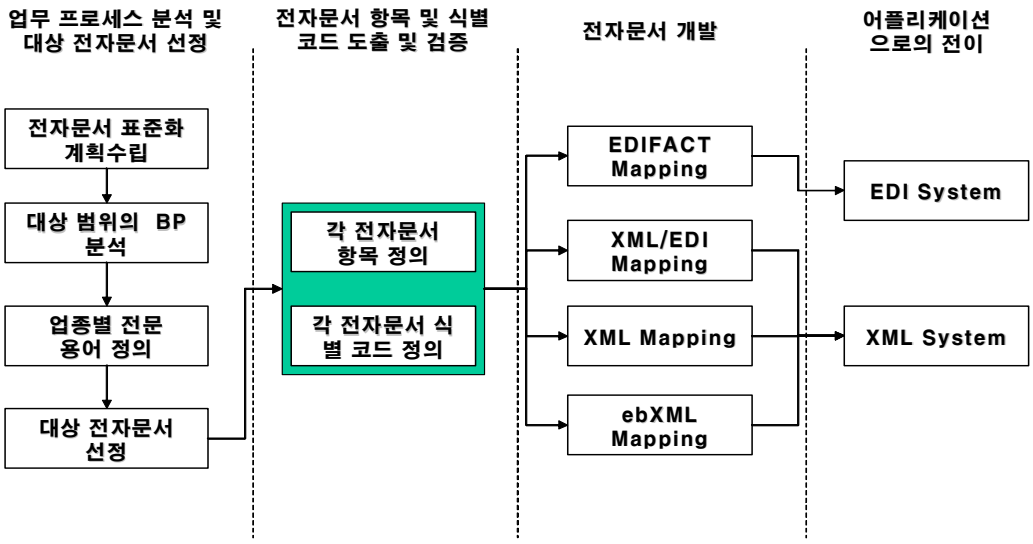
1.3.5 각 시스템간 전자문서를 교환할 때 : 아래의 그림에서 보는 바와 같이 전자문서 표준은 거래 당사자간 비즈니스를 수행하기 위한 서로간의 비즈니스 언어(Business Language)이다. 서로 다른 시스템이 대화하기 위해서는 통신 프로토콜이 서로 맞아야 한다. 이러한 프로토콜은 여러 사람들이 공통으로 합의한 규약, 규칙이며, 이 규칙에 어긋날 경우 서로간에 대화는 되지 않을 것이다. 이러한 통신 프로토콜의 일치를 기반으로 비즈니스 정보를 주고 받을 경우에도 정보와 관련된 서로간의 공통 규약이나 규칙을 준수하여야 하며, 이의 형태가 표준 전자문서의 유통이다.



<그림> 기업들간 거래에 있어서의 전자문서 표준 개념도

1.4 표준 전자문서 개발 절차 및 고려 사항

1.4.1 표준 전자문서 개발과 관련한 절차 및 방법은 각기 전자문서 개발자마다 다를 수 있으나 거의 대부분 유사한 과정을 거친다. 여기서는 전자문서 개발과 관련하여 크게 4단계로 나누었는데 아래 그림에서 볼 수 있듯이 업무 프로세스 분석 및 대상 전자문서를 선정하는 단계에서부터 출발하여, 전자문서 항목 및 식별코드 도출 및 검증과정을 거쳐서, 전자문서 표준 스펙에 맞게 구문(Syntax)을 할당하는 단계로 이러한 구문은 EDIFACT가 될수 있으며, ANSI X.12나 XML 구문으로 표현될 수도 있을 것이다. 마지막 단계로서는 어플리케이션으로의 전이단계인데, 여기서는 EDI시스템 혹은 기타 어플리케이션으로 개발한 전자문서를 구현하는 단계이며, 전자문서 개발자의 정확한 의도가 전달되어 정확하게 시스템내에서 문서의 구성이 되도록 보조해 주는 역할을 하게 된다. 각각의 단계별로 계속 검증이 이루어지는 것이 또한 중요하다.



<그림> 표준전자문서 개발절차

1.4.2 **업무 프로세스 분석 및 대상 전자문서 선정 단계** : 전자문서를 개발하기 위한 첫 번째 단계는 해당 분야의 업무 프로세스와 업무 요구사항을 분석하여, 필요한 전자문서가 무엇인지를 정확하게 밝히는 단계이다. 이는 개발할려고 하는 전자문서를 선정과 전자문서의 항목 등을 도출하는데 도움이 되며, 전자문서 개발의 타당성과 효율성을 제시하는 근거가 된다. 따라서 업무 프로세스분석 및 대상 전자문서 선정단계에서 검토되어야 할 항목으로 해당 분야의 공통업무 및 특성유형의 업무프로세스

분석, 업종내 전문용어의 정의, 업무요구사항분석, 대상전자문서의 선정으로 구분하여 검토되어야 한다.

1.4.2.1 해당 분야의 공통 업무 및 특정 유형의 업무 프로세스 분석 : 해당 분야 내에서 주요하게 이루어지는 업무 프로세스를 분류하고, 공통적으로 이루어지는 공통 업무 프로세스와 그 분야에서만 수행되는 특정 유형의 업무 프로세스를 분석하고, 업종내 각 업체별로 약간씩 다른 업무 환경 및 특성에 따라 달리 나타나는 업무 플로우 및 프로세스를 분석한다.

○ 해당 분야의 전체적인 워크 플로우 작성

○ 작성된 워크 플로우를 바탕으로 해당 분야의 표준 업무 프로세스 작성

○ 업무 프로세스 분석은 해당 분야의 업무 영역에서부터 가장 밑단의 비즈니스 거래단위까지 이루어져야 함

1.4.2.2 해당 분야 전문 용어 정의 : 해당 분야내 전문 용어 정의는 해당 분야 내에서 의미는 같지만, 업체마다 각기 다르게 부르거나 표시하는 용어에 대해서 통일하는 단계로 전자문서 표준화를 위한 기반 구축이다.

○ 해당 분야에서 사용되는 전문 용어 수집 및 분류

○ 해당 분야 전문 용어사전 편집

1.4.2.3. 업무 요구사항 분석 : 해당 분야의 실무 담당자들의 현재의 업무 절차에 대한 개선 사항 및 변경요구사항을 수집, 분석하는 단계이다.

○ 실무 담당자들에 대한 설문 조사 및 인터뷰

1.4.2.4. 대상 전자문서 선정 : 앞에서 분석한 업무 프로세스 분석 및 요구사항 분석을 거친 이후, 개발하여야 할 대상 전자문서를 선정하는 단계이다.

○ 업무 거래별 대상 전자문서 후보 선정

○ 전자문서화 할 경우의 기대 효과 분석

○ 선정된 대상 전자문서에 대한 실무 담당자의 검증

1.4.3 **전자문서 항목 및 코드 도출 단계** : 앞에서 도출된 대상 전자문서에 대한 항목과 항목에 대한 정의와 속성, 그리고 식별 코드를 도출해내는 단계이다. 이 부분은 전자문서를 구성하는 항목을 도출하고 정리하는 단계로, 전자문서의 내용을 구성하는 단계라 할 수 있다. 따라서 전자문서 항목 및 코드 도출단계에서는 전자문서의 항목을 선정하고 선정된 항목에 대한 정의 및 속성을 부여하는 단계이다.

1.4.3.1 전자문서의 항목 선정 : 전자문서를 구성하는 데이터 항목들을 선정하는 단계로 전자문서 개발과정에서 가장 중요한 단계이다.

○ 오프라인 업무에서 사용하는 업무 서식 수집, 분류

○ 대상 문서의 디자인(Option)

○ 해당 분야 전자문서 항목 마스터 테이블 마련

○ 각 대상 전자문서별 항목 테이블 마련

○ 리스트된 전자문서 항목에 대한 실무담당자 검증

1.4.3.2. 전자문서 항목에 대한 정의 및 속성 부여 : 선정된 전자문서 항목에 대해 명확한 정의와 속성을 부여하는 단계로 데이터 항목의 경우에는 항목에 대한 명확한 정의와 길이, 필수/선택, 비고 등을 기술하며, 식별코드의 경우에는 명확한 식별 코드리스트를 추가적으로 정의하여야한다.

○ 각 전자문서 항목에 대한 명확한 정의 및 속성 기술

○ 식별코드 항목에 대한 명확한 코드리스트 정의

1.4.4 **전자문서 개발 단계** : 전자문서를 구성하는 항목과 식별코드가 모두 개발이 되고, 검증이 된 이후에는, 개발하고자 하는 시스템의 사용범위에 따라서 국제/지역 전자문서 표준 혹은 업종별 전자문서 표준이나 국내 전자문서 표준에 맞춰서 전자문서 개발이 이루어져야 한다. 따라서 전자

문서개발단계에서는 전자문서 표준 및 규격에 대한 활용, 표준전자문서 유형의 재사용, 신규표준전자문서의 개발로 구분하는 단계이다.

1.4.4.1 전자문서 표준/규격의 활용 : 국내 혹은 국제적 유통 목적의 전자문서를 개발하고자 할 때는 이미 국제적으로 용인된 전자문서 표준이나 규격에 맞춰서 전자문서를 설계하여야 한다. 이런 전자문서 표준/규격에 맞게 거래를 하여야 거래 상대방간 시스템 연동에 문제가 없다

- ○ 시스템에 적용할 전자문서 표준/규격 선정
- ○ 도출된 대상 전자문서에 해당하는 표준 전자문서 선정

1.4.4.2 표준 전자문서 유형의 재사용 : 선정된 전자문서 표준/규격에는 이미 기 개발된 표준 전자문서 유형이 있다. 개발 대상 전자문서가 표준 전자문서와 중복될 경우에는 표준 전자문서 유형을 그대로 활용하는 것이 바람직하다. 예를 들면, UN/EDIFACT나 ANSI X.12와 같은 전자문서 표준에는 표준 전자문서유형이나 거래 세트등이 존재한다. UN/EDIFACT의 경우에는 현재 190여종의 표준전자문서(UNSM)가 있으며, ANSI X.12에는 280여종의 표준 거래세트(Transaction Set)가 있다. UN/EDIFACT 구현지침서를 보면, 이러한 문서 표준의 구조나 내용을 변경하지 않고 그대로 사용할 것을 권고한다. 그래서 기존 표준 전자문서를 재사용하기로 결정이 됐다면, 앞 단계에서 도출된 전자문서 항목 및 식별코드를 표준 전자문서의 각 부분에 매핑하는 작업을 하기만 하면 된다.

- ○ 전자문서 표준/규격에서 활용 가능한 표준 전자문서 선정
- ○ 대상 전자문서와 표준 전자문서와의 항목 비교 및 매칭
- ○ 대상 전자문서에서 활용하는 식별코드 리스트와 전자문서 코드리스트의 비교 및 매칭
- ○ EDIFACT와 ANSI X.12의 경우에는 MIG(Message Implementation GuideLine)를, XML의 경우에는 DTD 혹은 스키마를 개발

1.4.4.3. 신규 표준 전자문서의 개발 : 개발 대상 전자문서에 대한 메시지 유형

이 표준 전자문서 스펙에 없을 경우, 신규로 개발하여야 한다. 신규로 표준 전자문서를 개발하는 경우에 대한 주요 내용을 수록하였다. 이 경우 국내에서는 자체적으로 개발하여 그냥 사용하는 경우가 많은데, 국제간 거래를 위한 전자문서일 경우에는 국제 표준 관리기관에 신규 메시지 개발 요청을 하여야 하며, 국내간 거래를 위한 전자문서일 경우에는 전자거래진흥원에 신규 메시지 개발 요청을 하여야 한다.

- ○ 전자문서 표준관리 기관에 신규 메시지 개발 요청
- ○ 전자문서 표준관리 기관과 협의하여 공동으로 메시지 개발
- ○ 해당 업종과 관련된 업종 표준 현황 조사
- ○ 해당 전자문서와 유사한 국내/국외 전자문서 사용 실태 조사
- ○ EDIFACT와 ANSI X.12의 경우에는 MIG(Message Implementation GuideLine)를, XML의 경우에는 DTD 혹은 스키마를 개발

1.4.5 **어플리케이션으로의 전이 단계** : 전자문서를 개발하였다면 이후 그 전자문서를 가지고 시스템이나 어플리케이션에 전이하는 단계에 접어 들게 된다. 어플리케이션 개발자나 시스템 개발자는 개발된 전자문서를 가지고 화면 인터페이스 및 메시지를 개발하는데, 이 과정에서 전자문서의 활용과 관련된 정확한 이해와 구현이 필요하다. 따라서 전자문서의 어플리케이션에 대한 검증이 필요하며 활용에 중점을 둔 단계이다.

1.4.5.1 전자문서에 대한 어플리케이션의 검증 : 개발된 전자문서는 전자문서 개발자에 의해 개념적으로 만들어 진 것이다. 이를 어플리케이션이나 시스템 개발자에 의해서 물리적으로 개발되는 과정이 바로 이 단계인데 이 과정에서 전자문서의 구조와 내용에 대한 검증이 이루어 져야 하며, 표준 전자문서 스펙과 같은지 등을 검증하며, 전자문서가 가지고 있는 여러 값이나 코드 등에 대한 유효성 체크를 어떤 방식으로 할 것인지에 대한 적용 문제도 검토하여야 한다.

- ○ 개발된 전자문서의 구현방법과 관련해서 어플리케이션/시스템 개발자와 협의
- ○ 개발된 전자문서에 대한 어플리케이션 검증과 수정

2. 가이드라인과 관련된 일반적 지침

2.1 XML 태그명명 규칙

2.1.1 XML 태그명명 규칙은 기본적으로 ISO 11179 표준을 기반으로 하여 작성한다.

2.1.2 XML 태그명은 태그명의 정의를 통해서 유도되어져야 한다.

2.1.3 XML 태그명은 크게 기본 정보 개체 레벨과 복합 정보 개체 레벨로 나눌 수 있으며, 기본 정보 개체 레벨은 하나의 의미만을 가지고 있는 정보 단위로 EDIFACT의 경우에는 엘리먼트(Data Element)에 해당한다. 복합 정보 개체 레벨은 복합적인 의미를 지닌 객체를 표현할 때 사용하는 정보 단위로 EDIFACT의 경우에는 복합 데이터 엘리먼트(Composite Data Element) 혹은 세그먼트(Segment)에 해당한다.

2.14 복합 정보 개체명은 (한정어)객체명으로만 구성한다. 그리고 복합 정보 개체를 규정하는 복합 정보 개체 유형에 대한 명명은 (한정어)객체명 + . + Details 로 구성한다.

a) 복합 정보 개체에 표현용어는 필요없다.

b) 한정어는 선택사항으로 객체명을 한정할 경우에 사용하며, 생략하여도 무방하다.

2.1.5 기본 정보 개체명은 크게 (한정어)객체명, (한정어)속성명, 표현용어 등으로 구성한다. 기본 정보 개체명을 규정하는 기본 정보 개체 유형은 코어 컴포넌트 타입으로 표현용어를 규정한다.

a) 한정어는 선택사항으로 객체나 속성명을 한정할 경우에 사용하며, 생략하여도 무방하다.

b) 객체명은 하나 이상의 데이터 엘리먼트가 속해있는 논리적 데이터 그룹 혹은 집합을 표현한다.

c) 속성명은 객체가 지니고 있는 여러 가지의 속성을 표현한다.

d) 표현 용어는 한 정보 객체가 가지고 있는 값들의 형태를 표현한다.

e) 표현 용어는 XML 라이브러리에서 규정된 표현용어만을 사용하여야 한다.

2.1.6 객체명과 속성명은 여러 개의 단어로 구성될 수 있다. 이때, 단어와 단어 사이에는 공백을 두지 않는다.

2.1.7 엘리먼트명은 각 단어별 가장 첫 자리는 대문자로 시작하며, 어트리뷰트명은 소문자로 시작한다.

2.1.8 객체명, 속성명, 표현용어는 "."로 구분한다.

2.1.9 표현용어는 축약된 형태로 기술하여서는 안되며, 반드시 규정된 표현형 그대로 사용되어야 한다.

2.1.10 XML 라이브러리에서 관리하는 XML 태그명은 중복되어서는 안되며, 유니크(Unique)하여야 한다.

2.1.11 XML 태그는 기본적으로 단수형을 띠어야 한다. 그러나 비즈니스 컨텍스트가 결합되어 루핑(Looping)이 필요한 부분에서는 복수형을 사용하여 여러 다양한 개념의 단수형을 안에 포함할 수 있다.

2.1.12 XML 태그는 기본적으로 동사, 명사, 형용사만 포함한다. 태그명에 약어나 두문자어가 사용될 경우에 정의부분에 반드시 그 약어나 두문자어가 뜻하는 의미를 설명하여야 한다.

2.2 UID 할당 규칙

2.2.1 UID는 XML과 관련된 정보 개체들을 식별하기 위하여 일관된 규칙으로 할당하는 식별자이다. UID를 부여하기 위해서는 다음과 같은 사항들을 고려하여야 한다.

(1) 향후에 정보 개체들이 계속적으로 추가되거나 삭제되는 것을 고려하여야 한다.

(2) 전자거래진흥원을 식별할 수 있는 식별자가 있어야 한다.

(3) 쉽게 인식할 수 있어야 한다.

(4) 각 계층별로 구분되어 질 수 있어야 한다.

2.2.2 아래와 같이 계층적으로 UID를 할당하고자 한다.

	UID 할당 범위
기본 정보 개체 표현 유형	KIEC000100 ~ KIEC0999000
기본 코어 컴포넌트 정보 개체	KIEC100100 ~ KIEC1999000
기본 비즈니스 정보 개체	KIEC200100 ~ KIEC2999000
복합 코어 컴포넌트	KIEC300100 ~ KIEC3999000
복합 비즈니스 정보 개체	KIEC600100 ~ KIEC6999000
복수 복합 정보 개체	KIEC500100 ~ KIEC5999000
메시지	KIEC700100 ~ KIEC7999000

2.2.3 UID 할당과 관련해서는 가장 기본적으로 각 계층별로 영어의 알파벳순으로 매기는 것으로 하며, 새로운 정보 개체가 추가될 경우에는 그 새로운 정보 개체의 앞과 뒤의 UID의 중간 번호를 할당한다.

2.2.4 복합 비즈니스 정보 개체의 경우에는 참조하고 있는 복합 코어컴포넌트를 기준으로 UID를 순차적으로 할당한다.

2.2.5 메시지에 대한 UID는 국내 표준 전자문서의 경우, 비즈니스 프로세스별로 1차적으로 구분하고, 100단위로 순차적으로 할당하며, 국내 업종별(Vertical) 관리 표준의 경우에는 해당 메시지번호에서 5씩 순차적으로 할당한다.

2.2.6 신규 정보 개체나 컴포넌트를 개발하거나 기존의 컴포넌트를 수정하여 UID를 부여하고자 할 경우에는 반드시 전자거래진흥원에 요청하여 적정한 UID를 할당받아야 한다.

2.2.7 XML과 관련된 정보 개체나 컴포넌트에 있어서 UID는 중복되어서는 안되며, 반드시 유니크(Unique)하여야 한다.

2.3 네임 스페이스 활용 규칙

2.3.1 기본 이름공간을 설정할 때에는 다음과 같은 세 가지 방법을 사용할 수 있다.

1) 기본 이름공간으로 XMLSchema를 사용하고, targetNamespace를 한정
2) 기본 이름공간으로 targetNamespace를 사용하고, XMLSchema를 한정
3) 기본 이름공간을 사용하지 않고, targetNamespace와 XMLSchema를 모두 한정

2.3.2 다음은 <1안>을 사용하는 XML 스키마의 예를 보여주고 있다.

```
<?xml version="1.0"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
                targetNamespace="http://www.library.org"
                xmlns:lib="http://www.library.org"
                elementFormDefault="qualified">
    <include schemaLocation="BookCatalogue.xsd"/>
    <element name="Library">
        <complexType>
            <sequence>
                <element name="BookCatalogue">
                    <complexType>
```

```
                    <sequence>
                        <element ref="lib:Book"
                                        maxOccurs="unbounded"/>
                    </sequence>
                </complexType>
            </element>
        </sequence>
    </complexType>
</element>
</schema>
```

2.3.2.1 이 예에서는 XMLSchema가 기본 이름공간으로 사용되고 있으므로, 스키마를 구성하는 데 사용되는 모든 구성 요소(schema, element, complexType, sequence 등)에는 이름공간 한정자가 없다. 그러나 targetNamespace의 구성 요소(Library, BookCatalogue, Book 등)를 참조하기 위해서는 lib라는 이름공간 접두사를 사용해야 하는데, 여기서는 lib:Book에 대한 참조를 볼 수 있다.

1) 장점 : 스키마가 여러 이름공간의 구성 요소를 참조하는 경우 이 방법을 사용하면 항상 참조를 한정하므로 구성 요소를 일관성 있게 참조할 수 있게 된다.

2) 단점 : targetNamespace가 없는 스키마는 XMLSchema 구성 요소가 한정되도록 설계해야 한다. 이 방법을 적용하여 스키마를 설계하면 일부 스키마에서는 XMLSchema 구성 요소를 한정하고 다른 스키마에서는 XMLSchema 구성 요소를 한정하지 않게 되어 혼동을 일으킬 수 있다.

2.3.3 다음은 <2안>을 사용하는 XML 스키마의 예를 보여주고 있다.

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                    targetNamespace="http://www.library.org"
                    xmlns="http://www.library.org"
                    elementFormDefault="qualified">
    <xsd:include schemaLocation="BookCatalogue.xsd"/>
    <xsd:element name="Library">
        <xsd:complexType>
```

```
            <xsd:sequence>
                <xsd:element name="BookCatalogue">
                    <xsd:complexType>
                        <xsd:sequence>
                            <xsd:element ref="Book" maxOccurs="unbounded"/>
                        </xsd:sequence>
                    </xsd:complexType>
                </xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
</xsd:schema>
```

2.3.3.1 이 예에서는 스키마를 구성하는데 사용되는 모든 구성 요소가 xsd:를 사용하여 한정된 이름공간으로 표현되고 있다. 또한, targetNamespace를 기본 이름공간으로 선언하고 있으므로, targetNamespace의 구성 요소에 대한 모든 참조는 한정된 이름공간이 아니며, 따라서 Book에 대한 참조도 한정된 이름공간이 아니다.

1) 장점 : targetNamespace가 없는 스키마는 XMLSchema 구성 요소가 한정되도록 설계해야 한다. 이 방법은 스키마에 targetNamespace가 있는지 여부와 관계없이 작동하므로, 이 방법을 사용하면 XMLSchema 구성 요소의 이름공간을 항상 한정하므로 일관성 있게 스키마를 설계할 수 있다.

2) 단점 : 스키마가 여러 이름공간의 구성 요소를 참조하는 경우에 일부 참조에 대해서는 이름공간을 한정하지만 다른 경우(예: targetNamespace의 구성 요소를 참조하는 경우)에는 그렇지 않게 된다. 구성 요소를 참조할 때 이름공간 한정자를 다르게 사용하므로 혼동을 일으킬 수 있다.

2.3.4 다음은 <3안>을 사용하는 XML 스키마의 예를 보여주고 있다.

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                    targetNamespace="http://www.library.org"
```

```
                    xmlns:lib="http://www.library.org"
                    elementFormDefault="qualified">
    <xsd:include schemaLocation="BookCatalogue.xsd"/>
    <xsd:element name="Library">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="BookCatalogue">
                    <xsd:complexType>
                        <xsd:sequence>
                            <xsd:element ref="lib:Book" maxOccurs="unbounded"/>
                        </xsd:sequence>
                    </xsd:complexType>
                </xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
</xsd:schema>
```

2.3.4.1. 이 예에서는 targetNamespace의 구성 요소에 대한 참조뿐만 아니라 XMLSchema 구성 요소 모두를 명시적으로 한정하고 있다.

1) 장점 : <1안>과 <2안>의 장점을 모두 가지고 있다.

2) 단점 : 이름공간을 한정하여 모든 구성 요소와 참조가 명시적으로 보이므로 스키마를 읽을 때 혼란스러울 수 있다.

2.3.5 기존 전자문서의 사례

2.3.5.1 <1안> 사용 예

- ebXML

```
<schema targetNamespace="http://www.ebxml.org/namespaces/tradePartner"
        xmlns:xml="http://www.w3.org/XML/1998/namespace"
        xmlns="http://www.w3.org/2000/10/XMLSchema"
        xmlns:tns="http://www.ebxml.org/namespaces/tradePartner"
        xmlns:xlink="http://www.w3.org/1999/xlink"
        xmlns:xsi="http://www.w3.org/2000/10/XMLSchema-instance"
        xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
```

2.3.5.2 <2안> 사용 예

- OAGIS 8.0

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           targetNamespace="http://www.openapplications.org/oagis"
           xmlns="http://www.openapplications.org/oagis">
```

- xCBL 3.5

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
           targetNamespace="rrn:org.xcbl:schemas/xcbl/v3_5/xcbl35.xsd"
           xmlns="rrn:org.xcbl:schemas/xcbl/v3_5/xcbl35.xsd">
```

- OTA

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           targetNamespace="http://www.opentravel.org/OTA/2002/08"
           xmlns="http://www.opentravel.org/OTA/2002/08">
```

2.3.6 개발자의 설계 방식에 따라 1안과 2안은 각기 장단점을 가질 수 있기 때문에 본 가이드라인에서는 1안과 2안 모두 권고한다. 비록 최근 XML 스키마 설계 방향이 2안 방식으로 많이 진행되고 있어서 본 XML 라이브러리에서는 2안으로 네임스페이스를 규정하였지만, 스키마 전체 구조에 1안이 적합할 경우, 1안도 충분히 활용할 수 있다.

2.4 메시지/컴포넌트 확장 규칙

2.4.1 본 가이드라인에서는 메시지/컴포넌트 확장 규칙으로 크게 3가지 규칙을 규정하고 있다. 2가지 규칙은 OAG에서 제시하는 UserArea를 이용하는 방법과 Overlay 확장 규칙을 이용하는 것이며, 다른 1가지 규칙은 ebXML 코어 컴포넌트에서 규정한 Constraint Language에 있는 ContextRule 규칙을 활용하는 것이다. 이 ContextRule 규칙은 스키마 외부에서 스키마가 확장된 형태를 규정한 것으로 지금 당장 구현되기는 힘들며, ContextRule 규칙을 수용할 수 있는 ebXML Compliant 소프트웨어가 나와야 실질적으로 구현될 것이다. 물론 지금 당장은 OAG에서 제시하는 UserArea 영역을 확장하면서 외부에 있는 ContextRule 문서에 UserArea 확장 부분을 기술할 수 있을 것이다.

2.4.2 이러한 스키마 확장의 원칙은 다음과 같다.

① 교환 상대방과 쉽게 공유될 수 있고
② 별다른 추가 노력 없이도 버전 업그레이드를 수용할 수 있어야 함

또한, 스키마 확장의 설계 목표는 다음과 같다.

- 사용자 정의 확장은 핵심 XML 스키마 화일 중 어느 것도 수정할 필요가 없어야 하며 그에 의존해서도 안 됨
- 현재 표준에 대한 어떠한 확장도 현재 표준 이외의 이름공간에서 이루어져야 함

2.4.3 업종이나 부문별 표준의 경우에는 Overlay 확장을 권고한다. UserArea 요소를 많이 활용할 경우 업종 표준을 관리하기 힘들기 때문에 최대한 지양하며, 더 이상 확장할 필요가 없는 업체에서 업종 이나 부문별 표준을 이용하여 업체 전자문서를 개발하고자 할 경우에는 UserArea 요소를 부분적으로 사용하도록 한다.

2.4.4 **UserArea 요소를 통한 확장** : 확장할 내용을 별도로 분리되어 있는 UserArea 요소의 자식 요소로 추가하여 확장하며, UserArea 요소는 컴포넌트의 마지막 요소로 와야 한다.

- XML 스키마

```
<xs:complexType name="UserArea" block="restriction">
  <xs:sequence>
    <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

- 현재 표준에 있는 요소를 확장하는 XML 문서(동일한 이름공간 사용)

```
<UserArea>
  <SalesPerson>…</SalesPerson>
<UserArea>
```

- 현재 표준에 없는 요소를 확장하는 XML 문서(다른 이름공간 사용)

```
<UserArea>
  <SalesPerson>…</SalesPerson>
  <ai:VehicleWarrantyExpirationDate>
  …
  </ai:VehicleWarrantyExpirationDate>
<UserArea>
```

2.4.5 **오버레이 확장** : UserArea 요소로 분리되어서는 안 되는 확장 내용을 있을 경우는, 현재 표준에 있는 요소와 동격인 요소로 확장한다. 확장한 요소는 특정 요소나 타입의 가장 마지막에 위치하도록 한다.

① 새로운 이름공간, 예를 들어, “myns“ 내에서 OAGIS ”Invoice“ 타입(oa:Invoice)에 기반하여 확장한 전역 타입 하나를 다음과 같이 생성한다.

```
<xs:complexType name="Invoice">
  <xs:complexContent>
   <xs:extension base="oa:Invoice"/>
  </xs:complexContent>
</xs:complexType>
```

② 이 확장 중에 사용자가 원하는 만큼의 내용을 추가하게 되는데, 예를 들어 GrandTotal이라는 요소를 추가한다.

```
<xs:complexType name="Invoice">
  <xs:complexContent>
   <xs:extension base="oa:Invoice">
     <xs:sequence>
       <xs:element name="GrandTotal" minOccurs="0"/>
     </xs:sequence>
   </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

③ 이 새로운 이름공간에서 전역 요소 “Invoice"를 정의하고 이를 앞에서 만든 새로운 타입과 바인딩한 후, OAGIS Invoice 대체 그룹에 속하도록 정의한다.

```
<xs:element name="Invoice" type="myns:Invoice"
        substitutionGroup="oa:Invoice"/>
```

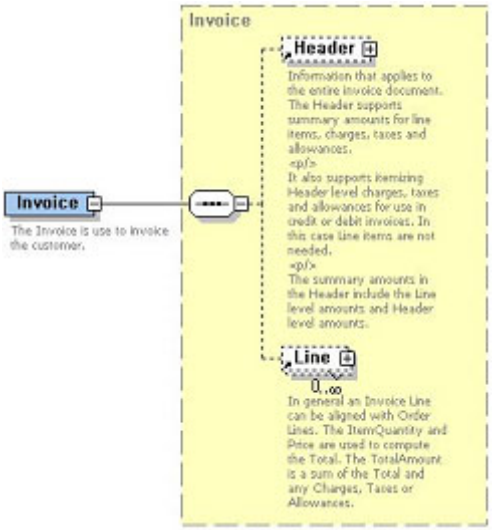
④ 이렇게 정의된 XML 스키마에 기반하여 생성된 실제 XML 인스턴스 문서 ProcessInvoice BOD는 다음과 같다.

```
<ProcessInvoice xmlns="http://www.openapplications.org/oagis"
                    xmlns:myns="http://www.my.net/myns"
                    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                    xsi:schemaLocation="http://www.my.net/myns
                                        ../BODs/ProcessInvoice.xsd"
                    revision="8.0" environment="Production" lang="en-US">
 <ApplicationArea>
   …
 </ApplicationArea>
 <DataArea>
   <Process/>
   <myns:Invoice>
     <Header>  …  </Header>
       <Line>   …   </Line>
     <myns:GrandTotal>1200000.00</myns:GrandTotal>
   </myns:Invoice>
 </DataArea>
</ProcessInvoice>
```

2.4.6 오버레이 확장의 예

- 현재 표준

```
<Invoice>
  <Header>…</Header>
  <Line>…</Line>
</Invoice>
```



현재 표준에 대해 이름공간이 ia인 회사에서 <Invoice> 요소의 자식으로 <GrandTotal> 요소와 <Header> 요소의 자식으로 <TimeCard> 요소를 확장한

- XML 스키마

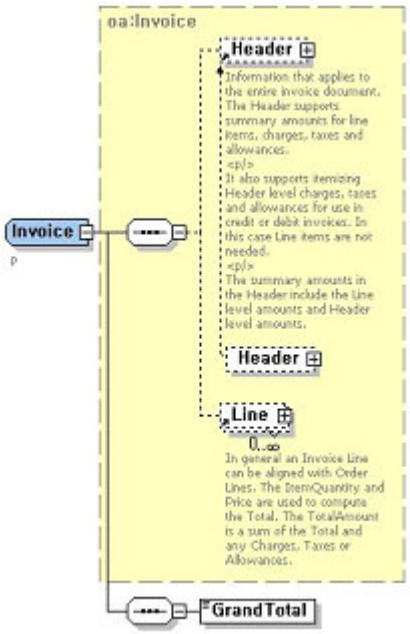
```
<xs:schema targetNamespace="http://www.oagi.net/oagis/ia"
           xmlns:ia="http://www.oagi.net/oagis/ia"
           xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns:oa="http://www.openapplications.org/oagis"
           elementFormDefault="qualified"
           attributeFormDefault="unqualified">
  <xs:import namespace="http://www.openapplications.org/oagis"
             schemaLocation="../../../../OAGIS/Resources/Nouns/Invoice.xsd"/>
  <xs:include schemaLocation="../IndustryA.xsd"/>
  <xs:include schemaLocation="TimeCard.xsd"/>
  <xs:complexType name="Invoice">
    <xs:complexContent>
      <xs:extension base="oa:Invoice">
        <xs:sequence>
          <xs:element name="GrandTotal" type="xs:string"/>
```

```
            </xs:sequence>
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="InvoiceHeader">
        <xs:complexContent>
          <xs:extension base="oa:InvoiceHeader">
            <xs:sequence>
              <xs:element name="TimeCard" type="ia:TimeCard"/>
            </xs:sequence>
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>
      <xs:element name="Invoice" type="ia:Invoice" substitutionGroup="oa:Invoice"/>
      <xs:element name="Header" type="ia:InvoiceHeader"
                  substitutionGroup="oa:Header"/>
  </xs:schema>
```

위의 스키마에서는 oa:Invoice를 기반으로 GradTotal을 추가한 ia:Invoice 타입을 정의하고, oa:InvoiceHeader를 기반으로 TimeCard를 추가한 ia:InvoiceHeader 타입을 정의한 다음, 전역 요소 Invoice를 ia:Invoice 타입의 oa:Invoice의 대체그룹으로 지정하고, Header를 ia:InvoiceHeader 타입의 oa:Header의 대체그룹으로 지정한다.

- <GrandTotal> 요소를 확장한 XML 인스턴스 문서

```
<ia:Invoice>
  <ia:Header>
    …
    <ia:TimeCard/>
  </ia:Header>
  <Line>…</Line>
  <ia:GrandTotal currency="USD">43000.00</ia:GrandTotal>
</ia:Invoice>
```



- 위의 확장에 대해 이름공간이 xyz인 회사에서
  <TotalDiscountsThisInvoice> 요소를 확장한 XML 스키마

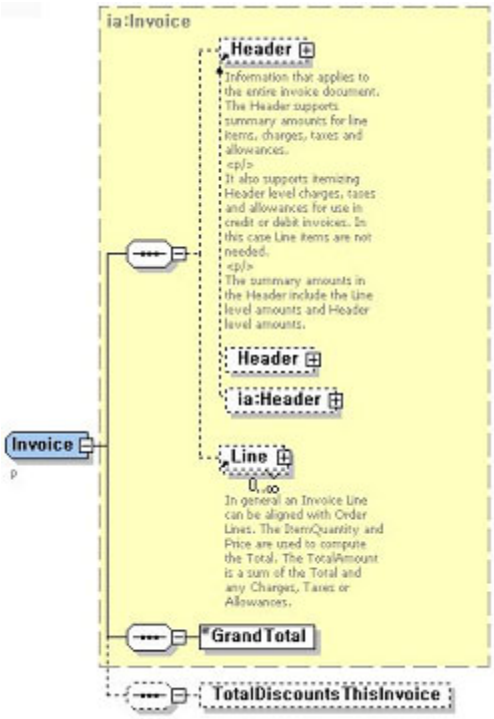
```
<xs:schema targetNamespace="http://www.XYZCorp.com/oagis/xyz"
           xmlns:ia="http://www.oagi.net/oagis/ia"
           xmlns:oa="http://www.openapplications.org/oagis"
           xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns:xyz="http://www.XYZCorp.com/oagis/xyz"
           elementFormDefault="qualified"
           attributeFormDefault="unqualified">
  <xs:import namespace="http://www.oagi.net/oagis/ia"
             schemaLocation="../../../IndustryA/Resources/Nouns/Invoice.xsd"/>
  <xs:import namespace="http://www.openapplications.org/oagis"
             schemaLocation="../../../../OAGIS/Resources/Nouns/Invoice.xsd"/>
  <xs:include schemaLocation="../XYZCorp.xsd"/>
  <xs:complexType name="Invoice">
    <xs:complexContent>
      <xs:extension base="ia:Invoice">
        <xs:sequence minOccurs="0">
          <xs:element name="TotalDiscountsThisInvoice" minOccurs="0"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
```

```
    </xs:complexType>
    <xs:complexType name="InvoiceHeader">
      <xs:complexContent>
        <xs:extension base="ia:InvoiceHeader"/>
      </xs:complexContent>
    </xs:complexType>
    <xs:element name="Invoice" type="xyz:Invoice" substitutionGroup="oa:Invoice"/>
    <xs:element name="Header" type="xyz:InvoiceHeader"
                substitutionGroup="oa:Header"/>
</xs:schema>
```

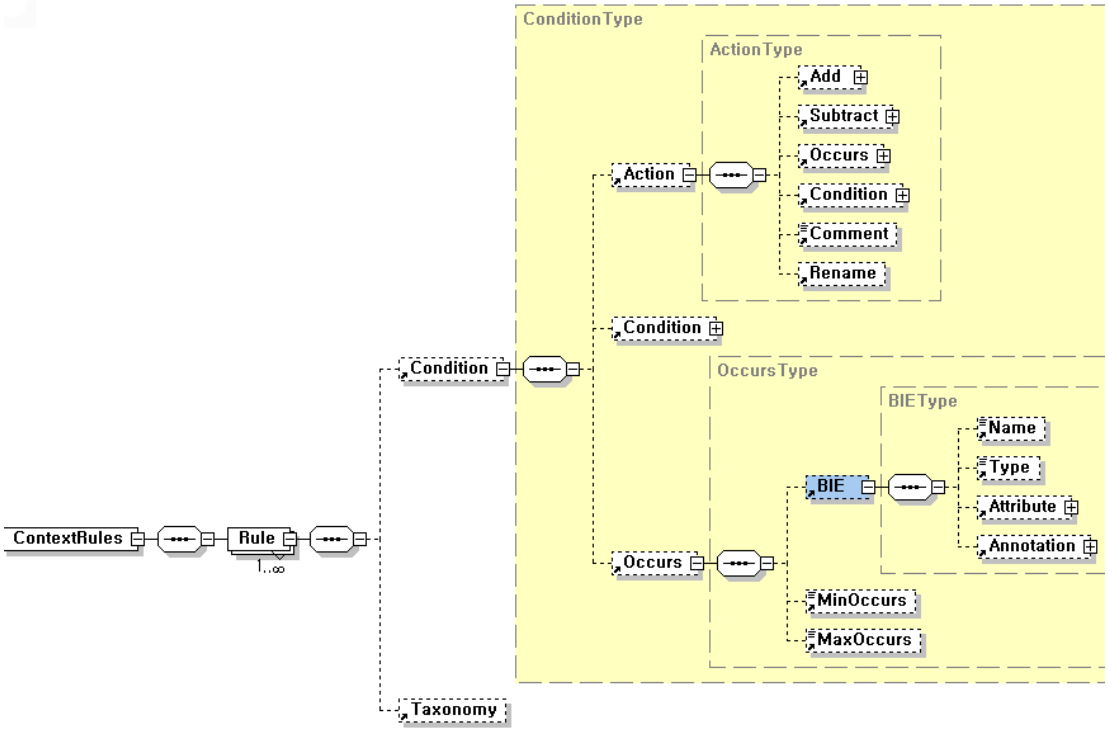
위의 스키마에서는 ia:Invoice를 기반으로 TotalDiscountsThisInvoice를 추가한 xyz:Invoice 타입을 정의하고, ia:InvoiceHeader를 기반으로 xyz: InvoiceHeader 타입을 정의한 다음, 전역 요소 Invoice를 xyz:Invoice 타입의 oa:Invoice의 대체 그룹으로 지정하고, Header를 xyz:InvoiceHeader 타입의 oa:Header의 대체그룹으로 지정한다.

- <TotalDiscountsThisInvoice> 요소를 확장한 XML 인스턴스 문서

```
<xyz:Invoice>
  <xyz:Header>
    …
    <ia:TimeCard/>
  </xyz:Header>
  <Line>…</Line>
  <ia:GrandTotal currency="USD">43000.00</ia:GrandTotal>
  <xyz:TotalDiscountsThisInvoice Currency="USD">2150.00
  </xyz:TotalDiscountsThisInvoice>
</xyz:Invoice>
```



2.4.7 ContextRule을 이용한 확장 : ebXML 코어 컴포넌트 스펙을 보면 Constraint Language에 대하여 기술되어져 있다. 이 언어는 전자문서의 조립과 컨텍스트규칙을 기술하는 XML 문서를 만들기 위한 구조를 정의해 놓은 것이다. 본 가이드라인에서는 XML 스키마 조립과 관련되어서는 아직 규정하지는 않으며, 대신 해당 비즈니스 컨텍스트에 따라서 컴포넌트의 변경 및 활용 방안을 ContextRule 문서에 세부적으로 기술할 것을 권고한다. 이 ContextRule 문서를 통하여, 이미 기 개발된 국내/외 XML 라이브러리 관련 정보를 체계적으로 정리할 수 있으며, 문서 조립 Tool이 개발될 경우, 큰 역할을 하게 될 것이다.


ConditionType
ActionType
Add
Subtract
Occurs
Condition
Comment
Rename
Action
Condition
OccursType
BIEType
Name
Type
Attribute
Annotation
BIE
MinOccurs
MaxOccurs
Occurs
ContextRules
Rule
1..∞
Condition
Taxonomy

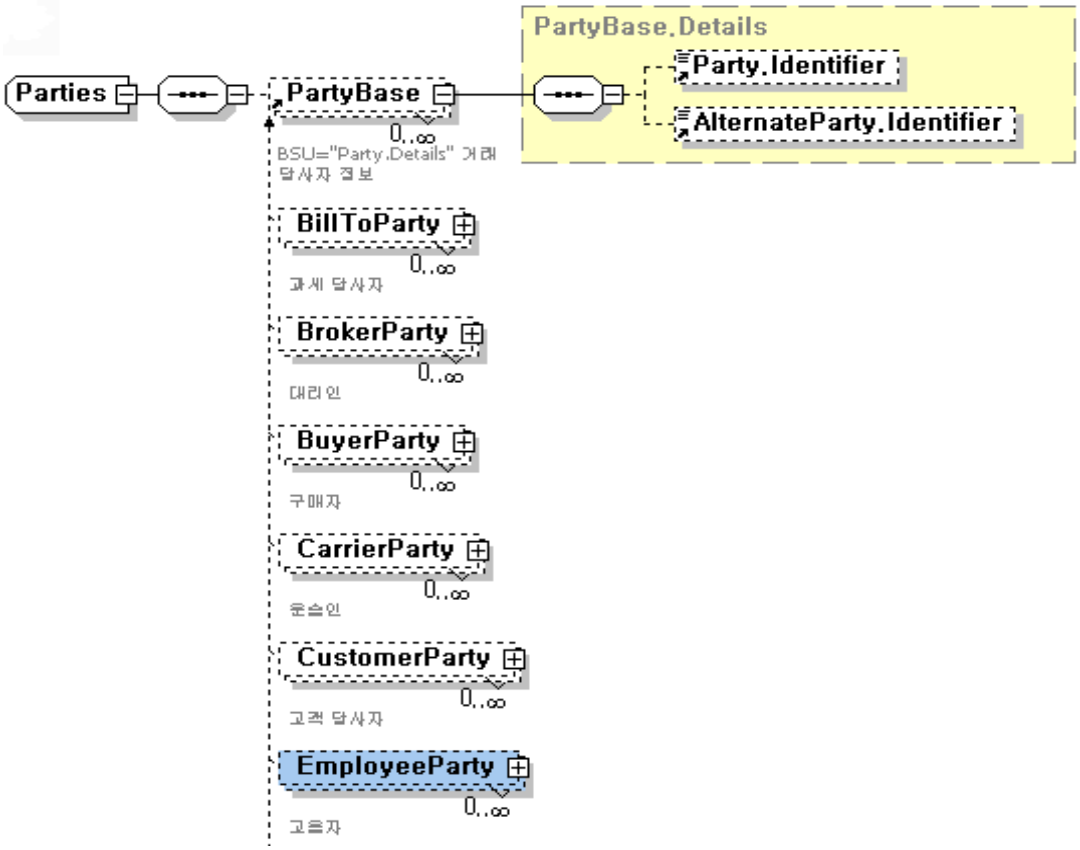
2.5 복수 컴포넌트 활용 규칙

2.5.1 EDIFACT 표준에서는 전자문서를 설계할 때 여러 개의 세그먼트를 묶어서 세그먼트 그룹을 만들어서 이 세그먼트 그룹을 Looping을 돌려서 여러 다양한 복수 정보 개체들을 표현하는 방식을 채택하고 있다. 일례로 EDIFACT 표준 전자문서에서 거래 당사자 정보를 표현할 때, 아래 그림과 같이 세그먼트 그룹을 묶어서 Looping을 돌려서, 구매자정보나 공급자 정보, 운송자 정보 등 여러 다양한 거래 당사자들을 표현한다.

	Segment group 2	**C**	**99**
NAD	Document/message details	M	1
LOC	Place/location identification	C	25
FII	Financial institution information	C	5
	Segment group 3	**C**	**99**
RFF	Reference	M	1
DTM	Date/time/period	C	5
	Segment group 4	**C**	**5**
DOC	Document/message details	M	1
DTM	Date/time/period	C	5
	Segment group 5	**C**	**5**
CTA	Contact information	M	1
COM	Communication contact	C	5

2.5.2 XCBL에서는 Looping을 돌려야 할 경우에는 ListOf - 라는 엘리먼트로 정의하여 표현한다.

2.5.3 본 가이드라인에서는 이렇게 여러 복수의 정보 개체들이 나타나는 경우에 하나의 복수 정보 개체를 정의하여 그 하위에서 단수의 정보 개체들을 나타나게 하는 방법으로 표현한다. 아울러 정보 개체를 정의하는 방법으로 주로 대체 요소를 이용한다. 아래 그림은 당사자 정보를 예로 들어서 보여주고 있다.



2.5.4 본 가이드라인에서는 현재 12개의 복수 컴포넌트를 정의하고 있으며, 각 업종/부문별로 추가할 필요성이 있을 경우, 진흥원에 통보하고 추가할 수 있다.

복수 컴포넌트명	정의	비고
Addresses	주소들	
Attachments	첨부 파일들	
Contacts	연락처들	
DocumentIds	문서 식별자들	
DocumentReferences	참조 문서들	
Locations	장소들	
Parties	당사자들	
PartyReferences	당사자 참조들	
Prices	가격 정보들	
Properties	특성들	
Ranges	범위들	
Taxes	세금들	

2.6 기본 정보 개체의 길이 할당 규칙

2.6.1 EDIFACT의 경우에 데이터 엘리먼트마다 각기 길이를 할당하고 있다. 할당하는 방식은 Fixed방식(예:an3)이 있고, 가변형 방식(예:an1..17)이 있다. 거의 대부분 가변형 방식을 사용하고 있다.

2.6.2 XML의 경우에 DTD로 문서구조를 정의하였을 경우, 데이터 길이에 대하여 제한을 둘수 없지만, 대부분 속성을 따로 사용자 정의하여 길이를 할당하는 방식을 취하고 있다. XML 스키마는 기본적으로 스키마에서 길이 제한을 가능하도록 하고 있다.

```
<simpleType name='form-input'>
  <restriction base='string'>
    <maxLength value='50'/>
  </restriction>
</simpleType>
```

2.6.3 본 가이드라인에서는 기본 정보 개체의 길이를 기본 정보 개체가 가지고 있는 표현 유형에 따라서 구분되도록 하였다.

표현 유형	길 이	표현 유형	길 이
Amount.Type	총길이:18, 소숫점:3	Name.Type	**35**
BinaryObjec.Type	제한없음	Number.Type	총길이:18, 소숫점:3
Code.Type	17	Numeric.Type	총길이:18, 소숫점:3
DateTime.Type	35	Quantity.Type	총길이:18, 소숫점:3
Identifier.Type	35	Text.Type	**70**
Indicator.Type	12	UserArea	제한없음
LongText.Type	제한없음	URI	제한없음
Measure.Type	총길이:18, 소숫점:6		

2.6.4 업종/부문별 전자문서 구성 데이터 항목을 정의할 경우, 데이터 항목에 대한 길이를 설정하게 된다. 이 때 사용하는 길이의 경우 대부분 어플리케이션에서 사용하는 DB의 필드 길이가 될 것이다. 이렇게 어플리케이션에서 사용되는 길이와 전자문서에서 사용하는 표현유형의 길이가 반드시 일치할 필요는 없다. 전자문서에서 사용하는 표현유형의 길이는 가변형이므로 어플리케이션에서 사용하는 길이가 전자문서에서 사용하는 표현유형의 길이 범위 안에만 있으면 된다.

2.6.5 만약 기본 정보 개체가 가지고 있는 비즈니스적인 의미가 표현유형이 가지고 있는 길이보다 클 경우에는 표현 유형을 수정하여 사용할 것을 권고한다.

<예제>

<EAN128.Identifier> -> <EAN128ID.Text>(유형은 LongText.Type)

3. XML 라이브러리 구성 요소

3.1 코어 컴포넌트 유형의 설계

3.1.1 코어 컴포넌트 유형 : 코어 컴포넌트 표현 유형은 기본적으로 ebXML에서 규정한 정보 개체 표현 유형을 준수하며, 본 가이드라인에서는 전자문서 설계의 효율성을 추가하기 위하여 몇 개의 표현 유형을 추가하였다.

표현유형	속성	표현형	비고
Amount.Type	- currency - currency.CodeListVersion	Amount	
BinaryObjec.Type	- binaryObject.Format - binaryObject.Mime.Type - binaryObject.Encoding.Type - binaryObject.URI	BinaryObject	
Code.Type	- codeList - codeList.Name - codeList.Agency - codeList.Agency.Name - codeListScheme.URI - codeList.URI - codeList.Version - language.Code	Code	
DateTime.Type	- format	DateTime	
Identifier.Type	- idSchemeAgency - idSchemeAgency.Name - idSchemeData.URI - idScheme - idScheme.Name - idScheme.URI - idSchemeAgency - language.Code	Identifier	

Indicator.Type	- format	Indicator	
LongText.Type	- language.Code	Text	
Measure.Type	- unit.Code - unit.CodeListVersion	Measure	
Name.Type	- language.Code	Name	
Number.Type	- format	Number	
Numeric.Type	- format	Numeric	
Quantity.Type	- unit.Code - unit.CodeList - unit.CodeListAgency - unit.CodeListAgency.Name	Quantity	
Text.Type	- language.Code	Text Value	
UserArea			
URI			

3.1.2 Amount 유형 : 가격을 표시할 때 사용되는 표현 유형으로 3자리 소숫점을 포함하여 총 18 digit로 이루어졌으며, 속성으로는 통화 코드와 통화 코드리스트 버전을 기술한다. 통화 코드로는 ISO 4217 표준을 활용하여 3자리의 통화 코드를 기술하면 된다.

<예제>

<Item.Amount currency="KRW">200000</Item.Amount>

3.1.3 BinaryObject 유형 : 이진 파일 정보를 직접 삽입하고자 할 때 사용되는 표현 유형으로 일단 기본적으로 base64Binary방식을 활용한다. 속성으로는 이진 파일 정보에 대한 Format, MimeType, 암호 유형, URI 등 여러 표현 속성을 기술한다.

3.1.4 Code 유형 : 코드를 기술할 때 사용하는 표현 유형으로, 전자문서의 내용을 기술할 때, 보통 데이터를 기술할 수도 있겠지만, 여러 데이터의 내용을 코드화하여 코드를 활용하는 경우가 많이 있다. 여기서는 해당 코드 값을 기술하며, 코드 값과 관련된 여러 속성을 기술하게 되어 있다. 속성으로는 코드리스트 식별자와 코드리스트 버전, 코드리스트명, 코드리스트 관리기관, 코드리스트 URI 등을 기술한다. 여기서 Code.Name에서는 코드값에 대한 정의를 선택적으로 기술하도록 한다.

<예제><PaymentTerms.Type.Code>22</PaymentTerms.Type.Code>

3.1.5 Date 유형 : 날짜를 표시할 때 사용되는 표현 유형으로 여러 가지 방식으로 표현이 가능하다. EDIFACT에서는 DTM 세그먼트에서 주로 날짜/시간/기간을 기술하며, 엘리먼트 2379에서 날짜/시간/기간에 대한 포맷을 규정하고 있다. 여기서는 기본적으로

일시 : DateTime(CCYY-MM-DDThh:mm:ss)

일자 : Date(CCYY-MM-DD)

시간 : Time(hh:mm:ss)

기간 : Period(CCYYMMDDThh:mm:ss/CCYYMMDDThh:mm:ss)

등으로 구분하여 기술하도록 한다. 즉, DateTime.Format의 열거값으로 위의 DateTime, Date, Time, Period로 구분하여 날짜 포맷을 기술한다.

<예제>

<Document.DateTime DateTime.Format="Date">2003-02-25</Document.DateTime>

3.1.6 Identifier 유형 : 식별자를 기술할 때 사용하는 표현 유형으로, 물품 번호나 거래 당사자 식별자 등을 기술할 때 사용한다. 식별자를 사용할 때 주의할 점으로는 식별자가 국제적으로나 국내적으로 표준화 되었을 경우 식별자 사용에 있어서 큰 문제가 없겠지만, 식별 구조를 거래 당사자들이 중복하여 구축하거나 사용하고 있을 경우, 거래당사자간에 식별자를 공유할 수 없는 문제가 생긴다. 그러기 때문에 식별자를 사용할 경우, 식별 구조와 식별자 관리 기관 속성을 기술하는 것이 좋다.

3.1.7 Indicator 유형 : 지시자를 기술할 때 사용하는 표현 유형으로, 전자문서의 내용을 기술할 때 보통 참, 거짓으로 표현하거나, 간단한 몇 개의 코드를 기술할 때 사용한다. 그래서 지시자 포맷으로는 W3C에서 규정한 Boolean형과 당사자가 직접 정의한 FreeForm으로 구분하여 기술한다.

<예제>

<Priority.Indicator Indicator.Format="boolean">true</Priority.Indicator>

3.1.8 LongText 유형 : 70바이트 이상의 평문을 기술할 때 사용하는 표현 유형으로, EAN-128 바이트 식별코드나 주소 자유 형식 등을 기술할 때 사용한다.

3.1.9 Measure 유형 : 측정 수치 등을 표시할 때 사용되는 표현 유형으로 6자리 소숫점을 포함하여 총 18 digit로 이루어졌으며, 속성으로는 단위 코드와 단위 코드리스트 버전을 기술한다. 단위 코드로는 UN/ECE 권고안 20번을 참조하면 된다.

<예제>

<NetWeight.Measure Unit.Code="KG">990</NetWeight.Measure>

3.1.10 Name 유형 : 35바이트 이하의 평문을 기술할 때 사용하는 표현 유형으로, 주로 이름이나 단문을 기술할 때 사용한다.

3.1.11 Number 유형 : 라인 번호나 개정 번호 등을 기술할 때 사용하는 표현 유형으로 순차적으로 증가하거나 양의 정수일 경우에 사용되는 표현 유형으로 소숫점 없이 총 6 digit로 이루어졌다. 속성으로는 번호 포맷을 사용한다.

3.1.12 Numeric 유형 : 정수값이나 소숫값 등 수치를 기술할 때 사용하는 표현 유형으로 3자리 소숫점을 포함하여 총 18 digit로 이루어졌으며, 속성으로는 수치 포맷을 기술한다.

3.1.13 Quantity 유형 : 개체의 수량이나 총 주문 수량 등 수량을 기술할 때 사용하는 표현 유형으로 속성으로는 단위 코드와 단위 코드리스트, 코드리스트 관리기관과 기관명을 기술한다. 특히 단위 코드는 반드시 기술하는 것을 권고한다.

3.1.14 Text 유형 : 35바이트 이상, 70바이트 이하의 평문을 기술할 때 사용하는 표현 유형으로, 주로 장문을 기술할 때 사용한다.

3.2 기본 정보 개체의 설계

3.2.1 기본 정보 개체는 EDIFACT표준에서 데이터 엘리먼트와 같은 역할을 한다. 즉, 하나의 기본 의미를 가지고, 컴포넌트나 메시지를 구성하는 기본 구성요소이다.

3.2.2 기본 정보 개체는 ebXML 방식을 기초로 하여 2개의 유형으로 구분하였다. 중립적인 의미를 가지면서 컴포넌트의 구성요소로서 재사용 가능한 기본 코어 컴포넌트 유형과 비즈니스적인 의미를 가져서 실제 독립적인 정보항목으로 역할을 할 수 있는 비즈니스 정보 항목으로 구분할 수 있다.

3.2.3 기본 코어 컴포넌트(Basic Core Component) : 기본 코어 컴포넌트는 앞의 용어 정의에서 나왔듯이 중립적인 단일 의미를 지니고 있으며, 복합 코어 컴포넌트(Aggregate Core Component)를 구성하는 기본 정보 개체라고 볼 수 있다. 현재 정확하게 기본 코어 컴포넌트에 대한 정의가 ebXML 코어 컴포넌트 스펙에 기술되어 있지만, 납득가능한 실제 사례를 찾아보기 힘들다.

3.2.4 본 가이드라인에서는 다음과 같이 기본 코어 컴포넌트를 규정하였다.

1) 객체와 객체의 표현형이 결합된 경우,

2) 속성과 속성의 표현형이 결합된 경우

로 한정하여 구분하였다. 아래 몇 개의 기본 코어 컴포넌트 사례를 예시하였다.

기본 정보 개체명	정의	객체 클래스	속성명	표현형
Accepted.Quantity	수취 수량		Accepted	Quantity
Account.Identifier	계좌 식별자	Account		Identifier
Account.Name	계좌명	Account		Name
Acknowledge.Code	확인 코드		Acknowledge	Code
Activity.Amount	활동 금액	Activity		Amount
Activity.Identifier	활동 식별자	Activity		Identifier
Actual.Amount	실제 금액		Actual	Amount

3.2.5 기본 비즈니스 정보 개체(Basic Business Information Entity) : 기본 비즈니스 정보 개체는 앞의 용어 정의에서 나왔듯이 비즈니스 컨텍스트가 적용되어 비즈니스적인 의미를 지닌 독립적인 항목이라고 정의하였다. 본 가이드라인에서는 보다 더 구체적으로 다음과 같은 경우를 기본 비즈니스 정보 개체로 정의한다.

1) 객체와 객체의 속성, 객체의 표현형이 결합된 경우,

2) 객체나 속성 어느 한 곳에라도 한정어가 사용된 경우

객체와 객체의 속성이 결합되었을 경우, 두 개의 요소 의미가 결합되었기 때문에 비즈니스 의미를 지닌 독립적인 항목이 될 수 있으며, 한정어로써 객체나 속성을 명확하게 한정하여 독립적인 항목이 되기 때문에 위의 두 경우에는 비즈니스 정보 개체라고 구분하였다.

기본 정보 개체명	정의	한정어	객체 클래스	한정어	속성명	표현형
OriginatedCountry. Code	원산지 국가 코드			Originated	Country	Code
OverShipTolerance. Quantity	초과 한도 유예 수량			OverShip	Tolerance	Quantity
OverTolerancePeriod. DateTime	초과 한도 유예 기간			OverTolera nce	Period	DateTime
Packaging.Level.Code	포장 등급 코드		Packaging		Level	Code
Packaging.Type.Code	포장 유형 코드		Packaging		Type	Code
Packaging.Unit. Quantity	단위 포장내 물품 수량		Packaging		Unit	Quantity

3.2.6 기본 개체 레벨에서 사용하고 있는 객체 클래스는 전자문서에서 주요한 역할을 하는 객체들로 구성하였다. 현재 XML 라이브러리에서 사용하고 있는 객체들은 아래와 같다. 이 이외에 각 업종별/부문별로 신규 전자문서를 개발하면서 요구되는 객체 클래스는 전자거래진흥원에 통보하여 추가할 수 있다.

객체 유형	객체 유형명
Container	컨테이너
Delivery	배송
Location	장소

Message	메시지
Order	주문
Organization	조직
Packaging	포장
Party	당사자
PaymentTerms	지불 조건
Person	개인
Price	가격
Project	프로젝트
Quote	견적
Shipment	운송
Tax	세금
Transaction	거래
Transport	운송

3.3 XML 컴포넌트의 설계

3.3.1 XML 컴포넌트는 전자문서를 구성하는 여러 기본 정보 개체들을 모델링 해 놓은 것으로 전자문서 개발시 중요한 구성 요소가 된다. 현재, 국제 표준화된 컴포넌트는 존재하지 않으며, 여러 표준화 단체에서 발간한 스펙에서 여러 다양한 방식으로 컴포넌트를 설계하고 있으며, 각각 저마다의 장단점을 지니고 있다.

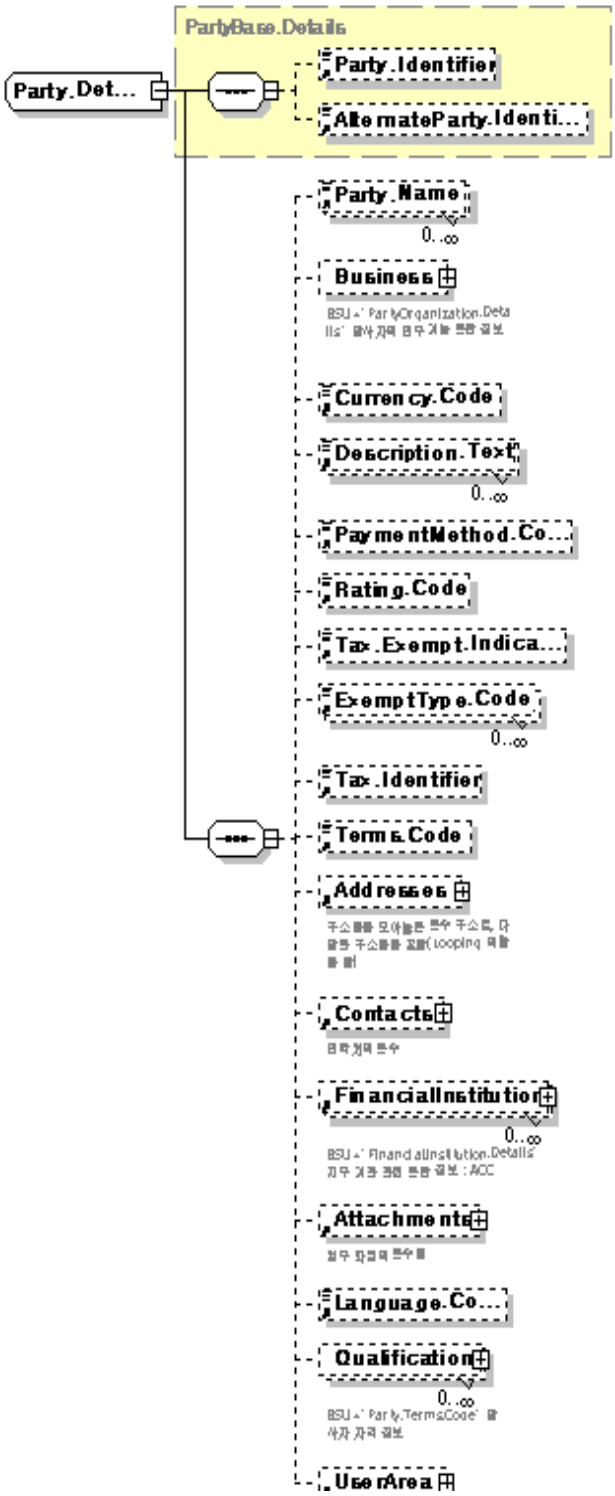
3.3.2 XML 컴포넌트를 설계하면서 기본으로 한 지침들은 다음과 같다.

(1) 현재 국제 및 국내 EDI전자문서에서 시맨틱적으로 가장 공통적이고 보편적으로 사용되는 개념을 그룹화하여 집합 코어 컴포넌트로 개발하며, 기본적으로 사용되는 중립적 개념을 기본 코어 컴포넌트로 사용한다
(2) 글로벌하게 적용할 수 있어야 한다
(3) 현재 UN/EDIFACT 와 ANSI X. 12에서 사용되는 기존 EDI Semantic을 기본으로 하며, 개발된 코어 컴포넌트는 기존의 시맨틱을 대부분 모두 수용 가능하여야 한다
(4) 여러 산업들과 도메인에 걸쳐 적용할 수 있어야 한다
(5) 코드 리스트와 관련해서는 직접 삽입방식 대신, 외부 코드 파일 참조 등을 권고하며, 비즈니스 프로세스별로 사용되는 각 전자문서에 사용되는 코드적용 규칙파일을 만들어서 사용할 수 있다. 그러나 당 가이드라인에서 이러한 코드적용 규칙파일에 대한 구체적인 정의는 하고 있지 않다.
(6) 간단하고 다른 언어로 명확하게 번역이 가능하여야 한다
(7) 코어 컴포넌트 내부에 한정어를 포함해서는 안된다. 대신 유형코드의 경우는 사안에 따라 포함 할 수 있다
(8) 코어 컴포넌트의 설계는 XML 기술에 맞게 잘 정의 되거나 안정된 관점에 의존하여 최적화되고, 가볍고, 단순한 구조를 유지하여야 한다

3.3.3 개발한 컴포넌트중 전자문서 상에서 중요한 역할을 하는 컴포넌트들 몇 개를 자세히 살펴보도록 하겠다.

3.3.4 거래 당사자 컴포넌트 설계 : 거래 당사자 컴포넌트는 전자문서에서 가장 중요한 컴포넌트 중의 하나로, 전자 거래에 참여하고 있는 당사자에 대한 상세한 정보들로 구성되어 있다. EDIFACT 표준이나 ANSI X.12 표준에서는 하나의 세그먼트로 거래 당사자를 표현하지 않고, 여러개의 세그먼트로 표현하고 있다.

본 가이드라인에서는 여러 표준들을 참고하여, 아래 그림과 같이 거래 당사자 컴포넌트를 정의하였다.



1) Party.Identifier : 거래 당사자 식별자로 거래 당사자를 식별할 수 있으며, EAN에서 정의하고 있는 13자리의 식별자를 사용할 수 있으며, 자체적으로 할당한 식별자를 사용할 있으며, 경우에 따라서는 사업자 등록증을 사용할 수도 있다. 이 경우 식별자의 속성중, IdScheme에 식별리스트를 쓰며, IdScheme.Agency에는 식별리스트 관리 기관 식별자를 부여.

2) AlternateParty.Identifier : 위의 거래 당사자 식별자 뿐만 아니라 추가적으로 거래 당사자 식별자를 기술하고자 할 경우 사용.

3) Party.Name : 거래 당사자명을 기술한다. 보통 회사명을 기술하지만, 만약 회사 대표자명과 회사명 둘다 기술하여야 하는 경우에는, Party.Name에 회사 대표자명을 기술하며, Business 컴포넌트에 있는 Organization.Name에 회사명을 기술.

4) Business 컴포넌트 : 거래 당사자가 회사일 경우, 회사에 대한 상세 정보를 기술.

5) Currency.Code : 거래 당사자가 기본으로 사용하는 통화 코드를 기술한다. 통화 코드는 ISO 4217 표준을 이용.

6) Description.Text : 거래 당사자에 대한 일반적인 평문 기술.

7) PaymentMethod.Code : 거래 당사자가 기본으로 하는 지불방법에 대해서 기술한다. 진흥원에서 분류해놓은 공통 코드를 활용.

8) Rating.Code : 거래 당사자에 대한 등급 분류 코드를 기술. 진흥원에서 분류해놓은 공통 코드를 활용.

9) Tax.Exempt.Indicator : 거래 당사자가 세금 면제인지에 대한 지시자로 값은 Boolean방식(true/false)으로 기술.

10) ExemptType.Code : 거래 당사자가 면제 일 경우 어떠한 유형의 면제인지를 기술하는 코드. 진흥원에서 분류해놓은 공통 코드를 활용.

11) Tax.Identifier : 거래 당사자의 세금관련 식별자로 Business 컴포넌트를 사용하지 않을 경우에 사용.

12) Terms.Code : 거래 당사자가 가지고 있는 여러 조건 코드를 기술하는 정보 개체로, 사용자 정의 코드.

13) Address 컴포넌트 : 거래 당사자에 대한 여러 주소들을 기술하는 복수 컴포넌트.

14) Contacts 컴포넌트 : 거래 당사자에 대한 여러 연락처들을 기술하는 복수 컴포넌트.

15) FinancialInstitution 컴포넌트 : 거래 당사자의 재무 기관 정보를 기술하는 복수 컴포넌트.

16) Attachments 컴포넌트 : 거래 당사자에 대한 첨부 파일 정보를 기술.

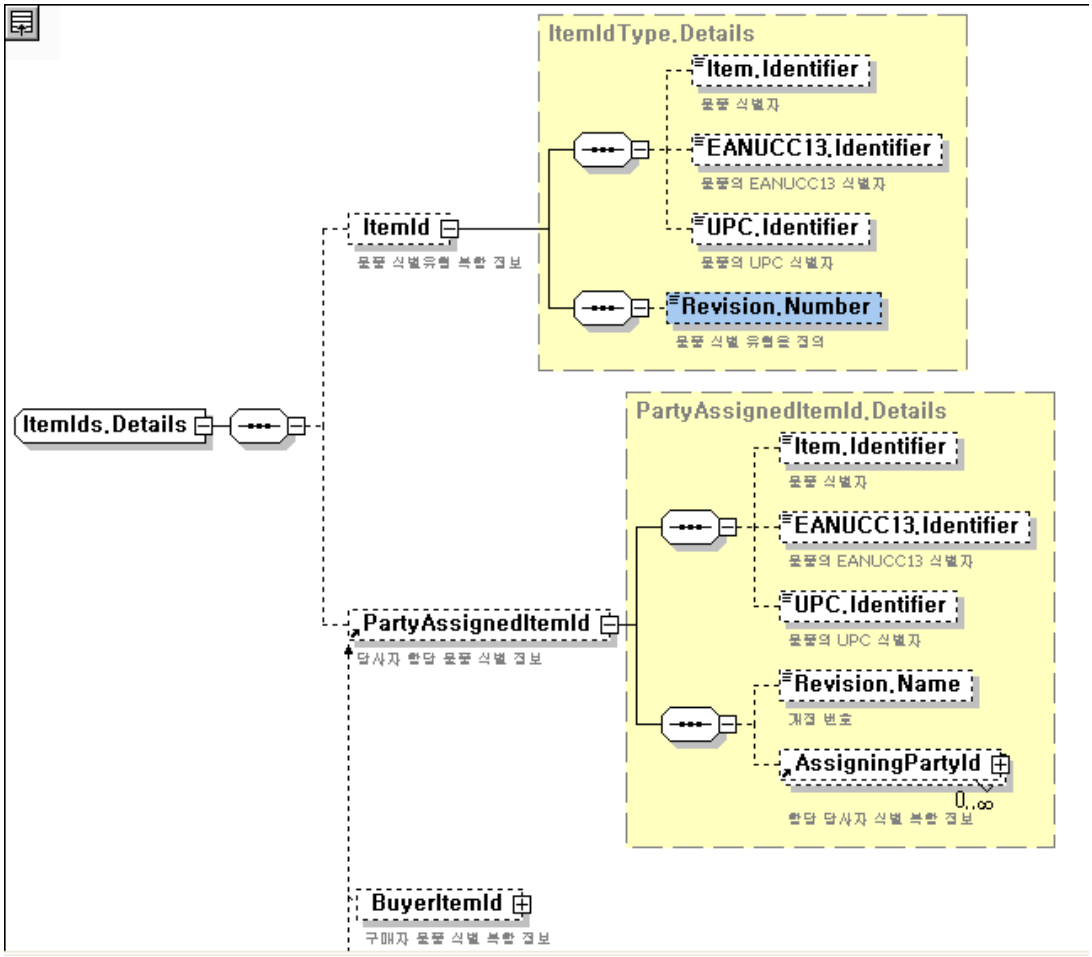
17) Language.Code : 거래 당사자가 사용하는 언어 코드.

18) Qualification : 거래 당사자가 가지고 있는 여러 자격조건이나 면허 등의 정보를 기술하는 컴포넌트.

19) UserArea : 사용자 정의 영역으로 확장 가능.

3.3.5 물품 식별 컴포넌트 설계 : 물품 식별 컴포넌트는 전자문서에서 물품을 식별하기 위한 컴포넌트이다. 물품 식별과 관련된 코드로는 현재 EAN-13 코드가 유통 분야에서 많이 사용되고 있으며, UPC 코드도 많이 사용하고 있어서 각기 독립적인 식별자로 구분하여 구성을 하였으며, 하위에는 당사자 할당 물품 식별코드 영역을 두어서 표준 코드 대신 당사자가 직접 할당한 코드를 부여할 수 있도록 하였다. 유통 분야에서는 각 거래 당사자별로 각각의 식별 코드를 할당하는 경우가 많아서 하나의 물품에 여러 개의 식별코드 라벨이 붙어 있는 경우를 볼 수 있는데, 이 경우에 활용할 수 있을 것이다.

본 가이드라인에서는 여러 표준들중, OAG에서 설계한 컴포넌트를 주로 참고하여, 아래 그림과 같이 거래 당사자 컴포넌트를 정의하였다.



1) Item.Identifier : 물품 식별자로 물품에 대한 식별번호를 기술한다. 식별자의 속성중, IdScheme에 식별리스트를 쓰며, IdScheme.Agency에는 식별리스트 관리 기관 식별자를 부여하는 것을 권고한다.

2) EANUCC13.Identifier : 유통 업종이나 기타 부문에서 EANUCC13 코드를 활용하는 경우에 EANUCC13 식별번호를 기술.

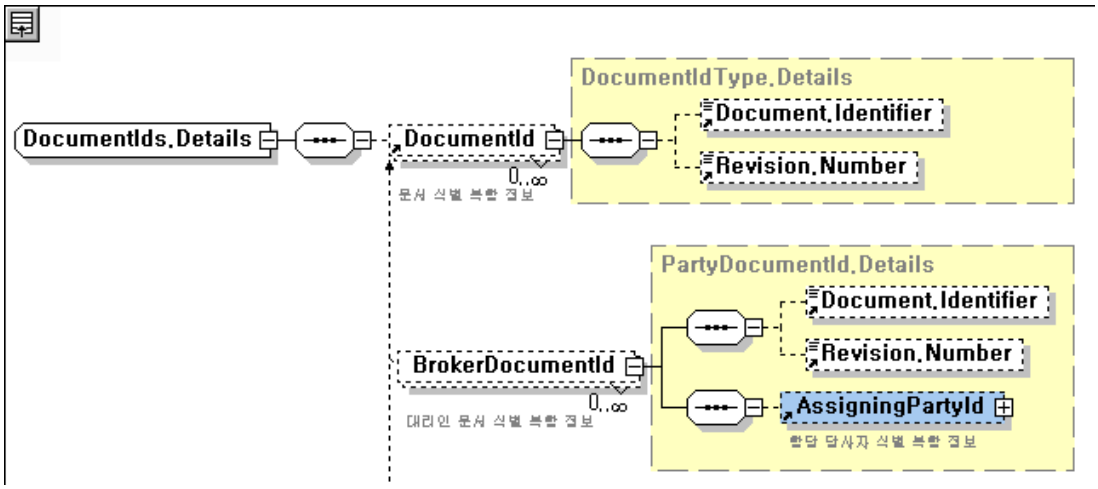
3) UPC.Identifier : 북미의 경우, UPC 식별번호를 많이 활용하는데, 이 경우 UPC 식별번호를 기술.

4) Revision.Number : 사용하고 있는 식별자의 개정 번호를 기술.

5) AssigningPartyId : 식별번호를 할당한 거래 당사자를 식별하는 정보 컴포넌트.

3.3.6 문서 식별 컴포넌트 설계 : 문서 식별 컴포넌트는 전자문서상에서 문서의 식별과 관련된 정보를 표현하는 컴포넌트이다.

본 가이드라인에서는 여러 표준들을 참고하여, 아래 그림과 같이 문서 식별 컴포넌트를 정의하였다.



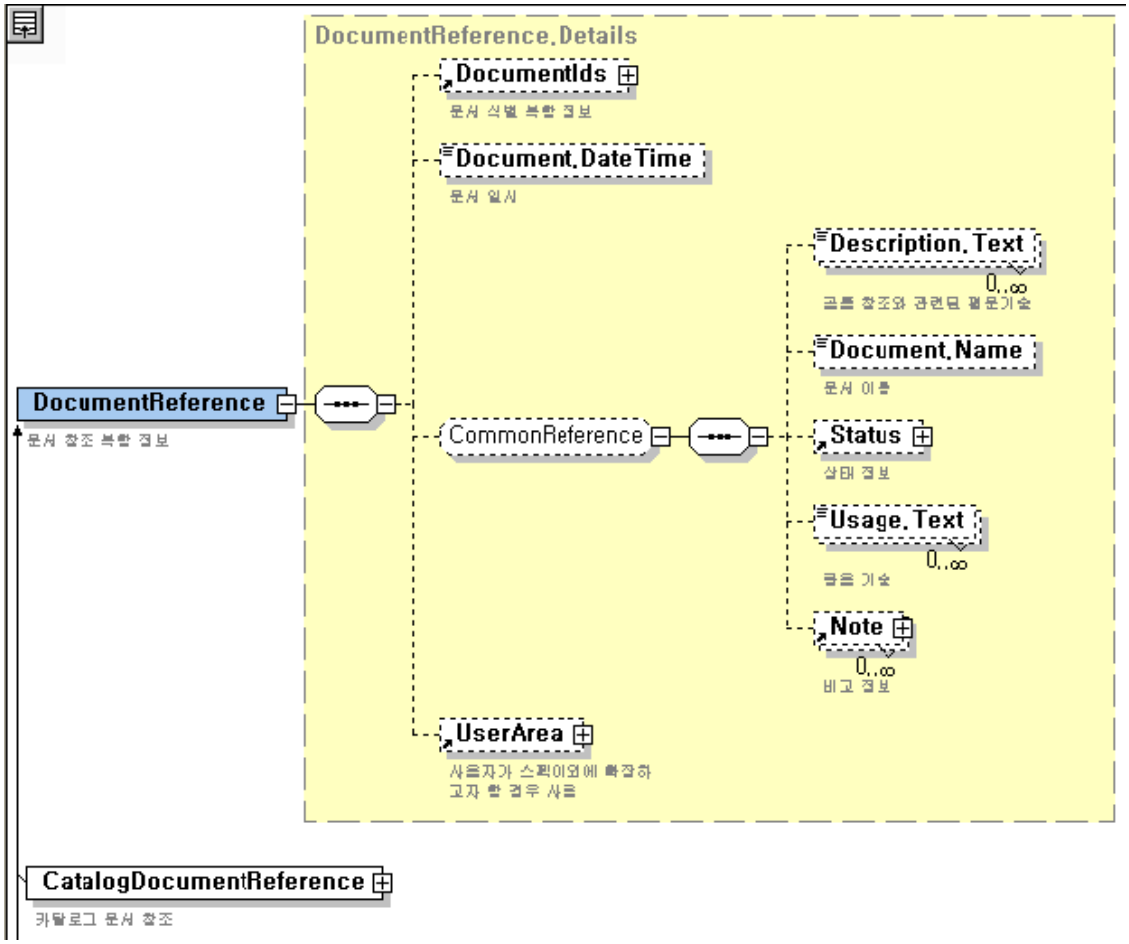
1) Document.Identifier : 문서에 대한 식별번호를 기술하는 정보 개체.

2) Revision.Number : 사용하고 있는 식별자의 개정 번호를 기술.

3) AssigningPartyId : 식별번호를 할당한 거래 당사자를 식별하는 정보 컴포넌트.

3.3.7 **참조 문서 식별 컴포넌트 설계** : 참조 문서 식별 컴포넌트는 전자문서상에서 참고하는 다른 문서의 식별과 관련된 정보를 표현하는 컴포넌트로 EDIFACT에서 주로 활용하는 세그먼트 그룹인 RFF-DTM과 유사한 역할을 한다.

본 가이드라인에서는 여러 표준들중, OAG에서 설계한 컴포넌트를 주로 참고하여, 아래 그림과 같이 참조 문서 식별 컴포넌트를 정의하였다.



1) DocumentIds 컴포넌트 : 문서 식별 컴포넌트로 해당 참조 문서에 대한 문서 식별번호를 기술하는 컴포넌트

2) Document.DateTime : 해당 참조문서의 참조 일자를 기술하는 정보개체

3) Description.Text : 해당 참조 문서에 대한 일반적인 정보를 기술.

4) Document.Name : 해당 참조 문서의 문서명 기술

5) Status 컴포넌트 : 해당 참조 문서의 상태를 기술하는 컴포넌트

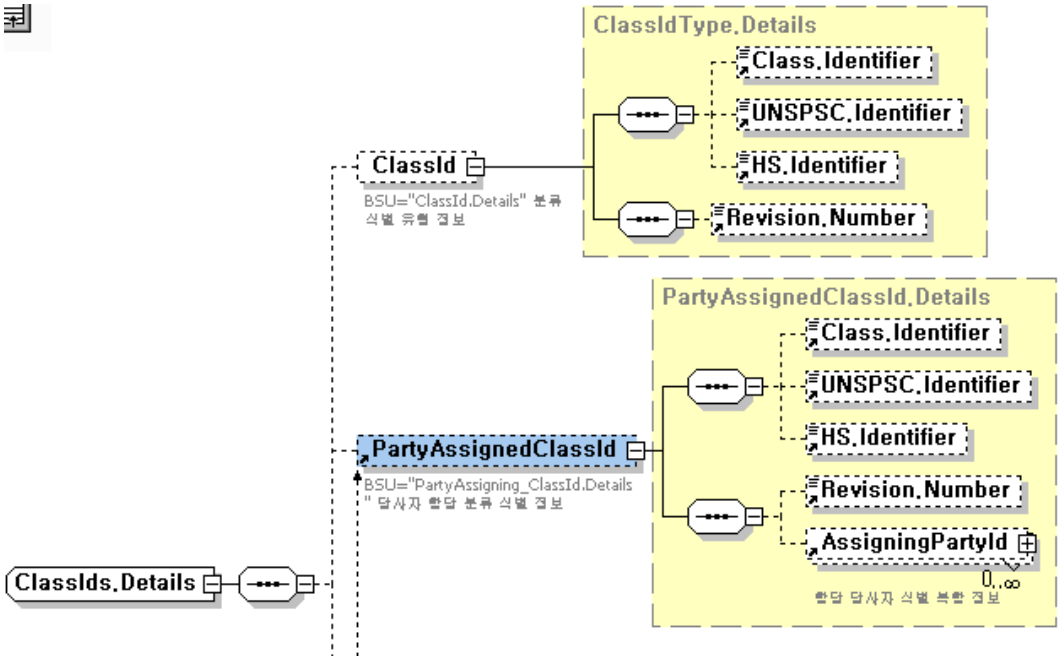
6) Usage.Text : 해당 참조 문서에 대한 활용과 관련된 평문을 기술

7) Note : 해당 참조 문서에 대한 비고 정보를 기술.

8) UserArea : 사용자 정의 영역으로 확장 가능.

3.3.8 **물품 분류 컴포넌트 설계** : 물품 분류 컴포넌트는 전자문서에서 물품을 분류하기 위한 정보를 표현하는 컴포넌트로 물품 분류와 관련된 코드로는 현재 UNSPSC 코드와 HS 코드가 많이 사용되고 있다. 하위에는 당사자 할당 물품 분류코드 영역을 두어서 표준 코드 대신 당사자가 직접 할당한 분류 코드를 부여할 수 있도록 하였다.

본 물품 분류 컴포넌트는 물품 식별 컴포넌트와 유사한 구조로 설계하였다.



1) Class.Identifier : 분류 식별자로 물품에 대한 분류번호를 기술한다. 식별자의 속성중, IdScheme에 식별리스트를 쓰며, IdScheme.Agency에는 식별리스트 관리 기관 식별자를 부여하는 것을 권고한다.

2) UNSPSC.Identifier : 물품 분류코드리스트중 가장 많이 사용하는 UNSPSC분류 코드를 사용할 경우 기술.

3) HS.Identifier : 무역부문에서 많이 사용하고 있는 HS 분류코드를 사용할 경우 기술.

4) Revision.Number : 사용하고 있는 식별자의 개정 번호를 기술.

5) AssigningPartyId : 식별번호를 할당한 거래 당사자를 식별하는 정보 컴포넌트.

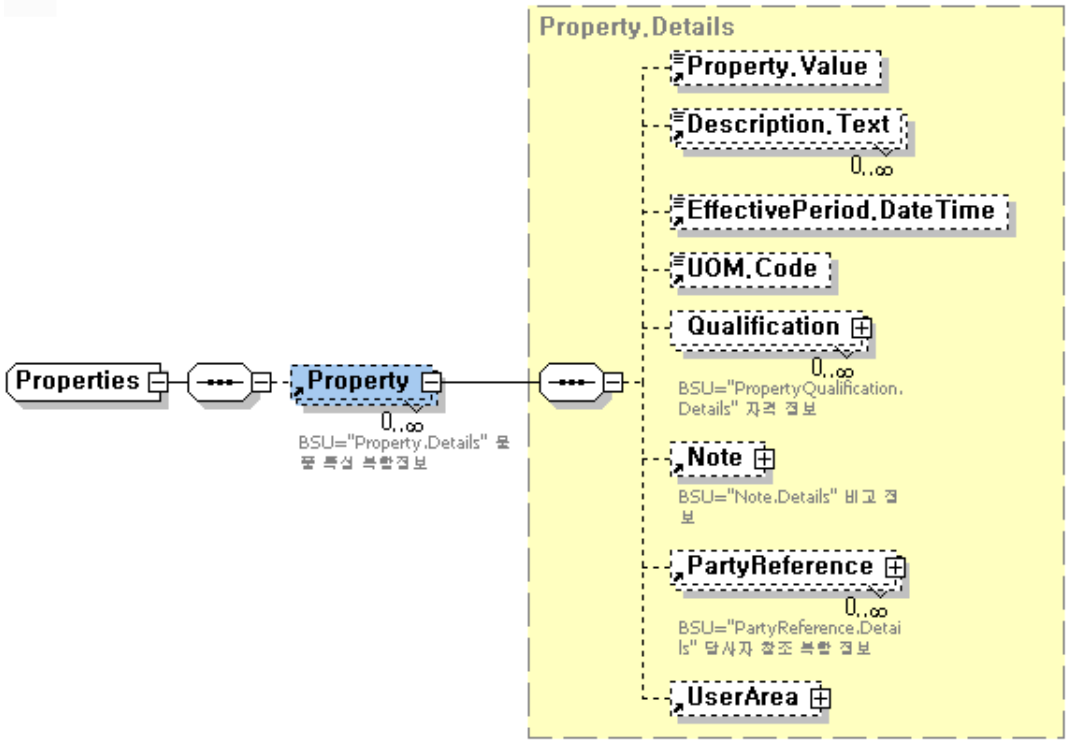
3.3.9 **물품 특성 정보 컴포넌트 설계** : 물품 특성 정보 컴포넌트는 전자문서상에서 기술하는 물품에 대한 특성들을 표현하는 컴포넌트로 EDIFACT에서의 IMD 세그먼트를 대체하는 역할을 할 수 있다. 아울러 물품 특성 컴포넌트는 복수형(Properties)으로 두어서 여러개의 Property를 기술할 수 있도록 하였다.

3.3.9.1 아울러 물품 특성의 경우 각 업종별, 부문별로 상당히 많은 정보가 있으며, 이를 모두 기본 정보 개체로 만들어서 활용하는데는 여러 어려움이 있기 때문에 어트리뷰트를 활용하여, 어트리뷰트에서 물품 특성의 이름을 기술하고 엘리먼트 값으로 특성값을 기술하는 방식을 취하였으며, 관련 물품 특성을 한정하는 EDIFACT 7081 코드값을 어트리뷰트에서 같이 기술하도록 하였다.

<예>

```
<Properties>
    <Property>
        <!-- 물품 속성 정보 -->
        <Property.Value name="규격" code="2">10*10*10 MDF</Property.Value>
    </Property>
</Properties>
```

본 가이드라인에서는 여러 표준들을 참고하여, 아래 그림과 같이 물품 특성 컴포넌트를 정의하였다.



1) Property.Value : 물품 특성 값을 기술하는 정보 개체로, 물품 특성명은 어트리뷰트에서 기술. 아울러 물품 특성을 한정하는 한정어 코드도 선택적으로 기술할 수 있음.

2) Description.Text : 물품 특성에 대하여 평문 기술

3) EffectivePeriod.DateTime : 물품 특성과 관련된 유효기간 기술.

4) UOM.Code : 물품 특성과 관련된 측정 단위 코드 기술. 진흥원에서 분류해놓은 공통 코드를 활용.

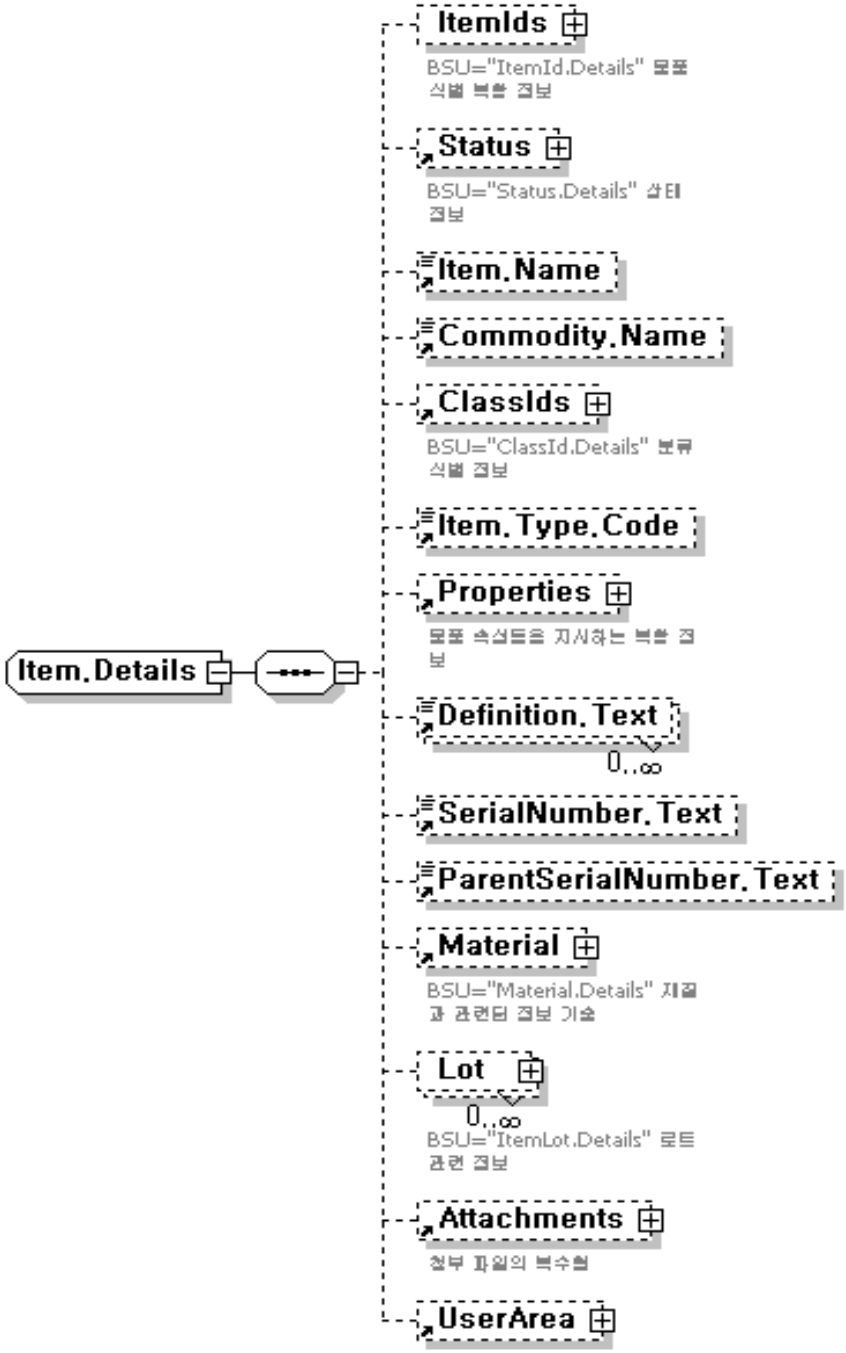
5) Qualification 컴포넌트 : 거래 당사자가 가지고 있는 여러 자격조건이나 면허 등의 정보를 기술하는 컴포넌트.

6) Note 컴포넌트 : 비고 정보를 기술하는 컴포넌트

7) PartyReference 컴포넌트 : 물품 특성과 관련된 거래 당사자 참조 정보를 기술하는 컴포넌트.

8) UserArea : 사용자 정의 영역으로 확장 가능 .

3.3.10 **물품 컴포넌트 설계** : 물품 컴포넌트는 전자문서에서 물품에 대한 관련 정보를 표현하는 컴포넌트로 물품 식별정보, 물품 분류정보, 물품 특성 등 여러 가지 다양한 하위 정보들을 포함하고 있는 컴포넌트이다.



1) ItemIds 컴포넌트 : 물품 식별번호를 기술하는 컴포넌트.

2) Status 컴포넌트 : 물품 상태를 기술하는 컴포넌트

3) Item.Name : 물품명을 기술.

4) Commodity.Name : 물품과 관련된 상품분류명을 기술.

5) ClassIds : 물품 분류 식별번호를 기술하는 컴포넌트.

6) Item.Type.Code : 물품 유형 정보를 기술

7) Properties 컴포넌트 : 물품 특성에 대한 정보를 기술하는 컴포넌트.

8) Definition.Text : 물품에 대한 규격이나 일반 정보를 기술.

9) SerialNumber.Text : 물품의 일련번호를 기술

10) ParentSerialNumber.Text : 물품의 상위 일련번호를 기술.

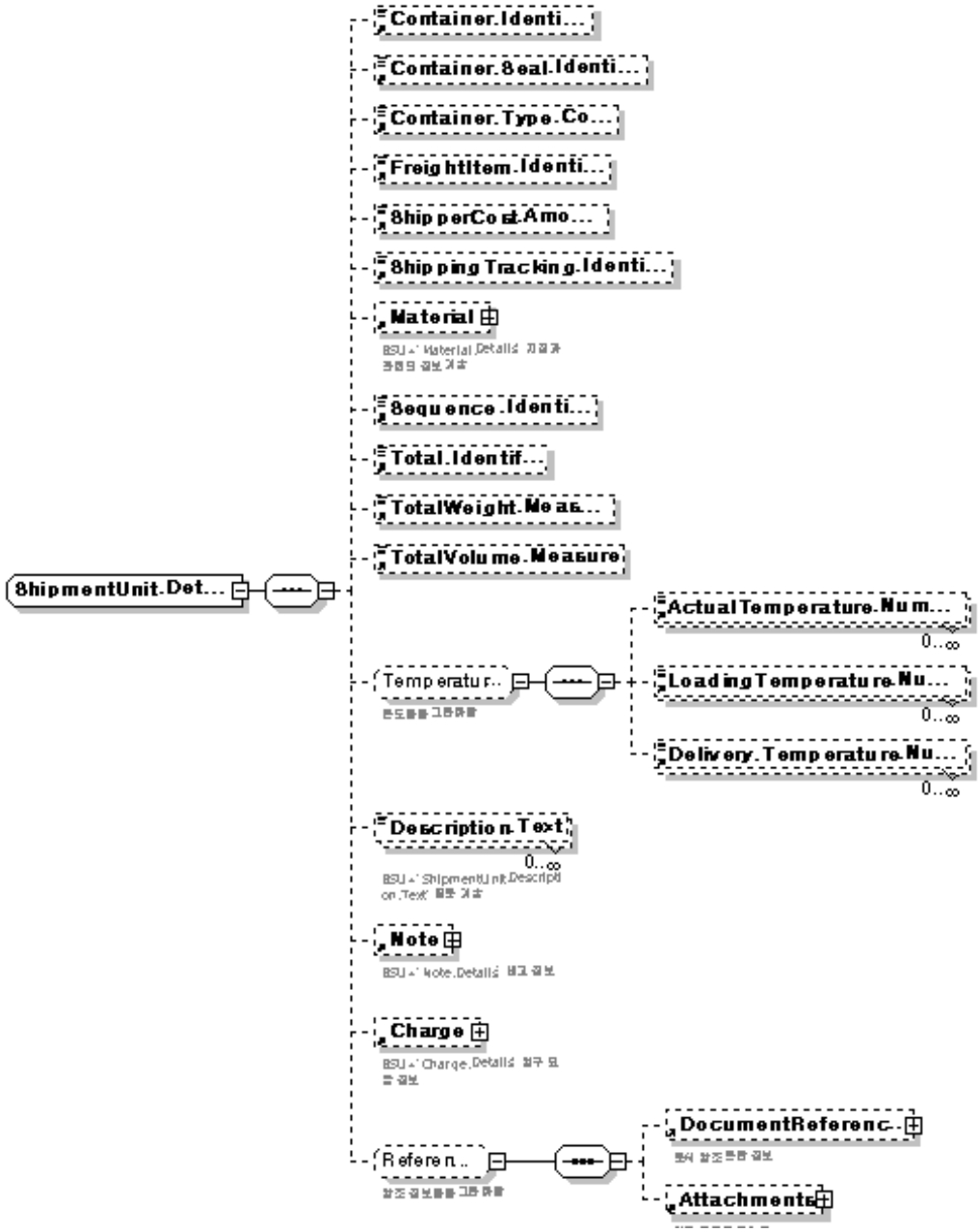
11) Material : 물품의 재질과 관련된 정보를 기술하는 컴포넌트.

12) Lot : 물품군과 관련된 정보를 기술하는 컴포넌트

13) Attachments : 물품과 관련된 첨부 파일 정보를 기술하는 컴포넌트

14) UserArea : 사용자 정의 영역으로 확장 가능 .

3.3.11 **운송단위 컴포넌트 설계** : 운송 단위 컴포넌트는 운송 물품을 운반하는 컨테이너나, 여러 운송 용기들과 관련된 정보를 표현하는 컴포넌트로 컨테이너 식별정보, 봉인 식별자, 기타 식별정보 등으로 구성되어 있다.



1) Container.Identifier : 컨테이너 식별 번호를 기술.

2) Container.Seal.Identifier : 컨테이너 봉인 번호를 기술

3) Container.Type.Code : 컨테이너 유형 코드를 기술

4) FreightItem.Identifier : 운송 물품의 식별 번호를기술

5) Shippercost.Amount : 운송자 비용을 기술.

6) ShippingTracking.Identifier : 운송중 추적 식별자를 기술

7) Material 컴포넌트 : 재질 정보를 기술하는 컴포넌트.

8) Sequence.Identifier : 일련 식별 번호를 기술.

9) Total.Identifier : 운송 물품의 전체 식별번호를 기술.

10) Temperatures : 운송 단위와 관련된 온도 정보를 기술하는 그룹

11) Description.Text : 운송 단위와 관련된 평문 기술

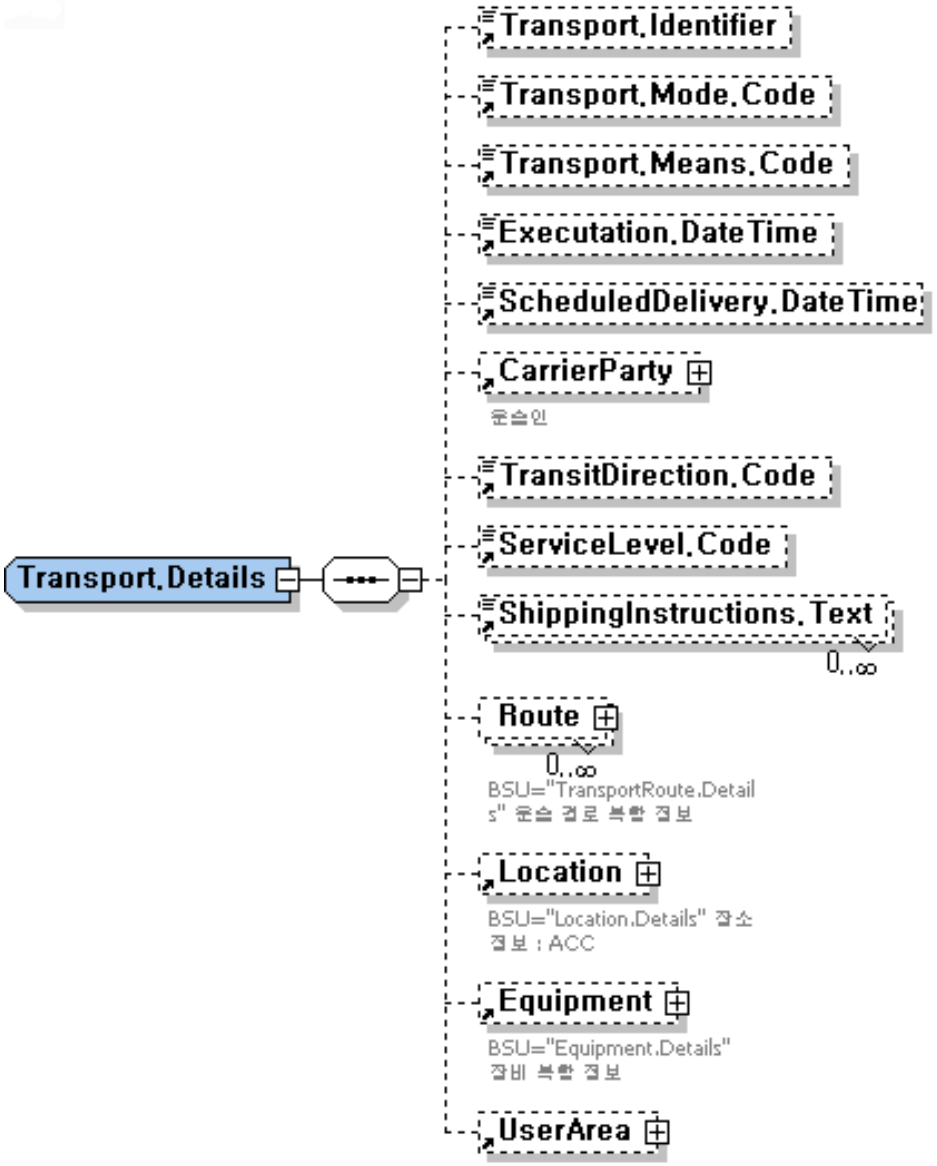
12) Note 컴포넌트 : 운송 단위와 관련된 비고 정보를 기술.

13) Charge 컴포넌트 : 운송 단위와 관련된 청구 요금 정보를 기술하는 컴포넌트.

14) DocumentReferences : 운송 단위와 관련된 참조 문서 정보를 기술하는 컴포넌트

15) Attachments : 운송 단위와 관련된 첨부 파일 정보를 기술하는 컴포넌트.

3.3.12 **운송 컴포넌트 설계** : 운송 컴포넌트는 전자문서에서 물품/서비스의 운송과 관련된 정보를 표현하는 컴포넌트로 운송 방법, 운송 형태, 운송자 식별, 관련된 장소 정보 등 여러 가지 다양한 하위 정보들을 포함하고 있는 컴포넌트이다.



1) Transport.Identifier : 운송 식별 번호를 기술.

2) Transport.Mode.Code : 운송 형태 코드를 기술하며, 해당 코드는 진흥원에서 분류해놓은 공통코드를 참조.

3) Transport.Means.Code : 운송 방법 코드를 기술하며, 해당 코드는 진흥원에서 분류해놓은 공통코드를 참조.

4) Executation.DateTime : 운송과 관련된 실행 일시를 기술

5) ScheduledDelivery.DateTime : 예정된 배송 일시를 기술

6) CarrierParty : 운송자에 대한 정보를 기술

7) TransitDirection.Code : 운송 방향 지시 코드를 기술하는 것으로 진흥원에서 분류해놓은 공통 코드를 참조

8) ServiceLevel.Code : 서비스 등급 코드를 기술하는 것으로 진흥원에서 분류해놓은 공통 코드를 참조

9) ShippingInstructions.Text : 운송 지시와 관련된 평문 기술.

10) Route : 운송 경로와 관련된 정보를 기술하는 컴포넌트

11) Location : 운송 장소와 관련된 정보를 기술하는 컴포넌트

12) Equipment : 운송과 관련된 장비 정보를 기술하는 컴포넌트.

13) UserArea : 사용자 정의 영역으로 확장 가능.

3.3.13 복합 비즈니스 정보 개체는 복합 코어 컴포넌트에 비즈니스 컨텍스트가 적용되어 한정어가 결합된 형태라고 할 수 있다. 크게 2가지의 형태로 복합 비즈니스 정보 개체가 생성되는데, 보통 일반적으로 복합 코어 컴포넌트를 유형으로 하여 생성되는 경우와 대체 요소(SubstitutionGroup)을 활용하여 복합 코어 컴포넌트를 대체하는 경우이다.

1) 첫 번째 방법

```
<xs:element name="OrderStatus" type="Status.Details">
        <xs:annotation>
                <xs:documentation source="http://www.kiec.or.kr/kis">주문 상태
                정보</xs:documentation>
        </xs:annotation>
</xs:element>
```

2) 두 번째 방법

```
<xs:element name="ImportTax" type="Tax.Details" substitutionGroup="Tax">
        <xs:annotation>
                <xs:documentation source="http://www.kiec.or.kr/kis">수입 세금
                </xs:documentation>
        </xs:annotation>
</xs:element>
```

3.3.14 복합 비즈니스 정보 개체를 정의할 때는 반드시 적용된 비즈니스 컨텍스트에 대한 정보도 규정하여야 한다.

3.3.15 현재 복합 비즈니스 정보 개체는 122개 정도 있으며, 이 이외에 각 업종별/부문별로 신규 전자문서를 개발하면서 요구되는 복합 비즈니스 정보 개체는 전자거래진흥원에 통보하여 추가할 수 있다.

3.4 XML 메시지의 개요

3.4.1 메시지는 실제 비즈니스 거래가 거래 당사자간 혹은 거래 당사자 Agent 사이에서 이루어질 때 송수신된다. 그러므로 전자문서 선정이나 전자문서에 들어가야 할 정보 항목들은 업무의 요구사항에 의해 결정되어 진다.

3.4.2 본 가이드라인에서는 각 업종/부문별로 가장 빈번하고, 공통적으로 사용하는 7개의 업무 프로세스에서 사용되는 19종의 메시지를 1차적으로 개발하였다. 세부적인 사항은 아래와 같다.

전자문서명	문서명(한)	정 의	해당 프로세스
Party	거래처정보	거래처정보는 거래 상대방간에 상거래 관계가 시작되는 시점에 제일 먼저 교환하게 되는 전자문서로서, 거래 상대방에게 로케이션 정보, 그리고 업체명, 주소, 담당자, 거래은행 계좌번호 등과 같은 관련 정보를 제공하는데 사용된다. 이와 같은 전자문서는 향후 관련 정보의 내용이 변경되거나 추가될 경우, 거래 상대방의 컴퓨터 마스터 파일을 즉시 갱신할 수 있도록 재 전송되어야 한다.	기본 관리
General Response	일반응답서	일반 응답서로 일반적인 평문 정보를 기술하는 문서로 어플리케이션 활용과 관련된 단문이나 일반적 응용시 사용하는 문서	
Quote	견적서	공급자가 구매자에게 물품/서비스를 제공하기 위하여 사전에 물품 가격이나 수량 등과 관련된 정보를 제공하는 문서	견적
RequestFor Quote	견적요청서	구매자가 물품/서비스를 구매하기 위하여 공급자에게 사전에 물품 가격이나 수량 등과 관련된 정보를 요청하기 위해 활용하는 문서	
Order	주문서	판매자와 구매자간에 상품의 주문과 이에 관련된 서비스에 대한 상세사항을 전송할 수 있도록 하는 전자문서이다. 아울러, 국내에서 상품구매시 이용되고 있는 매입전표나 거래명세서 역시 본 주문서를 통해 구현할 수 있다.	주문
Order Change	주문변경서	주문변경요청서는 구매자가 이미 전송한 주문서에 대한 변경을 요청하거나, 판매자가 주문응답서를 이용하여 제시한 주문내역의 확인이나 변경요청에 대한 수정을 하는데 사용할 수 있다.	

전자문서명	문서명(한)	정 의	해당 프로세스
Order Response	주문응답서	구매자가 송신한 주문서나 주문 변경요청서에 대해서 물품의 재고나 기타 상황을 검토하여 구매자의 주문에 대해서 응답하는 데 활용하는 전자문서	
OrderStatus Request	주문 상태 요청서	이미 사전에 송부한 주문서 내역과 관련된 상태를 질의하기 위해 구매자/구매대리인이 공급자/공급대리인에게 송부하는 문서	
OrderStatus Report	주문 상태 보고서	구매자/구매대리인이 송부한 주문 상태질의서에 대응하여 공급자/공급대리인이 주문 내역과 관련된 상태(완료, 진행중, 진행퍼센트 등)에 대해서 보고하는 문서	
SalesInformationReport	판매정보 보고서	위치, 기간, 상품인식, 가격, 화폐량, 수량, 시장정보, 판매분야와 같은 상품이나 서비스에 관련된 판매정보의 전송을 가능케 하는 전자문서이다.	판매
InventoryInformationReport	재고정보 보고서	재고현황과 관련된 정보를 제공하는 전자문서이다. 재고현황보고서는 거래당사자간에 직접 사용토록 지정된다.	
RequestForDelivery	납품요청서	구매자에 의해서 공급자에게 상품의 납품을 요청하는 문서로 주문서의 모든 물품 혹은 일부의 물품에 대해서 납품을 요청할 수 있다.	납품
Shipment Notify	선적통지서	거래 업체 상호간의 약정에 따른 상품의 납품 또는 발송에 대한 상세 내역을 통보하기 위한 전자문서	
Shipment Response (Confirmation)	선적응답서 (인수통지서)	상품의 인수와 관련한 업무적인 정보를 제공하는 전자문서이다. 상품인수확인, 발송통지서와 연계한 인수확인, 그리고 주문상품과 인수상품간의 이상 통지 등의 작업에 활용될 수 있다. 선적 통지서를 받고 바로 응답하지 않고, 물건을 받고 나서 선적통지된 물건과 실제 받은 물건을 비교 검토후 주로 발송	
AnnounceForReturns	반품통지서	한 업체가 다른 업체에게 특정한 원인에 기인한 제품의 반품을 상세하게 알려주는 문서이다.	

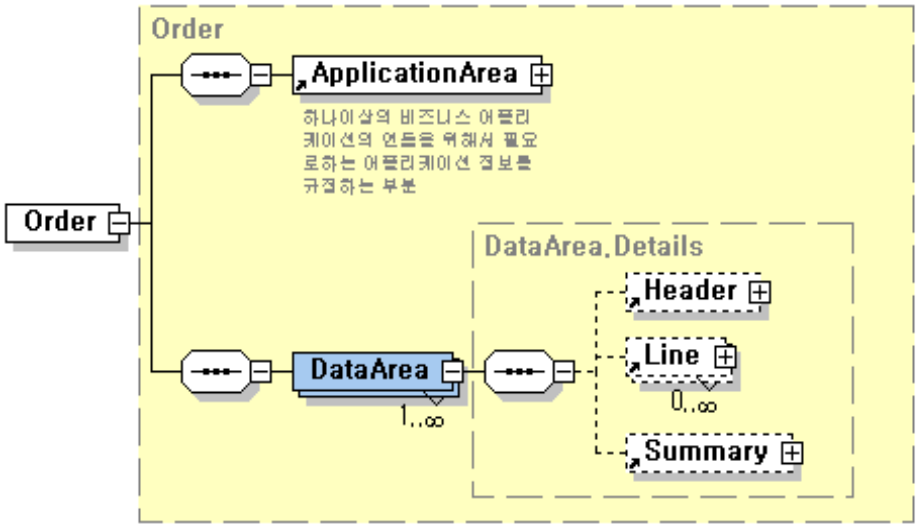
전자문서명	문서명(한)	정 의	해당 프로세스
RequestFor Inspection	검사요청서	납품하기전에 공급자가 제3자 검사대행기관에 검품을 의뢰하거나, 구매자가 공급자나 제3자 검사대행기관에 검품을 의뢰하는데 사용되는 문서	검품
ReportFor Inspection	검사보고서	고객이 의뢰한 검품 결과에 대해서 고객이나 관계된 당사자에게 송부하기 위해 사용되는 문서	
Invoice	송장	매도인과 매수인간에 합의된 계약조건하에서 공급되는 상품 또는 서비스에 대한 지불을 요구하는 전자문서이다. 송장 전자문서는 출금통지서(Debit Note Message)와 입금통지서(Credit Note message)에 대한 명세로서 사용되므로, 이 전자문서를 통해 송장/입금통지/출금통지 등에 광범위하게 활용된다.	결제
TaxInvoice	세금계산서	구매자와 공급자간에 서로간의 물품/서비스 교환에 의한 지불이 발생하였을 경우, 지불에 대한 영수/청구의 증명의 기능을 수행한다	

3.4.3 위의 19종의 전자문서 이외에 여러 다양한 전자문서들이 업종별/부문별로 사용되고 있다. 업종별/부문별 신규 메시지의 개발은 국내의 전자문서 표준관리 측면에서 반드시 전자거래진흥원과 유기적인 협력관계를 통하여 개발되어야 하며, 개발 이후 개발 결과물의 심사평가를 의뢰하여 검증을 받고, 향후 표준으로 등록하여야 한다.

3.4.4 전자거래진흥원은 업종/부문에서 신규 메시지 요청을 받을 경우에, 공동협력하여 계속적으로 메시지를 개발 및 공고할 것이며, 매년 XML 라이브러리를 업그레이드하여 공개하여, 누구나 이용하도록 할 것이다.

3.5 XML 메시지의 설계

3.5.1 본 가이드라인에서는 XML 메시지의 구조를 크게 어플리케이션 영역과 데이터 영역으로 구분하고, 데이터 영역을 다시 또 헤더부분과 본문인 라인부분, 그리고 요약부분인 Summary로 나누어 메시지를 구성한다. 이 메시지 구조는 OAGIS 8.0 구조를 참조하였다.



1) 어플리케이션 영역 : 메시지의 송 수신 및 메시지에 대한 메타 데이터 등을 표현한다. 그래서 이 영역은 주로 비즈니스 어플리케이션에서 직접 Control하는 부분이라 할 수 있다.

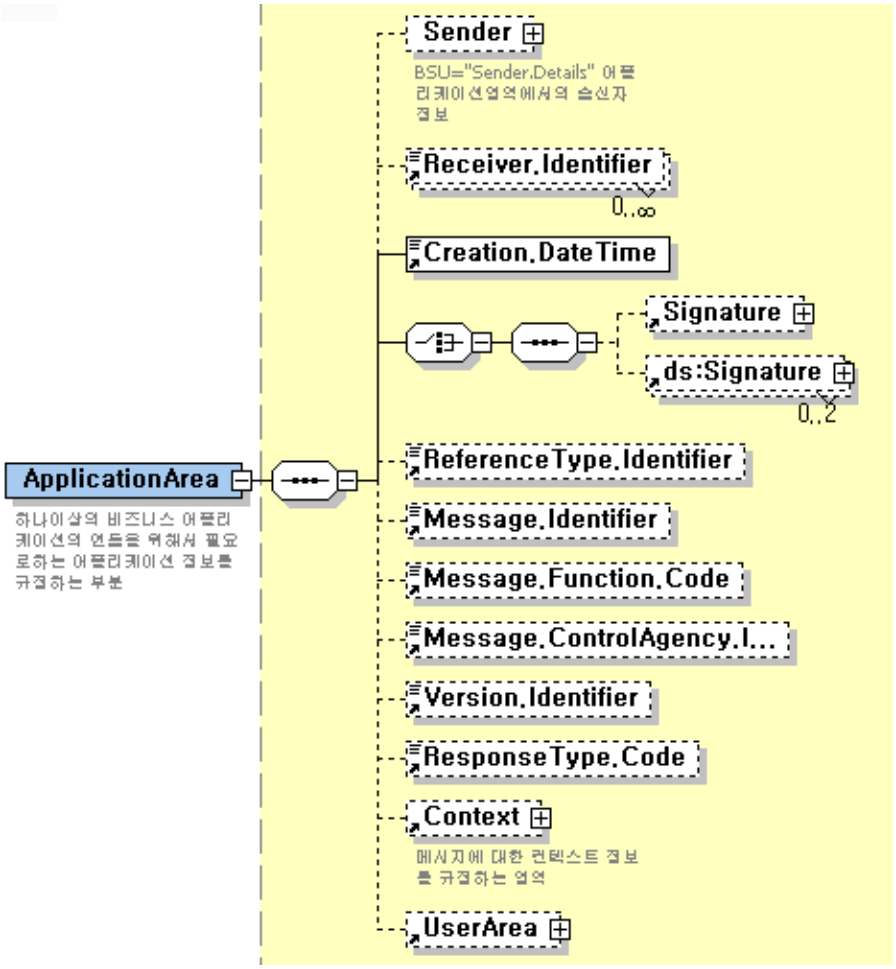
2) 데이터 영역 : 실제 전자문서의 내용이 들어가는 부분으로 하나 이상 나타난다.

3) 데이터 영역안의 헤더부분 : 전자문서에 전체적으로 적용되는 정보항목을 수록하고 있으며, 주요 포함 정보 항목은 문서 식별정보, 참조 문서 정보, 거래 당사자 정보, 운송관련 정보, 포장 정보, 할인/할증 정보, 세금 정보 등등 많은 정보들이 헤더부분에 나올 수 있다.

4) 라인부분 : 물품/서비스와 관련된 정보나 운송단위/운송물품과 관련된 정보를 수록한다. 여기서는 라인 물품과 관련된 가격 정보, 포장 정보, 납품 정보, 기타 정보들이 포함되며, 선택적으로 하위 물품정보와 물품의 스케줄 정보까지 포함할 수 있다.

5) 요약(Summary)부분 : 전자문서에서 기술된 물품/서비스에 대한 총계 정보를 나타내는 부분이다. 특히 요약부분은 주문서나 견적서와 같은 경우에는 사용할 수 있지만, 납품이나 기타 전자문서에서는 요약 정보가 필요하지 않은 경우도 있으므로 생략 가능하다.

3.5.2 어플리케이션 영역의 설계 : 어플리케이션 영역은 전자문서와 관련된 실제 내용이 들어가지는 않으며, 메시지의 송수신 관련 정보, 메시지의 속성 등에 대한 메타 데이터로 구성된다. EDIFACT의 경우에는 UNB, UNG, UNH 등과 같은 시스템 세그먼트에서 그 역할을 하고 있다. VAN-EDI시스템에서 EDIFACT 메시지는 송수신 정보까지 포함하고 있지만, 현재 SOAP나 기타 메시지 전송 프로토콜을 이용하여 Transport/Routing/Packaging을 수행하는 메시징 시스템에서는 이러한 어플리케이션 영역이 불필요할 수도 있겠지만, 메시지를 주고 받는 거래 상대방간에 서로 협약을 통하여 이 영역을 선택적으로 활용하도록 한다.



1) Sender 컴포넌트 : 발신자 정보를 기술하는 컴포넌트

2) Receiver.Identifier : 수신자 식별번호를 기술

3) Creation.DateTime : 메시지의 생성 일시를 기술

4) Signature & ds:Signature : 메시지 전자서명을 기록

5) ReferenceType.Identifier : 메시지의 참조 유형 식별자를 기록

6) Message.Identifier : 메시지 식별자.

7) Message.Function.code : 메시지 기능 코드로 EDIFACT 1225 코드리스트 참조

8) Message.ControlAgency.Identifier : 메시지 관리 기관 식별자 기술. 전자거래진흥원에서 관리하는 메시지일 경우, KIEC 기술하며, 각 업종에서 관리할 경우, 업종 대표기관의 약자를 기술

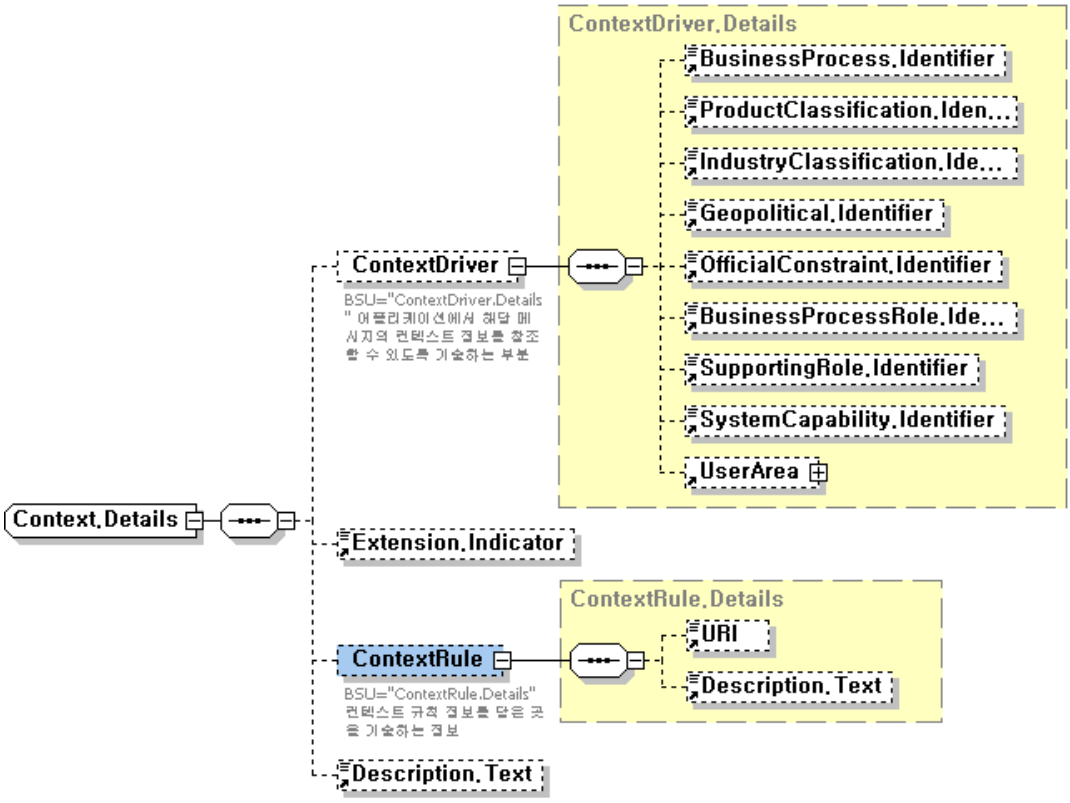
9) Version.Identifier : 메시지의 버전을 기술

10) ResponseType.Code : 해당 메시지의 응답 유형을 기술. EDIFACT 4343 코드리스트 참조

11) Context 컴포넌트 : 해당 메시지의 비즈니스 컨텍스트 정보를 기술하는 컴포넌트

12) UserArea : 사용자 정의 영역으로 확장 가능

3.5.2.1 어플리케이션 영역내 Context 정보 : 해당 메시지와 관련된 컨텍스트 정보를 표현할 수 있도록 하여 시스템간 상호 연동을 지원할 수 있도록 하였다. 이 컨텍스트 정보는 사용자가 입력하는 것이 아니라 어플리케이션에서 입력하고 식별할 수 있도록 한다. 아래 그림을 보면, 컨텍스트 정보 구조를 살펴볼 수 있을 것이다.



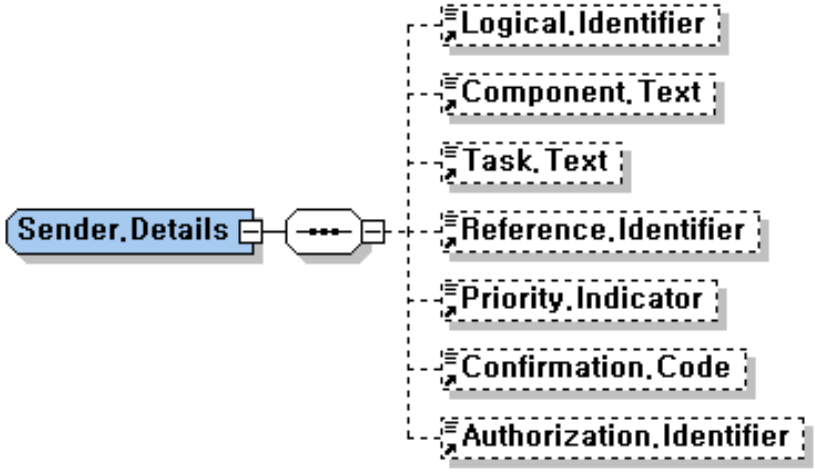
1) ContextDriver 컴포넌트 : 컨텍스트 드라이버 정보를 기술하는 컴포넌트로 해당 컨텍스트 정보에 식별정보를 기술. 예를 들어 해당 메시지가 어느 비즈니스 프로세스에서 사용되는지를 BusinessProcess.Identifier에 기술하며, 어느 업종/부문에서 사용되는 지를 IndustryClassification. Identifier에 기술하며, 메시지 적용범위가 국제적인지, 국내용인지를 구분하기 위해 Geopolitical.Identifier에 기술한다.

2) Extension.Indicator : 해당 메시지가 확장되었는 지를 지시하는 지시자로, boolean 형식(true/false)으로 기술

3) ContextRule 컴포넌트 : 컨텍스트 규칙 파일이 사용되었을 경우 파일에 대한 정보 기술

4) Description.Text : 해당 컨텍스트 정보와 관련된 평문 기술

3.5.2.2 어플리케이션 영역내 송신자 정보 : 해당 메시지를 송신하는 송신자에 대한 정보를 표현하는 컴포넌트로 메시징 프로토콜을 사용하는 시스템에서는 이 컴포넌트 대신 메시징과 관련된 헤더를 이용할 수 있다.



1) Logical.Identifier : 해당 메시지 송신자의 논리적 식별자 기술

2) Task.Text : 송신자의 작업과 관련된 평문 기술

3) Reference.Identifier : 메시지 송신자와 관련된 참조 식별자

4) Priority.Indicator : 메시지 송신자가 할당한 메시지에 대한 우선권 지시자

5) Confirmation.Code : 메시지 송신자가 기술하는 확인 요청 코드

6) Authorization.Identifier : 메시지 송신자의 인증 식별자 기술

3.5.3 데이터 영역의 설계 : 데이터 영역은 실제 전자문서의 내용이 들어가는 부분으로 한번 이상 나타난다. 본 가이드라인에서는 데이터 영역에 대해서 본 가이드라인에서 정의한 전자문서 이외에 여타 전자문서도 포함될 수 있도록 데이터 영역에 포함된 전자문서를 나타내는 속성으로 세가지 속성을 부여하였는데,

1) payloadType : 해당 데이터 영역의 전자문서의 유형을 구분하는 지시자로 EDI 일 경우, EDIFACT 메시지인지, ANSI X.12 메시지인지 구분하며, XML의 경우 W3C XML Schema인지, XML DTD 인지, RELAX NG 인지, 기타 스키마 유형인지 구분하는 지시자이다.

2) payloadScheme : 해당 데이터 영역의 전자문서 구조를 구분하는 지시자로 EDI Document 인지, XML Document인지를 구분하는 지시자이다.

3) controlAgency : 해당 데이터 영역의 전자문서 유형을 관리하는 기관에 대한 식별자를 기술하는 부분이다.

3.5.3.1 데이터 영역은 크게 전자문서의 헤더와 라인과 요약부분으로 구분하여 표현한다.

1) 헤더 부분(Header)

- 문서 전체에 공통으로 적용될 내용으로 구성한다.

2) 본문 부분(Line)

- 전송할 내용 자체는 모두 이 부분에 수록되도록 한다.
- 주로 메시지에서 표현되는 물품이나 서비스와 관련된 정보를 나타내는 부분이며, 헤더부분에서 공통으로 적용되는 배송이나, 지불 등과 같은 조건이 맞지 않은 경우에는 별도로 해당 정보를 표현한다.

3) 요약 부분(Summary)

- xCBL이나 EDIFACT의 요약은 총계정보 및 유효성 검증 정보로 구성된다. XML로 문서를 작성한다면 XML 유효성 검증기 및 어플리케이션 처리로 자동 집계 및 유효성 검증이 가능하므로 필요치는 않지만, 해당 항목이 거의 대부분 오프라인 문서에 존재하기 때문에 기본적으로 총액, 총 수량 등 기본 정보만을 포함한다.
- 그러나 납품 통지, 거래처 정보 등과 같이 필요하지 않은 경우에는 요약부분을 삭제하는 것을 허용한다.

4. 전자문서 개발 관련 Template 및 운영 전략

4.1 전자문서 개발 관련 Template

4.1.1 전자문서 개발 관련 템플릿을 제시하는 이유는 각 업종별/부문별로 전자문서를 개발하는 절차나 포맷이 제각기 다르기 때문에 겪는 혼란 때문이다. 업종의 표준화 작업은 쉽고 금방 끝나는 작업이 아니다. 표준이라는 것이 단기간에 만들고 끝나는 작업이 아니라 표준의 유지보수 작업이 계속적으로 이루어져야 하며, 업종내 업체들의 모든 현황을 고려하여야 한다. 그리고 어플리케이션 개발자나 실무담당자들이 쉽게 이해할 수 있도록 논리적이고 체계적인 템플릿이 필요하다.

4.1.2 본 가이드라인에서는 업종별/부문별 전자문서 표준화 담당자들이 원활하게 작업할 수 있도록 최적화된 템플릿을 제공하고자 한다. 템플릿중에는 필수 템플릿과 선택 템플릿으로 구분하여서 업종 전자문서 표준화를 위해서 반드시 작성하여야 하는 템플릿과 선택적으로 작성할 수 있는 템플릿을 구분하였다.

4.1.3 다음 아래와 같은 표를 보면 업종 전체적으로 사용되는 전자문서 개발 관련 템플릿을 설명하고 있다. 이 템플릿들은 개발하는 전자문서가 XML 이든지, EDI 인지 구분없이 전체적으로 개발하여야 한다.

템플릿명	템플릿 정의	필수/선택	비고
업종 마스터 항목 테이블	업종 표준으로 정의한 전체 항목을 한곳에 모아놓은 테이블	필수	
업종 전자문서 리스트	업종에서 개발 대상으로 선정한 전자문서 리스트	필수	
업종 코드 리스트	업종에서 사용하는 코드 리스트	필수	
업종 코드 테이블	위의 코드 리스트 각각에 대한 코드값까지 기술한 코드 테이블	필수	
공통 코드 테이블	업종을 초월하여 일반적으로 사용하는 국제코드나 공통 코드 테이블	필수	진흥원 공통 코드 차용
업종 전문용어 테이블	업종에서 전문적으로 사용하는 용어에 대한 정의를 해놓은 테이블	선택	
신규 개발 목록	진흥원에서 개발한 공통 라이브러리에는 존재하지 않고 업종에서 새로이 신규 개발한 컴포넌트, 정보개체, 식별코드 등을 기술한 목록	필수	

4.1.4 아래와 같은 표는 XML 전자문서를 개발하기 위한 템플릿이다. 만약에 EDIFACT 전자문서를 개발하고자 할 경우에는 전자거래진흥원에 요청을 하면 된다.

템플릿명	템플릿 정의	필수/선택	비고
전자문서 개요	해당 전자문서의 정의와 활용 방법 및 관리정보를 포함하고 있는 템플릿	필수	
전자문서 항목 테이블	전자문서를 구성하고 있는 항목에 대한 테이블	필수	
전자문서 UI(인터페이스)	해당 전자문서의 필수 항목들을 조합하여 인터페이스를 만들어 놓은 부분인데, 실무담당자와의 원활한 커뮤니케이션을 위한 템플릿	선택	
메시지 구현 Simple Spec	해당 전자문서를 구성하고 있는 컴포넌트와 정보개체를 기준으로 항목을 매칭 시켜놓은 테이블	필수	
정보 개체 리스트	해당 전자문서에서 사용하고 있는 컴포넌트, 기본정보개체, 코드를 기술하는 템플릿	선택	
컴포넌트 테이블	해당 전자문서에서 사용하고 있는 컴포넌트에 대한 세부적인 테이블	필수	
코드 테이블	해당 전자문서에서 사용하고 있는 코드에 대한 세부적인 테이블	필수	

4.1.5 아울러 위와 같은 템플릿을 작업한 것으로 끝나지 않으며, 위의 템플릿 이외에 기본적으로 XML 스키마와 XML 샘플 문서가 첨부되어야 할 것이다.

4.1.6 **업종 마스터 항목 테이블** : 업종 표준으로 정의한 전체 항목을 한곳에 모아놓은 테이블로 이 테이블에서는 한글로된 데이터 항목만을 사용한다. 즉, 전자문서 구문인 EDIFACT 구문이나 XML 태그가 포함되지 않고, 비즈니스 정보 단위를 항목으로 기술한다.

업종 마스터 항목 Table

(관리: +:신규 데이터 -:삭제 예정, 유형: S:String N:Number C:Code, 코드표:업종 식별코드 테이

관리	UID	항목명	정의	유형	길이	코드표	견적서	견적요청	주문서	주문변경

1) 관리 : 해당 정보 항목의 관리 상태를 기술하는 부분으로

+:신규 데이터 -:삭제 예정 으로 표시하여, 정보 항목의 추가나 삭제의 상태를 기술한다.

2) UID : 해당 정보 항목의 UID를 기술하는 부분으로 해당 업종에서 총괄하는 데이터 항목에 대한 UID를 기술한다. 여기서 기술된 UID는 전자문서 개발시 사용하는 전자문서 데이터 항목의 UID와 같아야 한다.

3) 항목명 : 해당 업종에서 사용하는 정보 항목을 기술하는 부분으로 비즈니스 정보 항목을 기술하여야 한다. 예를 들어, 거래당사자 식별자라고 하지 않고, 정확한 한정어까지 기술하여야 한다. 구매당사자 식별자, 구매당사자 주소, 구매당사자 사업자 등록번호, 공급당사자, 식별자, 공급당사 주소, 운송자 식별자 등

4) 정 의 : 해당 정보 항목에 대한 정의를 기술하는 부분이다. 해당 정보 항목에 대한 명확한 정의를 기술하여, 처음 접하는 사람도 이해할 수 있도록 구체적으로 기술한다.

5) 유 형 : 해당 정보 항목의 데이터 유형을 기술한다.

S:String, N:Number, C:Code 로 표시하여, 정보 항목의 데이터 유형을

기술한다.

6) 길 이 : 해당 정보 항목의 길이를 기술하는 부분이다.

7) 코드표 : 해당 정보 항목이 코드일 경우, 식별코드테이블에 있는 해당 코드표의 번호를 기술한다.

8) 전자문서 목록 : 해당 정보 항목이 속해 사용되는 전자문서를 체크하는 부분이다.

4.1.7 **업종 전자문서 리스트** : 업종에서 개발한 전자문서 리스트이다. 해당 전자문서에 대한 정의를 명확하게 기술하여야 한다.

대상 전자문서 Table

(관리: +:신규 메시지 -:삭제 예정)

관리	UID	전자문서명	정 의	참조메시지유형	유사메시지명

1) 관리 : 해당 메시지의 관리 상태를 기술하는 부분으로

+:신규 데이터 -:삭제 예정 으로 표시하여, 메시지의 추가나 삭제의 상태를 기술한다.

2) UID : 해당 메시지의 UID를 기술하는 부분으로 해당 업종에서 총괄하

는 메시지에 대한 UID를 기술한다.

3) 전자문서명 : 해당 업종에서 선정된 전자문서명을 기술한다.

4) 정 의 : 해당 전자문서에 대한 정의를 기술하는 부분이다. 해당 전자문서에 대한 명확한 정의를 기술하여, 처음 접하는 사람도 이해할 수 있도록 구체적으로 기술한다.

5) 참조 메시지 유형 : 해당 전자문서가 참조하는 메시지 유형을 기술한다.

6) 유사 메시지명 : 해당 전자문서와 비슷한 기능을 하는 메시지들을 기술한다.

4.1.8 **업종 코드 리스트** : 업종에서 사용되는 코드 리스트이다. 코드로 기술되는 모든 정보항목을 기술하여야 한다.

업종 코드 List

(관리: +:신규 코드 -:삭제 여경)

관리	UID	코드항목명	정 의	공통/업종 구분	코드 값 리스트

1) 관리 : 해당 코드 항목의 관리 상태를 기술하는 부분으로

+:신규 코드 -:삭제 예정 으로 표시하여, 코드항목의 추가나 삭제의 상태를 기술한다.

2) UID : 해당 코드의 UID를 기술하는 부분으로 해당 업종에서 총괄하는 메시지에 대한 UID를 기술한다.

3) 코드항목명 : 해당 업종에서 사용되는 코드항목명을 기술한다.

4) 정 의 : 해당 코드항목에 대한 정의를 기술한다.

5) 공통/업종 코드 : 해당 코드항목이 공통 코드인지, 업종코드인지 기술한다.

6) 코드값 리스트 : 해당 코드 항목이 어떤 코드 리스트을 참조하였지 기술한다.

4.1.9 **업종 코드 테이블** : 업종에서 사용하는 코드 리스트 각각에 대한 코드값까지 기술한 코드 테이블

업종 코드 테이블					
관리	UID	코드항목명	코드값	코드값 정의	비 고

1) 관리 : 해당 코드 항목의 관리 상태를 기술하는 부분으로

+:신규 코드 -:삭제 예정 으로 표시하여, 코드항목의 추가나 삭제의 상

태를 기술한다.

2) UID : 해당 코드의 UID를 기술하는 부분으로 해당 업종에서 총괄하는 메시지에 대한 UID를 기술한다.

3) 코드항목명 : 해당 업종에서 사용되는 코드항목명을 기술한다.

4) 코드값 : 해당 코드항목의 코드값을 기술한다.

5) 코드값 정의 : 해당 코드항목의 코드값에 대한 정의를 기술한다.

6) 비　고 : 해당 코드항목의 코드값에 대한 비고를 기술한다.

4.1.10 **공통 코드 테이블** : 업종을 초월하여 일반적으로 사용하는 국제 코드나 공통 코드를 기술하는 코드 테이블로 바로 위에서 기술한 업종 코드 테이블과 똑같은 테이블 구조를 가진다.

공통 코드 List					
관리	UID	코드항목명	코드값	코드값 정의	비　고

4.1.11 **업종 전문용어 테이블** : 업종에서 전문적으로 사용하는 용어에 대해서 상세하게 정의를 해놓은 테이블

업종 전문용어 List			
관리	업종 전문용어	정 의	비 고

1) 관리 : 해당 코드 항목의 관리 상태를 기술하는 부분으로

+:신규 코드 -:삭제 예정 으로 표시하여, 코드항목의 추가나 삭제의 상태를 기술한다.

2) 업종 전문용어 : 업종/부문에서 사용하는 전문적인 용어명을 기술한다.

3) 정 의 : 전문용어에 대한 상세한 정의를 기술한다.

4) 비 고 : 해당 전문용어에 대한 비고 기술

4.1.12 **신규 개발 목록** : 진흥원에서 개발한 공통 라이브러리에는 존재하지 않고 업종에서 신규 개발한 컴포넌트, 정보 개체, 코드 등을 기술한 목록

4.1.13 **전자문서 개요** : 해당 전자문서에 대한 상세한 정보 및 관리 정보를 기술하는 부분이다.

1. 기본정보
1.1 기능적 정의
판매자와 구매자간에 상품의 주문과 이에 관련된 서비스에 대한 상세사항을 전송할 수 있도록 하는 전자문서이다. 아울러, 국내에서 상품구매시 이용되고 있는 매입전표나 거래명세서 역시 본 주문서를 통해 구현할 수 있다.
1.2 적용 범위
본 전자문서는 W3C의 XML 스키마로 개발되었으며, 현재는 국내를 대상으로만 사용할 수 있다. 국제적으로는 상대방과의 상호 조건이나 상호 협약이 먼저 선행이 되고 나서 사용될 수 있을 것이다.
1.3 적용 원칙 및 활용 방법
1) EDIFACT와 ANSI X.12의 주문서 및 기타 전자문서 표준스펙의 주문서와 서로 연동할 수 있어야 한다. 2) 업종이나 국가를 불문하고, Horizontal하게 적용될 수 있어야 한다. 3) 확장성을 최대한 보장하여야 하며, 문서의 프레임워크가 체계적으로 정리되어 있어야 한다. 4) 주문서는 이전의 계약이나 구매 요청서를 참조하여 자동으로 Generate 될 수 있거나 시스템 자동화와 관련된 여러 참조 항목들을 가지고 있어야 한다. 5) 타 전자문서 표준의 주문서와 연동시 정보 손실이 일어나더라도, 최소화하거나 대치할 수 있어야 하며, 이와 관련된 사항은 전자문서 개발 명세에 반드시 포함하여야 할 것이다. 6) 주문서는 XML 전자문서 포맷으로 개발된다. 그러나 현재 XML 전자문서 표준이 국제적으로나 국내적으로 없는

1) 기능적 정의 : 해당 전자문서가 수행하는 기본 기능에 대해서 기술하는 부분이다.

2) 적용 범위 : 해당 전자문서가 적용되는 범위를 기술하는 부분이다.

3) 적용 원칙 및 활용 방법 : 해당 전자문서가 사용되는 적용 원칙 및 전자문서의 활용 방법에 대해서 기술한다.

4) 문서 흐름 : 해당 전자문서가 사용되는 비즈니스 거래형태와 거래 당사자들에 대한 정보를 기술한다.

5) 관련 컨텍스트 : 해당 전자문서의 비즈니스 컨텍스트를 기술한다.

6) 참고 자료 : 해당 전자문서에 대한 참고 자료를 기술한다.

7) 타 규격현황 : 해당 전자문서와 관련된 타 규격 현황을 기술한다.

8) 등록자 정보 : 해당 전자문서를 등록하고자 하는 등록자 정보를 기술한

다.

9) 관리자 정보 : 해당 전자문서를 관리하는 관리자 정보를 기술한다.

10) 변경 정보 : 해당 전자문서의 변경과 관련된 변경 연혁을 기술한다.

11) 첨부 자료 : 해당 전자문서와 관련된 첨부 자료를 기술한다.

4.1.14 **전자문서 항목 테이블** : 해당 전자문서를 구성하고 있는 전자문서 항목에 대한 항목 테이블로, 각 항목별 태그명과 정의, 유형, 길이, 필수/선택 등 여러 가지 정보를 기술한다.

전자문서 항목 Table (관리 : +:신규 항목 -:삭제 예정, 유형 : S:String N:Number C:Code, M/O: M:필수									
관리	UID	항 목 명			TAG명	정의	유형	길이	M/O
		대분류	중분류	소분류					

1) 관리 : 해당 전자문서 데이터 항목의 관리 상태를 기술하는 부분으로 +:신규 데이터 -:삭제 예정 으로 표시하여, 정보 항목의 추가나 삭제의 상태를 기술한다.

2) UID : 해당 전자문서 데이터 항목의 UID를 기술하는 부분으로 해당 업종에서 총괄하는 데이터 항목에 대한 UID를 기술한다. 여기서 기술된 UID와 해당 업종에서 총괄하는 데이터 항목의 UID와 같아야 한다.

3) 항목명 : 해당 전자문서에서 사용하는 정보 항목을 기술하는 부분으로 비즈니스 정보 항목을 기술한다.

4) 태그명 : 해당 전자문서 데이터 항목에 대한 태그명을 기술하는 부분이다.

5) 정 의 : 해당 전자문서 데이터 항목에 대한 정의를 기술하는 부분이다. 해당 전자문서 데이터 항목에 대한 명확한 정의를 기술하여, 처음 접하는 사람도 이해할 수 있도록 구체적으로 기술한다.

6) 유 형 : 해당 정보 항목의 데이터 유형을 기술한다.

S:String, N:Number, C:Code 로 표시하여, 정보 항목의 데이터 유형을 기술한다.

7) 길 이 : 해당 정보 항목의 길이를 기술하는 부분이다.

8) M/O : 해당 전자문서 데이터 항목에 대한 필수/선택을 기술한다.

9) Occurs : 해당 전자문서 데이터 항목의 발생 횟수를 기술한다.

10) 코드표 : 해당 전자문서 데이터 항목이 코드일 경우, 식별코드테이블에 있는 해당 코드표의 번호를 기술한다.

11) 비 고 : 해당 전자문서 데이터 항목에 대한 비고를 기술한다.

4.1.15 **전자문서 UI** : 해당 전자문서의 필수 항목들을 조합하여 인터페이스를 만들어 놓는 것으로, 실무담당자와의 원활한 커뮤니케이션을 위한 템플릿으로 정해진 규격은 없다. 아울러, 이 UI는 반드시 작성할 필요는 없다.

4.1.16 **메시지 구현 Simple Spec** : 해당 전자문서를 구성하고 있는 컴포넌트와 정보개체를 기준으로 항목을 매칭시켜놓은 테이블

관리	UID	전자문서 구조		전자문서 구성 데이터 항목
		Section	표준전자문서 Tag명	
		헤더영역	**DocumentIds**	
			LastModification,DateTime	
			Document,DateTime	
			Document,Name	
			Description,Text	
			Note	
			DocumentReferences	
			Attachments	
			OrderStatus	
			SpecialPriceAuthorization,Identifier	
			TaxWithholdingExempt,Indicator	
			License,Indicator	
			FreightClass,Identifier	
			ShipNote	
			DropShip,Indicator	
			BackOrdered,Indicator	
			ShipPriorToDueDate,Indicator	
			Priority,Indicator	
			Reason,Code	
			EarliestShip,DateTime	
			NeedDelivery,DateTime	

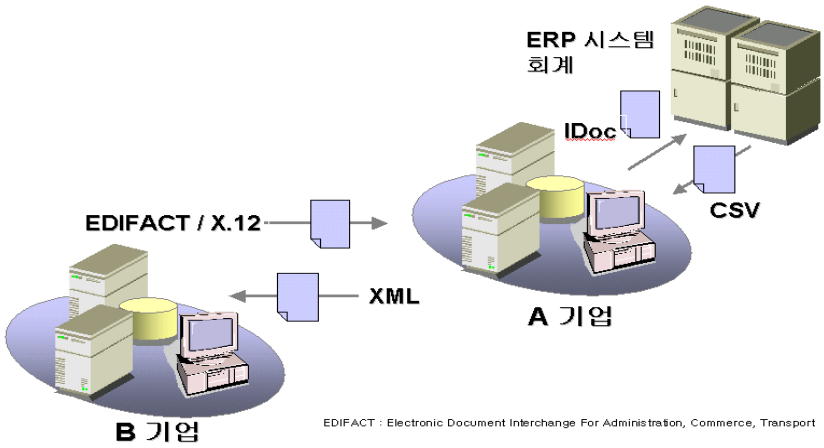
4.1.17 **정보 개체 리스트** : 해당 전자문서에서 사용하고 있는 컴포넌트, 기본 정보 개체, 식별코드를 기술하는 템플릿

4.1.18 **코드 테이블** : 해당 전자문서에서 사용하고 있는 식별코드에 대한 세부적인 테이블이다.

4.2 전자문서 레벨의 상호 연동 방법

4.2.1 기업간, 시스템간 전자문서 교환이 이루어지기 위해서는 서로간의 프로토콜, 메시징, 전자문서 구조 및 의미에 대한 거래 상대방간 약속이 이루어져야 한다. 다수의 거래자들이 1개의 중앙 시스템을 활용하는 경우에는 공통의 시스템을 사용하기 때문에 중앙시스템에 구축된 메시지 핸들링 서버에 대한 규약에만 따르기만 하면 된다. 그러나 2개 이상의 시스템간 전자문서 교환이 이루어질 경우에는 각 시스템의 프로토콜과 메시징, 전자문서 구조 및 의미에 대한 약속이 이루어져야 한다. 1:1 거래의 경우에는 서로간에 부적합 부분이 있더라도 약간의 수정작업을 통하여 연동하기 쉽지만, N:N의 거래일 경우에는 기하 급수적인 노력과 자금을 필요로 하게 된다.

4.2.2 본 가이드라인에서는 하드웨어나 어플리케이션영역은 제외하고, 전자문서 레벨에서의 상호 연동에 대한 방안을 제시하고자 한다. 전자문서 레벨의 상호 연동 방법이라고 했을 때 주 대상 영역은 전자문서 구조에 대한 서로간의 약속과 전자문서 내용에 대한 서로간의 약속이 체결되어져 거래당사자들이 약속된 방법대로 상대방에게 전자문서를 송수신 하여야 한다는 것이다.

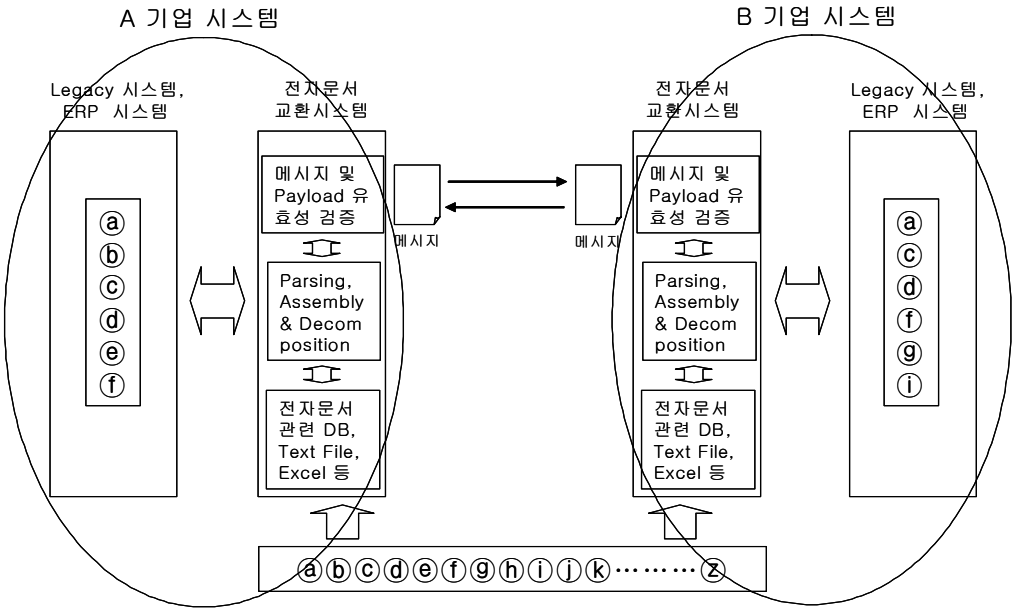


4.2.3 전자문서 레벨의 상호 연동 방법에 있어서 중요한 요소들로는 1) 전자문서 구조, 2) 전자문서 구성 데이터 항목, 3) 전자문서에서 사용되는 식별자 및 코드리스트의 동일성이다. 이 3가지 요소들이 모두 일치 할 경우

에는 시스템간 상호 연동이 원활하게 이루어 질 수 있지만, 실제로 이러한 경우는 극히 드물다. 위의 항목중 어느 하나라도 서로간에 다를 경우에는 연동을 위한 매핑 작업이나 유지/보수 작업을 하여야 한다.

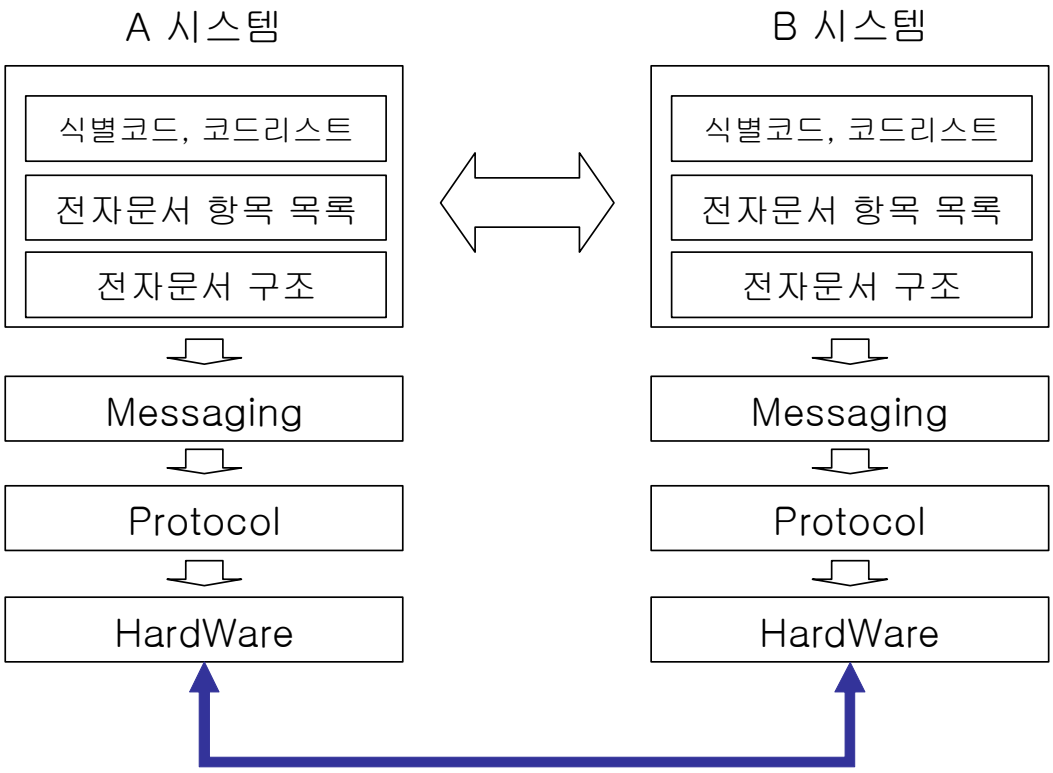
4.2.4 전자문서 구조가 동일하여야 한다는 것은 서로 다른 비즈니스 어플리케이션에서 전자문서를 처리하는 방법을 동일하게 한다는 것이다. 예를 들어 A라는 업체가 사용하는 구매 주문서와 B라는 업체가 사용하는 구매 주문서가 같은 전자 문서 구조를 가졌을 경우, 서로 다른 구매 주문서를 처리하기가 용이하지만, 서로 다른 전자문서 구조로 이루어져 있다면, A, B 업체 모두 상대방의 처리 방법이나 기타 다른 방법을 가져야 한다

4.2.5 전자문서 구성 데이터 항목이 동일하여야 한다는 것은 서로간의 다른 비즈니스 어플리케이션에서 사용하는 항목들이 다르다는 것을 의미한다. 사실 업종내 업체들간에 같은 데이터 항목들을 사용한다는 것은 사실 찾아보기 힘들다. 그러하기 때문에 사용하는 항목이 다를 경우에 전자문서 교환에서 정보의 손실은 불가피하다. 그렇지만, 본 가이드라인에서는 업종 표준 데이터 항목 목록을 각 시스템에서의 전자문서 교환시스템에서 모두 구현할 것을 권고한다. 그럴 경우에는 나중에 시스템간 연동작업이 더 필요할 때 유지 보수작업이 훨씬 용이해진다.



업종 표준 데이터 항목 목록(필수, 선택항목 포함)

4.2.6 전자문서에서 사용하는 식별자와 코드리스트의 동일성은 물품 식별번호나 분류 코드, 거래처 식별번호, 기타 각 업종별/업체별로 식별코드화 해놓은 정보가 각 시스템간 동일하여야 함을 의미한다. 사실 각 시스템 독자적으로 자재번호나 단품번호, 식별번호 등을 코드화하여 사용하는 경우가 많다. 특히 유통부문의 경우에는 생산자 식별번호, 운송업자 식별호, 유통업체 식별번호 등 하나의 물품을 가지고 다양하게 식별번호 체계를 가지는 경우도 있다. 이러한 식별 코드의 중복이나 복잡성은 시스템간 상호연동을 저해하는 중요한 요소가 된다. 그렇기 때문에 업종별 전자문서 표준화 작업에 있어서 중요 작업의 하나는 업종내 사용되는 식별 코드의 표준화 작업이다. 본 가이드라인에서는 식별 코드의 표준화 작업이 그렇게 순탄하게 이루어지는 경우가 많지 않지만, 궁극적으로는 업종내 식별코드의 표준화를 통한 상호연동을 권고하지만, 임시적으로는 각 식별 코드의 식별 코드리스트 식별과 코드리스트 관리기관 등을 명확하게 기술하여 식별코드의 혼란을 최소화할 것을 권고한다.



4.3 XML 라이브러리 운영 및 업그레이드 지침

4.3.1 XML 라이브러리는 크게 4개의 계층으로 구성된다. 가장 상위로 XML 메시지를 규정하며, 그 다음 단계로 컴포넌트, 컴포넌트를 구성하는 기본 정보 개체, 기본 정보 개체에서 사용되는 식별 코드리스트와 일반 코드리스트로써 라이브러리는 구성된다.

4.3.2 국내에는 이미 각 업종별/부문별로 자체적으로 XML 관련 메시지나 컴포넌트, 기본 정보 개체들을 개발한 분야도 있으며, 지금 개발하고자 하는 분야도 있다. 중복되는 메시지나 컴포넌트, 기본 정보개체가 있을 수 있으며, 다양한 문제가 있을 수 있다. 이러한 복잡성을 단순화 시키고, 체계적으로 관리하기 위해서는 공통 표준과 업종별/부문별 표준으로 구분하여 관리할 필요가 있다.

4.3.3 각 메시지에 대해서는 컨텍스트 구분을 해서 관리하며, 컴포넌트나 기본 정보 개체의 경우에는 네임스페이스를 통해서 구분한다.

예를 들어, 국내에 공통으로 적용할 수 있는 메시지나 컴포넌트, 정보 개체의 경우에는

Geopolitical.Identifier="KR"이 될 수 있으며,

IndustryClassification.Identifier="All Context" 가 될 수 있을 것이다.

그러나 업종 부문별로만 적용될 수 있는 버티컬 메시지나 컴포넌트, 정보 개체의 경우에는

Geopolitical.Identifier="KR"이 될 수 있으며,

IndustryClassification.Identifier="TextTile, Electronics" 로 구분하여 버티컬 영역을 식별하면 될 것이다. 아울러, 업종 부문별로 개발한 컴포넌트나 개체인 경우에는 네임스페이스를 기술하여 구분짓는 방식으로 관리하여야 한다.

4.3.4 각 업종별/부문별로 산출된 결과물에 대해서는 고유한 네임스페이스 할당을 통해서 구분하도록 한다.

예를 들어 섬유업종인 경우 xmlns:tex:http://www.kofoti.or.kr 이라고 하며, 전자업종인 경우 xmlns:elec:http://www.eiak.or.kr 이라고 할 경

우, 컴포넌트와 기본 정보 개체는 tex 와 elec를 단어앞에 사용하여, 구분하도록 한다.

4.3.5 XML 라이브러리가 반드시 하나일 수는 없다. 이미 외국의 경우에는 국제표준을 획득한 XML 스펙은 없지만, 국제 표준화 단체에 의해 릴리즈된 XML 스펙은 여러 가지 나와있으며, 국내의 경우에도 정부 조달시스템을 구축하면서, XML 라이브러리를 개발하였으며, 건설, 행정쪽에서도 각자의 방식대로 XML 라이브러리를 개발하였다. 이러한 라이브러리는 컨텍스트 구분자와 네임스페이스를 할당하여 구분 관리하여야 한다.

4.3.6 업종별로 구축된 라이브러리의 경우에는 그 업종의 특수한 업무나 조건 등에 맞춰서 개발하였기 때문에 타 업종에서 사용하기에 애로사항이 있다. 그러기 때문에 본 가이드라인은 업종간에 공통적으로 사용하기 위해 개발된 전자거래진흥원의 XML 라이브러리를 권고한다.

4.3.7 아울러, 전자거래진흥원 혹은 기타 조직에서 국내외 업종/분야의 XML 라이브러리 및 관련 자료를 총괄하여, 체계화되고, 정보화된 표준 운영 및 관리, 서로간의 중복 방지, 업종별/부문별 연동방안 마련, 국제 표준과의 호환 방안 마련 등의 업무를 수행하여야 할 것이다.

4.3.8 XML 라이브러리 업그레이드와 관련해서는 크게 두가지 유형으로 나눠서 구분할 수 있을 것이다.

1) XML 라이브러리에 대한 추가 작업이나 구조적인 변경이 아닌 경우에는 버전의 소숫점단위를 순차적으로 증가시킨다.
2) XML 라이브러리에 대한 대폭적이거나 구조적인 변경이 이루어 질 경우에는 버전의 정수단위를 순차적으로 증가시킨다.

4.3.9 XML 라이브러리 업그레이드는 다음과 같은 사항들의 변경 사항이 있을 경우 이루어 질 것이다.

1) XML 메시지의 신규 추가, 삭제, 개정
2) XML 컴포넌트의 신규 추가, 삭제, 개정
3) 기본 정보 개체의 신규 추가, 삭제, 개정

4) 코드리스트의 신규 추가, 삭제, 개정

5) XML 라이브러리 구성과 관련된 일반 지침의 개정

4.3.10 XML 라이브러리는 매년 수정, 변경된 사항을 한국전자문서 교환위원회(Korea EDIFACT Commitee)에 보고하여, 승인을 받은 후 연말경에 발행할 것이다.