□ 기술해설 □

CALS에서의 문서 표준 SGML

현대미디어시스템 현득창*
배재대학교 정회경**
광운대학교 이수연***

● 목 차 ●

1. 서 론

다가오는 21세기는 고도화된 정보화사회가 될 것이다. 이러한 고도의 정보화 추세에 가장 필수적인 요소가 기존 및 앞으로 발생되는 문서정보를 전자문서로 만들고 관리하는 작업이다. 이러한 측면이 가장 극명하게 나타나는 것이 공공기관의 각종 서류 및 자료, 기업에서 처리되는 각종 서류등이 현재는 기존 프린트된 문서들로 서로 교환하고 사용하기에 그 시간적 비용적인 손실이 매우 컸다. 따라서 각 공공기관이나 기업에서는 이러한 비효율성을 감소시키고자 효율적이고 비용 절감할 수 있는 방안으로 모든 문서처리의 전산화를 서둘러 실시하고 있는 실정이다.

미국에서는 이러한 일환으로 미국방성이 군수 지원과정에 소요되는 시간과 비용을 절감하기 위해 군수 조달 및 관리정보시스템(Computer-aided Acquisition Logistic Support : CALS) 전략을 수립하였고, 현재는 점차 민간, 특히 제조업 분야에 응용되고 있으며 따라서 의미도 경영 지원 통합 정보시스템(Continuous-Acquisition and Life-cycle Support)으로 변하고 있다. 그런데 이 CALS 전략이 바로 서류와의 전쟁에서 파생된 것이라고 말할 수 있다. 현재 우리나라에서도 단순한 지역적 의미의 전산화가 아닌 전역적인 의미에서의 통합정보시스템을 구축하기 위하여 CALS의 도입을 적극적으로 검토하고 있다.

한편, 문서정보를 전자화하고 이 기종 시스템간에 서로 상호교환하기 위해서는 문서를 표준적인 형태로 저장하는 것이 중요한데, CALS에서의 문서정보 표현에 관한 표준을 독자적으로 제정한 것이 아니라 지금까지의 표준을 선택적으로 채용하고 있다. 즉 이미 널리 채택된 표준을 CALS에서는 추구하고 있다. 따라서 CALS에서는 문서 양식 표준으로 현재 전세계적으로 가장 널리 채택되어 사용되고 있는 국제표준 ISO 8879 : SGML(Standard Generalized Markup Language)을 공식적으로 문서표준 SGML-28001로 채택하고 있다.

본 논고에서는 현재 CALS에서뿐만 아니라 각 기업 및 공공 기관에서 채용하려고 하는 SGML의 소개와 사용실례 그리고 미래에 대해 살펴본다.

* 학생회원
** 정 회 원
*** 종신회원

2. 문서정보 표현방식의 중요성

지금까지는 컴퓨터를 이용하여 문서정보를 작성할 때, 특정 시스템에 의존적으로 작성되어 이 기종 시스템에서 사용하기가 불가능하거나, 만일 이 기종 시스템에서 사용하려면 문서정보를 변형하여 사용하는 수 밖에 없었다. 이것은 문서 자체의 물리적 외형과 밀접하게 관련되어 있어서 문서정보가 상호간에 일치하지 않기 때문이다. 특히 문서에서의 정보는 매우 복잡한 형태의 외형정보를 갖고 있어 더욱 시스템 의존적이라 할 수 있다. 따라서 시스템 독립적인 정보의 표현방법이 매우 중요하게 된다. 따라서 정보표현방식의 선택은 신중해야한다.

통상, 문서정보는 두가지로로 크게 구분할 수 있다. 첫째로 문서 자체이다. 이것은 일반 텍스트 및 테이블, 그림, 사진, 그래픽과 같은 순수한 정보데이타를 의미한다. 두번째로는 문서의 구조적 정보이다. 현재까지는 문서의 구조정보는 문서를 이루고 있는 다양한 요소들의 외형과 위치의 차이(물리적 외형정보)에 의해 표현되었는데, 외형 그 자체로는 문서의 정확한 구조적 정보를 전달하지 못한다. 바로 이점이 문서정보의 상호교환 및 공유를 어렵게 했던 것이며, 이렇게 상호 교환과 공유에 발생되는 어려움을 해결하기 위한 방법으로 문서정보를 계층적인 논리구조관계를 갖는 오브젝트(장, 절, 단락, 표제 등)들로 표현하는 새로운 개념이 도입된 것이다. 즉, 문서의 물리적 외형적 구조보다는 논리적 구조를 가지고 정의함으로서 상호 교환 및 공유시 발생되는 문제점을 해결하여 효율적인 문서정보의 관리가 이루어지도록 하고 있다.

3. 문서정보의 표준 동향

문서정보가 논리적인 계층구조를 가지고 작성되도록 하기 위하여 ISO(International Standard Organization)에서는 데이타 객체 양식 표준으로써 ISO/IEC 8869 : Office Document Architecture(ODA)와 ISO/IEC 8879 : Information Processing-Text and Office Systems -Standard Generalized Markup Language (SGML) 제정하였다. ODA는 원래 European Computer Manufacturers Association (ECMA)에서 개발이 되었고, 문서의 상호 통신교환을 목적으로 문서의 제한된 논리구조와 배치구조(layout structure)를 병행적으로 구성할 수 있다. 이에 비해 SGML은 문서정보의 논리적 구조를 표현하는 마크업의 일관성을 강조하고 있다. 따라서 ODA와 SGML는 성격이 매우 다른 표준이며, 굳이 구분한다면, ODA는 비교적 간단한 문서구조를 갖는 문서를 작성할 수 있고, 복잡한 문서의 구조 표현에는 부적합한 면이 있으나, SGML은 개념적인 문서의 논리 구조만을 가지고 있어 어떤 복잡한 문서들도 작성할 수 있다. 현재 ODA는 유럽을 중심으로, SGML은 미국을 중심으로 사용되고 있으나 점차 SGML의 활용이 더 우위를 차지해 가고 있는 추세이다.

현재 인터넷상에서 활용되고 있는 HTML은 이러한 국제 표준인 SGML의 규격을 따르는 상용표준이라고 할 수 있다. 좀 더 정확히 표현한다면 HTML은 SGML의 한 응용이라고 할 수 있다. 현재 HTML은 특정 DTD을 설정하고 있으며, 계속 발전, 보강되고 있는 실정이다.

4. SGML의 사용현황

현재, 미국에서는 SGML에 관한 연구가 활발하여, 문서의 공유와 문서 교환을 위해서 문서 구조에 대한 표준화가 IEEE, 미국화학회, 수학회등 많은 학회와 미국 출판 연합회에 의하여 진행되어 EMP(Electronic Manuscript Project)가 개발되었다. 이것은 SGML 응용레벨의 표준으로서 ANSI 규격인 Z39.59로 되었으며 기본적인 문서 형태가 규정되어 있다. 또 하나의 큰 프로젝트로서 미국 국방성에 의하여 개발된 CALS 전략으로, 이것은 미국 국방성의 조달 물자에 대한 방대한 정보를 전자화하기 위한 것으로 1990년 부터 단계적으로 실시되고 있고, 그 규격은 문서 정보에 SGML을 채용하였다. 이와같이 미국과 캐나다에서는 SGML을 처리할 수 있는 시스템들이 개발되어

속속 발표되고 있는 실정이다.

아시아는 유럽이나 미국에 비하여 보급이 늦어졌지만, 일본에서는 1989년에 SGML 간담회가 발족되어 SGML의 보급활동을 시작하였으며, 일본의 특허청은 1990년부터 특허공문의 전자화에 SGML을 채용하였다. 그리고 학술논문의 전자화에 관하여 SGML에 의한 논문의 데이타베이스화의 검토가 시작되는등 SGML 응용의 움직임이 일어나고 있다. 일본의 표준활동으로서는 ISO 8879의 JIS(Japan International Standard)가 추진, 1992년에 정식표준으로 되었고, 앞에서 기술한 EMP에 관해서 JIS화가 진행되고 있다. 우리나라에서도 SGML의 중요성을 인식, SGML을 표준화하여, 공표하기만 하면된다. 하지만 아직까지는 SGML관련 응용을 개발하는 수준에 이르고 있지 못하고 있다.

이에 비해 HTML은 WWW가 널리 보급됨에 따라, 1993년부터 전세계적으로 널리 알려지기 시작했으며, 우리나라에서는 1995년에 활성화가 되어 HTML에 관해 관심을 갖게 되었다. 하지만 HTML이 SGML의 응용이라는 그 근원을 잘 알지 못하고 있고, 설사 그렇다고 할 지라도 SGML에 대한 연구가 거의 없었고 국내에 알려지지 못한 실정이기 때문에, 단순히 이용하는 수준에 머물고 있는 실정이다.

5. SGML(Standard Generalized Markup Language)

5.1 SGML의 배경

기존의 출판 위주의 텍스트에서 전자 문서형태로 변화해 오면서 문서의 전송과 교환이 중요한 문제로 대두되어 왔다. 이러한 전송이 중요한 문제가 됨으로서 문서에 대한 개념의 정립이 필요하게 되었다. 통상 정보의 구조는 문서로 되어 있는데 이를 시스템간에 전달하는 데에는 여러 가지 방법이 있다. 하지만 문서에서 텍스트와 함께 페이지 단위로 표현되는 정보, 즉 어떤 형식에 맞게 포맷되어 있는지를 나타내는 문서정보를 전송하여 다른 시스템에서 사용하고자 할 때 사용할 수 있는 방법이 없었다. 그것은 문서의 포맷 정보가 물리적인 외형과 너무 밀접하게 관련되어 있어서 상호간에 전송되지 못하였기 때문이다. 이렇게 문서의 외형적 구조에 의존하여 나타나는 구조를 배치구조라고 한다. 이것은 물리적 구성에 기본한 계층 구조화로 포맷된 형식 문서를 표현하는데 사용되며, 그림 1에서 하위 부분에 해당한다. 예를 들면, 간단한 문서라도 하나 이상의 페이지(page)로 구성되며, 이러한 페이지는 컬럼(column)들로 구성되어 있으며, 또한 칼럼은 줄(line)들로 구성이 되어 있다. 즉 우리가 눈으로 보면서 식별이 가능한 외형상의 특징들로 구성되어 있다.

통상 정보의 구조는 문서를 이루고 있는 다양한 요소들의 외형과 위치의 차이에 의해 나타나지만, 외형 그 자체가 구조적 정보를 전달하는 것은 아니다. 즉, 문서의 물리적 구성요소인 쪽, 프레임, 블럭 등이 구조적 정보가 아니라, 오브젝트의 의미(예 : 장, 절, 그림, 단락, 표제, 각주 등)에 기본하여 수정 가능한 형식 문서를 표현하는데 사용하도록 논리성에 의해 구조를 표시하게 된다. 그럼으로써 다른 외형적 특성을 가지고 있지만 사용자에게는 같은 의미를 줄 수 있다. 오브젝트의 의미로 표현된 논리적 구조는 그림 1에서 상위부분이다.

타이핑된 문서와 인쇄된 문서는 외형적으로 각자 폰트라든지, 포맷정보 등의 정보를 가지고 있으나, 그것이 문서의 논리적 구조 정보를 갖지는 못한다. 그러나, 문서에서 오브젝트의 의미 즉, 장(chapter)이라든지 절(section) 표제(heading)와 같은 것들로 표현을 하게 되면 논리적인 구조를 갖는 정보로 표현이 된다. 그림 1에서 보면 논리적 구조는 표제를 가지고 있다. 표제란 사용되는 문서가 어떤 형태이든 사용자로 하여금 다음에 오는 내용이 무엇인지를 알게 한다. 또 좀 더 복잡해지면 표제 밑에 여러 종류의 텍스트들과 인용문, 그림, 주석들이 포함되어서 구성된다. 이렇게 논리적으로 구성된 문서의 구조는 효율적인 문서 교환을 위해 문서내에 마크업 형태로 포함 시키는 개념이 등장했다.

원래, 마크업은 문서 내에서 사용자가 문서를 작성할 때 본문 이외의 추가적인 정보(폰트의 크기, 레이아웃 등)를 첨가하기 위해 포맷

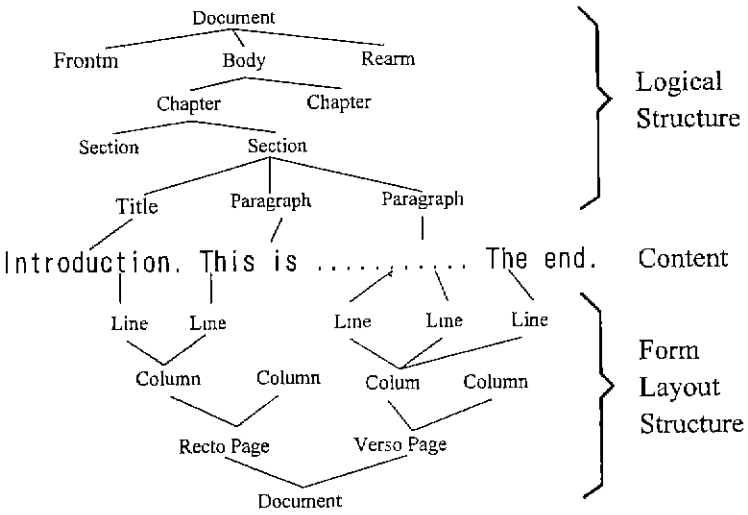


그림 1 문서의 논리구조와 배치구조

명령을 사용하는데 이러한 포맷 명령을 지칭하며, 또한 이러한 배치 성격의 마크업을 절차적 마크업이라 한다. 따라서 마크업 개념이 포맷 정보에서 문서의 논리 구조 정보로 바뀜으로서 문서 교환시 마크업 전달의 일관성을 유지할 수가 있게 되었다. 이런 문서의 논리 구조 정보와 속성을 기술하는 마크업을 범용 마크업이라 한다. 이러한 범용 마크업의 일관성을 강화하기 위해서 ISO에서는 국제 표준안으로 문서 구조를 정의한 SGML을 제정하게 되었다. 이것은 텍스트 안에 사용자가 정의한 텍스트 문자열을 끼워 넣음으로서 구조적 정보를 문서에 포함시킬 수 있게 고안되었기 때문에 SGML은 정보 제공자가 문서 구조의 자세한 내용을 한 시스템에서 다른 시스템으로 전송할 수 있게 하였다.

따라서, SGML은 어떤 표준 마크업 언어(Standard Markup Language)나 포맷터가 아닌 마크업의 개념만을 정의한 메타 언어(Meta Language)이다. 즉, 어떻게 마크업을 제정해야 하느냐를 서술하는 것이지, 마크업이 무엇인지, 그것이 무엇을 의미하는지를 나타내는 수단은 아니다. SGML은 마크업 언어에 대한 추상적 구문을 정의하고, 그 구문을 이용하여 한 문서의 구조를 묘사하는데 사용할 수 있는 범용 마크업 언어의 그룹을 생성하기 위한 표준 메카니즘을 제공하고 있을 뿐이다.

5.2 SGML 엔티티(SGML Entity)

SGML 문서들을 엔티티라는 단위로 표현할 수 있는데, 한 SGML 엔티티는 크게 세부분으로 분류된다. 사용될 언어 및 글자들의 집합을 선언하고, SGML 문서 요소의 수, 토큰 수등의 상한 값들 등을 정하는 SGML 선언부, 한 SGML 문서의 논리적, 계층적 구조를 표현하는 문서형 정의부, 그리고 실제 내용을 포함하고 있는 SGML 문서이다.

5.2.1 SGML 선언부(SGML Declaration)

한 시스템에서 다른 시스템으로 전송되는 SGML 문서는 SGML 선언부를 갖고 있어야 한다. 그러나 한 시스템내에서 사용되는 SGML 문서에서는 이것이 없어도 관계없다. 이 SGML 선언부는 SGML이 사용할 문자의 집합, 그리고 코딩 규칙등을 정의한다. 그림 2에 보이는 것은 기본적인 SGML 선언부로서, 이와같은 SGML 선언부를 적용하여 작성한 SGML 문서를 기본 SGML 문서라고 한다.

SGML 선언부는 컴퓨터내에서의 처리시에도 참조되지만, 인쇄된 형태로서 인간의 이해를 돕기도 한다. 즉 SGML 문서를 수신할 때에 시스템이 문자의 어떤 번역을 하여야 하는지,

아니면 알고리즘적인 변환을 하여야 하는지 등을 정의한다.

이 SGML 선언부는 문서에서 사용할 문자 집합을 정의하는 부분, SGML 문서의 처리능력에 대한 범위를 정의하는 부분, 주요 구체적 문법을 정의하는 부분, 그리고 SGML 문서가 갖는 특수한 특징을 표현하는 부분등으로 구성된다.

SGML 선언부에 대한 문법을 기술하면 "마크업 선언 개방(Markup Declaration Open)"으로 시작하며, "마크업 선언 폐쇄(Markup Declaration Close)"로 끝난다. 그리고 첫번째 행은 "SGML"로 시작하고 현재의 SGML 선언부에서 적용한 표준의 번호와 그 표준의 발표시기를 문자열로 형성하여 기록한다. 그 다음에는 앞에서 말한 정의 부분들을 차례로 작성하는데, SGML 문서내에서 사용될 글자의 집합을 정의할 때는 첫 번째 열에 "CHARSET"를 두고 다음 열에 "BASESET"와 "DESCSET"를 쓰는데 전자는 기본적인 문자 집합을 의미하는 것으로서 바로 우측에 표준이나 등록된 이름, 또는 숫자등을 내용으로 하는 인간이 읽을수 있는 문자열을 기록하여 문서를 수신할 때 이해할 수 있게 한다. 후자는 다음 행에 쓰며 그 우측으로 세개의 열로써 표현하는데, 첫번째 열은 사용할 문자의 2진 코드를 십진수로 바꾼 값이며, 다음열은 첫번째 열의 십진 코드부터 연속하여 선언하고 있는 글자의

```
<!SGML "ISO 8879-1986"
BASESET "ISO 646-1983//CHARSET
        International Reference Version (irv)//ESC 2/5 4/0"
DESCSET   0   9  UNUSED
          9   2   9
          11  2  UNUSED
          13  1  13
          14  18  UNUSED
          32  95  32
          127  1  UNUSED
BASESET "ISO Registration Number 109//CHARSET
        ECMA-94 Right Part of Latin Alphabet Nr. 3//ESC 2/13 4/3
DESCSET  128  32  UNUSED
          160  5  32
          165  1  "SGML User's Group logo"
          166  88  38
          254  1  127
          255  1  UNUSED

CAPACITY PUBLIC "ISO 8879-1986//CAPACITY Reference//EN"
SCOPE     DOCUMENT
SYNTAX    PUBLIC "ISO 8879-1986//SYNTAX Reference//EN"

                          FEATURES
MINIMIZE DATATAG  NO  OMITTAG  YES  RANK  NO  SHORTTAG YES
LINK     SIMPLE   NO  IMPLICIT  NO  EXPLICIT NO
OTHER    CONCUR   NO  SUBDOC    NO  FORMAL  NO

                          APPINFO NONE>
```

그림 2 기본적인 SGML 선언

수효이고, 그 다음 열은 문자에 대한 기술 내용이다. 이 문자 기술 내용은 첫 번째 열의 코드에 대한 설명으로서 BASESET에서 기술한 ISO등의 표준 규격의 코드이다. 이에 대해서 첫 번째 열은 본 SGML 문서 내에서 사용하는 코드이다.

다음에 선언되는 것은 CAPACITY이며 그림 2의 예에서는 ISO 8879의 "CAPACITY Reference"를 따른다고 되어 있다. 이곳에서는 엔티티 정의의 한계, 엔티티 내의 글자의 한계, 내용 모델내의 임의의 레벨에서 갖을수 있는 토큰의수, 속성 정의의 수등의 상한값을 정한다. 그 다음에 SCOPE등을 선언하는데 이것은 주요 구체 문법이 문서 전체에서 사용될 것인지, 아니면 문서의 텍스트에서만 사용될 수 있는지를 선언하는것으로 전자에는 DOCUMENT로 후자에는 INSTANCE로 한다. 다음에 주요 구체 문법을 선언하며 PUBLIC이라는 예약어를 사용하여 국제적으로 공인된 이름을 쓴다. 그렇게 하면 PUBLIC 이름으로 사용된 문법이 구체 문법이 되는 것이다. 그리고 끝에 FEATURES를 이용하여 SGML시스템이 갖추어야할 기능등을 표시한다.

5.2.2 문서형 정의부(Document Type Definition : DTD)

SGML 문서는 문서형 정의부에 문서 자체의 구조를 내포한다. 이 문서형 정의부의 기술 방법도 ISO 8879로써 문서형을 표현하기 위한 문법이 표준화되었다. 이 문서형 정의부는 어떤 특정한 형태를 갖는 모든 문서에 적용하는 마크업 선언들의 집합으로 구성되며 시작부분에 문서형 정의부의 고유이름을 갖는다. 문서형 정의부에 선언할 수 있는 선언부들중에서 중요한 선언부로 요소 선언(element declaration), 속성 선언(attribute declaration), 그리고 엔티티 선언(entity declaration)의 세가지를 들 수 있다. 요소 선언은 각 문서 요소 내부에서 존재할 수 있는 부문서요소에 대한 공통식별자(Generic Identifier)를 순서에 준하여 정의하며, 선언중인 문서 요소에 대한 공통식별자도 정의한다. 속성 선언은 문서 요소에 관련되어 특정될 수 있는 속성들과 그 속성들이 갖을 수 있는 값들을 정의한다. 그리고 엔티티 선언은 이 문서형의 문서들에서 참조 할 수 있는 엔티티들을 정의한다.

문서형 정의부를 작성하기 위하여 여러가지 마크업 기호가 필요한데 이 여기에서는 ISO 8879의 부록 B에서 사용한 것을 사용하였으며, 몇가지 대표적인 마크업 기호를 표 1에 보인다.

SGML 구문에 의하면 문서형 정의부는 mdo (markup declaration open. 표 1참조)기호로 시작하여 mdc(markup declaration close)기호로 끝난다. 이 mdo와 mdc사이에는 mdo 기호 바로 다음에 DOCTYPE라는 예약어와 정의중인 문서 형태에 대한 고유 이름인 공통 식별자를 차례로 기술한다. 이어서 dso(declaration subset open) 기호를 쓰고, 엔티티들을 선언하고, 문서요소들을 선언하며, 다음에 속성들을 선언한다. 그리고 dsc(declaration subset close)기호를 기입하고, mdc 기호로 끝낸다.

따라서 모든 DTD는 문서의 구조를 기술하기 때문에 DTD를 참조할 수 있도록 SGML의 첫 부분에 특수한 마크업 선언을 하는데 이러

표 1 마크업 기호

마크업 이름	기 호	의 미
mdo	<!	마크업 선언 시작
mdc	>	마크업 선언 끝
dso	[	선언 부분집합 시작
dsc	]	선언 부분 집합 끝
ero	&	엔티티 참조 시작
refc	;	엔티티 참조 끝
pio	<?	처리명령 시작
pic	>	처리명령 끝
grpo	(	그룹 시작
grpc	)	그룹 끝
stago	<	시작 태그 시작
stagc	>	시작 태그 끝
seq	,	순차적
opt	?	선택적
plus	+	1번 이상
rep	*	0번 이상
or	|	논리적 OR
and	&	논리적 AND

표 2 DOCTYPE 선언 부분

Example : <!DOCTYPE book [
....]>

순 서	심 볼	설 명	축 약 어
1	<!	마크업 선언 개방 구분자	MDO
2	DOCTYPE	DOCTYPE 키워드	
3	book	문서형 이름	
4	[	선언 부집합 개방 구분자	DSO
5	....	DTD 또는 마크업 선언	
6	]	선언 부집합 폐쇄 구분자	DSC
8	>	마크업 선언 폐쇄 구분자	MDC

```
<!DOCTYPE book [

<!ENTITY % ade.ph "str | cty | sbd | cny | pc | san | ead" -- address elements -->
<!ENTITY % bib "au | cau | msn | srt | loc | pdt | pp | atl | sct | obi" --
bibliographic-->
<!ENTITY % bmcps.d "vt" -- back matter cps elements -->
<!ENTITY % bmsec.d "awd | bib | notes" -- back matter sections -->
                    :
<!--      ELEMENT      MIN  CONTENT                -->
<!ELEMENT book          - - (fm, bdy, appm?, bm?) >
<!ELEMENT fm             O  O  (tig, (au | cau)*, (rep1, (rep2, rep3?)?)?, pubfm?,
                              (fwd | pf | ack | ded | abs | smtl)*, toc?)>
<!ELEMENT  (%fmsec.d;)  - O  %m.sec;>
<!ELEMENT  (%fmcps.d;)  - O  %m.cps;>
<!ELEMENT bdy          O  O  (part+ | chp+)>
<!ELEMENT part           - O  (no?, pt, (%s.zz;)*, chp+)>
                    :
<!--                     Attribute Definition Lists -->
<!ATTLIST art            "id ID IMPLIED"
                         "sizex NUTOKEN #IMPLIED sizey NUTOKEN #IMPLIED
                         unit CDATA
  IMPLIED">
                    :
]>
```

그림 3 문서형 정의부의 예

한 선언을 문서형선언(document type declaration)이라 한다. 이렇게 문서형 선언부는 DOCTYPE 키워드로서 시작되며 그 구조의 전형적인 예를 표 2에서 설명하고 있다.

그림 3은 공용으로 사용되는 한 DTD의 일부분을 보여준 것이다.

1) 엔티티 선언(entity declaration)

SGML 문서를 작성할 때 같은 문자들을 여러번 반복하여 입력할 필요가 있을때, 엔티티

를 이용하면 문자들을 입력하는 수고를 줄일수 있다.

ISO 8879에서 엔티티는 문자들의 집합이라고 정의되었는데 그 크기와 형태가 다양하여 한두 글자를 갖는 엔티티도 있는가 하면 한 문서의 일부분이나, 하나의 완전한 SGML 문서도 엔티티로 볼수 있다. 그 중에서 SGML문서에 포함된 엔티티에는 두가지의 중요한 형태가 있는데, 일반 엔티티와 파라미터 엔티티가 있다. 일반 엔티티는 그 대치될 문자로서 문서의 구조상의 어떤 의미등이 없는 단순한 문자들을 갖으며, 파라미터 엔티티는 대치될 문자가 다른 문서 요소의 공통식별자 즉 문서 요소 이름이 되는 경우로서 문서형 정의부 내에서만 사용된다. 그리고 일반 엔티티의 경우 대치될 엔티티 문자가 현재의 처리중인 SGML 문서 내에 존재하지 않는 엔티티를 외부 엔티티라고 한다. 이 외부 엔티티를 다시 한 시스템 내에서만 사용가능한 시스템 엔티티와 두개 이상의 시스템에서 사용될 수 있도록 공적으로 선언된 공용 엔티티로 구분한다. 이 관계를 그림 4에 보인다.

위에서 말한 여러 종류의 엔티티를 SGML 문서에서 사용하기 위해서는 엔티티 선언과 엔티티 참조가 필요한데, 그 방법이 엔티티 종류에 따라 다르다. 엔티티 선언은 엔티티의 내용을 정의하고, 엔티티 참조는 엔티티 내용이 문서내의 어디에 위치해야 할지를 지시한다. 그리고 엔티티 선언은 문서형 정의부내에 있어야 하며, 엔티티 참조는 문서내의 어디에나 올 수 있으며, 문서형 정의에서도 가능하다. 그러나 파라미터 엔티티는 문서 형태 정의에서만 참조할 수 있다.

일반 엔티티와 파라미터 엔티티는 선언과 참조에서 그 마크업 방법을 다르게 하겠다. 파라미터 엔티티를 선언하는 마크업은 mdo. ENTITY, pero. 엔티티 이름, 대체문자, mdc,의 순서대로 한다. pero는 %로 한다. 참조할 때는 pero, 엔티티 이름, refc의 순서로 마크업한다. 그 선언과 참조의 예를 들면

<! ENTITY %요소7 "요소1 | 요소4 | 요소5">

표 3으로 선언되고, 참조는 문서 요소내에서나 본문내에서 가능한데 다음은 문서 요소내에서의 엔티티 참조의 예이다.

<!ELEMENT 절5 (단락 | 예문 | %요소7;) >

이 중 "%요소7;"부분이 엔티티 참조이며 "%요소7;"이 있는 위치에 엔티티 요소7이 선언될 때 대체 문자로 이용된 "요소1 | 요소4 | 요소5"가 대신 들어간다. 여기서 요소7은 엔티티 이름이며, 엔티티 내용인 요소1, 요소4, 요소5들은 이 문서내의 문서 요소의 이름들로서 문서 요소 절 5의 부문서 요소가 되는 것이다.

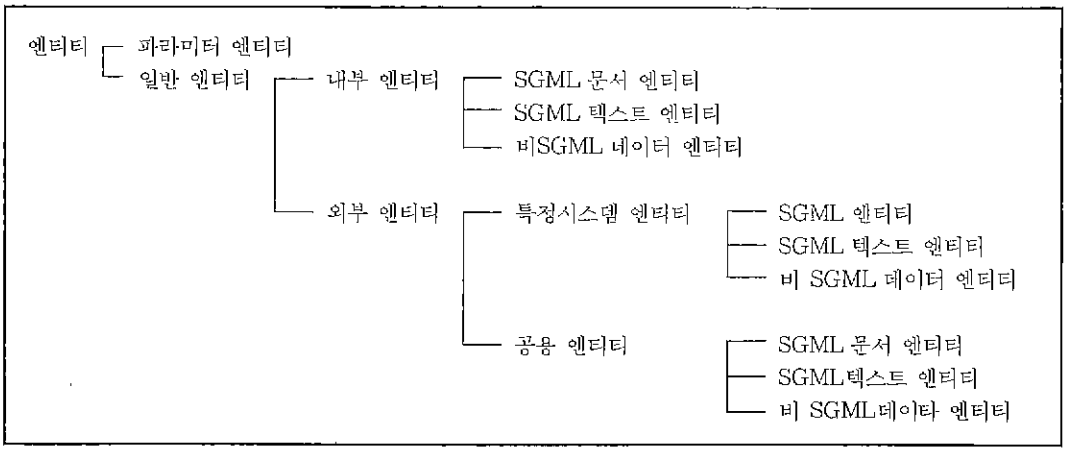


그림 4 엔티티의 유형

표 3 파라메타 엔티티 선언 부분

Example : <!ENTITY % bmcps.d "vt" --decument type generic identifier-->			
순 서	심 볼	설 명	축 약 어
1	<!	마크업 선언 개방 구분자	MDO
2	ENTITY	엔티티 선언 키워드	
3	%	파라메타 엔티티 참조 개방 문자	PERO
4	bmcps.d	이름(name)	
5	"or'	문자열 구분자	LIT or LITA
6	vt	파라메타 데이타	
7	"or'	문자열 구분자	LIT or LITA
8	>	마크업 선언 폐쇄 구분자	MDC

위의 예에서 대체 문자내에서 사용된 | 도 마크업 기호의 하나로서 | 기호 앞과 뒤중 적어도 하나 필요하다는 의미를 갖는다. 이것이 문서 요소 절 5의 내용으로 참조되면 결과적으로 절 5 요소의 내용 부분은 (단락 | 예문 | 요소1 | 요소4 | 요소5)가 된다.

공적으로 선언된 외부 엔티티중에서도 문서형 정의에서 참조될 엔티티는 파라미터 엔티티로 선언되며

<! ENTITY %공용엔티티명 PUBLIC "공적으로 선언된 문자열">

와 같은 형태이다. 여기서 PUBLIC은 공적으로 선언된 것임을 의미하는 예약어이다. 이것은 문서형 정의

%공용엔티티명 ;

의 형식으로 참조된다. 이러한 방법에 의하여 공적으로 선언된 문서들을 문서 형태 정의에서 이용할 수 있다.

일반 엔티티 중 SGML 텍스트를 갖는 경우의 엔티티 선언은 mdo, ENTITY, 엔티티명, 엔티티 내용, mdc의 순서로 즉

<! ENTITY 엔티티명 "대체할 엔티티 내용">

와 같이 마크업하며, 참조할 때는 ero, 엔티티명, refc로 하며

&엔티티명;

이 된다. 이 엔티티의 내용에는 일반 문자 및 마크업 문자도 포함시킬수 있다.

2) 요소 선언(element declaration)

문서 요소 선언은 크게 공통식별자나 이름 그룹등으로 이루어지는 문서 요소 형태 부분과 문서 요소의 실제 내용 또는 다른 부요소들에 대한 공통식별자들로 구성되는 내용 부분으로 구분된다.

문서형 정의부 내의 문서 요소 선언중 제일 먼저 선언되어야하는 것은 문서 형태 정의의 고유 이름을 공통식별자로 하는 것이며, 이 선언의 내용부분에는 문서의 구조상 상위 레벨에 있는 문서 요소들의 공통식별자들이 온다. 물론 이 때 사용한 공통식별자들은 차례로 문서 요소 선언을 하여야 하며, 임의의 문서 요소 선언은 내용 부분에 다른 문서 요소의 공통 식별자를 갖을수 있는데, 그러면 이 공통 식별자를 갖는 문서 요소는 선언중인 문서 요소의 부요소가 되는 것이다. 그리고 이 부요소에 대한 문서 요소 선언도 필요하며, 부요소의 문서 요소를 선언할 때에도 그 아래 레벨의 부요소를

내용부분으로할 수 있다. 결과적으로 문서형 정의부에 있는 문서 요소 선언들에 의하여 문서의 구조를 파악할 수 있으며, 이 구조는 트리 형태를 갖는다.

문서 요소 선언에 관한 구문을 살펴보면 mdo로 시작하고, 예약어인 ELEMENT를 쓴다. 그리고 이어서 현재 선언중인 문서 요소의 문서 요소 형태를 쓰는데, 이것은 하나의 공통 식별자 일 수도 있으며, 복수개의 공통 식별자일 수도 있다. 전자의 경우에는 그대로 쓰고 후자인 경우에는 표 1의 grpo(group open) 기호를 먼저 쓰고 공통 식별자들을 컴머로 구분하며 열거한 후 grpc(group close) 기호를 쓴다. 이러한 형태를 이름 그룹이라 한다.

다음에는 SGML 문서에서 본 문서 요소에 대한 시작태그 및 끝 태그를 생략할 수 있는지, 생략할 수 없는지에 대한 정보를 갖는 두 글자를 쓰는데 앞의 글자는 시작 태그에 대한 것이고, 뒤의 글자는 끝 태그에 대한 것이다. 이때 "-"자와 "O"를 사용하는데 "-"자는 생략할 수 없음을 의미하고, "O"자는 생략이 가능함을 의미한다. 아래에 몇가지 문서 요소 선언의 예를 들었다.

<!ELEMENT 제목 - O 내용1
<!ELEMENT (보내는사람, 받는사람) - - 내용2

윗줄의 예에서 "제목"은 문서 요소의 공통식별자로서 내용1을 갖는 문서 요소들을 지칭하는 이름이며, "- O"는 시작 태그는 생략할 수 없으며, 끝태그는 생략 가능함을 의미한다. 아래 예의 공통 식별자는 "보내는사람"과 "받는사람"의 둘이며, 이 두 식별자는 그 내용이 "내용2"로서 같음을 의미하고, 이름 그룹의 형태로 표기되었다. 이 예에서는 시작태그도 끝태그도 무두 생략할 수 없음을 말하고 있다.

내용 부분에 대한 구문은 일반적으로 부 문서 요소를 선언중인 문서요소내에 포함할 수 있는 것을 정의하는 내용 모델로 구성된다. 내용 모델에는 여러가지 부 문서 요소가 모델 그룹으로서 정의되어야 한다. 이름 그룹과 마찬가지로 모델 그룹도 grpo와 grpc 사이에 포함될 부 문서요소들의 이름(공통 식별자)을 열거한다. 이때 발생 지시자와 연결자를 사용하는데 발생 지시자는 이름 바로 뒤에 쓰며 한번만 발생해야 하는 문서 요소의 이름에는 쓰지 않는다. 발생 지시자로는 opt, plus, rep가 있는데, opt는 선택적임을 말하는 것으로 즉, 그 부 문서요소가 있을 수도 있고 없을 수도 있음을 의미하고, plus는 그 앞의 부 문서요소가 선언중인 문서 요소 가운데 꼭 필요하며, 두번 이상 있을수도 있음을 의미하고, rep는 본 문서 요소중에 없을 수도 있고 1번이상 존재할 수도 있음을 의미한다. 연결자는 한 부 문서요소이름과 다음 부 문서요소 사이에 필요한 것으로서 seq, or, and의 세가지가 있는데, seq 앞뒤의 부 문서 요소는 반드시 존재하되 이 모델 그룹에 등장한 순서에 맞추어 와야 한다는 것을 말하고, or 연결자 전후의 두 부 문서 요소는 둘중 어느 하나의 부 문서 요소만 있어도 되고, 둘다 있어도 좋음을 의미한다. and 연결자 전후의 부 문서 요소는 순서는 개의치 않지만 반드시 존재하여야 함을 의미한다. 이와 같은 모델 그룹을 사용한 문서 요소 선언으로

<!ELEMENT (보내는사람, 받는사람) -- (성명, 나이, 전화번호* | 주소)>

을 예로 할때, 여기에서 성명, 나이, 전화번호, 주소는 각각 부 문서요소의 공통 식별자이고, 전화번호 다음의 * 구분자는 전화번호 요소가 없을 수도 있고, 하나 이상 있을 수도 있음을 의미하며, 이 모델 그룹의 연결자들은 성명과 나이 요소는 순서대로 발생되어야 하고, 전화번호나 주소중 어느 한 부 문서요소는 그 다음에 발생해야함을 의미한다. 앞에서 설명하였듯이 전화번호와 주소 요소가 둘다 있을 수도 있으며, 그 경우 순서는 임의로 할 수 있다.

이러한 요소의 구조는 표 4에서와 같이 정의된다. 또한 DOCTYPE에서 선언되었던 이름을 갖는 요소는 반드시 정의되어 있어야 한다.

요소 선언은 문서형 선언부에서 문서형 이름과 어떤 선택-식별자를 따르는 문서형 선언 부집합 부분을 형성한다. 각 요소는 마크업 구분자(delimiter)의 집합내에서 예약어 ELEMENT로서 선언된다. 표 4의 선언에서 이름

표 4 엘리먼트의 선언 부분

Example : <!ELEMENT book – O (fm, bdy, appm?, bm?) >			
순 서	심 볼	실 명	축 약 어
1	<!	마크업 선언 개방 구분자	MDO
2	ELEMNT	엘리먼트 선언 키워드	
3	book	엘리먼트의 범용 식별지	GI
4	– O	생략된 태그 최소화	
5	(fm, bdy, appm?, bm?)	문자의 내용부분	
6	>	마크업 선언 폐쇄 구분자	MDC

(name)은 문서의 요소를 식별하는 범용 식별자이며, 내용부는 요소내에 입력되어지는 데이타 형태의 공식적인 선언이다. 일반 식별자의 위치에 이름 그룹(name group)을 사용함으로서 하나 이상의 요소명이 선언된 내용부의 집합과 연관되어지며, 또한 내용부에서는 그룹단위로 묶여질 수 있는 모델 그룹(model group)을 형성하여 처리되는 단위이다.

기본 문서의 요소는 문서형 선언내에서 이름(Name)이 최초 문서형 선언과 같은 요소 선언의 항목에 의해서 공식적으로 선언되어야 한다. 선택된 요소의 이름은 일반적으로 문서의 범위를 식별할 수 있도록 태그 집합을 사용하여 고안되었다. SGML 문서 내에서는 요소는 시작-태그(start tag)와 끝-태그(end tag)에 의해 구분되며, 입력된 끝 태그는 시작 태그(예, <book>)로서 사용된 이름과 일치하는 태그(예, </book>)이어야 한다. 이 태그는

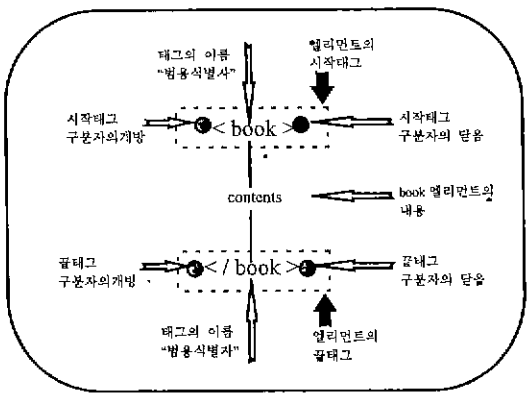


그림 5 태그 구분

문서의 끝이 SGML 프로그램에 의해 옳게 식별되었는지를 확인한다. 기본 문서 요소들은 한계를 정하는 두 태그들 사이에 끼워지며, 이러한 예를 그림 5에서 자세히 설명하고 있다.

3) 속성 선언(Attribute Declaration)

속성은 문서 요소의 성질을 표현하기 위하여 사용되는 것으로서 문서 혹은 문서 요소의 상태, 문서의 텍스트가 출력되는 형식, 외부로 부터의 데이터가 문서에 추가될 경우에 그 위치 및 크기등을 알린다. 속성 지정을 하기 위하여 속성은 속성의 이름과 속성의 값을 갖는다.

속성의 선언을 문시 형태 정의에서 관련되는 문서요소의 선언 아래 부분에서 한다. 보통 문서 요소의 선언들이 끝난 다음에 속성 선언들을 위치시킨다. 속성 선언의 일반 형식은 mdo, ATTLIST, 관계 문서 요소명, 속성 선언 리스트, mdc의 순서이며 속성 선언 리스트에는 각각 다른 특성을 지정할 수 있는 속성 정의가 리스트를 이루며, 하나의 속성 정의는 속성의 이름, 선언된 값, 디폴트값의 세 성분으로 구성된다. 속성 이름의 첫 글자는 이름 시작 문자로 시작되어야 하고, 그 후는 이름 문자로 구성되어 8자 이내로 되어야 하며, 선언된 값은 속성의 실제 값이 될 수 있는 후보등의 or 연결자에 의한 리스트이거나, 또는 SGML 문서 작성시 입력될 속성의 값의 형태를 말해주는 예약어이다. 이 예약어로는 CDATA, ENTITY, ENTITIES, ID, NMTOKEN, NAME, NUMBER등이 있으며, 대표적인 예약어와 이것들이 정하는 의미를 표 5에 보였다.

표 5 속성의 선언된 값에 대한 예약어

예 약 어	속성값의 형
CDATA	문자 데이타
ENTITY	부 문서 또는 데이타 엔티티 명
ENTITIES	부문서들 또는 데이타 엔티티명들
ID	문서 요소 식별자
IDREF	문서 요소 식별자의 참조값
IDREFS	문서 요소 식별자의 참조값 목록
NAME	SGML 이름
NAMES	SGML이름들
NMTOKEN	이름 토큰, 즉 이름 시작 문자가 아닌 숫자등도 첫글자가 될수 있음.
NUMBER	수
NUMBERS	수들
NUTOKEN	수 토큰, 즉 숫자로 시작하는 이름
NUTOKENS	수 토큰들

속성 정의중 세번째 성분인 디폴트 값은, 바로 앞의 선언된 값이 실제 속성값의 후보로 된 경우는 그 중의 한 값으로 정할 수 있으며, 다른 예약어들이 올수도 있다. 이 예약어들은 속성의 값이 설정되는 상황을 말해준다. 즉

FIXED라는 예약어는 값이 고정되어, 후에 다른값이 지정될 수 없음을 의미하고

REQUIRED는 관련 문서 요소가 사용될 때는 반드시 속성값이 필요함을 의미하며,

IMPLIED는 속성값이 지정되지 않는 경우에 프로그램이 임의의 값을 포함할 수도 있음을 의미하는 둥이다.

위의 두가지 예약어를 사용하는데는 다음과 같이 몇가지 제약이 따른다. 선언된 값의 ID와 NOTATION은 어떠한 속성 선언에서도 한번만 사용할 수 있다. 그리고 NOTATION은 관련 문서 요소의 내용이 EMPTY로 선언되었을 때만 사용이 가능하고,

CONREF는 EMPTY 내용으로 선언된 문서 요소와 관계될 때는 사용할 수 없다. 그리고 선언된 값이 ID로 선언되었을 때의 디폴트 값은 #REQUIRED이거나 #IMPLIED여야 한다. 아래에 실제 속성 선언의 몇가지 예를 들었다.

속성은 문서 요소의 성질을 표현하기 위하여 사용되는 것으로서 문서 혹은 문서 요소의 상태, 문서의 텍스트가

```
<!ATTLIST exam style CDATA #IMPLIED
    keep NMTOKEN "all"
    form (lines | runon) lines>

<!ATTLIST fig id ID #IMPLIED
    number CDATA
    frame (box | rules | none) none
    position (top | bottom | middle | asfound)
    asfound
    type (column | page)
    align (left | right | centered) centered >
```

이들 속성 선언에서 관련 문서 요소로 사용한 exam과 fig는 각각 문서 요소 선언이 되어 되어야 한다. 위의 예에는 세개의 속성 정의가 있으며 style, keep, form이 각각의 속성이름이며, CDATA, NMTOKEN, (lines | runon)이

표 6 속성 선언 부분

Example : <!ATTLIST Memo status (confiden : public) public >			
순 서	심 볼	설 명	축 약 어
1	<!	마크업 선언 개방 구분자	MDO
2	ATTLIST	속성 정의 목록 선언	
3	Memo	요소의 이름	GI
4	status	속성의 이름	
5	(confiden | public)	속성의 선언된 값	
6	public	속성의 기본 값	
7	>	마크업 선언 폐쇄 구분자	MDC

```
<!DOCTYPE report [
<!ELEMENT report - - (prolog,body, appendix?) >
<!ELEMENT prolog - - (title, author+, abstract) >
<!ELEMENT boody - - (intro, contents, conclusions) >
<!ELEMENT conclusions - - (experiment, result, conclusion, acknowledge?) >
<!ELEMENT (contents | experiment | result | conclusion) - - (section)+ >
<!ELEMENT (subsect | section) - - (paragraphs | subsect)*>
<!ELEMENT paragraphs - - (heading?, paragraph+) >
<!ELEMENT paragraph - - (#PCDATA | items | list)* >
<!ELEMENT appendix - - (reference?, annex?) >
<!ELEMENT
(title | author | abstract | intro | acknowledge | heading | items | list | reference | annex) - - (#
PCDATA) >
]>
```

그림 6 문서형태 정의부의 예

선언된 값인데, CDATA와 NMTOKEN은 표 5에 있는 예약어중의 하나이며 (lines | runon)은 실제 속성값으로 lines나 runon이 올수 있다는 것을 지정하고 있으며, 그 경우 디폴트 값은 lines로 정의 되었다.

위에서 설명된 문서 요소, 엔티티속성들의 선언 방법을 적용하여 문서형 정의부를 작성한 예를 그림 6에 보이는데, 문서형 정의부의 이름은 report로 하였으며, 문서 전체를 칭하는 report라는 이름을 갖는 문서요소가 처음으로 선언되었다. 물론 문서 요소, report는 부문서 요소로서 prolog, body, appendix의 세 문서 요소를 차례로 갖으며, 문서 요소 prolog는 그 내용 부분에 "title, author+, abstract"로 되어 있는데, 이것은 title이라는 문서 요소를 첫 번째 부문서 요소가 되고, 두번째 부문서 요소로는 author가 오는데 + 표시는 author 문서 요소가 한번 또는 두번 이상 존재하여야 함을 의미하기 위한 것이다. prolog 문서 요소의 부문서 요소로서 그 다음에는 abstract가 오는 것도 알수 있다.

다음에 문서 요소 title을 보면 내용 부문이 #PCDATA로 되어 있는데 이것은 title의 내용으로서 문자데이터를 받을수 있음을 뜻한다. 문서 요소 body는 그 내용이 "intro, contents, conclusions"로 구성되어 있으며, 이것은 위 세개의 내용 토큰이 순서대로 발생해야 함을 의미한다. paragraph는 "(#PCDATA | items | list)*"을 내용으로 하는데 연결자가 " | "이므로 순서에 관계하지 않는다. 그리고, 뒤의 "*"는 한번이상 발생할 수도 있음을 의미한다.

속성은 다음과 같은 경우에 사용된다. 문서나 요소의 상태를 식별하는데, 텍스트가 어떻게 포맷되었는지를 조절하거나, 시스템에의해 생성된 텍스트를 식별하는데, 크기나 문서에 더해지도록 외부에 저장된 데이타의 소스(source)를 정의하는데, 요소의 특별한 발생을 식별하거나 이전에 식별된 요소에 상호-참조(cross-reference)하는데 사용된다. 속성은 속성명과 속성값이 값지정자에 의해 결합되는데 속성값은 속성값 문자로서 입력된다.

5.2.3 SGML 문서

SGML 문서는 많은 수의 상호 연결된 문서 요소들로 구성된다. 각 문서 요소는 특정한 목적을 갖는 문자들을 포함하고, 그중 특수한 문자나 단어는 여러 다른 문서 요소의 구성 성분이 될 수도 있다. 한 예로 어떤 단락 안에 포함된 한 문구는 그 단락을 포함한 절의 내용이 되고, 그 절이 포함된 장(chapter)의 한 성분이 되기도 하는 것이다.

SGML을 효과적으로 이용하기 위해선 이들 문서 요소들의 논리적인 관계를 명확히 규명하여 위에서 말한 문서형 정의부를 작성하여야 한다. 그리고 ISO 8879에서 정의한 SGML 문서에 대한 마아크업 기술 방법과 문서형 정의부에 따라서 SGML 문서를 작성한다. 우선 문서 형태 선언을 하는데 그것은 한 시스템이 여러 종류의 문서형 정의부를 갖고 있을 때, 그 중 어떤 문서형 정의부에 맞는 문서인지, 즉 문서의 개괄적인 구조가 어떠한지를 나타내기 위함이다. 이것은 mdo DOCTYPE 문서형 정의부 이름 PUBLIC 설명문 mdc의 문서로 된 구문을 갖는다. 한예로

<!DOCTYPE report PUBLIC "-//kwang-woon//DTD report //EN">

를 들 수 있는데 여기서 report는 적용한 문서형 정의부의 이름이고, DOCTYPE과 PUBLIC은 예약어이다. 이후에는 문서 요소들의 객체가 단위가 되어 문서형 정의부에서 정의된 구조로 구성된다.

한 문서 요소는 시작 태그, 내용, 끝태그의 세 부분으로 구분되며, 다시 시작 태그는 STAGO(<), 문서 요소명, TAGC(>)로 마아크업되고, 끝 태그는 ETAGO(</), 문서요소명, TAGC(>)의 순서로 기술된다. 이 때, 시작태그와 끝태그는 문서형 정의부내에서 정의된 정보에 따라 생략하는 것도 가능하다. 그리고 시작태그와 끝태그 사이의 내용은 다른

```
<!DOCTYPE report [
<!ELEMENT report - - (prolog,body, appendix?) >
<!ELEMENT prolog - - (title, author+, abstract) >
<!ELEMENT boody - - (intro, contents, conclusions) >
<!ELEMENT conclusions - - (experiment, result, conclusion, acknowledge?) >
<!ELEMENT (contents | experiment | result | conclusion) - - (section)+ >
<!ELEMENT (subsect | section) - - (paragraphs | subsect)*>
<!ELEMENT paragraphs - - (heading?, paragraph+) >
<!ELEMENT paragraph - - (#PCDATA items list)*>
<!ELEMENT appendix - - (reference?, annex?) >
<!ELEMENT    (title | author | abstract | intro | acknowledge | heading | items | list | reference |
annex) - - (#PCDATA) >
]>
<report>
  <prolog>
    <title>SGML conference report</title>
    <author>Goldfarb</author>
  </prolog>
  <body>
    <intro>SGML에 대하여</intro>
          .....
          ....
          ..
          .
  </body>
  <appendix>
    <annex>.....</annex>
  </appendix>
</report>
```

그림 7 SGML문서의 예

문서 요소를 포함할 수도 있고, 그렇지 않고 일반적인 문자 데이터로 구성되기도 한다. 여기서 내용의 위치에 오는 문서 요소를 부문서 요소라 하고, 이것도 상위의 문서 요소와 동일한 구조로 마크업된다. 앞절에서 예로 들었던 report에 대한 문서형 정의부를 기초로 하여 하나의 SGML 문서를 그림 7에 보인다.

6. CALS IETM에서의 SGML 이용실례

SGML의 CALS 응용 형태로 현재 가장 많이 응용되고 있는 것이 문서정보의 한 특수한 유형이라고 할 수 있는 기술교범(Technical Manual)이다. 현재 기술교범은 우리나라에서만 총 2천여종에 이루고 있는 것으로 조사되고 있어 이제는 프린트된 기술교범에 의한 업무처리나 경험에 의한 정비, 조작등이 어렵게 되어가고 있다. 따라서 CALS에서는 기존 프린트된 기술교범을 현재 각 기간산업에서 필수적인 부가조건으로 부각되고 있는 전자화된 기술교범(Electronic Technical Manual : ETM)으로 만들고, 더불어 최종 사용자와 화면으로 대화하는 형태인 대화형 전자화된 기술 교범(Interactive Electronic Technical Manual : IETM)화에 많은 노력을 하고 있다.

IETM은 기존에 종이에 작성된 교범들을 컴퓨터를 이용한 전자식 교범으로 전환된 형태로

표 7 IETM의 기능

분류 \ 기능	IETM의 주요 기능
기본 기능	1) 고장진단 및 검색 2) 정비 절차 및 특수공구 시험장비 운용절차 3) 고장배제 수행절차 4) 보급관련 자료 검색 5) 장비 이력자료 입력 및 검색
보조 기능	1) 자체교육 수행 기능 2) 정비 관련 기록 관리 기능 3) 향후 타 전산시스템과의 연계(LAMIS, CALS)

써, 운용자/ 정비자가 휴대용 컴퓨터를 가지고 다니면서, 필요한 시점에 필요한 장소에서 전산 체계를 이용한 운용 및 정비활동을 지원받을 수 있는 교범체계라 할 수 있다. 표 7은 전자식 기술교범의 주요 기능을 보여 주고 있다.

이러한 전자식 기술교범은 그림 8에서와 같이 기술적 발전단계에 따라 6단계로 구분 할 수 있고 분류 3 이상 부터는 MIL 표준에 따라 구성되어야 한다. 분류 3에서 준수하는 MIL-M-87268 표준은 GCSFUI(General Content, Style, Format and User-Interaction)에 관한 표준이고, MIL-D-87269 은 IETM 데이터베이스의 사양을 규정한 표준이다. 그리고 분류 4에서는 앞의 두 표준과 더불어 QA(Quality Assurance)에 관한 요구사항을 명시한 MIL-Q-87270을 준수해야 한다 .

전자식 교범 시스템은 다양한 입력도구로부터 제작된 데이터들을 SGML 코드화 모듈과 대화식 내용 데이터베이스 생성 모듈에 의해서 SGML 형태로 데이터베이스에 저장된다. 저장된 데이터는 여러형태의 플랫폼이나 응용 프로그램에서도 어떠한 특별한 변환을 거치지 않고 공유가 가능해 진다. 프리젠테이션 모듈은 SGML의 덧붙이기 정보를 참조하여 다양한 출력 환경 및 출력 양식으로 제공된다. IETM 전체 시스템의 흐름은 아래 그림과 같다.

이 IETM은 어느 특정 산업에만 사용되는 것이 아니라 모든 기간 산업에서 필요로 하고 있고, IETM을 이용 현재 선박, 비행기, 자동차, 군수무기인 탱크 산업에서 개발물에 대한 유지, 보수, 교육 등에서 이용할 수 있다.

IETM 만들기를 위해서는 IETM 저작도구가 필요한데, 현재 상품화되어 시판되고 있는 상용 소프트웨어는, 미국의 InfoAccess사의 Guide Professional Publisher와 Huge사의 Advanced Integrated Maintenance Support System제품정도가 시판되고 있다. 또한 InfoAccess사는 저작된 IETM 데이터를 현재 전세계적으로 활성화되고 있는 internet상의 WWW(World Wide Web)에서 이용할 수 있는 HTML 문서로 변환 시킬 수 있는 HTML Transit을 개발, 시판 중이다. 국내에서는 현대미디어시스템이 InfoAccess사와 제휴하여

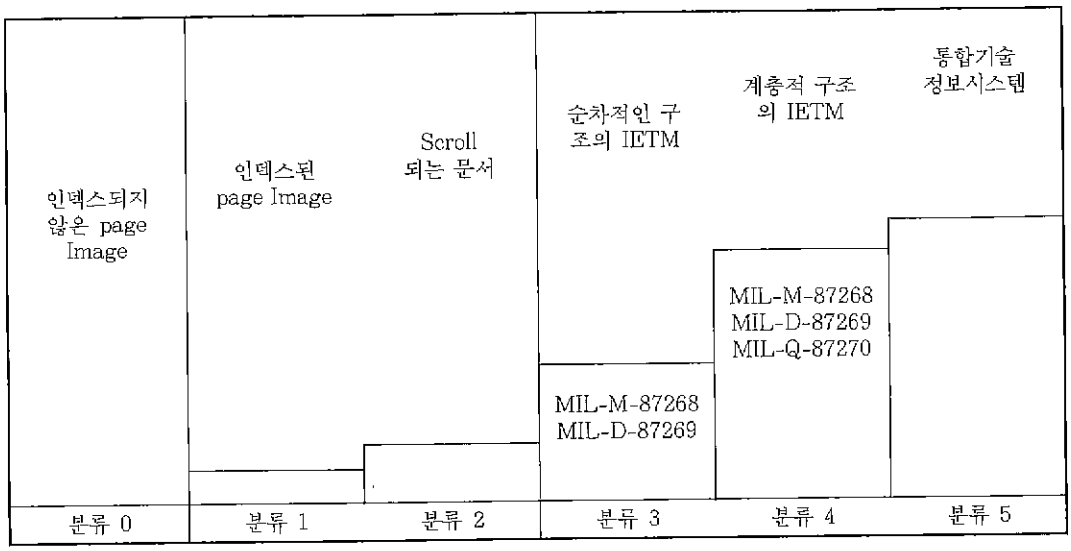


그림 8 대화형 전자식 기술교범의 발전단계

Guide Professional Publisher을 국내에 소개하고 판매를 하고 있으며 별도의 IETM 개발팀

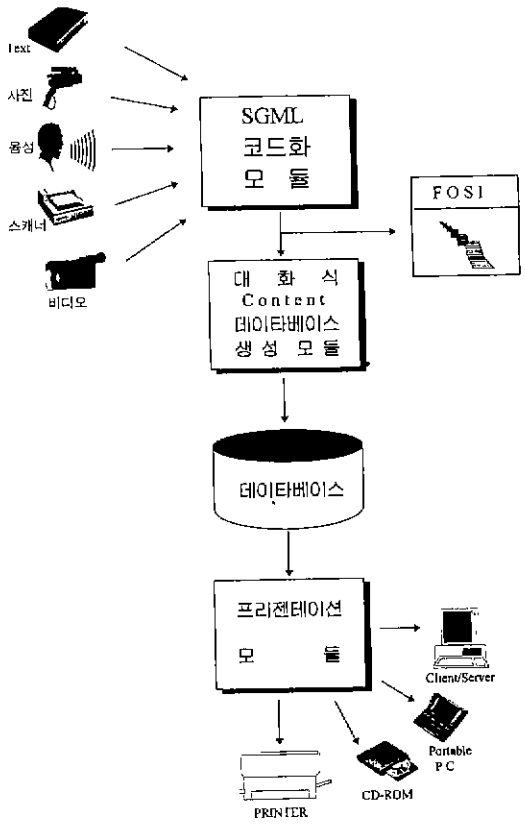


그림 9 IETM 전체 시스템의 구성

을 만들어 국산 IETM 저작 소프트웨어를 개발하고 있고 선경 및 삼성에서도 IETM 관련팀을 구성운영하고 있는 실정이다.

따라서 이 IETM 개발은 국가적인 차원에서 TM 멀티미디어/하이퍼미디어 S/W 기술을 개발함으로써 앞으로의 모든 기간산업의 중요한 한 부분을 담당하고 선도해 나갈 분야이기 때문에, 이 기술의 선점은 매우 중요하다고 볼 수 있다.

7. 결 언

SGML은 개방시스템 환경하에서 문서정보의 상호교환을 목적으로 만들어진 표준이며, 현재 전 세계적으로 채택되어, 이미 널리 활용되고 있는 실정이다. SGML은 앞에서 소개한 바와 같이 문서를 논리적인 정보로 표현할 수 있어, 문서의 색인화가 가능해짐으로 데이터베이스시스템 응용등에 효율적으로 사용할 수 있다. 또한 하이퍼미디어 문서의 근간으로 현재 ISO 10744 : HyTime(Hypermedia-Time based Structuring Language)이 제정되어 이를 근간으로하는 표준 응용들이 나타나고 있다. 그런데 이 HyTime의 구문이 바로 SGML로 구성되어 있다. 그리고 현재 인터넷상에서 널리 확산되어 사용되고 있는 HTML 역시 SGML의 한

subset에 불과한 것이다.

CALS에서는 SGML을 이용한 전자출판 및 IETM의 활용등 문서가 있는 곳에서는 어느 곳에서나 사용해야 하는 중요한 표준으로써 자리를 이미 차지 하고 있다. 국내에서는 아직 SGML의 소개가 널리 알려지지는 않았지만 국내의 CALS의 활성화의 계기로 점차 알려지기 시작하였다.

미래의 정보교환에 있어도 전자문서의 비중은 앞에서 언급한 바와 같이 매우 중요한 역할을 계속 담당할 것이다. 따라서 전자문서의 표현 표준은 그 중요성이 앞으로 기하급수적으로 커질 것이 자명한 일이다. SGML은 이러한 미래의 정보화 사회에 대한 조그만한 그러나 필수적인 준비라고 생각되며, 국내에서도 많은 SGML에 관련된 연구와 개발이 진행되리라고 기대한다.

참고 문헌

[1] 김철환, 김규수, "CALS-이론과 실제", 문원.

[2] 마스시마 가쓰모리, "CALS전략과 EC(Electronic Commerce)", 현대경제사회연구원.

[3] ISO 8879 Standard Generalized Markup Language, 1986.

[4] Martin Bryan, "AnAuthor's Guide to the Standard Generalized Markup Language". Addison-Wesley Publishing Company.

[5] Eric van Herwijnen, ' Practical SGML Second Edition" ,Kluwer Academic Publishers.

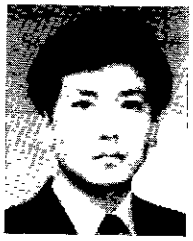
현 득 창

1992 광운대학교 공과대학원 전자계산기 공학 전공(공학석사)

1992~현재 광운대학교 공과대학원 전자계산기 공학과 박사과정

1994~현재 현대미디어시스템 IETM 개발팀장

관심분야 : 하이퍼미디어 문서 처리, SGML, HyTime

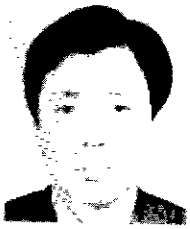


정 회 경

1993 광운대학교 공과대학원 전자계산기 공학과 전공(공학박사)

1994~현재 배재대학교 전자계산학과 전임강사

관심분야 : 개방형 문서처리, 하이퍼미디어/멀티미디어 문서 처리, SGML, ODA



이 수 연

1969 광운대학교 전자통신 공학과 졸업(공학사)

1977 연세대학교 전지통신공학과 졸업(공학석사)

1983 일본 교또대학교 정보공학과 졸업(공학박사)

1983~현재 광운대학교 컴퓨터공학과 교수

관심분야 : 하이퍼미디어/멀티미디어 문서 처리, SGML/HTML, HyTime