The Journal of Natural Science,
Pai Chai University, Korea
Volume 8: 93~100, 1995

역추적 기능을 이용한 ODA 문서 포맷터 설계 및 구현에 관한 연구

정희경, 조인준, 김진수
배재대학교 전자계산학과

A study on the design implementation of ODA document formatter using backtracking mechanism

H.K. Jung, I.J. Jo, J.S. Kim
Department of Computer Science, Pai Chai University

본 논문은 서로 다른 시스템들 간에 구조화된 멀티미디어 문서정보 교환을 위한 국제표준 문서구조인 ODA의 문서 포맷터 설계 및 구현에 관한 것이다. 포맷터는 공통배치구조로부터 특정배치구조를 생성함과 동시에 사용자의 상호작용에 따라서 특정배치구조와 특정논리구조의 관계를 설정토록 하였으며, 이를 위해 역추적(backtracking) 메커니즘과 배치 지시자(layout directive)의 처리규칙을 제안하였다. 특히 복잡한 내부구조를 투명하게 보임으로써 보다 용이하게 문서작성을 수행토록 사용자 인터페이스를 대화식 처리방식으로 구현하였다.

This paper describes the design and implementation of ODA document formatter with the capability of interchange for the structured multimedia document information between heterogeneous systems. We designed the formatter generating the specific layout structure by the generic layout structure and establishing relationship between specific logical/layout structure by interaction of user. For it, we proposed backtracking mechanism and processing rules of layout directive. Especially, we implemented an interactive method as user interface for ease creation of a document due to show user transparently complicate internal of structure.

Key words : ODA, ODIF, formatter, backtracking, multimedia document architecture.

1. 서 론

최근의 정보화 사회에서 구조화된 멀티미디어 문서정보를 서로 다른 시스템들과 응용들 간의 교환이 요구됨에 따라 여러 표준형식에 대해 많은 연구가 이루어지고 있다.[1,2]

이들 표준들은 IBM의 DCA(Document Content Architecture), DEC의 CDA(Compound Document Architecture) 등이 있고,[3,4] 국제 표준 형식으로는 SGML(Standard Generalized Markup Language)[5,6]과 ODA(Open Document Architecture)[7,8]가 대표적이다. SGML은 출판환경에서 문서구조의 교환과 방법을 제공하고, 이에 의해 저자들의 문서는 개념적인 논리구조를 마크업(markup)하여, 복잡한 문서를 작성할 수 있으므로 실제 많은 출판 환경에서 실용화되어 사용되고 있다. ODA는 구조화된 멀티미디어 문서를 나타내는 방법과 네트워크 상에서 전송이나 화일을 저장하기 위한 문서의 2진 부호화를 제공하며, 계층적이고 오브젝트 지향적(object oriented)인 문서구조를 제공한다.

위의 표준 형식중 개방 분산 환경 속에서 구조

화된 사무 문서를 편집, 포맷팅(formatting), 표현 및 처리가 가능하고, 배치구조를 설명하는 시맨틱스(semantics)를 제공하며, 정적인 데이터뿐만 아니라 오디오, 비디오 등의 동적인 데이터를 수용하는 ODA 시스템이 널리 사용되고 있다. 대표적인 ODA 시스템 개발 예로는 외국의 경우 미국의 EXPRESS(Experimental Research in Electronic Submission) 프로젝트에서 다양한 기종사이 멀티미디어 문서 교환을 위한 ODA 시스템을 개발하였으며, 유럽의 ESPRIT(European Strategic Programme for Research into Information Technology) 프로젝트내 PODA(Piloting of ODA)에서 ODA 변환기 및 툴킷을 개발하였다. 국내에서는 몇몇 연구소 및 학교 등을 중심으로 ODA 클래스 편집기, ODIF 변환기 등을 구현 발표하고 있다.[9,10,11]

이에 본 연구에서는 ODA에 기본한 텍스트, 도형 및 라스터 그래픽스 정보를 포함하는 문서를 서로 다른 시스템들간에 교환하기 위한 ODA 시스템에서 필수적으로 요구되는 포맷터를 설계 구현한다. 본 시스템은 문자 단위로 포맷팅하며, 공통 논리구조와 특정 논리구조 및 공통 배치구조를 입력으로 받아들여 특정 배치구조를 생성한다. 생성시 역추적 메커니즘과 배치 지시자들 사이 충돌을 우선순위에 따라 처리함으로써 공통 배치구조와 배치 지시자에 맞는 특정 배치구조를 생성하도록 한다. 또한 복잡한 내부 구조를 문서 작성자에게 투명하게 보임으로써 보다 쉽게 문서 작성을 수행하도록 하기 위해 사용자 접속을 대화식 처리 방식으로 구현한다. 구현 단계는 문서 응용 프로화일 중 단계 212)에 기본 하여 논리적 측면에서 텍스트, 도형, 라스터 그래픽스 정보를 가지며, 배치적 측면에서 주석(footnote), 두문(header), 미문(footer), 멀티컬럼(multi-column) 및 섹션(section) 번호 등이 가능한 문서를 작성할 수 있도록 한다. 구현 환경은 UNIX 환경 하에서 네트워크와 이식성이 좋은 X 윈도우와 Motif를 사용하여 구현하므로써 시스템의 이식성을 추구하고, 사용자에게 친숙하고 일관된 사용자 접속을 형성한다.[13]

2. 개방형 문서 구조 (ODA : Open Document Architecture)

ODA 문서구조의 기본 개념은 문서를 오브젝트 의미에 기본하여 계층구조를 갖는 논리구조와 문서 오브젝트를 물리적 구성에 기본하여 계층구조를 갖는 배치구조로 구성된다. 이들에 의한 문서형태는 논리구조만을 가진 수정가능한 문서형태와 배치구조만을 가진 포맷된 문서형태 또는 논리구조와 배치구조를 갖는 포맷된 수정 가능한 문서형태가 있다. 또한 ODA에서는 문서의 공통 특성을 기술하는 논리적/배치적 오브젝트들로 구성되는 공통 논리구조와 공통 배치구조를 갖으며, 이들 공통구조에 의해 생성되는 실제 문서의 특정 구조를 각각 특정 논리구조와 특정 배치구조라 한다.

오브젝트들과 오브젝트 클래스의 배치 또는 표현에 대한 정보는 스타일이라는 구성요소로 지정되는데, 배치처리에서 논리 오브젝트에서 배치 오브젝트를 생성하는 배치규칙을 제공하는 배치 스타일과 내용부의 가시화와 배치를 위해 논리 오브젝트와 배치 오브젝트에 적용되는 표현 스타일이 있으며 이들은 내용구조에 따라 각각 규정된다. 또한 문서는 하나의 문서 프로화일을 가지는데 이는 문서에 관계되는 모든 정보를 유지 관리한다.

ODA의 내용구조는 내용부 기본 배치 오브젝트에 관련된 최하위 구성요소로서 그래픽 요소, 제어함수, 내용과 관련된 부호화 방법, 표현속성으로 구성되며, 문서내용은 현재 문자, 도형 그래픽, 라스터 그래픽 내용구조를 정의하고 있으나 오디오 등의 시간 매개변수를 갖는 동적 미디어와 수치 데이터 등의 처리 미디어를 통합적으로 표현하는 구조로 연구 진행중이다.[14]

ODA의 문서처리 모델은 문서내용 편집과 공통 논리구조의 참조로 특정 논리구조를 편집하는 편집처리, 공통 배치구조의 참조로 특정 논리구조를 편집하는 편집처리, 공통 배치구조의 참조로 문서내용이 배치될 특정 배치구조를 생성하는 배치처리 및 표현 스타일의 표현속성을 적용하여 기본 배치 오브젝트의 표현속성과 배치 내용에 삽입된 제어함수에 의해 매체상에 문서를 나타내는 가시화 처리로 구성된다.[11]

3. 시스템 설계 및 구성

포맷터는 표현 매체 상에 사람이 인지할 수 있는 형태로 문서를 표현하는 가시화 처리에서 사용되도록 배치 지시자에 의해 제어되어, 공통 배치구조로부터 특정 배치구조를 생성하며, 이는 다시 다음과 같은 2가지 처리로 분류한다. 하나는 배치 스타일 정보와 공통 배치구조에 의해 배

치구조를 생성하는 문서 배치처리이고 다른 하나는 문서 배치구조에 의해 생성된 유효 영역에 내용부를 포맷팅하는 내용 배치처리이다.

포맷터는 공통 논리구조, 특정 논리구조, 공통 배치구조 및 배치/표현 스타일을 입력으로 받아들여 특정 배치구조를 생성하게 된다. 여과기는 교환하고자 하는 상대 시스템이 포맷된 또는 수정가능한 문서만을 요구시 여과기를 이용하여 해결한다. 내용은 3가지 형태인 텍스트, 도형, 및 라스터 그래픽스 내용을 지원한다. 포맷터의 구성도를 그림 1에 보인다.

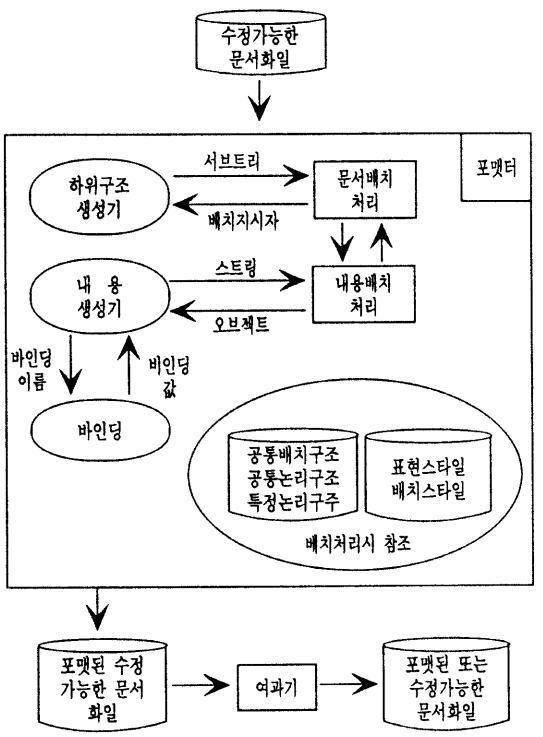

Fig. 1. Diagram of Formatter.

3.1. 문서 배치처리

문서 배치처리는 스타일 정보와 공통 배치구조의 참조로 특정 배치구조를 생성한다. 배치처리는 역추적 기능과 추측 기능을 이용하는데, 역추적 기능은 이미 공통 배치 구조에 의해 포맷된 구조가 잘못되어 다시 포맷팅하는 경우에 배치 오브젝트의 생성 과정을 재 추적하여 새로운 다른 구조를 생성한다. 추측은 선행배치, 모든 배치 지시자, 공통구조 정보로 부터 한 구조를 선택하는 것으로 배치처리 실행 전에 배치 지시자를 만족할 지 모르기 때문에 실제로 구현하는데 문제점이 있다.

그래서 본 시스템에서의 배치처리는 역추적 기능과 배치 지시자들 간의 규칙 처리로 해결한다. 공통 배치구조에 의해 특정 배치구조 생성시 지시자 없이 비 강제 배치 오브젝트를 생성하고, 이 생성이 잘못 생성되었을 때는 역추적 기능에 의해 이를 해결한다.

3.1.1 지시자 없이 비 강제 배치 오브젝트 생성

공통 배치구조에 의해 특정 배치구조를 생성 시에 강제 속성에 의해서는 무조건 생성되며, 배치 지시자에 의해 요구되지 않을 때 CHOICE, OPT 또는 OPT_REP 요소를 갖는 표현식일 때 하위 구조 생성자로부터 오브젝트는 생성될 수도, 안될 수도 있다. 이들은 다음과 같이 처리한다.

(1) CHOICE 요소 : 배치 지시자가 나타나지 않으면 첫 CHOICE 요소를 선택한다.

(2) OPT 요소 : 배치 지시자에 의해 요구되면 생성될 오브젝트처럼 OPT를 해석한다. 이에 대한 예를 그림 2에 나타내고 있다. 그림에서 현재 쪽에 논리 오브젝트가 배치되기를 원하면 INIT 쪽이 생성된다. 그러나 현재 쪽에 배치되기를 원하지 않으면, INIT 쪽은 생성되지 않으며, 첫 논리 오브젝트는 VPAGE에 배치된다.

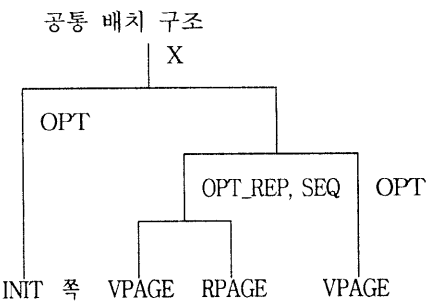

Fig. 2 Example of OPT Element Processing.

(3) OPT_REP 요소 : 배치 지시자가 없을 때 OPT_REP 오브젝트는 내용의 양에 따라 생성된다. 매번 어떤 내용을 배치하기 위한 공간을 필요시에 이 오브젝트를 반복해서 생성할 수 있다.

3.1.2 역추적 기능

(1) 역추적 기능이 요구되는 경우

① 배치 오브젝트에 포맷할 어떤 내용이 남아 있을 때, 논리 오브젝트의 배치 지시자가 나뉘지 않음 특성을 가지거나, 배치 오브젝트가

논리 오브젝트 내용을 포함할 정도로 크지 않을 경우, 논리 내용이 그림일 경우 혹은 논리 오브젝트의 표현 속성이 "선두고립행, 후미고립행"을 갖고 배치 오브젝트가 충분히 크지 않을 경우,

② 현재 배치 오브젝트가 이전의 논리 오브젝트를 참조하는 배치 지시자를 가지고 이전 논리 오브젝트 배치를 변형할 때,

③ 균형된 멀티컬럼 집합 프레임 일 때 하위 컬럼이 같은 높이를 가져야 되는 경우,

④ 형제 각주 프레임을 갖는 멀티컬럼 집합 프레임 일 때, 컬럼의 높이가 수평을 유지해야 할 때.

(2) 역추적 메커니즘

① 배치 모듈은 제거할 배치 서브트리와 대응되는 논리 오브젝트를 찾는다. 이를 위해 배치 구조와 논리구조의 유일한 링크인 내용부를 사용한다. 그리고 제거하기 전에 배치 서브트리가 강제적이 아닌지 검사한다.

② 역추적 처리가 시작되는 곳에서 논리 오브젝트는 플래그를 설정해 놓는다. 이 오브젝트는 다시 포맷될 가장 상위의 오브젝트가 된다.

③ 배치 모듈은 내용부를 제거하지 않고 배치 서브트리를 제거한다. 한번 내용부가 재 포맷되면, 다시 재 포맷하지 않는다.

④ 역추적 처리를 수행 시에 무한히 반복되는 것을 방지하기 위해, 역추적 처리가 수행될 때 역추적을 포함하는 배치 지시자나 논리 오브젝트는 스택(stack)에 푸시(push)하고, 역추적 처리가 수행된 뒤 논리 오브젝트가 스택에서 팝(pop) 된다.

3.1.3 배치 지시자 및 속성 처리 규칙

배치 지시자들 사이에는 서로 충돌이 일어날 가능성이 있다. 동일 논리 오브젝트에 적용될 배치 지시자들 사이 충돌은 표준에서 정의된 각 배치 지시자들의 우선순위에 의해 해결되지만, 다른 논리 오브젝트에 적용될 배치 지시자들 사이 충돌은 표준에 의해 해결되지 않기 때문에 각 지시자에 따라 해결한다. 다음에 각 지시자들에 대한 처리 방침을 설명한다.

(1) 배치 오브젝트 클래스

① 가장 높은 우선순위 단계를 가진다.

② 배치 오브젝트 클래스 지시자는 논리 오브젝트가 참조된 클래스의 배치 오브젝트 내에 전체적으로 배치되어야 된다.

(2) 새 배치 오브젝트(new layout object)

① 배치 오브젝트 클래스와 새 배치 오브젝트가 동일 논리 오브젝트에 적용되었을 때, 먼저 배치 오브젝트 클래스에 따라 배치 오브젝트를 탐색한다.

② 만일 이 배치 오브젝트가 새 배치 오브젝트에 알맞지 않으면 이 배치 오브젝트의 하위구조에서 탐색한다. 만일 오브젝트를 발견하지 못하면 새 배치 오브젝트를 무시한다.

(3) 배치 범주(layout category)

① 이 배치 지시자는 논리 오브젝트가 배치 범주에 따르는 허용된 범주들의 배치 오브젝트에 포맷되어야 함을 의미한다.

② 배치 지시자 "배치 범주"를 만나면 현재 배치 스트림은 대응되는 배치 스트림으로 스위치되고, 현재 배치 위치는 바뀐 현재 배치 스트림으로 스위치 한다.

③ 허용된 범주에서의 탐색시 "배치 오브젝트 클래스"나 "나뉘지 않음"이 상위구조 단계에서 정의되기 때문에 정지될 수 있다.

(4) 나뉘지 않음(indivisibility)

① 표준에서 "나뉘지 않음" 지시자는 가장 낮은 우선순위이다.

② 여러 배치 지시자가 동일 논리 오브젝트에 적용될 때 만일 "나뉘지 않음"이 가능하지 않으면 경고 신호를 발생하고 무시한다.

③ 복합 논리 오브젝트에 적용된 "나뉘지 않음"이 채워지기 불가능 시에 2 가지 경우가 발생한다. 한가지는 하위구조가 한 충돌 배치 지시지를 가질 때 오류와 함께 정지한다. 둘째 경우는 내용의 끝에 배치하기 위한 추가적 블록을 탐색시 나뉠 수 있는 논리 오브젝트를 다른 배치 오브젝트에 포맷한다.

(5) 채움 순서(fill order)

① 프레임의 위치와 치수에만 영향이 있다.

② 다른 배치 지시자와 충돌이 없다.

③ 하위의 프레임은 생성순서이므로 각주 프레임이나 "역 채움 순서(reverse fill order)" 프레임은 마지막 프레임일 필요는 없고, 이들의 최대 수직 치수가 갱신된다.

(6) 논리원(logical source)

① 논리원이 발견될 때 논리원에 의해 참조된 논리 클래스에 따르는 특정 논리 오브젝트가 생성된다.

② 상위(superior)가 없으면 이는 부분(partial) 특정 배치 구조이고, 그들의 서브트리가 생성된다.

(7) 표현 스타일

① 표준에 따라 표현 스타일은 단지 기본 구성요소에 적용되며, 내용 배치처리에 의해 운용된다.

② 선두고립행 크기와 후미고립행 크기는 문서 배치처리를 포함한다.

(8) 주석과 멀티컬럼

```
Footnote_multicolumn_process()
{
    컬럼 프레임 폐쇄;
    if(주석 프레임 생성 OR 주석 프레임
      갱신)
        각 컬럼 프레임의 수직 치수 = 컬
        럼 집합 프레임의 최대 치수;
    마지막 컬럼 수직 치수 = 모든 이전
    컬럼 크기 초과의 양이 감소;
    if (컬럼 집합 프레임 폐쇄)
        모든 컬럼 수직 치수가 첫 번째 컬
        럼의 블록에 할당한 후에 컬럼의
        첫 블록으로부터 역추적 처리;
    }
```

(9) 균형된 멀티컬럼

① 모든 컬럼이 같은 높이를 가져야 되며, 컬럼 집합 프레임이 폐쇄될 때 각 컬럼의 높이가 검사되어, 만일 그들의 값이 같으면, 완전하여 더이상 처리할 필요가 없다. 그러나 다르면 평균 높이가 계산된다. 각 컬럼 최소 수직 치수는 이 높이로 할당한다. 이때 컬럼의 첫 블록으로부터 역추적 처리한다.

② 평균높이(dim0) = 첫 컬럼 치수 - (컬럼 집합에서 수직 빈 영역/컬럼수)

③ 컬럼 수 계산 방법은 컬럼 집합의 하위 생성자가 SEQ 이면, 단지 컬럼수를 계수하며, 컬럼 집합의 하위 생성자가 OPT_RET이면, 이들 모두 생성되지 않기 때문에 단지 하위 구성요소들을 셀 수 없다.

- 컬럼 수 = (sfd - ote - ole) + max(ste, sle))/(fcd + max(ste, sle))
  - sfd : 상위 프레임의 치수
  - ote : 오프셀 후미 가장자리(offset trailing edge)
  - ole : 오프셀 선도 가장자리(offset leading edge)
  - ste : 분리 후미 가장자리
  - sle : 분리 선도 가장자리
  - fcd : 첫 번째 컬럼의 치수
  - max(a,b) : a, b 중 최대값

(10) 배치 오브젝트 치수와 위치

① 프레임 클래스는 수평/수직 치수에 대해 고정 혹은 변수(규칙 B) 치수를 가진다.

② 위치는 고정 또는 변수가 오프셀(offset), 분리(separation) 및 정렬(alignment)에 의해 제어된다.

③ 배치 처리동안 쪽, 프레임, 블록의 배치 오브젝트에 치수(ODA 속성에 의해)와 크기(오브젝트의 현재 크기에 의해) 파라메타를 첨가한다.

④ 블록이나 프레임이 생성 또는 개방시 치수가 클래스 치수 값으로 할당한다.

⑤ 오브젝트가 채워지고 크기를 갱신한다. 크기 갱신시 상위 치수는 치수가 클래스에서 설명된 상위에 도달 시까지 갱신된다.

3.2 내용 배치처리

내용 배치처리는 문서 배치처리에 의해 설명된 유효영역에 표현 스타일 정보를 이용하여 내용부를 포맷팅한다. 내용 배치처리의 흐름도를 그림 3에 보인다.

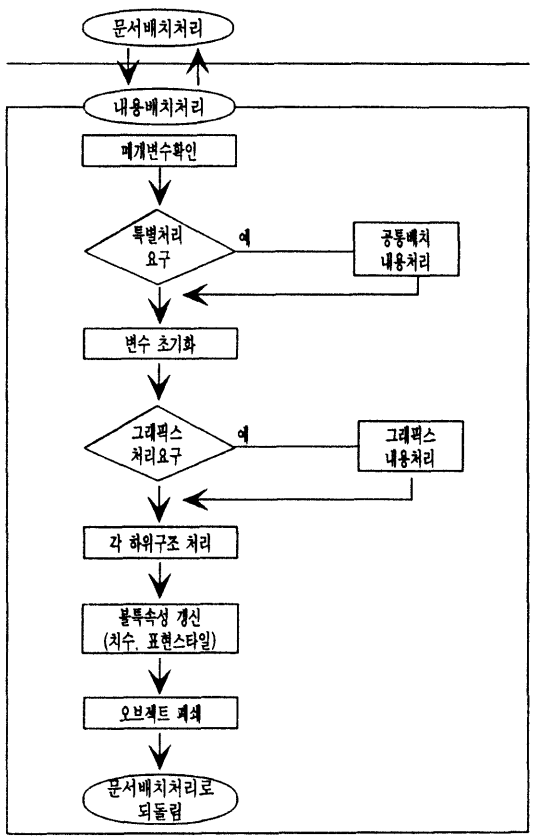

Fig. 3. Flowchart of Content Layout Processing

3.2.1 내용 배치처리 순서

(1) 포맷될 내용을 탐색 : 대부분 내용이 현재 논리 오브젝트나 그들의 클래스와 관련된 내용부에서 직접 접근할 수 있다. 경우에 따라서 내용 생성을 설명하는 오브젝트와 관련된 내용은 내용 생성 속성에 의해 설명된 표현식을 평가한 후 내용을 구성한다. 이들 표현식은 "바인딩(binding)"을 참조한다.

(2) 속성 평가 : 내용에 적용되는 표현 속성이 ISO 8613에서 설명한 ODA 강제값 결정에 따라서 처리한다.

(3) 내용 포맷팅 : 문자단위로 수행하며, 처리하는 동안 만나는 제어함수와 표현 스타일에 의해 처리한다. 채움은 단어 단위로 처리하고, 단어의 구분은 CR-LF, STAB, CR, BPH, LF, SOS-ST, 공백문자에 의해 구분한다.

3.2.2 각 내용 형태들의 표현 속성

(1) 문자 표현속성 : 포맷터에서 처리가능한 문자 표현속성을 표 1에 나타내고 있다.

Table 1 Character Presentation Attributes.

속 성	기 본 값
정렬	시작*, 끝, 중앙, 양끝정렬
문자방향	0°
문자경로	0°
문자 간격(고정)	120 BMU*(Basic Measurement Unit)
그래픽 문자 집합	ISO 6937/2, 8859/1
그래픽 묘사(rendition)	0, 1, 3, 4, 21, 22, 23, 24, 10~19(폰트)
행 배치 테이블	탭 20개 까지
행 진행방향	270°
행 간격(고정)	200 BMU
들여쓰기(indentation)	0*
선두고립행(orphan)	1*
후미고립행(window)	1*
초기 오프셋	들여쓰기*, 역방향

[*는 당연값]

(2) 도형 그래픽스 표현속성

도형 그래픽스 형태의 내용에서 블록의 치수계산을 위해 포맷터에서 운영한다. 도형 그래픽스 표현속성을 표 2에 나타내고 있다.

Table 2 Geometric Graphic Presentation Attribute.

속 성	기 본 값
이미지 치수	임의값
간격비 플래그	1
부호화 방법	CGM(2진 부호화)

(3) 라스터 그래픽스 표현속성

라스터 그래픽스 형태의 내용에서 블록의 치수계산을 위해 포맷터에서 운영한다. 라스터 그래픽스 표현속성을 표 3에 나타내고 있다.

Table 3 Raster Graphic Presentation Attribute.

속 성	기 본 값
이미지 치수	임의값
행 진행방향	270°
펠(pel) 경로	0°
펠 간격	0
간격비	1
클리핑(clipping)	임의위치
부호화 방법	T.6

3.2.3 바인딩

바인딩 속성은 오브젝트 클래스 기술의 비강제 속성이다. 오브젝트 기술을 위해 당연값(default value)이다. 이의 속성은 프린팅 가능한 스트링으로 구성된 바인딩 이름과 표현식으로 구성된 바인딩 값으로 구성된다. 바인딩 이름에 의해 참조된 표현식이 해석되어 스트링 값이 되돌려진다.

4. 구현 및 고찰

4.1 구현

본 시스템의 구현 환경은 SUN SPARC I 워크스테이션을 사용하였고, NCD 그래픽 터미널을 이더넷(ethernet)으로 연결하여 사용하였다. 이 시스템의 운영체제는 UNIX BSD 4.1.1이고, 사용자 상호접속을 위한 윈도우 시스템으로는 X11 R.4이며, 그래픽컬 사용자 인터페이스 소프트웨어로 OSF의 Motif 1.2를 사용하였고, 출력장치는 포스트스크립트(PostScript)가 내장된 프린터로 출력하였다.

포맷터에 입력되는 문서는 문서 클래스 편집기에 의해 작성된 공통구조 화일의 안내를 받아 특

정구조 편집기에 의해 작성된 특정구조 화일을 입력으로 받아 포맷된 문서를 생성하였다.

실제 두문, 미문, 번호 세그먼트, 탭 및 정렬 기능을 갖고 스타일 정보로 볼트체, 밑줄체, 이중 밑줄체, 위첨자(superscript)와 아래첨자(subscript) 속성을 갖고, 내용 정보로 텍스트, 도형 및 라스터 그래픽스 정보를 갖는 문서를 포맷팅하여 작성된 문서 예를 출력한 결과를 그림 4, 5에 보인다.

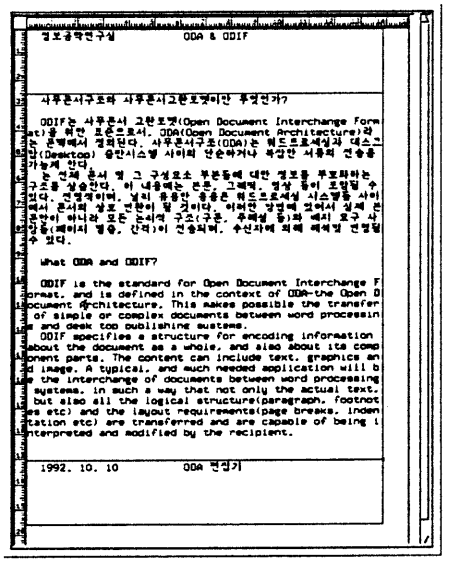

Fig. 4. Example 1 of document creation.

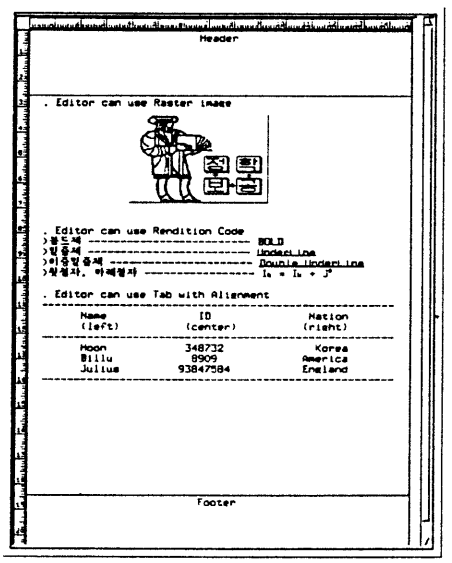

Fig. 5. Example 2 of document creation.

4.2 고찰

ODA 시스템 환경에서 요구되는 포맷터를 문자 단위로 포맷팅 하였으며, 공통 배치에 의해 포맷된 특정구조가 잘못되어 있는 경우 역추적 기능에 의해 재 포맷함으로써 이 문제를 해결할 수 있었으며, 역추적 기능이 무한히 반복되는 것을 방지하기 위해 스택을 이용하여 해결하였다. 또한 배치 지시자들 간의 충돌은 배치 지시자들을 처리하기 위한 규칙과 우선순위에 의해 해결하였다. 또한 대화식 처리 방식으로 사용자 접속을 구현하였다.

그러나 기존의 전자출판 시스템과 비교해 볼 때, ODA 시스템이 추구하는 문서 교환은 가능하지만 포맷팅 처리에서 동적으로 행단위와 파라그래프 단위 등의 처리 및 ODA에서 제공하는 스타일 정보의 포맷팅에 대한 확장과 함께 많은 기능에 있어서 앞으로 계속적인 연구가 필요하다.

5. 결 론

본 논문은 개방 문서처리 환경에서 이(異) 기종 시스템들 사이 문서 교환을 위해 널리 사용되는 문서구조인 ODA/ODIF에 기본한 문서처리 시스템에서 요구되는 포맷터에 관해 연구하였다. 포맷터는 역추적 메커니즘과 배치 지시자들의 처리 규칙에 의해 포맷팅함으로써 공통 배치구조에 의해 특정 배치구조 생성이 정확히 이루어 졌다. 또한 스택을 사용하여 역추적 기능이 무한히 반복되는 것을 방지할 수 있었다.

이후 ODA에서 제안하는 스타일 속성들을 위한 포맷팅 기능과 멀티미디어로의 응용을 위해 오디오, 비디오 등의 문서 형태에 대한 연구와 함께 성능 평가를 위한 연구도 요구된다. 국제 기능표준에 기본해 시스템을 구현하여 국제 경쟁력 강화에도 도움이 기대되며, 멀티미디어 문서정보 통신분야의 환경 구축에 크게 기여하리라 기대한다.

감사의 말씀

본 논문은 95년도 배재대학교 교내학술연구비 지원에 의하여 수행된 연구의 일부로 이에 감사를 드립니다.

참 고 문 헌

1. C. Meghini, F. Rabitti, C. Thanos, "Conceptual Modeling of Multimedia Document", IEEE Computer, pp. 23 - 30, Oct. 1991.
2. 阪田史郎, "クルーフウェアの 實現技術", Soft Research Center, pp.93~116, 148-165, 1992.
3. W. Herzner, "CDAM-Compound Document Access and Management", Eurographics Multimedia Workshop Proceeding, Stockholm, April 1991.
4. Jonathan Rosenberg, M. S. Sherman, Frank Giuffrida, "Translating among Processable Multimedia Document Formats using ODA", ACM Proceeding, pp. 61~70, Dec.1988.
5. Information Processing-Text and Office Systems-Standard Generalized Markup Language (SGML), International Organization for Standardization, Geneva, 1986, Includes Amendment 1, Reference ISO 8879 : 1986/A1, 1988.
6. Martin Bryan, "An Author's Guide to the Standard Generalized Markup Language", Addison-Wesley Publishing Company, 1991.
7. ISO 8613, Information Processing-Text and Office Systems-Office Document Architecture (ODA) and Interchange Format, International Organization for Standardization, 1988.
8. R. G. Herrtwich, L. Delgrossi, "ODA-Based Data Modeling in Multimedia Systems", TR-90-043, International Computer Science Institute, Aug. 1990.
9. "The CMU ODA toolkit Application Programmer's Interface", Information Technology Center of CMU, July 1988.
10. 임태훈외 3인, "ODA/ODIF를 기반으로한 이기종 시스템 사이에서의 문서교환", 한국정보과학회 봄 학술발표회논문집, vol. 19, no. 1, pp. 69 - 72, 1992.
11. 정회경, 이수연, "이 기종 시스템들 사이 문서 교환을 위한 ODIF 변환기 설계 및 구현", 한국정보과학회, Vol. 20, No. 5, May 1993.
12. ISO/IEC JTC1/SGFS N 303, pdISP 10610-Office Document Format FOD26-Enhanced Document Structure- Character, Raster Graphics and Geometric Graphics Content Architecture, EWOS, April 22nd, 1991.
13. Oliver Jones, Introcuction to the X Window System, Prientice Hall, 1989.
14. ISO/IEC JTC1/SC18/WG3 N 1898, Hyper ODA-Working Draft for Extending the ODA Standards to Support Hypermedia Applications, July 1991.