論文 94-19-9-5

객체지향 개념을 기반으로한 하이퍼텍스트 데이타 모델

正會員 李 在 舞* 正會員 林 海 喆**

An Extended Hypertext Data Model based on Object-Oriented Paradigm

Jae Mu Lee*, Hae Chull Lim** *Regular Members*

要 約

본 논문은 기존의 하이퍼텍스트 시스템의 문제점인 모델링 능력 미약 및 방향상실(lost in hyperspace, disorientation)문제를 최소화 하기 위하여 데이타 모델을 확장하고 이 확장된 모델이 객체지향 데이타베이스 시스템에서 표현되는 방법을 BNF 정의를 사용하여 형식화 하였다.

본 제안 모델은 모델링 능력 향상 및 개념적 항해를 위하여 링크에 의미를 부여하고, 노드와 링크를 기능에 따라 여러 타입으로 분류하여 효율적인 항해가 되도록 하였으며, 하이퍼텍스트 시스템에서 가장 문제가 되는 방향상실 문제를 방지하기 위하여 구조를 강화시키는 방법과 지능적인 항해 방법을 제안하였다.

ABSTRACT

We propose an extended hypertext data model based on object oriented paradigm that can easily the real world and semantics. We use the BNF notation to formalize the model.

In our model, We introduce conceptional navigation by associating semantics on links and drive intelligent navigation using weights on links to alleviate user disorientation problem which is currently somewhat vague.

We functionally classify the hypertext node into three types:Indexing node, Content node, Extract node and likewise classify the link into Alink type, Rlink type, Slink type. We believe that the typed node and typed link approach accommodate efficient query/search in hypertext.

I. 서 론

하이퍼텍스트 시스템(hypertext system)은 융통성 있는 그래프 구조를 갖기 때문에 자유로운 브라우징(browsing) 기능으로 관련 정보를 쉽게 접근할 수 있고, 비구조적 자료를 쉽게 관리할 수 있어서 여러 응용분야에 널리 사용되고 있다. 그러나 하이퍼텍스트 시스템은 아직 해결되지 않은 여러 문제점들이 남아 있다[1,2]. 본 연구에서는 하이퍼 텍스트 시스템이 지닌 여러 문제점 중 특별히 다음과 같은 몇 가지 문제점에 관심을 가지고 이의 개선방안을 제안한다.

*부산교육대학교
Pusan National Teacher's University
**홍익대학교 전자계산학과
Dpt. of Computer Science, Hongik Univ.
論文番號 : 93250
接受日字 : 1993年 12月 20日

1) 실세계를 노드와 링크의 단순한 모델로 표현하기 때문에 기존의 시스템에서는 의미를 표현하기가 어렵다[3].
2) 노드와 링크를 위한 타입 정의 기능이 없다[4].
3) 항해시 사용자에게 방향상실 문제가 일어난다. 방향상실의 문제는 사용자가 항해시 하이퍼텍스트 데이타베이스 상에서 현재 자신의 위치와 항해할 방향 감각을 잃어버리는 문제[5]와 자신이 돌아가야할 이전의 위치를 잃어 버리는 문제이다[6].

본 연구에서는 위 문제 1)을 해결하기 위하여 링크에 의미를 부여하고 이의 모델을 객체 지향 모델로 표현하여 모델링 능력을 향상시켰으며 문제 2)를 해결하기 위하여 노드와 링크를 기능에 따른 타입으로 분류하였다. 문제 3)의 방향상실 문제는 아직 해결되지 못한 하이퍼텍스트 시스템에서 해결되어야 할 주요한 문제이다. 방향상실 문제를 해결하기 위한 연구는 구조적인 해결[14][16][17], 항해 전략 개선[18], 사용자 인터페이스 강화[6][19], 질의 처리를 통하여 관련 정보 탐색등이 있다. 그러나, 현재 어느 방법도 완전한 해결책이 되지 못한다. 본 논문은 가장 근본적인 해결책인 구조적인 해결 방법으로 링크에 의미를 부여하여 개념적 항해를 통한 방법을 제안한다. 그리고 항해전략 개선 관점에서 링크에 가중치를 부여한 지능적 항해를 제안한다.

본 논문의 구성은 다음과 같다. 먼저 Ⅱ장에서는 기존 하이퍼텍스트 시스템의 문제점을 해결하기 위한 확장 모델을 정의하고 Ⅲ장에서는 확장된 하이퍼텍스트 데이타 모델을 객체지향 모델로 변환한다. Ⅳ장에서는 평가 및 고찰을 하고, Ⅴ장은 결론으로 맺는다.

Ⅱ. 확장된 하이퍼텍스트 모델

이 장에서는 앞에서 언급한 기존의 하이퍼텍스트 시스템이 가지고 있는 3가지 문제점, 즉 모델링 능력 미약, 노드와 링크를 위한 타입 정의 기능 부재, 항해시 일어나는 방향상실 문제 등을 최소화 하기 위하여 기존의 하이퍼텍스트 모델을 확장하였다. 확장모델에서는 링크에 의미를 부여하여 모델링 능력을 확장하였고, 노드와 링크의 기능을 역할에 따라 여러 타입으로 분류하였다. 본 제안 모델은 BNF형태의 형식언어로 정의한다. 본문에서 사용한 형식언어를 정의하면 다음과 같다.

:: = 정의한다
= 할당한다
| 논리합(OR)
* 논리곱(AND)
→ 인용한다(reference)

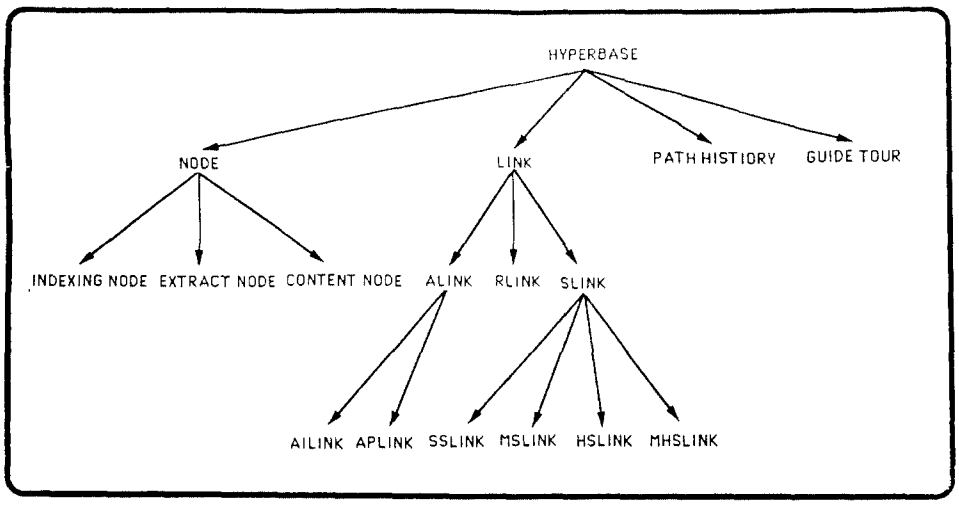

그림 1. 클래스 계층 구조
Fig 1. Class Hierarchy

* 이행적 폐쇄(reflexive transsitive closure)

하이퍼텍스트 시스템은 사용자가 텍스트에 비순차 접근을 가능하게 하는 상호 교차참조(active cross reference)가 가능한 데이타베이스이다. 하이퍼텍스트에서 노드는 문서가 되고 링크는 참조 기능을 갖는다. 이러한 하이퍼텍스트의 데이타베이스를 하이퍼베이스로 정의한다. 본 논문에서 제안한 하이퍼베이스의 클래스 계층 구조는 [그림 1]과 같다.

Hypertext :: = Nodes * Links
Nodes = "chunk of information"
Links = "cross reference"
Hyperbase = "database of hypertext"

1. 노드의 확장

노드는 하이퍼텍스트 시스템에서 표현되는 정보의 조각을 말한다. 본 모델에서는 노드의 개념을 단순히 언급되어지는 대상에서 확장하여 노드에 참조 기능을 추가하였다. 노드는 기능에 따라 내용 노드, 인덱싱 노드, 추출 노드로 분류한다. 내용 노드는 노드의 실제 자료 값을 가지고 있는 일반적인 노드로 버전 개념이 지원된다. 인덱싱 노드는 다른 실제 값을 가지고 있는 일반적인 노드로 버전 개념이 지원된다. 인덱싱 노드는 다른 노드를 참조하기 위한 노드로 다른 노드의 인덱스 값을 가지고 있다. 추출 노드는 기존의 노드에서 추출된 새로운 집합 노드를 언급한다. 노드에는 노드의 특성을 나타내기 위한 여러 속성들이 존재한다.

Nodes = Nid → (node * attribute)
Node :: = "information"
Nid :: = TOKEN
Nodes :: = 〈Indexing node〉 | 〈Content node〉 | 〈Conception node〉
Indexing node = "index information"
Content node = "content information"
Extract node = "result of operation"

1.1 인덱싱 노드

인덱싱 노드는 다른 노드를 참조하기 위한 노드로 노드의 내용에 다른 노드의 인덱스 값을 가지고 있

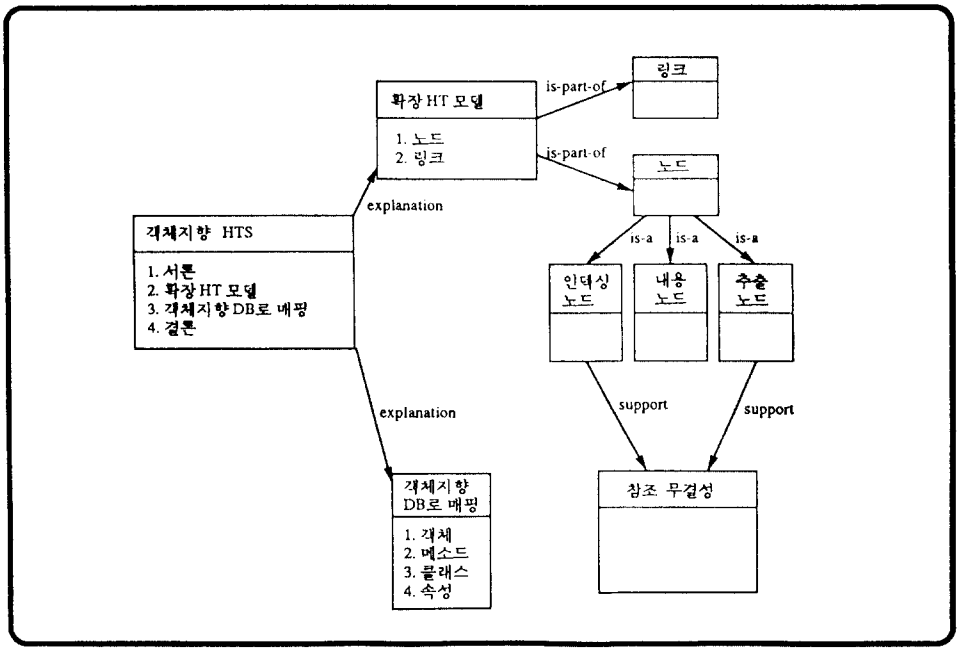

그림 2. 본 논문을 하이퍼텍스트 데이타 모델로 표현한 예
Fig 2. Example of Hypertext Data Model

다. 인덱싱 노드는 일반적인 문서에서 목차를 이용하여 보다 빠른 접근을 하는 것처럼 하이퍼텍스트 데이타베이스 상의 참조만을 위한 자료와 포인터를 가지고 있으므로 접근 경로를 줄이는 등의 효율적인 접근을 하기 위해 사용한다. 인덱싱 노드는 다른 인덱싱 노드를 포함할 수 있다. 따라서 복합 인덱싱 노드를 만들 수 있다. 인덱싱 노드내의 인덱스 값을 내용 노드가 수정되면 함께 수정되어야 하고 특히 삭제 시는 참조 무결성을 유지해야한다. [그림 2]는 본 논문의 내용을 예로 선정하여 하이퍼텍스트 데이타 모델로 표현한 예이다. [그림 2]에서 "객체지향 HTS" 노드와 "확장 HT 모델" 노드는 인덱싱 노드로 내용 노드의 타이틀을 가지고 있다. 이때 내용 노드의 값이 삭제될때 반드시 참조 무결성을 유지하기 위해, 이를 가리키는 인덱싱 노드의 인덱스 값도 수정되어야 한다.

1.2 내용 노드

내용 노드는 노드의 실제 자료 값을 가진 노드로 버전 개념이 지원된다. 노드의 크기는 하이퍼텍스트 시스템에 중요한 영향을 미친다. 즉, 노드의 크기가 작아지면 표현 능력이 좋고 질의 능력은 향상되지만, 하이퍼텍스트 데이타베이스가 복잡해지는 결과를 가져오고, 노드의 크기가 커지면 노드와 링크수가 줄고 간순화되지만 하이퍼텍스트 시스템의 표현 능력이 줄어든다. 본 논문에서는 노드의 기본 단위를 문서내의 기본 개념이나 주제로 정한다. 내용 노드의 예는 [표 1]과 같다.

표 1. 내용 노드의 예
Table 1. Example of content node

2. Extended hypertext data model The hypertext system is defined as a database that has active cross-references. Allowing the user to have nonsequential access to a text thereby making the reading process nonlinear. A hypertext can be modelled as a set of nodes and a collection of links where the nodes are documents and the links are cross-references.

1.2.2 슬롯(Slots)

노드의 서브 구조를 슬롯으로 정의한다. 이때 노드는 각기 내용을 갖고 그 자신의 유일한 이름을 갖는 슬롯으로 구성된 유일한 집합이다. 슬롯은 링크를 위한 연결점이 될 수 있으며 텍스트의 내용을 가진다. 슬롯내에는 핸들이 정의될 수 있으며, 앵커[9]와 데스티네이션을 명시할 수 있다.

Node = Slid → Slot
Slot ::= String * Attributes
String ::=CHAR*
Anchor, Destination = Nid : Nid * Slot
Slid ::= TOKEN
Slot ::= String * Handles * Attributes

1.2.3 핸들

슬롯내에는 핸들을 정의할 수 있다. [표 1]에서 hypertext system, database, cross-reference, nodes, links등이 핸들에 해당된다. 핸들은 슬롯내의 링크가 붙여 질 수 있는 집합으로 노드의 내용과 다른 노드들과의 연결을 가능하게 한다.

Handles = Hid → Region
Region ::= Position * Length
Position, Length ::= N

1.3 추출 노드

추출 노드는 내용 노드내의 인스턴스 집합에서 특정 속성 값에 의해 추출된 새로운 개념의 노드 집합으로 정의한다. 추출 노드에는 내용 노드의 실제 자료 값이 존재하는 것이 아니라, 내용 노드의 집합을 가리키는 포인터 집합만을 가지고 있다. 추출 노드는 내용 노드내의 자료 값의 중복을 피하고 접근 시간을 단축하기 위해서 사용된다. 또한 추출 노드도 인덱싱 노드처럼 인덱스 값이 삭제될 때 이를 가리키는 포인터 값이 함께 삭제되어야 한다. 추출 노드는 다른 노드를 가리키는 인덱스 값을 가지고 있다는 점에서 인덱싱 노드와 동일하지만, 인덱싱 노드는 노드 클래스를 언급하고 추출 노드는 한 클래스내의 인스턴스 집합을 언급한다는 점이 다르다.

2. 링크의 확장

하이퍼텍스트는 두 노드간을 링크로 연결한다. 링크가 활성화 될 때 사용자는 링크를 따라 다른 주제의 노드를 참조할 수 있다. 본 모델에서 제안한 링크의 연결 가능은 다음 3가지가 존재한다. 첫째, 두 노

노드간의 연결 둘째, 노드내의 슬롯과 노드간의 연결 셋째, 노드내의 핸들과 노드간의 연결이다. 링크는 하이퍼텍스트 시스템에 많은 영향을 미치는 가장 중요한 요소이다. 따라서 링크의 개선을 시도하였다. 인간의 사고는 연관의미를 나타내지는 못한다. 또한 하나의 원시 노드에서 다수의 목적 노드로 분기될 때 각 분기점을 구별할 필요가 있다. 따라서 본 논문에서는 링크에 의미를 부여하여 개념적 항해와 질의 기능을 강화할 수 있는 기반을 마련하였다. 또한 각 링크마다 항해시 방향 상실 문제를 해결하기 위한 목적으로 노드간의 연관 빈도에 따른 가중치를 두었다. 가중치는 디폴트 패스를 생성하여 사용자의 주요한 문제인 방향상실을 예방하는데 목적이 있다. 링크는 의미를 표현하고, 역(inverse)관계의 방향성을 자동 설정하여 양방향을 지원 하도록 하였다. 특히, 본 논문에서 링크 표현은 KMS, Notecards, Hypercard 등의 기존 모델에서 일반적으로 이용된 노드의 내용 값에 링크가 포함되는 내포링크(embedded link)가 아닌, InterMedia의 Web처럼 링크도 하나의 객체로 분류하는 독립 링크(independent link)를 사용하였다. 본 연구에서는 효율적인 항해 및 질의 처리를 위하여 링크를 기능에 따라 다음과 같이 분류하였다.

2.1 ALink 타입

is-a와 is-part-of관계를 나타내는 링크로 추상화를 표현하는 구조적인 링크이다.

1) AILink 타입 : 많은 링크로 인해 하이퍼텍스트상에서 발생하는 링크의 복잡성을 단순화하고 의미를 나타내기 위하여 추상화 개념인 일반화 개념을 나타내는 is a를 나타내는 타입이다. [그림 2]에서 노드들은 인덱싱 노드, 개념 노드, 내용 노드와 is a 관계를 가지고 있다.
2) APLink 타입 : 집단화 개념을 나타내는 is-part-of를 나타내는 타입이다. [그림 2]에서 노드와 링크는 "확장 HT모델"과 is part of 관계이다.

2.2 RLink 타입

링크에 두 노드간의 의미를 표현하는 링크이다. SLink 타입은 의미를 갖는 형태에 따라 SSLink타입, MSLink 타입, HSlink 타입, 그리고 MHSLink로 세분화 할 수 있다. SLink 타입의 사용 목적은 브라우징 능력 향상과 개념적 항해하고 이의 세분화 목적은 질의 능력 향상을 위해서이다. [그림 2]에서 "인덱싱 노드"와 "개념 노드"는 "참조 무결성"과 "support"의 링크를 갖는 SLink 타입의 링크이다.

1) SSLink 타입 : 노드간의 의미를 나타내며 다중값(set of value) 또는 하위 값을 갖지 않는 단순한 링크이다.
2) MSLink 타입 : 노드간의 의미를 표현하는 링크중 같은 의미를 표현할 수 있는 링크로 다중 값으로 표현된다. MSLink 타입의 목적은 특정 도메인에서 같은 의미의 링크 값이 존재할 때 의미상 중복되는 값을 표현하므로 에이리어싱(aliasing)[8] 및 항해시 질의 기능 향상을 위해서이다. 예를 들면, 링크의 의미가 "가르치다"로 표현된 경우 "지도하다"라는 관계를 갖는 경우, 이를 동일한 의미로 인식하지 못하는 문제가 발생한다. 이러한 문제를 해결하기 위하여 같은 의미의 링크들을 함께 표현한다.
3) HSLink 타입 : 노드간의 의미를 표현하는 링크중 주로 명사에서 동사 화한 동사로 링크간의 계승성을 표현할 수 있는 링크이다. 링크간의 내포관계로 표현된다. 예를 들면, [그림 3]에서 "현대자동차는 소나타를 생산한다", "코마코는 광고를 제작한다", "조선무역은 쌍감탕을 제조한다"는 의미는 "회사는 상품을 만든다"는 상위 개념으로 일반화가 가능하다. 이처럼 링크 의미를 일반화 시킬

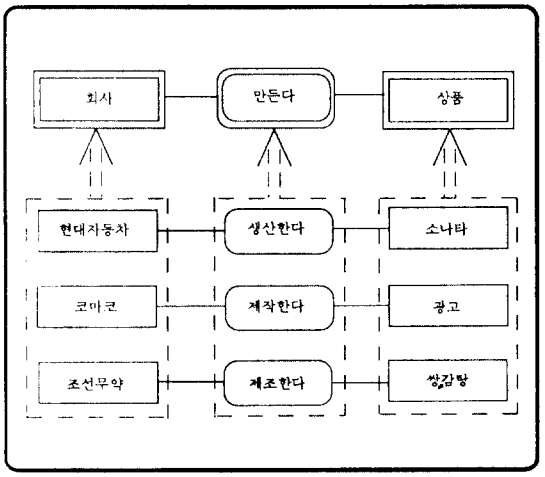

그림 3. HSLink 링크 타입의 예
Fig 3. Example if HSLink type

수 있는 동사를 HSLink 링크 타입으로 간주하고, 질의시 이들간의 관계를 고려하기 위하여 사용한다.

4) MHSLink 타입 : MSLink 타입과 HSLink 타입의 기능을 모두 갖고 있는 타입이다.

또한 링크의 구조에 따라 다음 링크를 두었다.

1.1 이진(Binary) 링크

일반적인 링크로 하나의 앵커와 하나의 데스티네이션을 연결하는 링크이다.

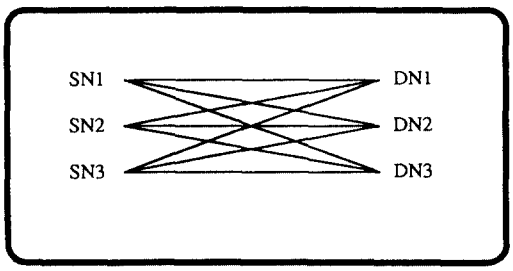

그림 4. 이진 링크
Fig 4. Example of Binary Link

1.2 배열(N-ary) 링크

다수의 앵커와 다수의 데스티네이션을 연결하는 링크이다. 본 모델은 링크에 의미를 부여하기 때문에 Lange 모델[9]에서 언급된 [그림 5]의 다중 링크에 의미를 부여한 [그림 6]과 같은 배열 링크이다.

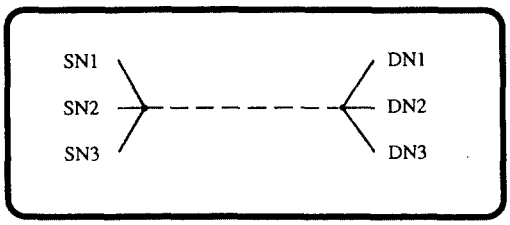

그림 5. 배열 링크
Fig 5. Example of N-ary Link

1.3 2차(2nd order) 링크

링크의 데스티네이션이 다른 링크의 앵커를 언급하는 링크이다. [그림 7]에서 노드 N5는 N2, N3, N4

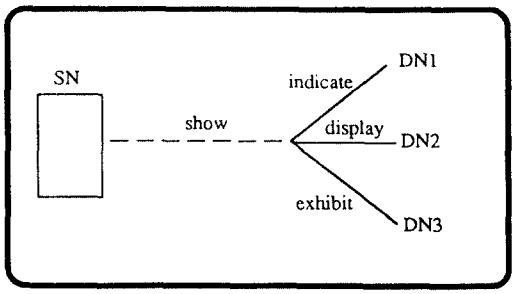

그림 6. 링크에 의미가 부여된 모델에서 배열 링크
Fig 6. Example of N-ary Link in semantic model

를 언급하는 N1을 복적 노드로 언급함으로 N2, N3, N4를 간접적으로 N1을 통해 복적 노드를 언급하고 있다.

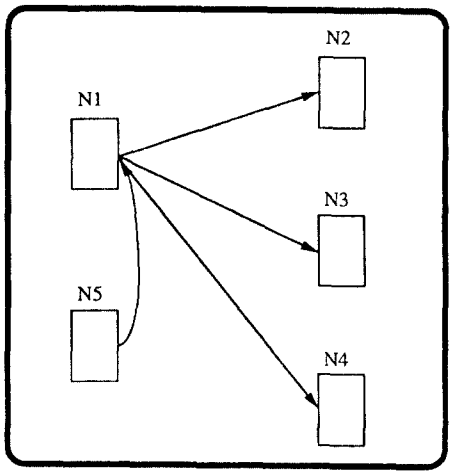

그림 7. 2차 링크
Fig 7. Example of 2nd order Link

Link ::= Connection * Attribute
Connections ::= Anchor * Destination
Anchor, Destination = Nid
Lid ::= TOKEN
Link type = Alink : RLINK : SLINK
ALINK = AILINK : APLINK
SLINK = SSLINK : DSLINK : HSLINK : DHSLINK

3. 경로 히스토리(path history)

사용자의 과거 항해했던 기록을 기억하고 정보 리스트이다. 이는 사용자가 자신이 항해했던 정보와 이전의 위치를 알려주기 위함이다. 이 정보 리스트에는 사용자의 이름, 경로 히스토리, 사용 시간 등이 저장된다.

4. 안내 정보(guide tour)

하이퍼텍스트는 사용자에게 항해의 선택을 주는 장점이 있지만 이로 인하여 사용자에 오히려 방향상실 문제를 유발한다. 항해 방향을 시스템이 정해놓고 어떤 개념에 따라 서적처럼 일관된 정보를 제공해 주는 것이 안내 정보이다. 이 안내 정보에는 주제에 따른 관련 노드의 순차적 집합이 있다.

Ⅲ. 하이퍼텍스트 시스템의 객체지향 모델로의 변환

1. 객 체

객체 지향 모델은 모든 실세계를 객체로 표현한다. 객체는 그 상태를 나타내는 하나 이상의 속성과 객체의 상태를 조작하는 메소드를 갖는다. 또한 객체는 유일하게 구별할 수 있는 OID 값을 가지며, 이 OID는 접근할 객체를 가리키는 유일한 수단이 된다. 하이퍼텍스트 시스템의 표현에서 노드와 링크, 노드 클래스, 링크 클래스, 노드, 링크 객체가 갖는 속성 등을 객체로 간주하며 이들 객체간의 항해는 객체가 갖는 OID 값으로 행한다.

Object ::= State * Operation
Hyperbase = Objid → Object
State = Node obnect : Link object
Objid = Nid : Slid
Objid ::= TOKEN

1.1 노드 객체

노드 클래스에는 노드 객체의 속성을 나타내기 위한 여러 속성들이 존재한다. 내용 노드 객체를 위한 속성에는 속성명과 To 링크 값과 From 링크 값을 나타내는 속성, 분석의 검색을 위한 키워드, 버전 값, 접근권, 데이타 내용을 위한 속성 등이 존재한다.

Node Object ::= 〈Node name〉〈To〉〈From〉
〈Key words〉〈Version number〉
〈Access right〉〈Data content〉
Node name ::= string*
To, From = Oid
Keywords = Empty : Keyword*
Version number = N
Access right = Author
Author = String*
Data content = string*

1.2 링크 객체

링크 객체는 의미를 나타내는 링크를 표현하기 위하여 정의된다. 링크 객체에는 링크 타입, source node, destination node, 노드간의 의미, 링크의 가중치를 위한 변수등이 존재한다. 객체의 속성 값은 단일 값이 일반적이나 에이리어징의 예방과 질의 기능 향상을 위하여 같은 의미의 링크는 다중 값으로 표현하였다.

Link Object = 〈Link type〉〈Source node〉〈Destination node〉〈Semantic〉〈Weight〉
Source node, Destination node = OID
Semantic = String*
Weight = N

2. 속성(attribute)

속성은 기본적으로 속성명과 값으로 매핑되며 사용자가 속성명을 정의할 수 있다. 속성 값의 도메인은 텍스트형, 숫자형, 문자형이 될 수 있으며, 다른 클래스가 될 수 있다. 속성의 종류에는 노드를 위한 노드명, TO 링크, From 링크와 링크 객체를 위한 링크 타입, source 노드, destination 노드, 의미(semantic), 가중치(weight)등이 존재한다. 노드 클래스의 속성중 노드명의 도메인은 문자열(character string)을 갖고, From링크와 To 링크 도메인은 링크 클래스이고 노드의 속성을 나타내기 위한 이외의 속성이 존재할 수 있다. 링크 클래스를 위한 속성중 링크 타입의 속성은 문자 값을 가지고, 원시 노드 속성의 도메인은 입력되는 원시 노드의 클래스 값을 가지며, 목적 노드 속성의 도메인은 출력되는 노드 클래스 값을 갖는다. 의미 속성은 링크의 의미 값을 가지며 같은 의미의 값을 표현하기 위하여 다중 값을 가질 수 있다. 가중치의 속성은 빈도에 따른 정수 값을 가지며, 지능적 항해를 위하여 동적으로 변화된다.

Attribute ::= 〈Attr name〉〈Attr value〉

Attr name = CHAR*
Attr value = (Attr value) * Attr value
Attr value = CHAR TYPE ; NUMERIC TYPE ; TEXT TYPE
TEXT TYPE = CHAR*

3. 클래스(class)

같은 속성과 메소드를 갖는 객체는 클래스가 된다. 클래스에는 여러 객체의 집합이 존재할 수도 있으며 단지 한 객체만이 존재하여 클래스이면서 동시에 객체가 될 수 있다. 그리고 각 클래스는 여러 개의 특수화(specialize)된 서브 클래스를 가질 수 있다.

Class ::= Nodes class | Link class | History class | Guide class
Class = "set of object"

4. 메소드(method)

하이퍼텍스트 시스템에서 메소드는 링크의 선택에 따라 디스플레이(display)되는 노드로 활성 데이타베이스(active database)의 트리거(trigger) 개념과 같이 룰(rule)로서 인캡슐레이션되어 링크 객체 OID와 목적 노드의 OID를 트리거 한다. 따라서 사용자가 선택한 링크간의 항해는 메소드로서 행하고, 저자의 의도대로 행할시는 항해 시마다 가중치가 변하여 동적인 디폴트 경로를 출력한다. 또한 객체내 값의 연산은 객체에 붙어 있는 메소드로 행한다.

메소드가 하는 일은 다음과 같다.

1) 자신의 인스턴스에서 내용을 디스플레이하고 사용자가 선택한 단어의 OID를 트리거한다.
2) 링크의 가중치를 자동적으로 변환한다.
3) 출력 디폴트 패스의 OID를 생성한다.
4) 객체내의 타입 또는 값의 입력, 수정, 삭제 등의 연산을 한다.
5) 노드 객체 인스턴스 값의 수정시 참조 무결성 유지를 위하여 자동적으로 이를 가리키는 인덱싱 노드 및 추출노드의 인덱스 값을 수정한다.

위 기능에 따라 다음과 같은 메소드를 정의한다.

1) 노드 객체 메소드
   CREATE NODE : 새로운 노드를 생성한다.
   HIGHLIGHT NODE : 특정 노드를 하이라이트 한다.
   REMOVE NODE : 노드를 제거한다.
   CHOOSE LINK : 링크를 선택한다.

2) 링크 객체 메소드
   CREATE LINK : 새로운 링크를 생성한다.
   REMOVE LINK : 링크를 제거한다.
   FOLLOW LINK : 링크를 따라 항해한다.
   RETURN OID : 데스티네이션 노드의 OID를 생성한다.
   PATHHISTORY : 링크 패스의 정보를 제공한다.

3) 슬롯 메소드
   ADD SLOT : 새로운 슬롯을 노드에 추가한다.
   REMOVE SLOT : 슬롯과 그 내용을 제거한다.

4) 인스턴트 메소드
   CREATE INSTANCE : 노드 인스턴스를 만든다.
   DELETE INSTANCE : 노드 인스턴스를 삭제한다.

5) 핸들 메소드
   ADD HANDLE : 슬롯내에 핸들을 추가한다.
   REMOVE HANDLE : 핸들을 제거한다.

6) 속성 메소드
   ADD ATTRIBUTE : 슬롯에 속성의 타입을 추가한다.
   REMOVE ATTRIBUTE : 속성이 제거된다.
   ASSIGN ATTRIBUTE : 속성 값이 속성명에 할당된다.
   READ ATTRIBUTE : 속성 값을 읽는다.

5. 복합 객체(composite object)

복합 객체 개념은 현재 대부분의 하이퍼텍스트 시스템에서 지원하지 않는 개념으로 미래의 시스템에서 반드시 지원되어야할 개념으로 많은 논문들에게 언급되고 있다[1]. 본 제안 모델은 객체 지향 개념을 이용함으로 자연스럽게 복합 객체의 클래스를 언급한다. 이미 객체의 도메인은 프리미티브 클래스인 경우 값을 갖거나, 비프리미티브(nonprimitive) 클래스인 경우 다른 객체의 OID 값을 가지고 있다. 다른 노드 객체를 링크하는 포인터도 결국 OID 값으로 클래스 복합 계층(composition hierarchy)을 따라 항해하게 된다. 복합 객체간의 언급은 다른 객체도 공유할 수 있는 공유 독립 복합 참조(shared indepen-

dent composite reference)[10]를 따르는 것이 하이퍼텍스트 시스템에 적합하다.

6. 클래스 계층(class hierarchy)

노드 클래스 및 링크 클래스도 서브 클래스를 가질 수 있다. 이때 모든 속성의 타입과 메소드가 완전 계승된다. 또한 다중 계승(multiple inheritance)을 이루고, 클래스는 루트가 존재하는 방향성을 갖는 비순환 그래프를 이룬다. 따라서 모든 노드 객체는 클래스 클래스로부터 접근 가능하다. 또한 다중 계승 상황에서 속성 메소드의 이름 충돌이 일어나면 우선 순위를 따른다. 하이퍼텍스트 시스템에서 클래스 계층 구조에서 얻을 수 있는 장점은 단순화를 이룰 수 있고, 계승성으로 인하여 모델링 능력을 향상시킬 수 있다는 것이다.

7. 개념적 스킴

본 모델의 개념적 스킴은 [그림 8]과 같다.

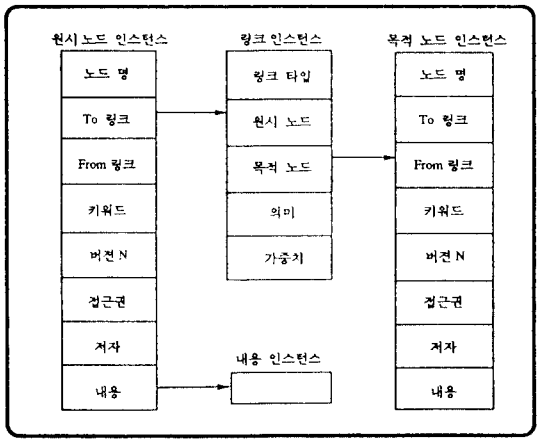

그림 8. 개념적 스킴
Fig 8. Generic Model

[그림 8]에서 노드 인스턴스는 원시 노드 인스턴스와 목적 노드 인스턴스가 존재하는데 각 인스턴스에는 노드 명, To 링크, From 링크, 키워드, 버전 N, 접근권, 저자, 내용 등의 속성들을 갖는다. 노드명의 도메인 타입은 문자형이고 To 링크는 출력되는 링크 클래스의 OID 값을 문자형으로 갖고, From 링크는 입력되는 링크 클래스에서 OID 값을 문자형으로 갖고 있다. 키워드는 문서 객체에서 주요한 단어를 알고리즘으로 추출하여 문자형으로 다중 값을 갖고 접근권은 read/write의 다중 값을 문자형으로 갖고 저자는 그 문서의 저자 이름을 나타내는 문자형을 갖는다. 내용 속성은 중복되는 문서를 효과적으로 표현하기 위하여 내용 노드의 인스턴스를 가리키는 OID 값을 갖는다. 노드 클래스는 서브 클래스를 가질 수 있고 이때 메소드와 속성 타입이 계승된다. 링크 클래스는 링크타입, 원시 노드, 목적 노드, 의미, 가중치 등의 속성을 갖는다. 링크 타입은 도메인 타입으로 문자형을 갖고, 원시 노드는 입력 노드 클래스의 OID 값을 가지고 있으며, 목적 노드는 출력 노드의 OID 값을 다중 값으로 가지고 있고 의미는 문자형을 다중 값이나 단일 값으로 갖는다. 가중치는 정수의 도메인 값을 갖는다. 목적 노드 인스턴스는 원시 노드 인스턴스와 동일한 속성을 가지며 원시 노드 인스턴스와 같은 도메인 타입을 갖는다. 노드 클래스는 원시 노드 인스턴스 값을 가지면서 동시에 목적 노드 인스턴스 값도 가질 수 있다.

8. 모델의 예

위의 모델을 설명하기 위한 실제 예는 [그림 9]와 같다.

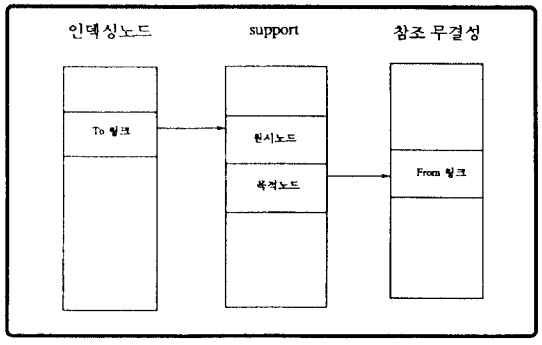

그림 9. 개념적 스킴 예
Fig 9. Example of Generic Model

[그림 9]는 [그림 2]에서 "인덱싱 노드"는 "참조 무결성"을 "support"하나는 관계를 나타낸다. "인덱싱 노드"는 원시 노드 인스턴스가 되고 "참조 무결성"은 목적 노드 인스턴스가 되며, "support"는 링크 인스턴스이다. 인덱싱 노드의 "to 링크"의 속성 값은 "support" 객체를 가리키는 OID 값을 가지고 있으며, "support"내의 목적 노드의 속성 값은 "참조 무

결성" 객체를 가리키는 값을 가지고 있다. 이들 객체들은 서브 클래스를 가질 수 있으며, 슈퍼 클래스를 가질 수도 있다.

Ⅳ. 평가 및 고찰

Leggett[11]는 R-mode[12], Lange mode[9], Dexter model[13]모델을 비교하였다. 본 장에서는 Leggett의 비교를 기초로 하여 본 제안 모델과 3개의 참조 모델을 비교를 한다.

표 2. 본 제안모델과 기존의 하이퍼텍스트 모델과의 비교
Table 2. Comparison of Model

	타 입	링 크	앵 커	형기화?
R-모델	시스템 정의를 위한 메타 모델	관계는 정의할 수 있지만 링크가 없다	앵커가 없다	않함
Lange 모델	하이퍼텍스트 모델	허상 참조 (DanglinLinks) 발생 가능	앵커 영역(regions)	VDM
Dexter 모델	하이퍼텍스트 시스템 모델	허상 참조 불가	앵커	Z
제안 모델	하이퍼텍스트 모델	허상 참조 불가	앵커	BNF

[표 2]에서 비교한 구조적 특성외에 제안 모델이 기존 모델들과의 특별히 구별되는 점은 링크에 타입이 존재하고, 링크 표현을 독립 링크로 하였으며, 링크에 의미와 가중치가 부여된다는 점이다. 또 다른 특징은 객체 지향 모델을 이용함으로서, 하이퍼텍스트 시스템에서 다음과 같은 잇점이 있다.

1) 항해적인 모델이므로 하이퍼텍스트 시스템의 항해를 자연스럽게 표현할 수 있다.
2) 하이퍼텍스트의 대부분을 이루는 복합 객체를 표현하는데 용이하다.
3) 계층성을 이용함으로 링크 수를 줄이고 단순화를 이룬다.

객체지향 개념을 이용한 하이퍼텍스트 시스템으로는 Intersect[3] [4], Mac-Web[14], Lange 모델 등이 있다. 객체 지향 개념을 하이퍼텍스트 데이타 모델의 설계시 이용하는 방법에는 두가지 형태가 있는데, 첫째 방법은 하이퍼텍스트 데이타 모델을 직접적으로 객체지향 개념 하에 표현하는 방법으로 이의 모델은 Intersect, Mac-Web등이 있으며, 두번째 방법은 하이퍼텍스트 데이타 모델을 설계하고 이를 객체 지향 모델로 매핑하는 방법으로 Lange 모델 등이 있다. 첫번째 방법은 수행시 효율성이 좋다는 장점이 있으나, 호환성 없고 구현이 어렵다는 단점이 있다. 반면에 두번째 방법은 수행시 효율성은 떨어지나 구현하기 쉽고, 기존의 모델들과 호환성 있게 표현할 수 있다는 장점이 있다. 본 제안 모델에서는 구현을 용이하게 하기 위하여 두번째 방법을 선택하였다.

기존의 하이퍼텍스트 시스템에서 최근 관심이 있는 것은 링크의 역할인데, 일반적으로 링크를 복잡성과 방향상실 문제를 가져오는 해로운 존재로 생각하고, 이의 기능을 축소하는 추세이다[5]. 그러나 본 연구에서는 역으로 링크의 의미를 표현하도록 하여 모델링 능력을 강화하고, 링크까지도 질의 대상으로 포함시켜 강력한 질의 기능을 갖도록 하였다. Mac-Web 모델에서처럼 개념적 항해를 통하여 항해시의 방향상실 문제를 해결하도록 하였다. 또한 링크마다 언급 빈도에 따른 가중치를 두어 항해시의 방향상실 문제를 해결하도록 하였다. 또한 HSLink 타입의 경우 일반화를 통하여 단순화를 이룰 수 있는 장점을 가지고 있다. 이렇게 링크의 기능을 확장하기 위하여 링크를 독립링크로 표현하였다. 이로 인한 문제점으로는 객체수가 많아지고 참조수가 늘어갈 수 있다는 단점이 있다. 노드는 슬롯과 핸들을 정의하여 융통성 있게 크기를 설정함으로 객체의 단위를 크게하고 단순화 시킴으로 노드 객체의 수를 어느 정도 감소시킬 수 있다. 그리고 노드는 노드간의 유이성에 따라 집단화를 이루어 상위 개념으로 추상화로 단순화를 이루어 복잡성을 어느 정도 해결할 수 있다.

안내정보 및 링크의 가중치는 어느 방향으로 진행되어야 하는지를 모르는 사용자에 항해 방향을 제시해주는 역할을 할 수 있다. 그리고 인덱싱 노드는 접근 경로 및 참조수를 감소시키는 역할을 하는데 유용하다. 추출 노드는 질의 처리시 실행 면에서 효율성이 있다. 링크의 타입들은 각기 역할에 따라 다른 처리를 하는데 유용하다. 방향상실 문제 해결 면으로 보면, 링크에 의미를 부여하여 개념적 항해를 통한 구조적 해결과 링크에 가중치를 두어 지능적 항해전략으로 해결하였다.

Ⅴ. 결 론

본 논문은 기존의 하이퍼텍스트 시스템의 문제점을 연구하고 이를 해결하기 위한 확장 모델을 정의하고, 이의 적합한 모델로 객체 지향 모델을 선택하여

효율적인 설계를 하였다. 본 데이타 모델의 특징은 다음과 같다.

1) 노드를 기능별로 분류하였고 특히 제안한 인덱싱 노드 및 개념 노드는 노드의 탐색 경로 깊이를 줄이는데 효과적이다.
2) 링크에 의미를 부여하여 기존의 노드, 링크의 단순한 모델로는 표현하기 어려운 의미를 부여함으로 모델링 능력을 향상시키고 개념적 항해를 통해 질의 능력 향상 및 방향상실 문제의 원인인 약한 구조를 강화할 수 있었다. 또한 원시노드에서 언급되는 링크가 여러개 존재할 때 사용자에게 링크의 의미를 통하여 다수의 링크 중 하나를 선택하므로서 더 상세한 정보를 제공한다.
3) 객체 지향 모델을 이용하므로 기존 모델의 약한 표현력을 높이고 그래프 구조로 항해가 자연스럽게 표현되고 복합 구조를 통해 하이퍼미디어 시스템에서 많은 복합 구조를 쉽게 표현할 수 있으며, 재사용성을 이용하기 때문에 링크 수를 줄이고 단순화를 이룰 수 있다.
4) 하이퍼텍스트의 주요한 문제인 방향상실 문제를 링크에 의미를 부여한 개념적 항해 방법과 링크에 가중치를 부여한 지능적 항해 방법을 통하여 방향상실 문제를 해결하였다.
5) 객체지향 데이타베이스 시스템을 이용하므로 선언적인 객체지향 질의어를 통하여 레코드 인스턴스 단위의 항해에서 집합단위의 효율적인 항해가 가능하다.

본 논문의 문제점으로는 독립 링크로 표현하므로 객체수가 많아지고 참조수가 증가되는 문제가 발생할 수 있다는 점이다. 이를 위해서는 설계시 노드 객체의 크기 및 링크 수의 제한으로 어느 정도 해결할 수 있으며 효율적인 구조의 단순한 전략 등이 필요하다. 본 논문에서 다루지 못한 저장 구조 및 질의 처리 방법과 복잡성의 단순화 방법등은 추후 연구 과제로 남겨놓는다.

참 고 문 헌

1. Frank. G. Halasz, Reflexions on Notecards : Seven issues for the next generation of hypermedia systems, CACM, 31(7), pp. 836-852, JUL, 1988.
2. K. Tanaka, N, Nishikawa, S, Hirayama, K. Nanba, Query pairs as hypertext links *IEEE, Data Engineering*, pp. 456-463, 1991
3. B. Wang P. Hitchcock, A method of combining oject oriented database with hypertext to support complicated documentation environments, *Proceedings Multimedia Information Systems*, pp. 52-66, JUL,. 1992.
4. B. Wang, P. Hitchcock, Intersect : A general purpose hypertext system based on an object oriented database, *DEXA*, pp. 459-464, 1991.
5. J. Conklin, Hypertext : An introduction and survey, *IEEE Computer*, pp. 17-40, SEP, 1987.
6. J. Nielsen, The art of navigating through hypertext, *CACM*, 33(3), pp. 296-310, 1990.
7. Roy RADA, *Hypertext : from text to expertext*, MaGraw-Hill Book Company, pp. 35, 1991
8. Ben Shneiderman, Greg Kearsley, *Hypertext hands-On*, Addison-Wesley Publishing Company, pp. 43, 1989.
9. Danny B. Lange, A formal of hypertext, *Proceedings of Hypertext standardization workshop*, NIST, pp. 145-166, JAN, 1990.
10. Won Kim, *Introduction to object-oriented database*, The MIT Press, pp. 24
11. John. J. Leggett, Comparision of the three models *Proceeding of the Hypertext Standardization Workshop*, NIST, pp. 145, JAN, 1990.
12. Richard Furuta, P.David stotts, The trellis hypertext model *Proceedings of the Hypertext Standardization Workshop*, NIST, pp. 83-93, JAN, 1990.
13. Frank. G. Halasz, Mayer Schwartz, The dexter hypertext reference model *Proceedings of the Hypertext Standardization Workshop*, NIST, pp. 95 131, JAN, 1990.
14. Jocelyne NANARD, Marc NANARD, Using structured typeds to incorporate knowledge in hypertext, *Hypertext 91 Proceedings*, pp. 329-343, 1991.
15. Laura de young, Linking considered harmful, *Proceeding of the First European Conference on Hypertext*, pp. 238-249, INRIA, France, NOV,

1990.

16. M.Benstein, Hypertext and technical writing, *Proceeding of the First European Conference on Hypetext,* INRIA, France, NOV, 1990.
17. A. Rodorigo, Botofogo, Ben Shneiderman, Identifying aggerates in hypertext structure, *Proceeding of the Hypertext 91 Conference,* ACM press, TEXAS, 1991
18. P. Clitherow, D.Riecken, M, Muller, Visar:a system for inference and navigation of hypertext, *Proceeding of the Hypertext 89 Conference,* ACM press, Baltimor, 1989
19. C. Marshall, guide tours and on-line presentation : How authors make existing hypertext intelligible for readers, *Proceeding of the Hypertext 91 Conference, DEC, 1991, Santoni, Texas.*

李 在 舞(Jae Mu Lee) 정회원

1983년 2월 : 숭전대학교 계산통계학과 졸업
1985년 2월 : 홍익대학교 전자계산학과 석사
1994년 8월 : 홍익대학교 전자계산학과 박사
1987년~현재 : 부산교육대학교 실과교육자 부교수
※주관심분야 : 하이퍼텍스트 시스템, 객체지향 데이타베이스 멀티미디어 시스템, ICAI

林 海 喆(Hae-Chull Lim) 정회원

1976년 : 서울대학교 계산통계학과 졸업
1978년 : 한국과학기술원 전자계산학과에서 석사학위 취득
1988년 : 서울대학교 컴퓨터공학과에서 박사학위 취득
1978년~1981년 : 현대 엔지니어링 대리
1981년~현재 : 홍익대학교 전자계산학과 교수로 재직
※주관심분야 : 불완전 데이타베이스, 객체 지향 데이타베이스, 분산 데이타베이스