論文95-32B-5-2

객체 지향 멀티미디어 데이타베이스를 위한 멀티미디어 질의어

(A Multimedia Query Language for Object-Oriented Multimedia Databases)

羅 然 默 *, 李 錫 浩 **, 金 奎 喆 *

(Yunmook Nah, Sukho Lee, and Kyuchull Kim)

요 약

본 논문은 시공간 상에서 모노미디어 데이타가 통합된 형태로 멀티미디어 데이타를 정의하고 조작할 수 있는 멀티미디어 질의어 MQL(Multimedia Query Language)을 제안하였다. MQL은 객체-관계 모델이라고 하는 멀티미디어 데이타 모델과 멀티미디어 데이타에 대한 연산을 정형적으로 기술하는 멀티미디어 객체 해석을 기반으로 개발되었다. MQL은 클라스 정의와 객체 검색, 삽입, 갱신, 삭제 연산을 표현할 수 있는 구문을 제공한다. MQL을 이용하여 복합 시공간 구조, 동기화와 순차성과 같은 다양한 관련성을 이용한 사용자의 질의 요구를 표현할 수 있다.

Abstract

In this paper, we propose a multimedia query language MQL which defines and manipulates multimedia data as integration of monomedia data in time and space. The MQL is designed for a multimedia data model, called the object-relationship model, and based on the multimedia object calculus which formally describes operations on multimedia data. The SQL-like syntax for class definition and object manipulation, such as retrieval, insert, update, and delete, is defined. We show how the MQL can represent the user queries using composite temporal-spatial class structures and various relationships, such as equivalence and sequence.

I. 서 론

멀티미디어 하드웨어와 소프트웨어 기술의 발전과 더불어 멀티미디어 데이타에 대한 관심이 고조되고 있다. 사무 정보 시스템, CAD/CAM, 지질 정보 시스템 등의 새로운 응용은 전통적인 정형 문자/숫자 데이타보다는 비정형 멀티미디어 데이타를 요구하고 있다.

멀티미디어 응용과 관련한 연구와 개발이 활발히 진행되어 MULTOS[8], MINOS[2], ORION[14], OMEGA[7], VODAK[5]과 같은 많은 실험적인 멀티미디어 데이타베이스 관리 시스템이 만들어졌다. 그러나, 이러한 연구의 대부분은 텍스트, 이미지, 음성 등의 모노미디어 데이타를 저장하고 검색하는 데 치중하고 있고, 이러한 모노미디어 데이타가 통합된 진정한 의미의 멀티미디어 데이타의 표현 및 처리 기능 부분은 미흡하다고 할 수 있다. 데이타 언어의 경우도 객체-지향 질의어에 모노미디어 데이타를 생성, 삭제, 압축하는 객체 조작 메소드만을 추가 지원하고 있다. 예를

* 正會員, 檀國大學校 컴퓨터工學科
(Dept. of Computer Eng., Dan-Kook Univ.)
** 正會員: 서울大學校 컴퓨터工學科
(Dept. of Computer Eng., Seoul Nat'l Univ.)
※ 이 논문은 1994년도 한국학술진흥재단의 공모과제 연구비에 의하여 연구되었음.
接受日字: 1994年9月2日, 수정완료일: 1995年4月29日

들어, 모노미디어 입출력 메소드를 지원하는 ORION 시스템은 '(capture capture-device captured-object)' 메세지를 이용해 모노미디어 객체를 읽어 들인다. 멀티미디어 문서 시스템인 MULTOS에서는 'FIND DOCUMENTS WHERE 〈condition〉', 'RETRIEVE IMAGES 〈condition〉' 형태의 검색 질의어만을 지원하며 검색 이외의 연산을 위한 일반 질의 기능이 결여되어 있다. VODAK의 멀티미디어 확장에서는 모노미디어 데이타를 조작하기 위한 'attach-Video', 'playVideo' 등의 메소드만을 지원하고 있다. 일부 시스템은 모노미디어 데이타를 내용이나 자연어 설명 정보를 이용해 검색하는 데 치중하고 있고, 일부는 모노미디어 데이타의 신속한 검색에 관심을 두고 있다.

멀티미디어 데이타를 정형 데이타가 아닌 텍스트, 이미지, 음성 등의 비정형 데이타로 보기도 하지만, 보다 정확히 정의하면 다양한 모노미디어 데이타와 전통적인 정형 데이타를 시간과 공간상에서 통합한 것으로 볼 수 있다. 현존하는 멀티미디어 데이타베이스 시스템들은 멀티미디어 데이타의 중요한 의미의 하나인 모노미디어간 관련성(inter-monomedia relationships), 즉 시공간 관련성(temporal-spatial relationships)의 표현이 결여되어 있다. 멀티미디어 기능이 보강된 상용 관계 DBMS를 포함한 기존의 시스템들은 주로 모노미디어 데이타를 저장하고 관리하는 기능 만을 지원하고 있으며, 이러한 데이타를 시간과 공간 상에서 통합하는 일은 사용자가 특별한 도구나 프로그래밍 언어를 이용하여 스스로 수행하여야 한다.

이 논문은 시공간 상에서 모노미디어가 통합된 형태로 멀티미디어 데이타를 정의하고 조작할 수 있는 멀티미디어 질의어 MQL(Multimedia Query Language)을 제안한다. MQL은 멀티미디어 데이타를 시간과 공간 상에서 정형 데이타와 모노미디어 데이타가 통합된 것으로 정의하는 객체-관계 모델(object-relationship model)[10,11]이라고 하는 멀티미디어 데이타 모델을 위한 것이다. MQL의 연산적 의미(operational semantics)는 멀티미디어 데이타에 대한 연산을 정형적으로 기술하는 멀티미디어 객체 해석[12]에 의해 주어진다. MQL은 클라스 정의와 객체 검색, 삽입, 갱신, 삭제 연산을 표현할 수 있는 SQL 형태의 구문을 제공한다. MQL을 이용하여 복합 시공간 구조, 동기화와 순차성과 같은 다양한 관련성을 이용한 사용자의 질의 요구를 표현할 수 있다.

본 논문의 초점은 사용자의 질의 요구를 MQL로 어떻게 표현할 수 있는 지를 보이는 데 있다. 이에 관련된 세부 메소드의 구현 방법과 질의 처리 방법은 다루지 않기로 한다.

본 논문의 구성은 다음과 같다. 2장에서는 MQL의 기반이 되는 멀티미디어 데이타 모델과 멀티미디어 객체 해석을 간략히 설명한다. 3장에서는 사용자 정의 클라스를 정의하는 MQL의 정의문를 기술한다. 4장에서는 멀티미디어 객체를 조작하기 위한 MQL의 조작문을 제안한다. 마지막으로 5장에서 결론을 맺는다.

Ⅱ. 멀티미디어 데이타 모델과 멀티미디어 객체 해석의 개관

멀티미디어 데이타 모델인 객체-관계 모델(ORM: object-relationship model)[10]은 객체-지향 개념을 바탕으로 멀티미디어 응용이 요구하는 데이타의 다양한 표현 구조와 그들 사이의 복잡한 관련성을 데이타베이스에 표현할 수 있도록 한 데이타 모델이다. ORM에서는 실세계가 객체(objects)와 관계(relationships)로 이루어진다. Little은 멀티미디어 데이타의 후행처리(postprocessing)시 구성 요소 데이타 간의 시간 관계를 표현하기 위하여 Allen이 제안한 7가지 시간 관계를 사용하고 있다[16]. ORM은 이 7가지 시간 관계와 MHEG의 동기화 타입을 시간 순차 관계(R_{ts}), 시간 동기화 관계(R_t), 지연 객체(delay object)를 이용하여 모두 표현할 수 있다[10]. ORM을 통한 개념적인 모델링 결과는 논리적 단계에서 객체-지향 모델의 클라스와 인스턴스로 표현된다. 클라스 객체와 클라스 관계는 클라스로, (클라스 객체를 제외한 나머지) 객체와 객체 수준의 관계는 인스턴스 객체로 사상이 된다.

멀티미디어 객체 해석[12]은 객체-관계 모델에 의해 모델링된 멀티미디어 데이타에 대한 연산을 정형적으로 기술하기 위해 사용된다. 멀티미디어 객체 해석은 메세지 전달 패러다임(message passing paradigm)을 기반으로 멀티미디어 데이타에 고유한 연산을 추가하여 확장한 객체 해석이다. 멀티미디어 객체 해석은 멀티미디어 데이타의 특성들, 즉 복잡한 시공간 구조와 다양한 클라스/객체 수준의 관계를 최대한 반영한다. 멀티미디어 객체 해석은 멀티미디어 데이타에 대한 연산을 새로운 연산자와 메소드를 이용하여 표현한다. 멀티미디어 객체 해석의 검색 결과는 시공간적으로 통합된 멀티미디어 데이타가 되며, 시공간 상에 직접 보여지게 된다.

멀티미디어 객체 해석을 이용하여 질의 요구를 표현할 수도 있지만 사용자가 사용하기에는 복잡하고 어려운 편이다. 그러므로, 멀티미디어 객체 해석은 질의의

의미(semantics)를 정형적으로 기술하기 위해 사용되고, MQL은 사용자가 용이하게 질의를 표현할 수 있도록 하기 위해 사용된다. 멀티미디어 객체 해석과 MQL의 상관 관계에 대한 정형적인 기술, 해석식을 질의어의 형태로 사상하는 규칙은 참고 문헌을 참조하기 바란다[17].

Ⅲ. MQL의 데이타 정의문

객체-지향 관점에서 보면 멀티미디어 데이타는 복잡한 구조와 상호 관계를 갖는 인스턴스 객체로 볼 수 있다. 유사한 성질을 갖는 인스턴스 객체들이 모여 클라스를 이룬다. 클라스는 소속 인스턴스 객체들의 공통 구조를 표현할 수 있지만 인스턴스 객체들간의 다양한 모든 관련성을 표현할 수는 없다. MQL의 데이타 정의문은 멀티미디어 데이타베이스를 구성하는 클라스와 클라스 관계를 정의하는 데 사용된다. 인스턴스 객체 수준에서 기술 가능한 관련성(버전이나 하이퍼미디어의 링크)들은 클라스 정의에는 반영되지 않고 인스턴스 객체에 메세지를 보내 표현하게 된다.

MQL의 클라스 정의 구문은 다음과 같다.

CREATE CLASS ⟨class-name⟩
MODE DEPENDENT |LOAN |RELATIONSHIP |VIEW
SUPER ⟨superclasses⟩
EQUIV ⟨equivalent-classes⟩
FOR ⟨participant-list⟩
ESSENTIAL ⟨essential-classes⟩
AS ⟨select-statement⟩
⟨dummy-attr-name⟩ : ⟨domain-type⟩
DESCRIPTOR ⟨descriptive-attributes⟩
METHOD ⟨instance-method-definition⟩
CATTRIBUTE ⟨class-attr-definition⟩
CMETHOD ⟨class-method-definition⟩

기본적인 도메인 타입은 문자, 숫자 데이타를 위한 Primitive 클라스(Int, Real, Char, String), 모노미디어 데이타를 위한 MultimediaPrimitive 클라스(Text, Graphic, Image, Audio)에 의해 지원된다. 이러한 클라스를 시공간적으로 조합해 사용자 정의 멀티미디어 클라스가 정의된다. 시공간적인 조합 도구로는 Meta 클라스(SpatialCompositon, Temporal Sequence, RandomSequence 등)가 이용된다. Meta 클라스는 시공간과 관련한 메소드를 제공하는 역할도 한다. Meta 클라스에 의해 다음과 같은 도메인 타입이 지원된다.

- 공간 구성(spatial composition): sc [a_1:t_1, a_2:t_2, ..., a_n:t_n]
- 공간 순차(spatial sequence): ss⟨ a_1:t_1, a_2:t_2, ..., a_n:t_n⟩
- 시간 순차(temporal sequence): ts⟨ a_1:t_1, a_2:t_2, ..., a_n:t_n
- 병행 모임(parallel collection): p [a_1:t_1, a_2:t_2, ..., a_n:t_n]
- 각 타입에 대한 임의 경우: sc{t}, sc{ t_1|t_2|...|t_n}, ss{t}, ss{ t_1|t_2|...|t_n, ts{t}, ts{t_1|t_2|...|t_n}

여기서 '[ ]'는 투플 형태, '⟨⟩'는 순차(sequence) 형태, '{}'는 집합 형태, '|'는 선택(choice) 형태로 된 도메인이 된다. a_i와 t_i는 각각 속성명과 해당 데이타 타입을 의미한다. 각 도메인 타입에 대해 붙은 'sc', 'ss', 'ts', 'p'는 태그(tag)로 해당 도메인 값으로 사용되는 객체들이 각각 공간 구성, 공간 순차, 시간 순차, 시간 동기화 관계를 갖게 됨을 의미한다. '임의 경우'란 이질적인 객체의 모임을 나타낸다.

기본 도메인 타입(앞에서 $t_i, i=1,..,n$에 해당)은 제약조건과 더불어 다음과 같이 선언된다.

[**REF**] ⟨basic-domain-type⟩ [**LKEY |UNIQUE**] [**DEP**]
IN |WITH |AT ⟨range-or-location-constraints⟩

클라스 D가 클라스 C의 속성 A의 도메인으로 선언되었다고 하자. 'REF D'는 클라스 C가 클라스 D에 대해 집단화 관계(aggregation relationship)가 아닌 연관 관계(association relationship)로 관련됨을 의미한다. 'LKEY'는 속성 A가 C의 논리키(logical key)임을 의미한다. 논리키란 한 클라스내에서 소속 객체를 유일하게 식별해주는 속성을 의미한다. 논리키의 값은 중복될 수는 있지만 널값을 가질 수는 없다. 'UNIQUE'로 명시되면 널값은 물론 중복된 값도 가질 수 없다. 'DEP'는 클라스 D가 클라스 C에 존재 종속임을 의미한다. 옵션인 IN이나 WITH-절은 속성 A가 만족시켜야할 도메인 값에 대한 제약을 지정한다. 옵션인 AT-절은 공간 상에서의 디폴트 상대 위치, 즉 공간 제약을 지정한다.

MODE-절은 클라스 유형을 구분하는 데 사용된다. 모드가 지정되지 않으면 독립적인 클라스(independent class)가 된다. 'DEPENDENT' 모드는 소속 객체가 다른 클라스의 객체에 존재 종속된 종속 클라스(dependent class), 'LOAN' 모드는 소속 객체가 다른 클라스의 객체들을 이용해 만들어지는 차용 클라스(loan class), 'RELATIONSHIP' 모드는 소속 객체가 클라스간의 연관 관계를 표현하는 접합 객체인 관계성 클라스(relationship class), 'VIEW' 모드는 소속 객체가 다른 객체상에 가상적으로 정의된 뷰 클라스(view class)임을 의미한다. 예를 들어, 독자적인 존재가 불가능한 종속 클라스 Date는 다음과 같이 정

의될 수 있다.

```
CREATE CLASS Date MODE DEPENDENT
 SUPER Object
 [ year:Int, month:Int, day:Int ]
```

차용 클라스는 기존 객체를 이용하여 새로운 시공간 통합 구조를 정의할 때 사용되는 클라스로 관계 데이타베이스의 뷰보다 발전된 개념이다. 차용 클라스의 경우, 베이스 클라스(base class)들은 FOR-절에 나열되며 본질적 베이스 클라스(essential base class)는 ESSENTIAL-절에 나열된다. 여기서 베이스 클라스는 차용 클라스에 객체를 제공하는 클라스이며, 본질적 베이스 클라스는 차용 클라스의 소속 객체의 존재를 결정하는 클라스이다. 예를 들어, Prof, Dept, Lab 클라스를 베이스 클라스로, 이 중에서 Dept 클라스를 본질적 베이스 클라스로 사용하는 차용 클라스 IntroToDept는 다음과 같이 정의할 수 있다.

```
CREATE CLASS IntroToDept MODE LOAN
 SUPER Object
 FOR Prof, Dept, Lab ESSENTIAL Dept
 p [ voiceExpl:Audio, deptIntro:DeptIntro DEP ]
```

Prof 클라스와 Project 클라스간의 다대다(n:m) 연관 관계를 표현하는 Work 클라스는 다음과 같이 정의할 수 있다.

```
CREATE CLASS Work
 MODE RELATIONSHIP SUPER Object
 FOR Prof, Project
 [ effort:Int ]
```

Person 클라스에 'age >= 20'이라는 제약을 가해 만들어지는 뷰 클라스 Adult는 다음과 같이 정의할 수 있다.

```
CREATE CLASS Adult MODE VIEW
 SUPER Person
 AS SELECT p FROM Person p
  WHERE p.age >= 20
```

옵션인 DESCRIPTOR-절은 설명 속성을 정의하는데 사용된다. 한 객체의 상태는 본질부와 기술부로 나눌 수 있다. 본질부는 시공간에 직접 표현될 본질적인 속성들, 시공간 관계, 연관 관계 등의 정보를 저장한다. 기술부는 시공간과는 직접 관련지 않은 나머지 속성들, 버젼 관계, 하이퍼 순차 관계, 종속 계수 등의 정보를 저장한다. 기술부도 정보 검색의 대상이지만, 멀티미디어 객체의 시공간적인 프레젠테이션시에는 제외가 된다. 예를 들어, 사용자가 지정한 사건(event)으로 구성되는 MyVideo 클라스는 다음과 같이 정의할 수 있다.

```
CREATE CLASS MyVideo
 events:ts{Event}
CREATE CLASS Event
 p [ frames:ts{Image}, soundTrack:Audio ]
 DESCRIPTOR (title:String LKEY)
```

그림 1은 대학의 학과를 소개하기 위한 시공간적으로 통합된 멀티미디어 데이타를 정의하는 예제 스키마를 보이고 있다. 학과 소개는 학과와 소속 연구실에 대한 간략한 소개와 향후 전망으로 이루어 진다. 여기서 DeptIntro는 IntroToDept의 도메인으로, LabIntro는 DeptIntro의 도메인으로, LabReview는 LabIntro의 도메인으로 사용되고 있다. 이 클라스들간의 관련성은 시공간적인 집단화로 볼 수 있다. 이러한 시공간 집단화 관계를 통하여 각 클라스들의 객체들이 계층적으로 결합된 구조를 복합 시공간 클라스 구조(composite temporal-spatial class structure)라고 한다. 예를 들어, 그림 1의 경우, 클라스 IntroToDept, DeptIntro, LabIntro, LabReview는 계층적으로 결합하여 클라스 IntroToDept의 복합 시공간 클라스 구조의 한 인스턴스를 만들어 낸다.

```
CREATE CLASS IntroToDept SUPER Object
  p[voiceExpl:Audio, deptIntro:DeptIntro DEP]
CREATE CLASS DeptIntro SUPER Object
  ts<deptReview:sc[deptName:String LKEY
                          AT 10@10,
                   deptHistory:Text
                          AT 30@10 150@350],
     introToLabs:ts{LabIntro DEP},
     prospect:Text>
CREATE CLASS LabIntro SUPER Object
  ts<labReview:LabReview DEP,
     labOrga:Graphic,
     projExpl:Text, labPictures:ts{Image}>
CREATE CLASS LabReview SUPER Object
  sc[labName:String LKEY AT 10@170,
     profPicture:Image AT 50@30 100@180,
     profName:String LKEY AT 130@30,
     profProfile:Text AT 50@200 150@380]
```

그림 1. 학과 소개를 위한 스키마
Fig. 1. A schema for department introduction.

그림 2는 다양한 연관 관련성과 등치 관련성을 보이는 스키마 정의를 보이고 있다. 등치 관련성은 다단계 문서 구조를 표현하는 데 사용될 수 있다. 등치 관련성은 양쪽 객체의 의미가 동등함을 표현하는 외에 동일한 데이타의 공유를 최대화하는 역할을 한다. 클라스 Prof는 클라스 Person의 서브클라스로 클라스 Paper에 대해 'writes'라는 연관 관계를 가지고 있다. 클라스 Paper는 클라스 PaperLayout과 등치 관계(equivalence relationship)를 가지고 있다.

```
// Person and Prof - spatial

CREATE CLASS Person SUPER Object
  sc[title:[name:String LKEY,
            birthDate:Date,friends:{REF Person}],
     picture:Image, profile:Text,
     audioProfile:Audio]
  METHOD (age:Int (^Date today year
                   - birthDate year.))

CREATE CLASS Prof SUPER Person
  sc[title:[name:String LKEY,birthDate:Date,
            friends:{REF Person}, prof#:Int,
            writes:{REF Paper}, affiliate:REF Dept,
            supervise:REF Lab],
     picture:Image, profile:Text,
     audioProfile:Audio]

// logical structure - not temporal-spatial

CREATE CLASS Paper SUPER Object
 EQUIV PaperLayout
 [cover:[title:String LKEY, author:s{REF Person},
         abstract:Text],
  body:s{Section}, annotation:{Text DEP}]

// layout structure - spatial

CREATE CLASS PaperLayout SUPER Object
 EQUIV Paper
 pages:ss{Page DEP}

CREATE CLASS Page SUPER Object
 MODE DEPENDENT
 page:sc{Text|Image|Graphic|Int}
```

그림 2. 연관 관련성과 등치 관련성이 포함된 스키마
Fig. 2. A Schema including association and equivalence relationships.

클라스 정의는 다음의 DROP문에 의해 제거된다.

DROP 〈user-class〉 [*]

'*'가 사용되면 지정된 클라스와 그에 속한 모든 서브클라스들이 같이 제거된다. 클라스의 삭제는 클라스 계층을 따라 상향식으로 이루어져야 한다.

Ⅳ. MQL의 데이타 조작문

이 장에서는 시공간적으로 통합된 멀티미디어 데이타에 대해 적용 가능한 MQL의 데이타 질의문의 구조를 제안하고 기존의 관계 질의어, 객체 지향 질의어에 비해 어떻게 다른 지를 설명하기로 한다. MQL은 사용자 친밀성과 구문의 단순성을 위해 그 구문의 외형은 관계 데이타베이스의 표준 질의어인 SQL과 유사하게 되어 있다. 그러나 기본적으로 그 이론적인 배경을 시공간 통합 멀티미디어 데이타를 위한 멀티미디어 객체 해석을 기반으로 하고 있으며 멀티미디어 데이타 처리를 위한 기능이 강조되어 있다.

1. 주요 특징

MQL 조작문의 특징은 다음과 같다.

- 객체 지향 질의어 기능과 멀티미디어 질의어 기능을 가지고 있다.
- SQL과 유사한 구문을 사용함으로써 사용자가 쉽게 이해하고 이용할 수 있도록 한다.
- 검색 결과가 단순히 객체 식별자가 아니고 시공간 구조를 갖는 멀티미디어 데이타이다.
- 멀티미디어 객체 해석의 복잡한 표기법을 사용자가 알 필요가 없도록 한다.

MQL은 객체 지향 질의어를 통해 얻을 수 있는 정보 외에 시공간 멀티미디어 데이타에 고유한 메소드를 질의어 내에서 사용하여 보다 유용한 정보들을 얻을 수 있도록 한다. 멀티미디어 데이타를 구성하기 위한 태그된 집단화별로 각 시스템 클라스에서 그와 관련한 메소드들이 지원된다.

SQL과 유사한 구문을 사용하는 이유는 현재 SQL이 관계 데이타베이스의 표준 데이타 언어임은 물론 최신 객체 지향, 또는 관계/객체-지향 DBMS에서도 거의 예외없이 채택되고 있기 때문이다. 멀티미디어 데이타베이스를 위한 표준으로 제정을 추진 중인 언어(SQL/MM)도 기본적인 형태로 SQL을 취하고 있다.

현재 복잡한 객체 검색을 지원하는 객체 지향 DBMS의 데이타 언어를 보면, 그 검색 결과가 주로 객체 식별자(의 리스트)이고, 세부 정보는 그 객체 식별자를 통해 항해 방식으로 찾아가도록 하고 있다. 예를 들어, SQL/X[4]의 경우 검색 결과가 '클라스이름: 번호' 형태의 식별자를 사용하고 있다. 이해 반해,

MQL의 기본적인 검색 결과는 시공간적으로 통합된 (시공간적으로 직접 프레젠테이션되는) 멀티미디어 데이타이고, 사용자가 지정하는 경우만 객체 식별자 형태의 검색 결과를 제공하게 된다.

MQL은 멀티미디어 객체 해석으로 표현 가능한 질의를 사용자에게 편리한 형태로 기술될 수 있도록 고안한 것이다. 멀티미디어 객체 해석으로 표현된 사용자의 질의 요구는 쉽게 MQL로 표현될 수 있다. 멀티미디어 객체 해석은 사용자의 질의 요구를 명확하게 표현하기 위해 사용되는 정형적인 도구이지만, 이것을 DBMS가 내부적인 질의어 처리 과정에서 사용할 수도 있다. 이러한 경우에도 역으로 MQL을 쉽게 멀티미디어 객체 해석으로 표현할 수 있다. 그림 3은 'ALPHA project'를 수행하는 연구실이 속한 학과를 검색하라는 질의를 멀티미디어 객체 해석과 MQL로 각각 표현하고 있다.

```
(IntroToDept/p) p
(∃i(p.deptIntro.introToLabs/i
  (i.projExpl contains 'ALPHA project')))

SELECT *
FROM IntroToDept
WHERE deptIntro.introToLabs
  (projExpl CONTAINS 'ALPHA project')
```

그림 3. 멀티미디어 객체 해석과 MQL로 표현한 질의어

Fig. 3. A query represented in multimedia object calculus and MQL.

2. 검색 질의문의 구성 요소

1) 검색 질의문의 형태

객체간의 참조 관계를 표현하기 위해서는 객체의 속성의 값으로 객체 식별자를 치환하여야 한다. 시스템이 내부적으로 사용하는 객체 식별자 대신 MQL에서는 OSQL[6]과 같이 ':'로 시작하는 세션 변수(session variable)를 사용한다. 세션이란 한 번에 실행되는 작업 단위로 보통 하나 이상의 질의어로 구성된다. 검색을 위한 구문은 다음과 같다.

```
[〈session-variable〉 := ]
SELECT [ + ] 〈target-list〉 [ASOID]
FROM 〈range-expression〉
WHERE 〈WFF〉
```

검색문을 BNF로 표현하면 다음과 같다.

```
〈select-statement〉 ::= [〈session-variable〉 := ]
  〈select-clause〉
  〈from-clause〉
  [〈where-clause〉]
〈session-variable〉 ::= :〈variable-name〉

〈select-clause〉 ::= SELECT [ + ] 〈select-specification〉 [ASOID]
〈select-specification〉 ::= * | 〈select-result-type〉
〈select-result-type〉 ::= 〈select-item-list〉
  | 〈structured-select-items〉
〈structured-select-items〉 ::= [〈select-item-list〉]
  | {〈structured-select-items〉}
  | (〈select-item-list〉)
  | 〈tag〉(〈structured-select-items〉)
〈select-item〉 ::= [〈new-attribute-name〉:] 〈path-expression〉
  | [〈new-attribute-name〉:] 〈structured-select-items〉

〈from-clause〉 ::= FROM 〈from-path-expression-list〉
〈from-path-expression〉 ::= 〈from-class-item〉 | 〈from-class-item〉.〈from-method-path-expression〉
〈from-class-item〉 ::= 〈from-class-item-all〉
  | 〈from-class-item-each〉
〈from-class-item-all〉 ::= 〈from-class-target〉 [〈object-variable〉]
〈from-class-target〉 ::= 〈class-names〉
  | 〈class-names〉 〈set-operator〉 〈from-class-target〉
  | 〈class-names〉, 〈from-class-target〉
  | (〈from-target〉)
〈class-names〉 ::= 〈class-name〉
  | 〈class-name〉*
  | 〈class-name〉#
〈set-operator〉 ::= UNION | INTERSECTION | DIFFERENCE
〈from-class-item-each〉 ::= 〈class-object-variable-pair〉
  | 〈class-object-variable-pair〉 〈set-operator〉 〈from-class-item-each〉
  | 〈class-object-variable-pair〉, 〈from-class-item-each〉
〈class-object-variable-pair〉 ::= [(]〈class-names〉〈object-variable〉[)]

〈from-method-path-expression〉 ::= 〈from-method-item〉
  |〈from-method-item〉.〈from-method-path-expression〉
〈from-method-item〉 ::= 〈scalar-method-name〉
  | 〈scalar-method-name〉+ [〈object-variable〉]
  | 〈set-method-name〉 [〈object-variable〉]

〈where-clause〉 ::= WHERE 〈predicate〉
〈predicate〉 ::= 〈atomic-formula〉
  | 〈atomic-formula〉 AND 〈predicate〉
  | 〈atomic-formula〉 OR 〈predicate〉
  | NOT 〈predicate〉
  | 〈quantifier〉(〈predicate〉)
  | 〈quantifier〉(〈select-statement〉)
〈quantifier〉 ::= ANY | ALL
```

'SELECT+'는 주어진 조건을 만족하는 오직 하나의 결과 객체만을 보고 싶은 경우 사용한다. 검색된 멀티미디어 정보는 시공간 상에 직접 보여 진다. 참조 관련성을 기록하기 위해 객체의 식별자가 필요한 경우에는 'ASOID' 옵션을 사용한다.

2) 객체 변수

클라스의 한 멤버 객체, 또는 객체 집합의 한 원소 객체를 나타내는 변수를 객체 변수(object variable)라고 한다. 예를 들어, 'Prof p'는 p가 Prof 클라스의 한 인스턴스를 나타내는 객체 변수임을 의미한다. 혼동이 없을 경우는 클라스 이름을 객체 변수로 대용할 수 있다.

객체 변수가 정의되는 클라스가 복합 시공간 클라스 구조를 갖는 멀티미디어 데이타인 경우는 객체 변수가 복합 시공간 클라스 구조의 한 인스턴스를 나타낸다. 예를 들어, 'IntroToDept p'는 p가 IntroToDept 클라스를 루트로 하는 복합 시공간 클라스 구조의 한 인스턴스를 나타냄을 의미한다. 모든 학과 소개를 검색하라는 질의는 다음과 같이 간단히 표현할 수 있다.

SELECT p FROM IntroToDept p

객체 변수 p는 IntroToDept 클라스의 복합 시공간 클라스 구조의 한 인스턴스로 오디오 객체와 계층적으로 시간 동기화되는 공간 객체의 시간 순차 객체가 된다. 검색 목표가 클라스의 복합 시공간 클라스 구조의 한 인스턴스를 의미하는 객체 변수일 경우, '*'로 대신할 수 있다. 예를 들어, 앞의 질의는 다음과 같이 표현해도 동일하다.

SELECT * FROM IntroToDept

3) 메세지 경로식

임의의 객체 또는 클라스 O에 대해 연속적으로 메세지가 적용된 것을 'O.MSG1.MSG2. .. . MSGn'으로 표기하며 메세지 경로식(message path expression)이라고 한다. 여기서 속성 이름은 그 자체가 메소드를 표시하는 속성 메소드(attribute method)이다. 예를 들어, 'IntroToDept p'가 정의되었다면,

p.deptIntro.deptReview.deptName

는 메세지 경로식을 나타낸다. 여기서, p는 객체 변수, deptIntro, deptReview, deptName은 검색 속성 메소드이다.

객체 변수 p가 지정하는 속성 이름 A가 연관 복합 시공간 클라스 구조(복합 시공간 클라스 구조에 연관 관련성에 의해 관련된 객체도 추가한 구조)내에서 유일하면 p로 시작하는 속성 메소드로만 구성된 경로식은 중간의 속성 메소드를 생략하고 'p.*.A' 형태로 표기할 수 있다. 객체 변수도 생략될 수 있는 경우면 '*.A'로 표기한다. 검색 목표가 경로식 형태일 경우 객체 변수가 없어도 목표의 의미가 명확하면 객체 변수를 생략할 수 있다. 예를 들어, 앞의 메세지 경로식은

p.*.deptName

으로 대신할 수 있다.

'MSGi q' 형태의 메세지를 포함한 메세지 경로식을 한정 메세지 경로식(quantified message path expression)이라고 한다. 이 경우 변수 q는 MSGi 까지를 적용한 결과 객체 집합의 한 인스턴스를 나타내는 속박 변수(bound variable)이다. 한정 메세지 경로식에서 다른 곳에서 쓰이지 않는 속박 변수는 생략될 수 있다. 'CE Dept.'의 'DB Lab.' 정보를 보이라는 질의는 FROM-절에 한정 메세지 경로식을 사용하여 다음과 같이 표현할 수 있다.

```
SELECT i
FROM IntroToDept p.deptIntro.introToLabs i
WHERE p.*.deptName = 'CE Dept.'
        AND i.*.labName = 'DB Lab.'
```

4) 연산자

연산자에는 클라스 연산자, 객체 연산자, 비교를 위한 θ 연산자, 논리 연산자가 있다. 클라스 연산자로는 UNION, INTERSECTION, DIFFERENCE가 있다. 논리 연산자에는 AND, OR, NOT이 있다.

객체 연산자는 다음과 같다. A, B를 객체 변수라 할 때 '〈A,B〉'는 접합 객체를 의미한다. B가 집합 객체이면 객체 A에 객체 B의 원소 객체들이 접합된다. 예를 들어, 교수 'S.Lee'의 정보와 연구 논문들의 첫 페이지를 검색하라는 질의는 접합 연산자를 이용하여 다음과 같이 표현된다.

```
SELECT 〈p, p.writes.pages[1]〉
FROM Prof p
WHERE p.name = 'S.Lee'
```

위 질의에서 'p.writes.pages[1]'는 멀티미디어 객체 해석으로 표현하면 'p.writes.EQUIV.pages.AT(1)'가 된다. EQUIV는 등치 관계에 있는 객체를, AT(1)은 순차 객체의 첫번째 원소 객체를 복귀시키는 메소드이다. MQL에서는 검색 목표가 경로식 형태일 경우 등치 관계에 있는 객체의 검색 메소드 EQUIV를 생략할 수 있다(FROM-절, WHERE-절에서도 동일). 'AT(i)' 메소드는 '[i]' 형태로 표기할 수 있다. 그러므로, 'p.writes'는 교수 p에 의해 작성된 논문 객체들의 집합을 복귀시키고, 'p.writes.EQUIV'는 이 논문들의 레이아웃 객체들의 집합을 복귀시키고, 최종적으로 AT(1) 메소드가 이 논문 레이아웃 객체들의 첫번째 원소, 즉 첫번째 페이지들을 복귀시키게 된다.

Y를 순차 객체, a와 b를 양의 정수, c와 d를 시간을 나타내는 수치라 하면, 'Y[a:b]'는 순서에 의한 프로젝션을, 'Y[c:d]'는 지속 시간에 의한 프로젝션을 나타낸다. 예를 들어, 'K.Hong'이 첫번째 저자인 논문의 3-5 페이지를 검색하라는 질의는 순서에 의한 프로젝션을 이용해 다음과 같이 표현할 수 있다.

```
SELECT p.pages[3:5]
FROM Paper p
WHERE p.author[1].name = 'K.Hong'
```

지속 시간에 의한 프로젝션의 예로, 'CE Dept.' 소개의 20초부터 40초까지의 내용을 보이라는 질의는 다음과 같다.

```
SELECT p [ 20sec:40sec ]
FROM IntroToDept p
WHERE p.*.deptName = 'CE Dept'
```

비교를 위한 θ 연산자는 다음과 같다. 객체와 객체의 비교에는 IDEQ, VEQ, DVEQ, REQ 연산자가 사용된다. 숫자와 숫자의 비교에는 일반적인 6가지 비교 연산자(=,〈〉,〉,〉=,〈,〈=)가 사용된다. 문자열과 문자열의 경우도 마찬가지이다. 텍스트에 대한 문자열의 포함 여부, 오디오에 대한 다른 오디오의 포함 여부 확인에는 CONTAINS 연산자가 사용된다. 예를 들어, 제목이 'Object-Relationship Model'인 논문중 'time synch'라는 키워드를 포함한 페이지들만 검색하라는 질의는 텍스트에 대한 문자열의 포함 여부를 확인하는 연산자를 이용하여 다음과 같이 표현할 수 있다.

```
SELECT g
FROM Paper.pages g.TEXT t
WHERE title = 'Object Relationship Model'
        AND t CONTAINS 'time synch'
```

객체 집합에 특정 클라스의 객체가 포함되어 있는지를 확인하는 데도 CONTAINS 연산자가 사용된다. 일반적인 형태는 '{Object} CONTAINS class_name'이다. 예를 들어, 제목이 'Object-Relationship Model'인 논문중 그림이 있는 페이지들만 검색하라는 질의는 논문 레이아웃의 각 페이지 객체(공간 구성 객체)에 대해 GRAPHIC 클라스에 속한 멤버 객체의 존재 여부를 묻는 CONTAINS 연산자를 이용하여 다음과 같이 표현할 수 있다.

```
SELECT g
FROM Paper.pages g
WHERE title = 'Object Relationship Model'
        AND g CONTAINS GRAPHIC
```

객체의 집합 객체에 대한 포함 여부 확인에 IN 연산자, 집합간의 비교에 =, 〈〉, SUBSET, SUBSETEQ, SUPERSET, SUPERSETEQ의 6가지 연산자가 사용된다.

5) 메소드

메세지 경로식에는 다양한 메소드가 포함될 수 있다. 멀티미디어 데이타 연산에 필수적인 메소드들은 시스템 클라스에서 정의 지원되고, 이 외에 사용자가 정의한 메소드를 추가적으로 사용할 수 있다. 예를 들어, 최상위 시스템 클라스 Object의 경우 EQUIV, SYNCH, DURATION 등의 메소드를, RandomSequence 시스템 클라스의 경우 COUNT, TEXT, GRAPHIC, IMAGE, AUDIO 등의 메소드를, Sequence 시스템 클라스의 경우 AT(index), LAST 등의 메소드를 지원한다. RandomSequence 클라스에서 GRAPHIC 메소드는 원소 객체 중에 GRAPHIC 시스템 클라스에 속한 인스턴스 객체들을 복귀시키는 메소드이다. 각 메소드의 세부적인 구현 사항은 본 논문에서는 생략하기로 한다.

DURATION은 시간상에서 지속 시간(duration)을 지니는 객체의 지속 시간을 구하는 메소드이다. 지속 시간은 단일 객체의 경우 그 객체에서 바로 구할 수 있고, 시공간 집단화에 의해 통합된 객체의 경우 집단화에 참가한 객체들의 지속 시간을 계층적으로 더해 구하게 된다. 예를 들어, 각 학과 소개의 소요 시간을 구하라는 질의는 다음과 같다.

```
SELECT DURATION FROM IntroToDept
```

이 질의에서, 'p.DURATION'은 IntroToDept 클라스의 복합 시공간 클라스 구조 인스턴스내의 지속 시간 속성값들을 계층적으로 (상향식으로) 더해서 구해진다.

두번째 저자가 'S.Lee'인 논문의 초록을 검색하라는 질의는 순차 객체의 특정 위치의 객체를 복귀하는 AT(index) 메소드를 이용하여 다음과 같이 표현할 수 있다.

```
SELECT abstract
FROM Paper
WHERE author [ 2 ].name = 'S.Lee'
```

여기서 'author'는 순차 객체가 되고, 'author [2]'는 그 순차 객체의 2번째 원소가 된다.

집합 성질을 갖는 객체의 경우 일반 데이타베이스처럼 COUNT, SUM, AVG, MAX, MIN 메소드를 이용해 관련 정보를 얻게 된다.

검색 질의어는 특정 클라스에 속한 소속 인스턴스 객체를 처리 대상으로 한다. 클라스 구조에 대한 정보는 클라스를 수신자(receiver)로 하는 메소드를 이용해 구할 수 있다. 예를 들어, 특정 클라스의 수퍼 클라스는 SUPER 메소드를 보내, 특정 클라스의 소속 객체의 수는 COUNT 메소드를 보내 얻을 수 있다. 클라

스 객체 자체가 처리 대상일 때는 예외적으로 클라스(이름)를 수신자로 하는 메세지를 통하여 검색 요구를 표현하게 된다. 예를 들어, 클라스 Prof의 수퍼클라스를 구하라는 질의는 다음과 같다.

```
Prof.SUPERCLASS
```

클라스 Prof의 소속 객체의 수를 구하라는 질의는 다음과 같다.

```
Prof.COUNT
```

3. 변경 질의

1) 계층적으로 통합된 멀티미디어 데이타의 삽입

계층적으로 통합된 멀티미디어 데이타에 대한 삽입은 상향식으로 수행된다. 즉, 복합 시공간 클라스 구조의 하위 객체(보통은 모노미디어 객체)를 먼저 삽입한 후, 단계적으로 상위 계층의 객체를 삽입하게 된다. 멀티미디어 프리미티브 객체, 즉 모노미디어 객체에 대한 삽입문은 이미지 스캐너/편집기, 그래픽 편집기 등 해당 모노미디어 편집 도구를 호출하게 된다. 모노미디어 데이타를 생성하는 삽입문의 구조는 다음과 같다.

```
INSERT ⟨multimedia-primitive-class⟩
⟨session-variable⟩
```

일반적인 사용자 정의 클라스 객체에 대한 삽입문의 구조는 다음과 같다.

```
INSERT INTO ⟨user-class⟩
(⟨target-structures⟩) ⟨session-variable⟩
VALUES (⟨structured-values⟩)
```

예를 들어, IntroToDept 클라스에 'CE Dept.'를 소개하는 멀티미디어 데이타를 삽입하는 경우를 고려해 보자. 'CE Dept.'에는 두 개의 연구실 'DB Lab.'과 'PL Lab.'이 속해 있고, 각 연구실은 'S.Lee'와 'Y.Kim'이 관리한다고 가정한다.

```
1단계: LabReview 소속 객체 삽입
  INSERT Image :slpict
  INSERT Text :sltext

  INSERT INTO LabReview() :db
  VALUES (sc['DB Lab', :slpict, 'S.Lee',
      :sltext])
2단계: LabIntro 소속 객체 삽입
  INSERT Graphic :dborga
  INSERT Text :dbproj
  INSERT Image :dbpict1
  INSERT Image :dbpict2
  ...

  INSERT INTO LabIntro() :dbintro
  VALUES (ts⟨:db, :dborga, :dbproj,
      ts{:dbpict1, :dbpict2, ..}⟩)
3단계: DeptIntro 소속 객체 삽입
  INSERT Text :cehist
  INSERT Text :cepros

  INSERT INTO DeptIntro() :ce
  VALUES (ts⟨sc['CE Dept.',:cehist],
      ts{:dbintro,:plintro},:cepros⟩)
4단계: IntroToDept 소속 객체 삽입
  INSERT INTO IntroToDept () :ceintro
  VALUES (p[(INSERT Audio :cevoice),:ce])
```

단계 4에서 ':cevoice'가 지시하는 오디오 객체는 ':ce'의 계층적 구조에 따라 여러 개의 서브객체(subobject)로 분해된다. 사용자는 계층적으로 분해된 ':cevoice' 객체의 단말 노드 수준에서 대화식으로 오디오 서브객체를 삽입(즉, 녹음)하게 된다.

2) 삭제

삭제문의 구조는 다음과 같다.

```
DELETE ⟨target⟩ WHERE ⟨WFF⟩
```

예를 들어, Prof 클라스에서 이름이 'S.Lee'인 교수를 삭제하는 질의는 다음과 같다.

```
DELETE Prof WHERE name = 'S.Lee'
```

특정 객체 삭제시 그에 존재 종속된 객체들도 연쇄 삭제된다. 예를 들어, IntroToDept에 속한 특정 객체를 삭제하면 그에 존재 종속된 DeptIntro, LabIntro, LabReview 객체들도 연쇄 삭제된다. IntroToDept 클라스에서 'CE Dept.' 정보를 삭제하라는 질의는 다음과 같다.

```
DELETE IntroToDept
WHERE *.deptName = 'CE Dept.'
```

3) 갱신

갱신문의 구조는 다음과 같다.

```
UPDATE 〈range-expression〉
SET 〈update-path-expression〉
WHERE 〈WFF〉
```

멀티미디어 객체의 일부 미디어 데이타를 변경하는 경우는 모노미디어 INSERT문으로 새로운 미디어 데이타를 생성한 후 사용자 멀티미디어 데이타에 대해 UPDATE문을 사용할 수도 있고, UPDATE문 내부에 모노미디어 INSERT문을 삽입하여 사용할 수도 있다. 예를 들어 'CE Dept.' 소개 데이타 중 'DB Lab.'을 소개하는 음성을 변경하는 질의는 다음과 같다.

```
UPDATE IntroToDept p.deptIntro.introToLabs i
SET i.SYNCH(INSERT Audio :newvoice)
WHERE p.*.deptName='CE Dept.'
AND i.*.labName='DB Lab.'
```

V. 결 론

본 논문에서는 객체 지향 멀티미디어 데이타베이스에서 일반 사용자가 사용할 수 있는 고급 멀티미디어 질의어 MQL을 제안하였다. 이 MQL이 기여하는 바는 다음과 같이 요약할 수 있다.

- 멀티미디어 데이타간의 다양한 관계(시공간 관련성, 다단계 구조를 위한 등치 관련성 등)를 이용한 정보 검색을 가능하게 한다. (여기서 다단계 구조란 멀티미디어 문서 모델의 개념 구조, 논리 구조, 레이아웃 구조를 의미한다.)
- 기본적으로 객체 지향 질의어의 기능을 가지고 있으므로 객체 지향 DBMS의 질의어로도 활용이 가능하다.
- 멀티미디어 데이타를 위한 SQL의 표준안으로 현재 연구 중인 SQL/MM 표준안 설정 자료로 활용할 수 있다.

일반 사용자는 MQL을 사용하여 멀티미디어 데이타에 대한 검색, 삽입, 삭제, 갱신 연산을 표현할 수 있다. MQL의 데이타 정의문은 멀티미디어 데이타 모델인 객체-관계 모델에서 지원하는 클라스와 클라스 간의 관계를 정의하는 기능을 갖는다. MQL의 데이타 조작문은 다양한 새로운 메소드가 포함된 메세지 경로식과 연산자를 이용하여 시공간적으로 통합된 멀티미디어 데이타와 다양한 객체간의 관련성을 통해 유용한 정보 검색을 가능하게 한다.

MQL의 구문은 사용자 친밀성과 구문의 단순성을 위해 그 외형은 관계 데이타베이스의 표준 질의어인 SQL과 유사하게 되어 있다. 그러나 기본적으로 그 이론적인 배경을 시공간 통합 멀티미디어 데이타를 위한 멀티미디어 객체 해석을 기반으로 하고 있으며 멀티미디어 데이타 처리를 위한 기능이 강조되어 있다. 즉, MQL은 멀티미디어 데이타의 시공간 구조와 다단계 구조 등을 이용한 정보 검색을 지원해 단순히 모노미디어 입력/압축/출력 메소드 만을 지원하는 객체 지향 데이타베이스의 멀티미디어 확장 시스템에서 가능한 것보다 다양한 정보를 제공하게 된다. 또한 MQL의 검색 결과는 단순한 객체 식별자가 아니고 시공간 구조를 갖는 멀티미디어 데이타가 된다.

MQL은 멀티미디어 질의어 기능 외에 기본적인 객체 지향 질의 기능도 가지고 있으므로 객체 지향 데이타베이스의 질의어로도 활용할 수 있다. 또한, 기존의 상용 DBMS에 멀티미디어 지원 기능을 추가하기 위해 활용될 수도 있고, 멀티미디어 데이타베이스 시스템을 새롭게 고안해 구현하는 데도 활용할 수 있다. 이러한 경우의 하나로 현재 MQL은 ALPHA 멀티미디어 DBMS[15]을 구축하기 위한 기반 질의어로 사용되고 있다. ALPHA 시스템은 ORM을 기반 모델로, MQL을 사용자 질의어로 사용하는, 사용자 인터페이스부터 저장 시스템까지 완전히 새로 설계하여 구현하고 있는 DBMS이다.

ISO/IEC JTC1에서는 SQL 3을 주로 참고하여 멀티미디어 데이타 지원이 가능한 SQL/MM(SQL/MultiMedia)을 전문텍스트, 그래픽, 정지 이미지 등 10여개의 부분으로 구분하여 표준안을 제정중이다. SQL/MM은 객체 지향 기능이 보강된 일반적인 관계 데이타 모델을 대상으로 하고 있고, MQL은 ORM 멀티미디어 모델을 대상으로 하고 있어 적지 않은 차이를 보일 것으로 예상된다. SQL/MM 표준안이 확정되는데로 SQL/MM과 MQL을 비교 평가하고 보완하는 작업이 수행되어야 할 것으로 보인다.

참 고 문 헌

[1] Bertino, E., Negri, M., Pelagatti, G., and Sbattella, L., "Object-Oriented Query Languages: The Notion and the Issues," IEEE TKDE, Vol.4, No.3, pp.223-237, June 1992.

[2] Christodoulakis, S., Ho, F., and Theodoridou, M., "The Multimedia Object Presentation Manager of MINOS: A

Symmetric Approach," in Proc. ACM SIGMOD, pp.295-310, 1986.

[3] Keim, D.A., Kim, K-C, and Lum, V., "A Friendly and Intelligent Approach to Data Retrieval in a Multimedia DBMS," in Proc. DEXA, pp.102-111, 1991.

[4] Kim, W., "Introduction to SQL/X," in Proc. the 2nd Far-East Workshop on Future Database Systems, pp.2-7, 1992.

[5] Klas, W., Neuhold, E.J., and Schrefl, M., "Using an Object-Oriented Approach to Model Multimedia Data," Computer Commnunications, Butterworth, Vol.13, No.4, pp.204-216, 1990.

[6] Lyngbaek, P., "OSQL: A Language for Object Databases," HPL-DTD-91-4, Hewlett-Packard, Jan. 1991.

[7] Masunaga, Y., "An Object-Oriented Approach to Multimedia Database Organization and Management," in Proc. Int. Sympo. on Database Systems for Advanced Applications, pp.190-200, 1989.

[8] Meghini, C., Rabitti, F., and Thanos, C., "Conceptual Modeling of Multimedia Documents," IEEE Computer, Vol.24, No.10, pp.23-30, 1991.

[9] Mylopoulos, J., Bernstein, P.A., and Wong, H.K.T., "A Language Facility for Designing Database-Intensive Applications," in ACM TODS, Vol.5, No.2, pp.185-207, June 1980.

[10] Nah, Y. and Lee, S., "Two-level Modeling Schemes for Temporal-Spatial Multimedia Data Representation," in Proc. DEXA, pp.102-107, 1992.

[11] Nah, Y., Lee, S., and Hwang, S., "Multimedia Composite Object having Temporal-Spatial Structures," in Proc. Int'l Computer Science Conference (ICSC) Data and Knowledge Engineering: Theory and Applications, pp.314-320, 1992.

[12] Nah, Y., Lee, S., Park, Y., and Hwang, S., "A Representation of Operations on Temporal-Spatial Multimedia Data," in Proc. Int'l Sympo. on Next Generation Database Systems and Their Applications, pp.241-248, 1993.

[13] Shipman, D.W., "The Functional Data Model and the Data Language DAPLEX," ACM TODS, Vol.6, No.1, pp.140-173, Mar. 1981.

[14] Woelk, D., Luther, W., and Kim, W., "Multimedia Applications and Database Requirements," in Proc. IEEE-CS Office Automation Sympo., pp.180-189, 1987.

[15] 이석호 외, "멀티미디어 DBMS ALPHA의 설계 및 구현", 한국정보과학회 논문지, 21(7), pp.1181-1188, 1994년 7월.

[16] Little, T.D.C. and Ghafoor, A., "Interval-Based Conceptual Models for Time-Dependent Multimedia Data," IEEE Trans. on Knowledge and Data Eng., Vol.5, No.4, pp.551-563, August 1993.

[17] 나연묵, 멀티미디어 응용을 위한 객체-관계 모델에 관한 연구, 서울대학교 대학원 컴퓨터공학과 박사 학위 논문, 1993년 2월

저 자 소 개

羅 然 默(正會員)

1964년 1월 19일생. 1986년 2월 서울대학교 전자계산기공학과(공학사). 1988년 2월 서울대학교 전자계산기공학과(공학석사). 1993년 2월 서울대학교 컴퓨터공학과(공학박사). 1991년 2월 ~ 1991년 8월 IBM Watson 연구소에서 비즈니스 프로세스 모델링에 관해 연구함. 1993년 8월 ~ 현재 단국대학교 컴퓨터공학과 전임강사로 재직중. 관심분야는 데이타베이스, 멀티미디어 시스템, 객체지향 시스템, 데이타 모델링 등임.

金 奎 喆(正會員)

1956년 1월 12일생. 1978년 2월 서울대학교 물리학과(이학사). 1980년 2월 서울대학교 물리학과(이학석사). 1986년 5월 U. Wisconsin at Madison 전기및컴퓨터공학과(M.S.). 1992년 5월 U. Wisconsin at Madison 전기및컴퓨터공학과(Ph.D.). 1993년 3월 ~ 1993년 8월 삼성반도체 선임연구원. 1993년 8월 ~ 현재 단국대학교 컴퓨터공학과 전임강사로 재직중. 관심분야는 VLSI 설계 및 테스팅, 자동 테스트 패턴 생성, Fault Diagnosis 등임.

李 錫 浩(正會員) 第 30 卷 B編 第 4 號 參照

현재 서울대학교 컴퓨터공학과 교수.