고딕 GIS 도구를 이용한 시공간 객체의 표현과 이력관리

(Representation and History Management of Spatio-Temporal Objects using a Gothic GIS Tool)

백 주 연 † 이 성 종 †† 류 근 호 †††

(Ju Yeon Paik) (Seong Jong Lee) (Keun Ho Ryu)

요 약 지리정보시스템 내에서 공간 객체는 시간의 흐름에 따라 속성 정보가 변화하거나 위치, 또는 객체 상호간의 위상 관계가 변화할 수 있다. 그러나 기존의 지리정보시스템은 시간에 따라 변경되는 비공간 속성 정보 및 공간 속성 정보를 삭제하고 현재 시점에서 유효한 값으로 대체함으로써 변경된 정보의 이력을 관리할 수 없고 시간질의를 비롯한 시공간 질의를 제공할 수 없는 문제점이 있다.

이 논문에서는 이러한 문제점을 해결하기 위하여 시간에 따라 변화하는 공간 객체의 이력을 관리하기 위한 시공간 객체 모델을 제안한다. 제안된 모델은 지리정보 시스템 도구인 고딕을 이용하여 시공간 클래스로 설계 구현했으며, 객체의 이력관리를 위해서는 객체 이력을 단일 개체내에 포함하여 관리하는 방법을 사용하였다. 그리고, 이력 질의 및 시공간 질의 기능을 수행할 수 있도록 시간 연산자, 시공간 연산자, 시공간 질의 연산 등을 고딕에 추가하였다.

Abstract In Geographic Information System, spatial object can be changed in the attribute information, spatial location and the topological relation between them with the change of time. However, traditional GIS deletes the old value of aspatial and spatial information and replaces them with new value. Therefore, it is difficult to manage the history of changed spatial object and can not support the spatio-temporal queries including temporal queries.

In this paper, we propose a spatio-temporal object oriented model to solve this problem. We implement the proposed model with spatio-temporal class using Gothic GIS tool. The historical information of an object is stored into the object itself for the effective history management. And, in order to provide the queries for the history of an object and spatio-temporal relationship, we add temporal operators, spatio-temporal operators, and spatio-temporal query operations into Gothic, and improve the facility of the Gothic.

1. 서 론

전통적인 공간 데이타베이스의 대표적인 응용인 지리정보시스템은 지리상의 공간 정보 및 공간 객체에 포함된 비공간 데이타를 표현하고 공간에 따른 자료 검색 및 갱신이 가능하다. 지리 정보 시스템 내에서 공간 객체는 시간에 따라 위치가 변화하거나 공간 객체 상호간의 위상 관계가 변화할 수 있으며 토지 구역의 경우 시간의 흐름에 따라 소유주가 바뀌거나 그 토지 모양이 변할 수 있다[1]. 그러나 기존 지리정보시스템은 시간에 따라 변경되는 비공간 속성 정보 및 공간 속성 정보를 삭제하고 새로운 정보값 만을 관리함으로써 변경된 정보의 이력을 관리할 수 없고 이력 질의 및 시공간 질의를 제공할 수 없는 문제점이 있다[2-10].

이 논문에서는 기존의 시공간데이타베이스 및 시간지

· 이 연구는 과학기술처 국가지리정보시스템 기술개발 사업의 연구비 지원과 한국과학재단의 99년도 특정기초연구의 일부로 수행되었음

† 비 회 원 : LG-EDS 시스템 연구원
jypaik@lgeds.lg.co.kr
†† 비 회 원 : 충북대학교 전자계산학과
sjlee@dblab.chungbuk.ac.kr
††† 종신회원 : 충북대학교 컴퓨터과학과 교수
khryu@dblab.chungbuk.ac.kr

논문접수 : 1997년 12월 23일
심사완료 : 2000년 2월 1일

리정보시스템을 중심으로한 관련연구를 통하여 기존 모델의 특성 및 장단점을 비교 서술하고, 객체 단위의 이력관리를 위한 이원시간[11]을 지원하는 시공간 모델을 제안한다. 제안한 시공간 모델은 시간에 따라 변화하는 공간 상태를 표현하기 위하여 시공간 객체 지향 모델[12,13]을 기반으로 하되 객체가 변경될때마다 객체 식별자를 할당하지 않고 변화 이력자체를 객체가 포함하도록 하여 이력 질의시 단 하나의 객체 식별자만으로 객체의 모든 이력을 검색할 수 있도록 한다.

제안한 모델은 객체지향 개념을 지원하는 지리정보시스템 도구인 고딕(Gothic)[14,15]을 이용하여 구현 검증하며, 객체 자신의 변화이력을 포함하는 시공간 클래스와 그들간에 존재하는 관계성에 기반한 질의를 위한 시간, 공간 및 시공간 연산자를 가진다. 제안된 모델을 고딕을 이용하여 구현함으로써 기존의 지리 정보 시스템이 가지는 공간 정보의 처리기능 이외에 시간에 따른 공간 객체의 변화정보를 효율적으로 처리함과 동시에, 시간질의 기능, 객체 이력질의, 그리고 시간에 따른 공간 데이타 분석기능을 제공할 수 있다.

논문 내용의 효율적인 전개를 위하여 다음과 같이 구성하였다. 먼저 2장에서는 문헌을 통한 관련 연구를 시공간데이타 모델별 특징과 질의 기능 등에 따라 비교하고, 이 논문에서 제안한 모델을 구현하기 위한 객체지향 지리정보시스템 도구인 고딕을 객체모델, 버전관리기법, 질의 유형을 중심으로 기술한다. 3장에서는 고딕에 적합한 객체별 이력을 포함하는 시공간 클래스의 계층과 시공간 객체를 정의하며 시공간 데이타모델을 제안하며, 이를 바탕으로한 시공간 객체의 이력 관리를 위한 알고리즘의 설계 및 구현을 4장에서 설명한다. 5장에서는 이력지원 시간 및 시공간 연산을 정의하고, 정의된 연산자의 설계 및 구현에 관한 사항을 정리한다. 마지막으로 6장에서는 논문의 요약 및 앞으로의 연구 사항을 기술한다.

2. 관련연구

2.1 시공간 데이타모델

기존의 시공간 데이타모델에 대한 기존의 관련문헌[7,10,13,16,17,18]을 보면 시공간 데이타모델에는 아직까지 특별히 우세한 방법이 존재하지 않으며 각기 다른 모델과 구별되는 독창성을 가진다[19,22]. 따라서 모델마다 접근방식에 따른 특성과 장단점, 지원 가능한 질의의 종류가 모델마다 서로 다르며 이는 다음과 같이 요약할 수 있다.

스냅샷 모델(snapshot model)은 가장 단순한 모델로서 시간 흐름에 따른 현실 세계의 각각의 상태를 시간 스탬프를 가지는 개별적인 레이어에 의해 공간 데이타 모델내에 표현하였다[6,7,16]. 각각의 레이어는 하나의 주제에 대하여 시간적으로 동일한 단위의 집합으로 간주되며, 직관적이고 단순한 반면, 두 개의 스냅샷 레이어상에 동일 자료가 중복되어 표현되고 시간상의 변화가 명확하지 않다[6, 9]. 시공간 복합 데이타 모델(space-time composite data model)은 다수의 스냅샷 레이어를 하나의 시공간 복합 레이어에 중첩시키는 방법으로 동일 이력을 갖는 공간 객체의 구분이 불가능한 문제가 있다[6,7]. 시간 스탬핑 모델(data models based on simple time-stamping)[20,21]은 각 객체에 대하여 객체의 생성 및 삭제에 대한 시간 스탬프쌍을 부여함으로써 객체의 시간 정보를 관리하는 방법으로 삭제되지 않은 현재 데이타의 삭제 시간 스탬프는 'NOW', 'CURRENT' 또는 'NULL' 등의 특별한 값을 가진다[20]. 이 모델에서는 동일한 객체의 서로 다른 버전이 여러개의 테이블에 존재함으로써 단일 객체의 이력 관리가 어렵고 특히 객체변경에 대한 직접적 정보를 얻기가 불가능하다[22]. Peuquet는 각각의 사건을 리스트로 구성하고, 각 사건에 대한 시간 스탬프(timestamp), 공간 속성 변화(feature change), 공간 위치 변화(locational change)를 링크드 리스트로 구성한 사건 기반 시공간 데이타모델(ESTDM: event oriented spatio-temporal data model)[9]을 제안하였다. 이력 그래프 모델(history graph model)[20]은 공간 데이타 집합에서 변화의 성질을 식별하기 위하여 시간 객체의 이력을 발생시키는 객체의 생성(creation), 변경(alteration), 삭제(cessation), 재생(reincarnation), 병합(merge/annexation), 분할(splitting/deduction)의 여섯가지 형태로 이벤트를 구분하며, 이력 객체간의 파생관계를 직접 표현할 수 있는 장점이 있다. Yuan은 시공간 데이타를 표현하기 위해 의미(semaintics), 공간(space), 시간(time) 영역을 서로 다른 테이블로 구성하고 이들간의 연결링크를 제공함으로써 공간정보의 처리 및 현상을 표현하는 삼원모델(three domain model)[10]을 제안하였는데 의미 영역은 시간 및 공간에 대해 독립적인 유일하게 식별 가능한 객체 정보를 포함하며, 공간 및 시간영역은 각각 공간 객체 정보와 시간 정보를 포함한다. 이 모델은 단순한 변화뿐 아니라 객체 이동과 같이 공간적 변화를 쉽게 표현할 수 있다. Erwig[5] 등은 추상적 데이타타입을 이용하여 이동점(moving point), 이동영역(moving region)에 대한 데이타타입을 정의함으로써 새로운 형태의 시공간 데이타모델인 이동객체의

시공간 데이타모델(spatio-temporal data models with moving object)을 제시하였다. 각각의 데이타타입은 점, 영역 등의 개체와 함께 이동 연산을 포함하며 시간 요소는 공간 개체에 통합됨으로써 변경 및 이동을 동시에 표현할 수 있다. 시공간 개체-관계모델(spatiotemporal entity-relationship model)은 객체의 위치, 공간 변이 속성, 공간관계, 관점에 따른 공간 뷰 등의 개념을 기본으로 한다[23]. 객체의 위치 정보는 형태, 크기, 위치, 원점 요소로 구성되며, 공간 변이 속성 값은 객체 자체에 대해서가 아니라 다만 위치에 따라서만 변화하는 특성을 가진다. 시공간 객체지향 모델(spatio-temporal object oriented data models) 객체, 클래스, 상속, 다형화등의 객체지향 패러다임을 기본으로 동일 객체의 모든 이력 버전을 하나의 단일 개체내에 포함하도록 한다[4,20,21,24].

앞서 기술한 대부분의 모델이 시간을 이산적 요소로 표현하며 소수의 모델[5,25]만이 연속적 요소로 표현한다. 시간 참조는 대부분의 절대적 시간 참조 방식을 지원하고 있으며 시간순서 모델링에 있어서는 모든 모델이 선형 시간만을 기본으로 하고 있다. 시간의 단위는 모두 다중 시간 단위를 지원하고 있으며 시간 개념에 있어서는 유효시간은 모든 모델이 지원하는 데 반해 거래시간을 동시에 지원하는 모델[12,13,20]은 많지 않다. 그리고, 시간스탬프를 부여하는 단위는 [5,23,25]등이 객체 단위로 부여하는 방법을 따르고 있으며, 레이어[6,16], 속성[6,7], 이벤트[9] 단위로 시간 스탬프를 부여하고 있다. 그리고 각 모델의 질의는 크게 시간질의와 시공간 질의로 구분되며, 각 질의는 다시 각각 단순, 영역, 관계 또는 사건 질의로 세분화 된다. 시간 질의만을 놓고 볼 때 대부분의 모델이 단순 시간질의는 지원하고 있으나 시간영역질의와 시간관계질의는 최근에 제시된 데이타모델[5,10,12,13,20]에서 지원된다. 시공간 질의에 있어서도 역시 대부분의 모델이 단순 시공간 질의는 지원하고 있으나 시공간 영역질의, 시공간 사건질의를 지원하는 모델은 최근의 데이타모델[10,12,20]에서 제한적으로 지원되고 있다.

지금까지의 시공간 데이타 모델들은 동일한 이력을 갖는 이력 객체간의 구별 문제[6,7], 동일한 객체의 이력이 중복 관리문제[6,9], 단일 객체의 이력 객체간 식별자가 서로 다르거나 이력 관리 단위가 지나치게 큰 문제[12,13,14,15]를 가지고 있다. 따라서 이 연구에서는 이러한 문제를 해결하기 위하여 객체 지향 모델을 기반으로 하는 시공간 데이타 모델을 제안하며 특히 단일 객체의 전체 이력을 객체별 이력 관리 테이블로 관리함으로써 단일 객체의 이력 객체간 객체 식별자가 서로 다른 문제점을 해결하도록 한다.

2.2 고딕 GIS 도구

이 논문에서 제안하는 객체 시공간 데이타모델은 영국의 Laser-Scan사에서 개발한 객체지향 지리정보시스템 도구인 GIS 도구를 사용하여 구현하도록 한다. 고딕 GIS 도구의 데이타베이스에서 모든 데이타는 객체 단위로 관리되며, 기본 데이타셋과 클래스, 클래스 간의 계층적 개념을 제공하여 동일 클래스로부터 서로 상이한 구조와 객체 연산을 가지는 다수의 객체를 구성할 수 있다[14,15]. 고딕에서 실세계의 정보(instances)는 객체에 저장되는데 하나의 객체는 반드시 하나의 클래스로부터 파생되며, 클래스는 데이타셋으로부터 기본적인 구성 정보를 상속받는 계층적 구조를 형성한다. 계층적 구조에서 개별적 객체의 기본이 되는 클래스는 사용자가 정의할 수 있으나 고딕에서 제공되는 object, rwo, geometric, spatial, _goth_pre_simple, simple, simple_point, simple_line, simple_area 등 기본적으로 이용되는 기본 클래스를 이용하여 필요한 클래스를 정의한다[14]. 그리고, 고딕은 데이타셋 단위의 버전관리 기법을 지원하고 있는데, 버전은 데이타셋의 복사본(as copy of a dataset) 개념으로 파악될 수 있으며[15], 데이타셋의 모든 내용을 중복 저장하지 않고 변화된 내용만을 별도로 관리하기 때문에 저장공간의 불필요한 낭비를 줄이고 있다. 버전관리를 할 때 변경 이전의 버전을 부모 버전, 파생버전을 자식 버전이라고 하며 동일한 부모버전으로부터 서로 다른 다수의 자식 버전을 구성할 경우 이를 버전의 분기라고 한다. 또한 분기된 다수의 하위 버전을 하나의 버전으로 구성하는 버전 통합을 지원하고 이에 따른 연산을 제공한다.

한편, 고딕 응용 시스템 개발 환경(Application Development Environment: ADE)에서 공간 및 비공간 데이타에 대한 질의는 화면 출력 프레임, 질의 프레임을 이용하여 수행된다. 프레임은 고딕에서 지원되는 응용프로그램 개발 언어인 lull 코드와 응용 어플리케이션간의 추상화된 단계로써 기본적으로 객체 지향적 접근방법을 이용한다[14,15]. 현재 고딕의 질의 프레임에서는 클래스 기반 질의, 일반 속성 기반 질의, 공간 속성 기반 질의가 가능하다. 클래스 기반 질의는 해당 클래스에 속하는 객체들을 찾는 질의이고, 일반 속성 기반 질의는 비공간 속성에대한 질의로써 표1과 같은 속성 연산자들을 사용한다. 속성 연산자는 비교 연산자와 논리 연산자로 구분된다.

표 1 고딕의 속성 연산자

구 분	연 산 자
비교 연산자	=, <, >, <=, >=, not equal, like, begins, ends, contains, is null, not null
논리 연산자	or, xor, and, and not, not and

공간 속성 기반의 질의는 주어진 영역(원, 사각형, 다각형)내에 존재하는 객체를 찾거나 주어진 선분과 교차하는 객체를 찾는 유형의 질의로써 표 2와 같은 공간 연산자를 이용한다.

표 2 고딕의 공간 연산자

공간 연산자	searching mode
equals, contained, contains, intersect, on, off, touching, nearest,	include-point, in-radius, cross-vector, cross-line, in-region, in-area, totally-in-area, nearest

그러나, 고딕은 공간 데이타를 관리하기 위한 도구이기 때문에 공간 객체의 시간적 변화 이력을 관리하기 위한 기능을 제공하지 않고 있다. 다만 데이타셋 단위의 버전관리 기능을 제공함으로써 필요한 경우 전체 데이타셋의 변경 내용들을 기록 관리한다. 따라서, 고딕에서는 스냅샷 모델과 같이 객체 단위의 이력관리가 불가능하며, 시간 개념을 지원하는 시간 연산자를 제공하지 못하고 따라서 시간 질의 및 시공간 질의 기능을 지원하지 못하고 있다[27].

3. 시공간 객체 모델

이 장에서는 시간에 따라 변화하는 시공간 객체를 표현하기 위한 시공간 데이타모델을 구현 도구인 고딕에 적합한 형태로 제안한다. 제안된 모델에서 모든 시공간 객체는 변화 이력을 단일 개체내에 포함함으로써 이력 질의시 단 하나의 객체 식별자만으로 객체의 모든 이력을 참조할 수 있다.

시공간 객체 모델을 정의하기 위해서는 먼저 시간에 따라 변화하는 공간 객체를 표현하기 위해서는 시간성을 표현할 수 있는 클래스가 필요하다[27,28]. 그러나, 공간정보 중심의 고딕은 시간을 표현하기 위한 클래스를 제공하지 않는다. 시공간 클래스는 다음과 같이 정의한다.

【정의 1】 시공간 클래스

$C_{ST} = (A, M)$

$A = \{A_{general} \cup A_{spatial} \cup A_{temporal} \cup A_{history}\}$

$A_{temporal} = < I_t, I_v >,$

$I_t = <T_{start}, T_{stop}>,\ I_v = <V_{from}, V_{to}>,$

단, $T_{start} \leqq T_{stop},\ V_{from} \leqq V_{to}$

$A_{history} = <(\exists t_1) \cdot\cdot (\exists t_n)\ (O_{ST}^{t1}, O_{ST}^{t2}, ..., O_{ST}^{tn})>$

단, $n \geqq 1$ (n은 O_{ST}의 변경횟수)

$M = \{M_{constructor} \cup M_{destructor} \cup M_{pre\_modify} \cup M_{post\_modify}\}$ □

시공간 클래스 C_{ST}는 속성정보 A와 메소드 M으로 구성된다. 속성정보 A는 다시 비공간 속성인 $A_{general}$, 공간정보 속성인 $A_{spatial}$ 외에 시간정보속성 $A_{temporal}$ 으로 세분화 되며, 메소드는 고딕의 객체관리 메소드, 디스플레이 메소드 등의 기본적인 메소드 외에 시간 변이 정보를 관리하기 위한 이력관리 메소드를 포함한다. 또한 추가적으로 객체 자신의 이력정보를 저장하기 위한 이력속성 $A_{history}$를 갖는다. 시공간 클래스 C_{ST}에서의 시간 표현은 좀 더 정확한 객체의 이력 관리를 위하여 유효(I_v) 및 거래(I_t)의 이원시간을 지원한다[27,28]. I_t는 시공간객체가 데이타베이스에 저장되는 거래시간을 의미하며, I_v는 데이타가 실세계에서 의미있는 시간으로 받아들여지는 유효시간을 의미한다[28,29,30]. 따라서, 이 두 시간은 반드시 서로 일치하는 것이 아니며, I_t가 I_v에 앞서거나 또는 그 반대가 될 수도 있다[29]. 시공간 클래스에 표현되는 공간 정보는 고딕에서 제공하는 점, 선, 면의 Simplex와 이들의 다양한 조합인 Simplical Complex로 표현되며 이들은 2차원상의 유크리드 평면(Euclidean plane)상에 표현할 수 있다.

시공간 클래스로부터 정의된 시공간 객체를 대상으로 하는 시공간 관계 연산은 다음과 같이 정의한다.

【정의 2】 시공간 관계 연산

$R_{ST} = C_{ST_i}\ OP\ C_{ST_j}$ 이며, $R_{ST} = \{R_{temoral} \cup R_{spatial} \cup R_{spatio\text{-}tempoal}\}$ 이다.

여기서 $C_{ST_i} = C_{ST_j}$

또는 $C_{ST_i} \neq C_{ST_j}$이다.

그리고 OP는 관계연산 및 논리연산을 수행하는 연산자이다. □

공간관계 $R_{spatial}$은 고딕의 공간 연산을 그대로 이용하며 시간관계연산 $R_{temporal}$과 시공간 관계연산 $R_{spatio\text{-}temporal}$은 제 5장에서 설명한다.

시간에 따른 객체의 변화를 표현하기 위한 시공간 객

체 모델 Q_{ST}는 다음과 같이 정의한다.

【정의 3】 시공간 객체 모델

$Q_{ST} = <C, R>$

여기서 $C \in C_{ST}$ 이고 $R \in R_{ST}$이다. □

고딕의 기본 클래스와 새롭게 추가되는 시간 클래스의 계층 관계를 표현하는 시공간 클래스 계층은 다음과 같이 정의할 수 있다.

【정의 4】 시공간 클래스 계층

C가 클래스의 집합이며 O가 클래스 C에 의하여 생성된 객체의 집합이라 하자. O에 대한 클래스 계층은 (C, ≐)로 표현하고, 상속관계는 ≐로 표현되는 클래스 구조이다. □

즉 시공간 클래스 계층은 시공간 클래스가 고딕의 기본 클래스와 시간 클래스로부터 다중상속을 받음으로써 구성되는 클래스간의 계층 관계를 표현한다[27,28]. 시공간 클래스 C_{ST}(temporal)는 (Csimple, ≐)로 표현되는 구조이며 simple 클래스와 ≐의 관계에 있다. 이 논문에서 시간 클래스는 고딕 클래스 계층중에서 simple의 하위 클래스로 추가함으로써 구성하였다. 따라서 temporal 클래스 자신이 시공간 클래스가 되거나 simple_point, simple_line, simple_area으로부터 다중상속 받음으로써 다양한 타입의 공간 유형을 가질 수 있다.

시간 구성요소를 포함하는 시공간 클래스의 계층은 계층적 구조로 표현되며 각각의 클래스는 상위 클래스로부터 속성과 메소드를 상속받고 또 자신만의 속성이나 메소드를 정의할 수 있다. 클래스 datetime은 주어진 단위 시간상의 한 시점을 나타내며 interval은 시작과 끝 datetime이 명시되지 않은 시간의 양(amout of time)을 표현하며, period는 시작과 끝 datetime이 명시되어지는 시간의 지속기간(duration)이다[28]. 또한 datetime set과 element는 각각 datetime과 period의 집합으로 이루어진 클래스다.

정의 1과 같이 정의된 시공간 클래스로부터 사용자는 필요한 시공간 클래스를 생성할 수 있다. 그림 2의 parcel 클래스는 공간상에서는 area 타입으로 존재하며, 시간선상에서는 시작과 끝 datetime을 갖는 period 형태로 존재한다. 또한 그림 1의 period 클래스가 갖는 속성들과 메소드 모두를 상속받으며 parcel 클래스 고유의 일반 속성인 string 타입의 owner와 integer 타입의 addr, cost를 가진다.

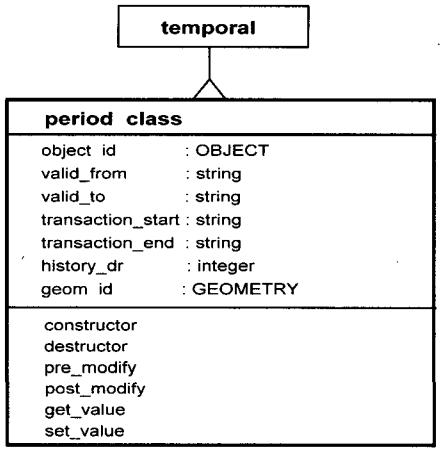


그림 1 시공간 클래스(period)

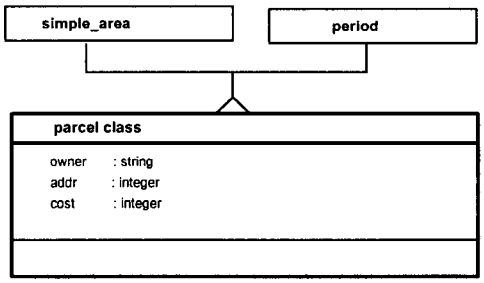


그림 2 사용자 정의 시공간 클래스 (parcel)

시공간 클래스 C_{ST}로부터 파생된 객체를 시공간 객체 O_{ST}라 한다. O_{ST}는 부모클래스 C_{ST}의 모든 속성(A)과 메소드(M)를 상속받으며 자신만의 고유 속성과 메소드를 가질 수 있다.

【정의 5】 시공간 객체

$O_{ST}=((C_{ST}.A, C_{ST}.M) \cup (O_{ST}.A, O_{ST}.M))$

$O_{ST(parcel)} = ((C_{simple\_area}, C_{ST(period)}), \doteq)$

$= (\{A_{general(owner,addr,cost)},$
$A_{spatial},$
$A_{temporal(Vfrom,Vto,}$
$Tstart, Tstop),$
$A_{history}\},$
$\{M_{constructor}, M_{destructor},$
$\cdots,$
$M_{set\_value}\})$

또한 객체의 n번째 변경상태 $O_{ST}.history(O_{ST}\ t_n)$는 간단히 $O_{ST}h(n)$으로 표기한다.□

시공간 객체는 하나 이상의 시공간 클래스로부터 생성되며 이들 각각은 자신의 변화사항을 history_dr에 저장한다. history_dr값은 객체 식별자처럼 시스템에서 주어지며 사용자가 지정할 수 없도록 하였다. 이것은 객체의 변경사항을 유지하기 위한 테이블로 사용자가 객체의 이력사항을 조회하고자 하는 경우 이 테이블로부터 정보를 얻을 수 있다. 객체는 T_{start}값으로서 객체가 생성될때의 시스템 클럭을, T_{stop}값으로서 forever값을 가지며 변경이 발생했을 때 그 당시의 시스템 클럭을 T_{stop}값으로 재저장하게 된다. 전술하였듯 T_{stop}값에 명확한 시스템 클럭이 존재한다는 것은 이력 데이타라는 것을 의미한다.

4. 시공간 객체의 이력관리

4.1 객체단위의 버전

고딕은 다음과 같은 질의에 적절한 해답을 제공할 수 없다. 이는 공간 객체의 시간성이 배제되었기 때문이기도 하지만, '필지', '하수관'과 같은 객체 단위의 버전을 제공하지 않기 때문이다.

【질의 1】 1980년 이래로 이 필지는 형태와 소유주 측면에서 어떠한 변화를 거쳤는가?

【질의 2】 1985년 누수공사가 진행되었던 하수관은 어디이며 그 당시의 하수관 경로는?

이와 같은 질의를 만족시키기 위해 객체 단위로 시간에 따른 이력사항들을 기록, 관리하는 방법이 필요하다. 이 논문에서는 데이타셋 단위로 버전을 관리하던 고딕 시스템을 객체별 이력 구분이 가능하게 하되 동일 객체의 모든 이력을 하나의 단일 개체 내에 포함할 수 있도록 각 객체에 단 하나의 객체 식별자만을 할당하며 모든 이력사항은 history_dr이라는 별도의 저장공간을 두어 관리하도록 하였다. 제안한 객체단위 버전 관리는 데이타셋 단위가 아니라 시공간 객체 단위로 이루어지기 때문에 개별적인 시공간 객체의 이력을 관리할 수 있으며 객체 단위의 시간 연산 및 시공간 연산을 지원할 수 있다[27].

4.2 객체의 이력관리

앞서 기술하였듯이 공간객체의 이력을 관리하기 위하여 시공간 클래스 구조내에 이력관리 속성을 추가하였다. 이력관리 속성 history_dr은 객체 이력을 관리하기 위한 테이블을 가리키는 포인터 역할을 한다. 그림 3과 같이 객체 이력관리를 위해서는 고딕에서 제공하는 reflex method를 사용하였으며 이 메소드들은 constructor, destructor, pre_modify, post_modify등으로써 객체의 최초 생성 및 삭제, 변경 등 객체에 사건이 발생하는 모든 경우에 수행된다[15]. 이러한 메소드들은 cache transaction을 이용하여 수행속도를 향상시키도록 구현하였다.

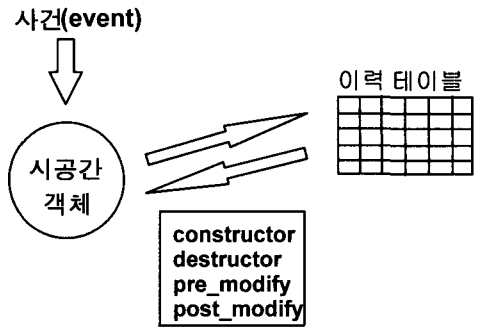


그림 3 reflex method를 이용한 객체 이력관리

이력관리 메소드의 동작은 알고리즘 1과 같은데 먼저 객체가 생성될 때 동작하는 constructor 메소드인 $M_{temporal\_constructor}$ 는 최초 객체 생성시 동작하며 거래시간 설정을 위하여 시스템으로부터 현재의 시간을 읽고 적절한 시간타입으로 변경하고 거래 시간의 시간값을 거래 시작 시간 값은 현재의 시간, 거래 종료 시간 값은 'forever'로 지정한다(단계 1-4). 그런 다음 이력 저장을 위한 description record를 생성(단계 5)하고 dr_id를 $A_{history}$의 값으로 설정한다(단계 6). 한편, 객체에 대한 변경이 발생할 경우 동작하는 메소드인 $M_{temporal_pre_modify}$는 의해 객체 자신의 history_dr 값을 획득하여(단계 1), history_dr의 가장 마지막 레코드에 (단계 2) 시스템으로부터 현재 시간값을 읽어 거래 종료 시간으로 설정하고(단계 3) 변경 전 객체를 이력관리 테이블에 변경 이전의 저장한다(단계 4-8).

5. 이력 관리 연산자의 설계와 구현

지도상의 객체간의 관계를 알아내고 다양한 유형의 질의를 위해서는 시간, 공간, 시공간 연산자의 정의 및 설계가 선행되어야 한다[30]. 여기서 시공간 연산자는 시간 연산자와 공간 연산자에 의해 표현 될 수 없는 시간과 공간에 의한 객체의 이력 질의를 위한 것으로써 공간 연산자는 고딕에서 제공하는 기본적 방향, 거리, 위상 연산자들을 이용하도록 한다.

5.1 시간 연산자

시간 관련 질의를 지원하기 위한 기본적인 요소로써

알고리즘 1 이력관리 알고리즘

```
function integer temproal_constructor
input : vac_id, obj_id, vr_id
output : status
begin
1   Get current time from system clock
2   Convert current time to adequate type
        (in this system, we restrict time granularity to month)
3   Set transaction_start to current time
        and transaction_stop to 'forever'
4   Get attribute number and type of this object
5   Make a description record as history storage (get dr_id)
6   Assgin history_dr value to dr_id
7   index := 1;
8   while (index <= attribute number) do
    begin
            add column to history_dr
            set the column type, header, key
            index := index + 1;
    end;
return GOTH_NORMAL;
end;
```

```
function integer temproal_pre_modify
input : vac_id, obj_id, vr_id
output : status
begin
1   Get history_dr value of this object
2   Get maximum row index of history_dr
3   Set transaction_stop of maxrow to current time
4   Add row to history_dr
5   Get attribute number of this object
6   index := 1;
7   while (index <= attribute number) do
    begin
8     Store each attribute value of modified object to maxrow
          including geometry information into history_dr
      index := index + 1;
    end;
    Set transaction_start of maxrow+1 to current time
          and transaction_stop to 'forever'
return GOTH_NORMAL;
end;
```

Allen이 정의한 13개의 시간 연산자[26]를 구현하였다. 구현한 시간 연산자는 시공간 질의어를 이용한 질의처리시 질의 서술절로 사용할 수 있다. 여기에서 구현에 필요한 시간 연산자를 다음과 같이 정의한다.

【정의 6】 시간 연산자

주어진 시간 간격 <from, to>와 $R_{temporal}$의 관계에 있는 객체를 추출해 내는 연산자이다.

(단, from≦to이고, $O_{ST}.V_{from} \leqq O_{ST}.V_{to}$ 를 만족해야 한다.)

$R_{temporal}$={OP_{before}, OP_{equal}, OP_{meets}, $OP_{overlaps}$, OP_{during}, OP_{starts}, $OP_{finishes}$}

OP_{before}(from, to) = ($O_{ST}.V_{to}$ < from)

OP_{equal}(from, to) = (($O_{ST}.V_{from}$ = from) & ($O_{ST}.V_{to}$ = to))

OP_{meets}(from, to) = (($O_{ST}.V_{to}$ = from) | ($O_{ST}.V_{from}$ = to))

$OP_{overlaps}$(from, to) = ((from < $O_{ST}.V_{to}$) & (to > $O_{ST}.V_{to}$))

$OP_{durings}$(from, to) = ((from < $O_{ST}.V_{from}$) & (to < $O_{ST}.V_{to}$))

OP_{starts}(from, to) = ((from = $O_{ST}.V_{from}$) & (to ≠ $O_{ST}.V_{to}$))

$OP_{finishes}$(from, to) = ((to = $O_{ST}.V_{to}$) & (from ≠ $O_{ST}.V_{from}$))

(같은 방법으로 V_{from}/V_{to}에 T_{start}/T_{stop}을 대응시킬 수 있다) □

위 7개 시간 연산자와 OP_{equal}을 제외한 그 역관계에 의해 총 13개의 시간 관계가 존재한다.

5.2 시공간 연산자

시공간 연산자는 앞서 정의한 시간 연산자와 공간 연산자의 구성요소로서 이루어지며 다음과 같이 정의하였다.

【정의 7】 시공간 연산자

주어진 시간 간격 <from, to>와 $R_{spatio-temporal}$의 관계에 있는 객체를 추출해 내는 연산자이다. (단 from < to이며 V_{from} < V_{to} 이다)

$R_{spatio-temporal}$ = {$OP_{contract}$, OP_{expand}, OP_{appear}, $OP_{disappear}$, OP_{rotate}, $OP_{displace}$, $OP_{transformate}$, OP_{union}, OP_{split}, $OP_{reallocate}$, OP_{stable},}

$OP_{contract}$(from, to) =
((∃f_i,1≤f_i<n(from≤$O_{ST}^{h(fi)}.V_{from}$<to)) &
(∃t_i, f_i<t_i≦n (from ≦ $O_{ST}^{h(ti)}.V_{to}$ ≦ to)) &($O_{ST}^{h(fi)}$.area(geom)>$O_{ST}^{h(ti)}$.area(geom)))

OP_{appear}(from, to) =
(from ≦ $O_{ST}^{h(1)}.V_{from}$ ≦ to)

OP_{stable}(from, to) =
(∃f_i, 1≤f_i<n (from ≦ $O_{ST}^{h(fi)}.V_{from}$)
& ($O_{ST}^{h(ti)}.V_{to}$ ≦ to))

(같은 방법으로 V_{from}/V_{to}에 T_{start}/T_{stop}을 대응시킬 수 있다) □

위에서 정의된 시간 연산자와 고딕에서 제공하는 공간 연산자의 조합으로 시간에 따른 공간 객체의 변화에

관한 시공간 연산이 불가능하거나 복잡한 경우의 연산을 시공간 연산자에 의해 구분하고 설계하였다. 이 논문에서는 이러한 연산자들을 크게 시간에 따른 공간 속성의 변화 특성에 따라 크기변화, 존재변화, 위치 변화, 모양 변화, 구조 변화, 불변화로 재구분하였고 각각의 세부 연산자들을 표 3과 같이 정의하였다.

표 3 시공간 연산자 분류

구분	크기변화	존재변화	위치변화
연산자	축소(contract) 확대(expand)	생성(appear) 소멸 (disappear)	회전(rotate) 전위 (displace)
구분	모양변화	구조변화	불변화
연산자	모양변형 (transformate)	병합(union) 분할(split) 재할당 (reallocate)	유지(stable)

5.3 시공간 연산자의 구현

5.3.1 시간 연산 함수

정의된 연산자는 HP 9000 C100 기종의 hp-ux 10.0 운영체제에서 구현하였으며, 사용언어는 C와 고딕의 스크립트 언어인 lull이고, Motif 툴킷에서 제공되는 UIL(User Interface Language)을 이용하여 그래픽 사용자 인터페이스(GUI : Graphic User Interface)를 제공하도록 하였다. 제안된 시공간 모델에 적용되는 함수적 시간 연산은 【정의 6】을 기반으로 한다.

【질의 3】 1996년 8월 23일 이전에 지번 48의 지가는 얼마이었는가?

와 같은 질의는 다음의 temporal_select 함수를 이용하여 수행된다. class_id는 지적 클래스를 가리키는 식별자이고 value_id는 지번이라는 속성을 가리키는 식별자이다.

```
temporal_select (vac_id, class_id,    value_id,
                          └"지적"   └지번:"48번지"
                 temporal_operators,
                          └T_BEFORE
                 temporal_value1, temporal_value2,
                                  └1996,8,23",    └"1996,8,23",
                 is_null,         set_id);
                    └TRUE,     └output
```

설계된 시간연산을 위한 함수는 사용자가 그래픽 인터페이스를 이용하여 시간 질의를 하거나 질의어에 의해 시간 연산을 하고자 할 때 호출된다. temporal_select 함수의 output으로서 리턴되어진 set_id는 주어진 질의를 만족시키는 객체 식별자들을 가지고 있다.

설계된 시간 연산자에 의한 질의를 지원하는 화면은 다음과 같이 구성하였다. select 버튼이 눌리어졌을 때 temporal_select라는 함수를 호출하여 시간값의 비교 및 질의에 적합한 객체를 선택한다. 그림 4는 위의 【질의 3】에 대한 수행 예이다.

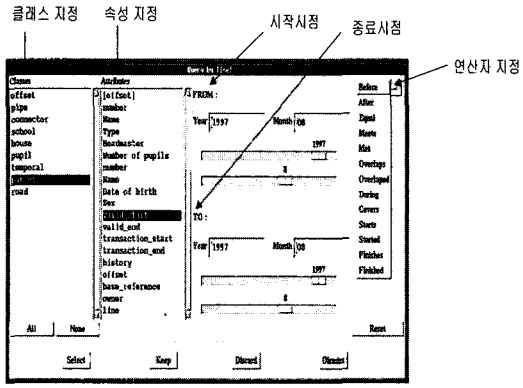


그림 4 시간 연산자의 수행

5.3.2 시공간 연산 함수

시공간 연산 함수는 【정의 7】에서 정의한 내용을 기반으로 구현한다.

【질의 4】 과거 10년동안 면적이 축소된 학교를 검색하라

와 같은 시공간 연산은 다음과 같은 시공간 함수에 의해 수행된다.

```
ST_select(vac_id, class_id,  ST_operator,
                          └school  ST_CONTRACT
          temporal_value1, temporal_value2,
            └"1987,10,20"    └"1997,10,20"
          is_null, ret_set_id);
            └true     └output
```

위의 【질의 4】와 같은 질의처리를 위한 시공간 연산은 알고리즘 2의 시공간 연산 알고리즘에 따라 수행된다. 먼저 현재 데이타셋으로부터 대상 객체들을 획득하여(단계 1-2) 각각의 객체들에 대해 입력되어진 시간 간격 동안에 객체가 변경되었는지를 판단해야 하며(단계 3-4) 변경되었다면(단계 5) 해당 연산을 만족시키는 객체를 찾아야 한다(단계 7). 이때는 각 객체의 history_dr로부터 주어진 시간 간격내에서 변경이 이루어지기 전(f_i)과 변경이 이루어진 후(t_i)의 포인터를 각각

알고리즘 2 시공간 연산 알고리즘

```
function integer ST_select
input : frame_id, t1,t2, operator
output : status
begin
1    Get object_ids from current dataset
          and push object_ids into obj_set
2    Pop an object from obj_set
3    for each object
     begin
4         if (this object is the last object) then
               highlight objects which is in selection_set
5         else
          begin
6              Compare_time(t1,t2,fi,ti);
7              if ((t1 ≦ (fi, ti) ≦ t2) and (fi ≠ ti) and
                   (fi < ti)), just only (1 ≦ I ≦ n) then
               begin
8                  Check_st_op(fi, ti, operator, obj_id,is_null);
          //Check this object is changed by this operator//
9                  if ( is_null ≠ true) then
                       put this object into selection_set
               end;
10             Get next object from obj_set
          end;
     end; // for //
return GOTH_NORMAL;
end;
```

```
function integer Compare_time
input : t1, t2
output : fi, ti, status code
begin
     Get history_dr value of this object
     Get maximum row index of history_dr
     fi, ti := 0;
     index :=1;
     while (index <= max row) do
     begin
          if (valid_start(index) ≦ t1 ≦ valid_stop(index)) then
          begin
               fi := index;
               index := max row;
          end;
          index := index + 1;
     end;
     if (fi == 0) then Find the nearest value of t1
          from history_dr and set fi to index at that time
     index := 1;
     while (index <= max row) do
     begin
          if (valid_start(index) ≦ t2 ≦ valid_stop(index)) then
          begin
               ti := index;
               index := max row;
          end;
          index := index + 1;
     end;
     if (ti == 0) then Find the nearest value of t2
          from history_dr and set ti to index at that time
return GOTH_NORMAL;
end;
```

찾는다. 이러한 기능을 지원하도록 하도록 구현한 함수가 Compare_time 이며 이 함수는 시공간 연산자가 호출되기 전에 수행되어 변화가 일어났을 경우에만 시공간 연산자를 이용한 루틴을 수행하도록 한다(단계 8). 시간 비교 루틴(Compare_time)으로부터 시간에 따른 공간적 변화가 존재함을 확인하였으면 ST_select함수의 연산자로서 입력되어진 ST_operator에 따라 각각 다른 루틴이 실행된다(단계 9).

5.4 시공간 객체의 이력지원 질의

5.4.1 시공간 연산자에 의한 질의

시간 연산자의 구현은 시간값의 유형을 사건(event)과 시간간격(interval)을 지원하도록 하였으며 사건은 특정 시점을 포함하면서 해당 시공간 연산이 일어난 객체를 질의할 때 사용된다. 그림 5는 시공간 연산자에 의한 시공간 질의(【질의 5】)의 수행예를 보여준다.

【질의 5】 1930년 1월과 1960년 1월 사이에 토지정리 사업으로 영역이 확장된 필지는?

위와 같이 주어진 시공간 질의를 시공간 연산자를 이용하여 수행하기 위해서는 그림 5와 같이 먼저 검색하고자 하는 객체의 해당 클래스와 시공간 연산자를 선택하고 시간값을 유형을 사건 또는 시간간격의 형태로 입력한 후 select 버튼을 눌러준다. 그림에서는 1930년 1월부터 1960년 1월의 시간간격이 검색하고자 하는 시간값으로 입력되었으며 그 시간동안 영역이 확장된 토지(parcel)를 찾기 위하여 Expand라는 시공간 연산자를 이용한다. 그리고 수행 결과는 그림 6의 (b)와 같다.

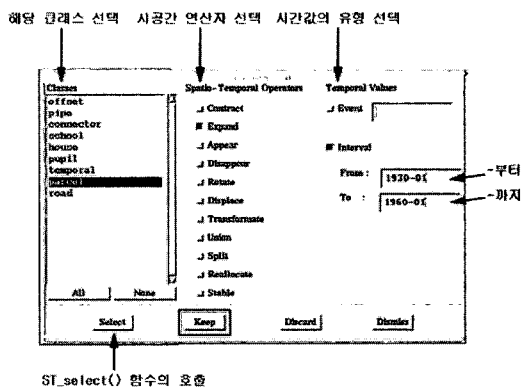
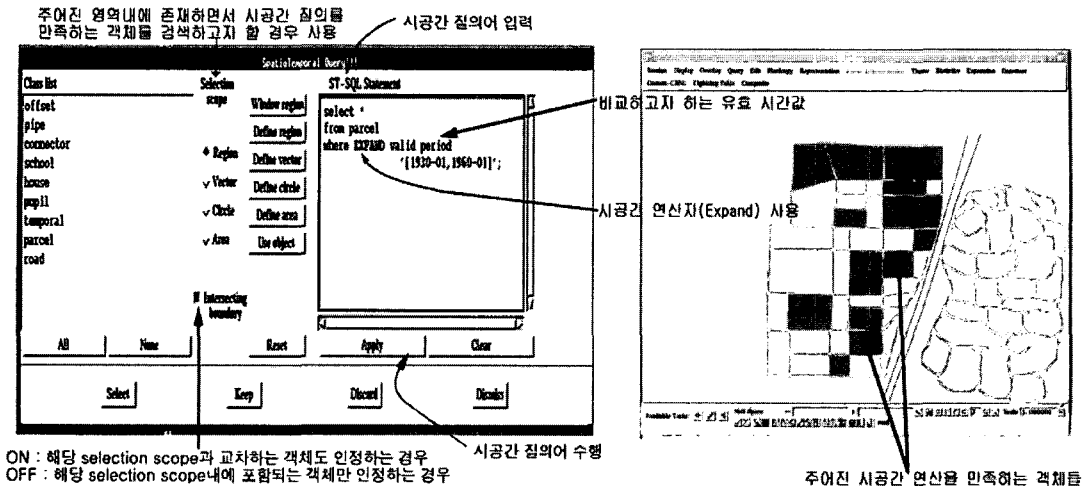
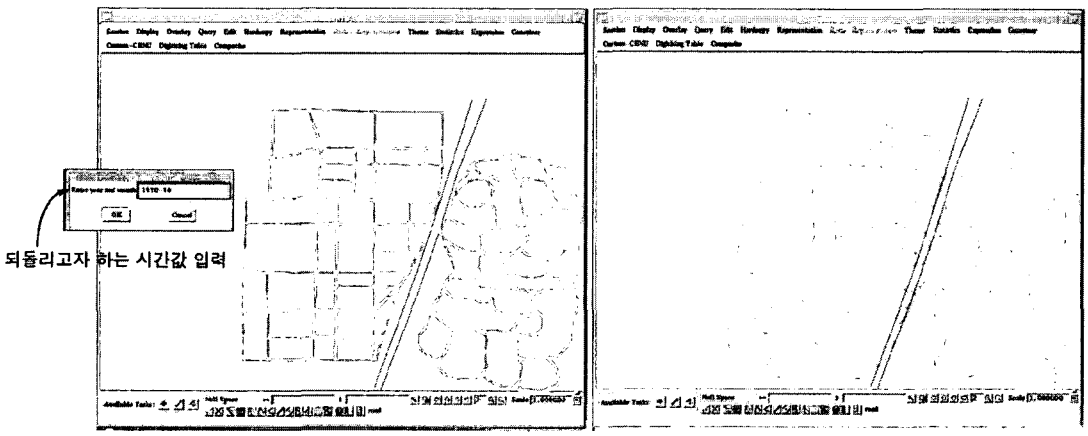


그림 5 시공간 연산자에 의한 질의



(a) 질의 (b) 결과

그림 6 시공간 질의어에 의한 질의



(a) 질의 (b) 결과

그림 7 시간에 의한 회귀연산

5.4.2 시공간 질의어에 의한 이력 질의

시공간 연산자에 의한 시공간 질의는 시공간 질의어를 이용하여 그대로 구현될 수 있다. 그림 6은 이력지원을 위한 질의시에 시공간 연산자를 질의 구분내에 포함하여 질의를 수행하는 예를 보여준다.

그림 6의 시공간 질의 입력창에는 앞의 【질의 5】를 시공간 질의어로 사용하고 있으며, 여기서 'EXPAND'라는 시공간 연산자를 질의 서술절에 사용하였으며 질의시 시간값의 유형은 시간간격(interval)으로써 'valid period(1930-01, 1960-01)'의 형태로 기술되고 있다.

5.4.3 시간값에 의한 회귀 연산 지원

기존의 지리정보 시스템이 현재의 상태만을 표현했던 반면, 시공지리정보시스템은 과거 특정 시점의 도면 상태를 조회할 수 있다. 예를 들어 "1980년 청주시의 도로망의 전체 상태는?", "토지 정리 사업이 있기전 사창동의 필지들의 분포상태는?" 등과 같은 질의는 시간값에 의한 회귀 연산에 의해 해결할 수 있다. 이러한 유형의 질의를 위하여 검색하고 싶은 연도와 월을 입력하면 당

시 시점의 전체 공간 상태를 확인할 수 있다.

그림 7은 앞서 제시한 이 연구의 토지구획 정리 사업을 회귀연산을 이용하여 검색하는 것을 보여준다. 그림에서는 현재 시점인 1980년의 토지 상황(도로를 중심으로 서구는 토지구획 정리가 완료되어 있다)으로부터 1930년 10월의 상황으로 시간을 특정한 시점으로 되돌리는 검색의 수행과 결과를 표현하고 있다. 결과화면를 통하여 알 수 있듯이 1930년 10월에는 1980년과 달리 도로를 중심으로한 동구와 서구 모두 토지정리 사업이 이루어지지 않았다.

6. 결 론

시간지리정보시스템은 기존의 지리정보시스템이 제공하는 효율적인 공간정보의 관리 기능을 제공할 뿐 아니라 시간의 흐름에 따라 변화하는 이력을 동시에 관리함으로써 다양한 응용 분야에서 사용될 수 있다. 시간에 따라 변화하는 공간 객체인 시공간 객체를 모델링하기 위해서는 객체의 변화 이력을 표현하기 위한 시공간 클래스 및 이력 저장 구조가 필요하고 저장된 객체의 이력을 질의하고 분석할 수 있도록 시간 연산자 및 시공간 연산자가 제공되어야 한다.

이 논문에서는 객체에 대한 효율적인 이력 관리를 위한 시공간 데이타 모델로써 이원 시간을 지원하는 객체지향 개념의 시공간 데이타모델을 제안하고 지리정보시스템 도구인 고딕을 이용하여 구현하였다. 시공간 데이타 모델은 고딕의 기본 클래스 외에 시간클래스를 추가 정의함으로써 시공간 데이타 표현이 가능하고 시간에 따른 객체 변경사항을 질의하도록 하였으며, 특히 객체 자신의 이력을 포함하며 단 하나의 식별자로 객체의 모든 이력을 참조할 수 있도록 하여 이력 질의의 효율을 기했다. 또한 사용자에 의한 공간 객체의 이력 질의시 요구되는 시간, 시공간 연산자를 설계 구현하였다. 특히, 기존의 고딕과 달리 데이타 셋 단위의 객체 단위의 이력을 지원하도록 함으로써 자동적인 이력관리 및 이력 객체를 대상으로한 시간 연산과 시공간 연산이 가능하도록 하였다. 이러한 기능을 이용함으로써 기존의 GIS 도구를 시간에 따른 이력 관리를 필요로 하는 응용분야에 널리 사용할 수 있다.

한편, 이 논문에서 다루지 못한 시공간 데이타베이스 구축시 기능상의 문제로 인한 객체 삭제 방법의 제공, 이력 검색시 시간 인덱스 및 시공간 인덱스를 제공함으로써 검색 성능을 증진하는 방법, 객체 변경시 전파에 따른 문제들의 해결방법에 대한 연구는 향후 과제로서 진행되어야 한다.

참 고 문 헌

[1] M. J. Egenhofer, J. Herry, "A Topological data model for spatial databases," Proceeding of the First International Symposidum on Large Spatial Databases, Santa Babara, 1989

[2] J. T. Candy, "Development of a prototype temporal geographic information system," Simon Fraser Univ., master's thesis, 1995

[3] Christopher Claramunt, Marius Theriault. "Managing Time in GIS:An Event-Oriented Approach," Recent Advance in Temporal Databases, 1995.

[4] M. J. Egenhofer, A. Frank, "Object-Oriented Modeling for GIS," Journal of the Urban and Regional Information Systems Association, pp. 786-795, 1994

[5] M.Erwig, R.H.Guetting, M.Schneider, and M.Vazirgiannis, "Spatio Temporal Data Types: An approach to Modeling and Querying Moving Objects in Databases," Chorochronos, Technical Report, CH-97-08, 1997

[6] G. Langran and N. R. Chrisman, "A framework for temporal geographic information," Int. Journal of Geographic Information Systems, 3(3), 1989

[7] G. Langran, "Time in Geographical Information Systems," ed. Taylor & Francis, London, 1992

[8] Leslie David Montgomery, "Temporal geographic information systems technology and requirements: Where we are today," Master's thesis, The Ohio State Univ. 1995

[9] Donna J. Peuquet and Miu Duanm "An event-based spatiotemporal data model (ESTDM) for temporal analysis of geographic data," Int. Journal of Geographic Information Systems, 1995

[10] M. Yuan, "Temporal GIS and Spatio- Temporal Modeling," 3rd Int. Conference/Workshop Integrating GIS and Environmental Modeling, Jan 21-25, 1996.

[11] Richard T. Snodgrass, "The TSQL2 Temporal Query Language," Kluwer Academic Publishers, 1995

[12] M.F. Worboys, "A Unified Model for Spatial and Temporal Information," The Computer Journal, Vol.37, No.1, pp.27-34, 1994

[13] M.F. Worboys, "GIS : A Computing Perspective," Taylor & Francis Pub., pp. 301-314, 1995

[14] Laser Scan Ltd., "Writing and developing

applications using GOTHIC ADE," PRG_602_TRG Issue 2.0, 1995
[15] Laser Scan Ltd., " Training Course : Gothic Concepts," CPT_205_TRG Issue 2.0, 1995
[16] C. Armenakis, "Estimation and Organisation of Spatio-Temporal Data," Proceedings of the Canadian Conference on GIS '92, Ottawa, Canada, 1992
[17] A.Frank, "Theories and Methods of Spatio-Temporal Reasoning in Geographic Space," Lecture Notes in Computer Science 639, Springer Verlage, 1992
[18] K.K.Al-Taha, R.T.Snodgrass, and M.D. Soo, "Bibliography on Spatio-Temporal Databases," ACM SIGMOD Record, Vol.22, pp.59-67, 1994
[19] F. Bonfatti, and P.D. Monari, "Spatiotemporal Modeling of Complex Geographical Structures," Computer support for Environmental Impact Assessment: Proceedings of the IFIP TC5/WG5.11 Working Conference on Computer Support for Environmental Assessment, CSEIA 93, Como, Italy, Elsevier, 1994
[20] A.Renolen, "History Graphs: Conceptual Modelling of Spatiotemporal Data," In GIS Frontiers in Business and Science, Vol.2, International Cartographic Association, 1996
[21] Gary J. Huntet and Ian P. Williamson, "The Development of a Historical Digital Cadasral Database," Int. Journal of Geographic Information Systems, 4(2), 1990
[22] A. Pavlopoulos, and B. Theodoulidis, "Reveiw of SpatioTemporal Data Models," TimeLab Technical Report TR-98-3, 1998
[23] N.Tryfona and Th.Hadzilacos, "Logical Data Modeling for Spatio-Temporal Applications: Definitions and a Model," Internationa Databae Engineering and Application Symposium, IEEE Press Proceedings, 1998
[24] E. Rojas-Vega and Z. Kemp, "An Object-Oriented Data Model for Spatiotemporal Data," Proceedings of the Ninth Annual Symposium on Geographic Information Systems, Vancouver, Canada, 1995
[25] P. Sistla, O.Wolfson, S.Chamberlain, S.Dao, "Modelling and Querying Moving Objects," Proceedings of the 13th International Conference on Data Engineering, Birmingham, UK, 1997
[26] J. F. Allen, "Maintaing Knowledge about temporal intervals," Communication of the ACM, Vol. 26, Num. 11, pp. 832-843, Nov. 1983
[27] 강병극, 백주연, 이성종, 류근호, "시간과 버전을 지원하는 객체지향 모델의 통합", 한국정보처리학회 학술발표논문집 제 3권, 제 2호, 1996년 10월
[28] 김삼남, "시간지원 패러다임에서 개체-관계 모델의 객체지향 모델 변환", 충북대학교 전자계산학과, 박사학위 논문, 1997년 8월
[29] 정경자, "사용자 접속 언어를 이용한 시간지원 질의 처리기의 구현", 충북대학교 전자계산학과, 석사학위 논문, 1993년 2월
[30] 이성종, 김동호, 류근호, "시공간 데이타 모델", 한국정보과학회 논문지(B), 제 26권, 제 11호, 1999년 11월

백 주 연

1996년 충북대학교 컴퓨터과학과 졸업(이학사). 1998년 충북대학교 대학원 전자계산학과 졸업(이학석사). 1998년 ~ 현재 (주)LG EDS 시스템 공공 1사업부 GIS팀 인천시 UIS파트. 관심분야는 시간 데이타베이스, 시공간 데이타베이스, Temporal GIS, UIS 등

이 성 종

1994년 충북대학교 컴퓨터과학과 졸업(이학사). 1996년 충북대학교 대학원 전자계산학과 졸업(이학석사). 1996년 ~ 현재 충북대학교 대학원 전자계산학과 박사과정. 관심분야는 시간 데이타베이스, 시공간 데이타베이스, Temporal GIS, 객체지향 데이타베이스 등

류 근 호

1976년 숭실대학교 전산학과 졸업(이학사). 1980년 연세대학교 산업 대학원 전산전공(공학석사). 1988년 연세대학교 대학원 전산전공(공학박사). 1976년 ~ 1986년 육군 군수 지원사 전산실(ROTC 장교), 한국전자통신연구소(연구원), 한국방송대학교 전산학과(조교수)근무. 1989년 ~ 1991년 Univ. of Arizona. Research Staff (TempIS 연구원, Temporal DB). 1986년 ~ 현재 충북대학교 컴퓨터과학과 교수. 관심분야는 시간 데이타베이스, 시공간 데이타베이스, 지식기반 정보검색, Temporal GIS, 객체 및 지식베이스 시스템