한국지리정보학회지 7권 3호 : 35~47(2004)

e-로지스틱스에서 효율적인 차량관제를 위한 질의 처리기 구현

김동호[1*] · 김진석[1] · 류근호[2]

Implementation of Query Processor for Efficient Vehicle Monitoring and Control in e-Logistics

Dong-Ho KIM[1*] · Jin-Suk KIM[1] · Keun-Ho RYU[2]

요 약

최근 부각되고 있는 텔레매틱스와 위치기반서비스의 효율적인 구축을 위해서는 실시간으로 수집되는 방대한 분량의 위치 데이터의 저장, 질의, 프리젠테이션 등을 포함하는 이동체 기술이 필요하다. 특히 물류분야에서 이동중인 차량의 데이터에 대한 효과적인 검색과 분석을 위해서는 응용별 특화된 형태의 데이터를 획득할 수 있는 질의구문이 요구되며, 전통적인 데이터베이스 질의어와 비교할 때 그 구조는 복잡한 양상을 보인다. 이에 대한 효과적인 대안으로는 SQL과 같은 표준화 데이터베이스 언어를 통한 접근을 고려할 수 있다. 따라서 이 논문에서는 e-로지스틱스 기반의 차량관제에서 요구하는 질의에 대하여 SQL를 확장한 이동체 질의어(MOQL)를 제안하고, 이를 효과적으로 처리하기 위한 질의처리기를 설계 및 구현한다.

주요어 : e-로지스틱스, 이동체, 질의언어, 질의처리

ABSTRACT

Telematics and LBS is one of rapidly emerged technology domains. In order to efficiently construct them, moving object technology which manages huge volume of real-time location data is required. Especially, the query which obtains special sorts of information closely related to the detailed applications is required in order to effectively retrieve and analyze the location data for moving object in logistics domain. It has also complex query structure comparing to the conventional database query. The approach using the standard database query language, like SQL, can be considered as an effective alternative choice. In this paper, we not only propose a new query language, entitled as MOQL based on SQL, for the query processing of the vehicle monitoring and control in e-Logistics but also design and implement the query processor.

KEYWORDS : e-Logistics, Moving Object, Query Language, Query Processor

2004년 7월 15일 접수 Recieved on July 15, 2004 / 2004년 9월 7일 심사완료 Accepted on September 7, 2004

1 한국전자통신연구원 우정기술연구센터 u-Post연구팀 u-Post Research Team, Postal Technology Research Center, ETRI
2 충북대학교 전기전자 및 컴퓨터공학부 School of Electrical & Computer Engineering, Chungbuk National University
* 연락저자 E-mail : kdh@etri.re.kr

서 론

인터넷을 기반으로 하여 새로이 형성되고 있는 가상의 물류 기업 활동 및 서비스 체계를 e-로지스틱스(e-Logistics)로 정의할 수 있으며 이를 위한 프레임워크는 다양한 유형의 물류정보시스템 간의 B2B 통합, 물류 정보 및 차량의 흐름에 대한 실시간 모니터링과 예외상황 발생에 따른 지능화된 경고/조치를 위한 가시화(Visibility), 최적화된 물류계획 수립 및 물류계획의 동적/지능적 재조정을 위한 물류 최적화 시스템 등으로 구성된다.

이동체 관리 시스템(moving object management system : MOMS)은 차량, 비행기, 선박과 같은 다양한 유형의 이동체로부터 생성되는 시공간 데이터를 저장하고 질의하며 관련 정보의 가공 및 표현을 지원하는 시스템이다. 또한 e-로지스틱스 환경에서 실물의 가시성 확보를 바탕으로 지능적이고 민첩한 실행통제 활동을 실현하기 위한 도구인 e-로지스틱스 지능화 시스템을 구성하는 요소이다.

이동체 관리 시스템은 그림 1에서 보여진 바와 같은 시스템 구성도로서 다음의 세부적인 기능으로 정리할 수 있다.

차량 위치 데이터 수집 기능: 다양한 유형의 장치에 대한 인터페이스를 통하여 이동중인 차량의 위치 값을 실시간으로 수집하고 표준 좌표체계로 변환하는 기능과 차량 제어를 위한 메시지 송신 기능을 제공한다(김동호 등, 2003). 즉, 기존에 구축된 다양한 유형의 차량 추적 시스템에 대한 통합 인터페이스를 지원하여 차량 위치 수집 장비와 전자지도의 유형에 대한 호환성 극대화를 제공하고, 문자 메시지를 실시간으로 차량에 전송함으로써 능동적인 제어를 가능케 한다.

차량 위치 데이터 관리 기능: 방대한 이동체 위치 데이터에 대한 효율적인 저장 및 검색 성능을 빠르고 효율적으로 향상시키기 위한 내부 기능을 지원하고, 사용자가 원하는 데이터를 쉽게 검색할 수 있는 질의 기능을 제공한다(안윤애 등, 2002; Kim 등, 2002). 이동체에 관련된 질의처리를 위해서는 이동체 위치 데이터에 대한 시공간 연산자가 요구되며, 이를 통하여 기존의 차량 관제 시스템과 차별되는 기능을 제공한다. 즉, 응용 분야의 사용자가 요구하는 이동체의 위치 정보 또는 이동체의 궤적 정보를 관리하기 위해 다양한 종류의 시공간 기하 및 위상관계 연산을 지원한다.

지능형 물류 응용 서비스 기능: 이동체 데이터베이스에 저장되는 데이터를 대상으로 전문가의 지식을 데이터베이스화하고 이를 기반으로 이동중인 차량의 위치 값에 대한 상태를 도출하여 상황에 따라서 필요한 동작을 수행하는 기능을 말한다. 이동체 관리 시스템은 실시간으로 운행되는 차량에 대한 위치 정보를 기록하므로 물류 지식 관리기의 능동적 궤적 상태 해석을 통해 화물의 수/배송이 운행 계획에 따라 이루어지는가를 실시간으로 확인할 수 있으며, 차량이 수송 경로에서 지연되거나 단축 운행을 할 경우에 상황에 대한 메시지를 송수신할 수 있는 기능을 지원한다.

차량 위치정보 프리젠테이션 기능: 유·무선 환경에서 GIS(geographic information system)기반의 위치정보와 차량관련 부가정보를 표현함으로써 사용자와 시스템간의 인터페이스 제공하여 커뮤니케이션이 가능하도록 한다. 사용자의 질의에 따라 실시간으로 도출된 지식을 바탕으로 배송현황, 차량현황, 운송계획 등을 제공함으로써 물류효율성을 증대할 수 있다(이혜진, 2003). 또한 차량 위치정보는 GIS의 전자지도와 함께 표준화 언어인 GML(geographic markup language) 및 SVG(vector graphics)를 기반으로 인코딩 수행 후 타 시스템에 제공되거나 차량관제 모니터링과 검색 형태로 표현함으로써 물류환경에서 실물의 가시성을 향상시킨다.

그림 1의 구성도에서 계층-2에 위치한 이동체 엔진(moving object engine)은 수집되는 이동체 위치 데이터를 관리하는 핵심 모듈로서 수집된 이동체 위치 데이터에 대하여 패킷 디코딩을 포함하는 로딩 기능과, 저장되지 않은 이동체 위치 데이터에 대한 불확실성 처리 기능 및 이동체 위치 데이터 색인 기능, 그리고 이동체 데이터를 질의하기 위한 모든 이동체 연산이 지원된다.

하지만 이전의 이동체 관리 시스템은 고정된 형태의 이동체 질의만을 고려하여 정의하였기 때문에 최종 사용자가 이동체 위치에 대한 고정적인 질의만이 가능하였다. 또한 시스템 관점에서 다수의 입력 질의에 대한 처리를 위한 고려가 미비하기 때문에 대규모 데이터와 복잡한 질의 처리에 대한 부가적인 방안이 제시되어야 한다.

따라서, 본 논문에서는 관련 문서에서 제시된 SQL(structured query language)기반의 이동체 질의구문을 수용한 질의어에 대한 효율적인 처리 기법을 제시하고자 한다. 즉, 이 문서에서는 이동체 엔진에 포함되어 사용자로부터 입력되는 다양한 형태의 SQL기반의 질의에 대한 구문분석과 의미분석 및 실행을 위한 알고리즘을 제시하며, 입력된 단일 질의 또는 다중 질의에 대하여 효율적인 처리를 위한 기법을 설계 및 구현하였다.

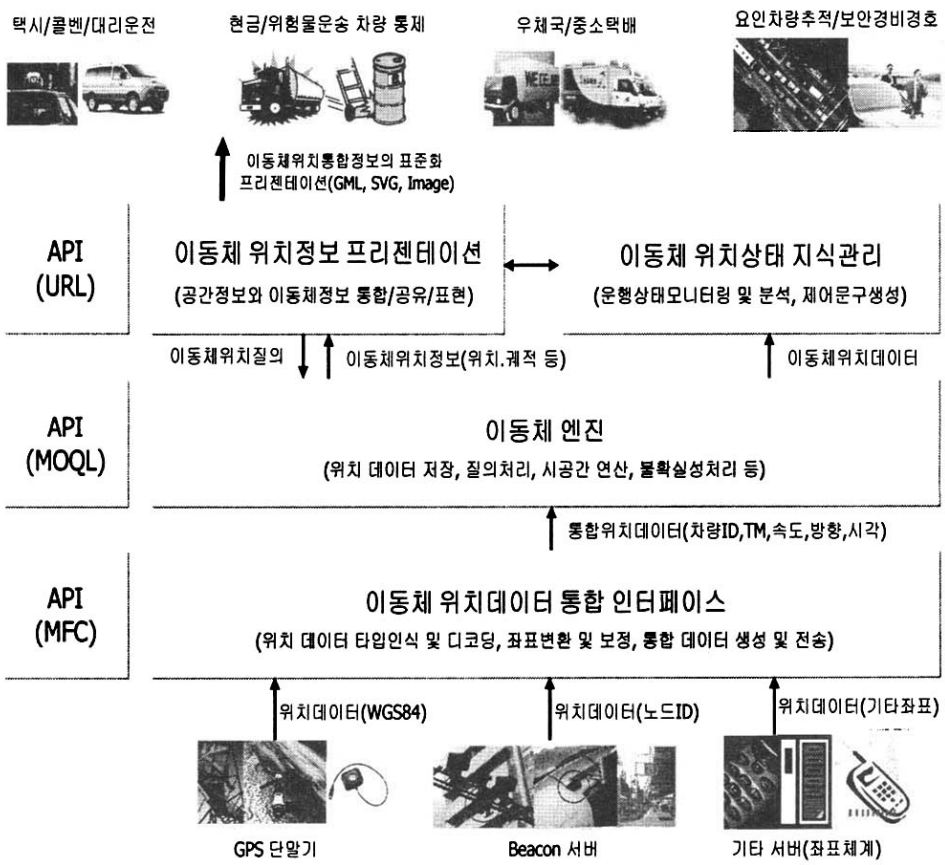


FIGURE 1. MOMS architecture

관련 연구

1. 기본적인 질의처리 방안

일반적인 질의처리에서는 입력된 SQL 질의어를 대상으로 토큰 분해, 구문 검사, 의미 분석, 실행 계획 생성 및 실행의 절차를 진행한다. SQL 질의어로부터 생성된 실행계획은 트리 구조를 가지며, PROJ(Projection), JOIN(Join), SELE (Selection)과 같은 단위 연산으로 구성된다.

이러한 실행계획을 하위 노드부터 차례로 방문하여 각각의 노드 타입에 따른 연산을 수행하여 그 결과를 자신의 상위 노드의 입력 값으로 전달하는 형태로 수행하고, 최상위 노드의 연산 결과는 입력된 SQL 질의어에 대한 실행 결과가 된다.

즉, 하위 노드의 연산 결과는 임시 테이블 생성 후 저장되며, 상위 노드로 전달하며, 이를 토대로 입력된 SQL질의에 대한 처리비용을 산정하고 최소화하기 위한 부가적인 절차가 선택적으로 진행된다.

예를 들어 'SELECT T1.I1, T2.I2 FROM T1, T2;' 과 같은 SQL질의어가 입력되면, 최하위 두 개의 노드(SCAN)는 데이터베이스에 저장된 두 개의 테이블 T1, T2를 각각 스캔 처리하여 결과를 임시 테이블로 만들고 이 결과를 상위 노드(JOIN)에 전달한다. JOIN 노드는 입력받은 두 개의 테이블을 조건에 따라 JOIN 연산을 수행하여 결과로 한 개의 임시 테이블을 만들어 상위 노드(PROJ)에 전달한다. 최종적으로 루트 노드(PROJ)는 입력받은 임시테이블로부터 해당 속성(I1, I2)만을 추출하여 출력한다.

위와 같은 실행 계획을 대상으로 진행되는 보편화된 질의처리 방안은 실행 계획에 존재하는 노드들의 위치를 조정하여 동일한 결과를 산출하면서도 데이터베이스의 액세스 회수를 감소시키며 처리 결과로서 생성되는 임시 테이블의 개수와 크기를 최소화하는 방법이다.

이와 함께, 입력되는 SQL질의어를 처리할 때 단일의 질의문장을 여러 번 처리하거나 하나의 문장으로 통합하고 한번의 실행 후 질의 결과를 수평·수직 분할하여 출력하는 관점에 따라서 질의처리 유형이 구분된다.

2. 이력 관리를 위한 처리 기법

기존의 관계형 데이터베이스와 달리 시간지원 데이터베이스에서는 유효시간과 거래시간 및 이벤트 시간을 통하여 현실세계에 존재하는 객체에 대한 이력정보를 제공할 수 있다. 즉, 전형적인 관계형 데이터베이스에서는 객체에 대한 정보는 정규형에 의해 오직 하나만 존재하며 갱신연산에 의해 과거의 값은 항상 최신의 값으로 변경된다. 하지만 응용분야에 따라서 비록 객체에 대한 갱신연산이 발생한 경우에라도 과거의 이력 값을 별도로 저장하고 필요한 경우에만 접근하도록 할 필요가 있다.

결국 시간지원 데이터베이스에서는 사용자로부터 시간 값을 새로운 질의조건으로 입력받아 처리를 한다. 세부적인 관점에서 객체에 대한 이력을 표현하는 방법에는 시간 값을 부여하는 방식에 따라 튜플-타임스탬핑과 속성-타임스탬핑으로 구분하며 부여되는 시간 값의 특성에 따라 유효시간 테이블과 거래시간 테이블로 세분화된다.

시간지원 데이터베이스는 일반적인 데이터베이스에 비해 삭제 연산은 논리적으로 변경 및 추가 연산으로 변환되기 때문에 자료의 분량이 급증하는 특성을 가진다. 또한 시간지원 데이터베이스에 입력되는 질의도 과거와 현재 및 미래에 대한 시간 범위를 포함하고 있으며, 두 개 이상의 테이블에 대한 다양한 형태의 조인 연산이 존재하여 그 처리비용도 거대한 특성을 가진다.

하지만 기존의 관계형 데이터베이스에서 제시된 바와 같은 최적화 기법과 유사하게 입력된 사용자 질의에서 기술된 시간 조건절(temporal predicate clause) 분석을 통해 검색 영역을 상당히 축소시킴으로써 처리비용을 감소시키는 기법이 제안되었다.

3. 공간 객체를 위한 처리 기법

공간 객체에 대한 연산을 포함하는 질의는 일반적인 관계형 질의 처리기법에 비해 복잡한 특성을 가진다. 이것은 공간 객체의 데이터 유형에 내재된 기하학적 요소와 위상관계적 요소를 포함하는 특성에 기인한다. 여기서 기하학적 요소란 공간상의 거리, 넓이, 둘레길이 등의 성질을 의미하며, 위상관계적 요소란 객체들간의 포함관계, 이웃관계 등의 성질을 의미한다.

전형적인 관계형 질의가 특정 값에 대한 검색을 위주로 하며 일차원적인 범위 검색에 한정됨에 비하여, 공간 객체에 대한 질의는 방대한 데이터 분량과 검색 범위와 연산 특성을 필요로 하기 때문에 상당히 많은 질의처리 기법들이 제안되었다.

이러한 공간 객체 질의에 대한 처리기법은 주로 공간 색인 기법과도 밀접한 관련성을 가지며, 공간 객체의 차원 변환에 의한 처리 방법과 여과-정제에 의한 처리 방법으로 구분된다. 특히 여과-정제에 의한 처리기법에서는 주어진 질의로부터 데이터베이스를 검색할 후보 영역을 생성하여 관련된 일부 객체 집합만을 선별한 다음, 선별된 객체를 대상으로 기술된 공간 연산을 만족하는 객체를 재 탐색하는데, 대표적으로 R-트리를 이용한 질의처리 기법을 들 수 있다.

전형적인 공간 질의처리 방법에서는 입력된 SQL질의어로부터 공간 연산을 포함하는 서브질의와 비공간 연산만을 포함하는 서브질의로 분해하고 각각을 개별적인 처리기를 두어 처리한 후 질의 결과를 합성하는 이원형 질의처리 구조를 가진다.

4. 이동체 관리 기술

이동체 관리를 위한 대표적인 응용 시스템 연구에는 DOMINO, CHOROCHRONOS, DEDALE, Battlefield Analysis등이 있다. DOMINO (Wolfson 등, 1999)는 이동 객체의 수송에 대한 웹 기반 실시간 궤도 응용의 개발을 촉진하는 이동 객체 소프트웨어 도구로서, DBMS 기술을 활용하여 만든 실시간 위치 추적 시스템 프로토타입이다. 그러나 DOMINO 프로토타입은 이동 객체의 현재 위치, 속도, 방향정보를 이용하여 미래의 이동 위치를 예측하는 방법에 주로 초점을 맞추고 있다. 따라서 과거 시점을 포함하는 이동 객체의 완전한 이동 경로인 궤적을 관리할 수 없는 단점을 가진다. CHOROCHRONOS 프로젝트에서는 시공간 데이터베이스의 특수한 형태인 이동 객체에 관한 연구가 집중적으로 수행되었고, 이동 객체의 데이터 모델링 및 인덱싱에 관한 많은 연구 결과를 발표하였다. 특히 이 연구는 GPS 기반의 수송 관리 시스템과 멀티미디어 시스템에 적용한 응용 시나리오를 제시하였다.

DEDALE(Grumbach 등, 1999)은 제약사항 데이터베이스 모델을 이용하여 시공간 데이터를 모델링하고 질의처리를 하기 위해 개발된 프로토타입으로, 기존의 시공간 데이터의 모델뿐만 아니라 이동 객체의 궤적 등과 같은 데이터 모델 및 질의 표현도 제공하였다. DEDALE은 고급 수준의 개발 및 최적화를 위한 질의어의 확장을 위해, 공간 질의에 대한 최적 계산 기법의 사용 대신 기하 데이터에 대해 선형 제약사항 추상화를 제공하였다. 그러나 DEDALE 프로토타입은 데이터베이스에 시간의 변화에 따른 이동 객체의 (x, y) 좌표 값이 직접 저장되지 않고, 특정 구간의 궤적을 표현하는 선형 제약사항(linear constraint)의 공식이 저장된다. 이로 인해 빈번하게 이동하는 객체의 실시간 위치 추적을 위한 응용 시스템 개발에는 부적합한 특징을 가진다. Battlefield analysis(Park 등, 2001)는 모의 전장에서 이동하는 부대 및 탱크들의 움직임을 예측하여 이를 의사결정에 활용할 수 있도록 개발된 전장분석 프로토타입이다. 전장분석 프로토타입은 이동 객체 관리기와 추론 엔진을 접목시키고자 하는 데 초점이 맞추어졌다. 특히, 시공간 이동 객체의 연산 결과를 추론 엔진에서 활용하는 새로운 이동 객체 추론 모델을 제시하였다.

이동체 질의 타입 및 표현

1. 질의 타입

이동체 엔진에서는 수집된 이동체 위치데이터의 효율적인 관리와 함께 QG-100에서 QG-700까지 7가지 형태로 정의되는 다양한 유형의 사용자 질의 그룹에 대한 처리를 지원하고 있다(이현아 등, 2003). 각각의 질의 그룹들은 차량 번호와 시간 및 공간 조건으로부터 차량의 위치나 궤적 정보를 획득한다. 아래에서 VID는 차량 번호, T는 유효 시간, S는 공간 정보, p는 위치 정보가 제약조건을 만족할 가능성을 나타내며, n은 n=0인 자연수이다.

[QG-100] $VID + T_1 \rightarrow S_1$
[QG-200] $VID + T_1 + T_2 \rightarrow ST_n$
[QG-300] $VID + T_1 + T_2 + S_1 + S_2 \rightarrow ST_n$
[QG-310] $VID + T_1 + T_2 + S_1 + S_2 \rightarrow ST_n$
[QG-400] $T_1 + S_1 \rightarrow VID_n + S_n$
[QG-500] $S_1 + S_2 \rightarrow VID_n + ST_n$
[QG-510] $VID + S_1 + S_2 \rightarrow ST_n$
[QG-600] $VID + T_1 + p \rightarrow S_1$
[QG-700] $VID + S_1 \rightarrow T_1$

2. 질의 연산자

이동체 관리 시스템은 고정된 값의 공간 데이터와 시간 데이터뿐만 아니라 유동적인 값의 형태를 가진 이동체 위치 데이터에 대한 관리를 포함하고 있기 때문에 질의들을 만족하는 데이터를 사용자에게 제공하기 위해서는 이전에 정의된 순수한 SQL문장만을 이용하기에는 많은 어려움이 존재한다.

사용자 질의는 복잡한 시공간 데이터 및 연산으로 구성되기 때문에 SQL2와 같은 이전의 문법을 통해 작성된 질의문은 대부분 중첩-질의 형태로서 복잡하고 장황한 문장 형태를 가지며 일부는 표현이 불가능할 수도 있다. 또한 입력된 질의에 대한 처리비용 관점에서도 불필요한 테이블의 반복적인 액세스와 과도한 메모리 사용이 야기될 수 있다.

NOW : 현재 시간을 표현하며, SQL 문의 SYSDATE로 대치하여 사용할 수 있으며, DB가 존재하는 시스템의 현재시간을 반환한다.

HERE : 질의가 입력된 시점의 시간에 위치한 이동체의 공간상의 위치 데이터를 반환한다.

BUFFER : 한 거점으로부터 일정한 반경에 포함되는 이동체의 위치데이터를 표현한다. 반지름은 질의어의 매개변수이며, 이동체의 공간(S), 시간(T) 정보는 일반 SQL 연산을 통하여 획득되어 연산자의 매개변수로 사용된다. 반환되는 연산 결과는 차량 식별자와 위치정보이다.

PROBABILITY : 데이터베이스에 저장되지 않은 이동체의 위치 데이터 표현한다.

SECTION : 두 거점 사이의 이동 경로 표현한다.

BOUNDARY : 두 거점을 꼭지점으로 하는 최소 영역 사각형(MBR)내의 이동 경로를 표현한다.

STARTTIME : 이동체가 거점을 출발하는 시간 표현한다.

ENDTIME : 이동체가 거점에 도착하는 시간 표현한다.

3. SQL 표현

7가지 유형의 이동체 질의는 이동체 연산과 기존의 SQL2에서 정의된 연산을 통해 다음과 같이 표현할 수 있다. 이를 통해 이동체 연산을 포함하는 새로운 SQL인 이동체 질의어(moving object query language : MOQL)의 표현성과 처리비용 절감의 효과를 직관적으로 확인할 수 있다.

[QG-100] 서울81바3578의 현재(2003-2-7 17:00) 위치는?

```
SELECT POSITION
FROM Vehiclehistory
WHERE ID='sl81ba3578' AND VALID AT NOW ;
```

[QG-200] 서울81바3578의 2003-2-7 16:30~현재(17:00)까지의 궤적은?

```
SELECT TRAJECTORY
FROM Vehiclehistory
WHERE ID='sl81ba3578' AND
  VALID FROM 'Feb 07, 2003 16:30' TO NOW;
```

[QG-300] 서울81바3578의 2003-2-7 17:00~현재(17:30)까지 현대백화점~길동사거리 구간의 궤적(부분궤적)은?

```
SELECT TRAJECTORY
FROM Vehicleshistory
WHERE ID =' sl81ba3578' AND
  VALID FROM 'Feb 07, 2003 17:00' TO NOW AND
  SECTION FROM '현대백화점' TO '길동사거리' ;
```

[QG-310] 서울81바3578의 2003-2-7 17:00~현재(17:30)까지 현대백화점~길동사거리 구간(MBR)의 궤적은?

```
SELECT TRAJECTORY
FROM Vehiclehistory
WHERE ID = 'sl81ba3578' AND
  VALID FROM 'Feb 07, 2003 17:00' TO NOW AND
  BOUNDARY FROM '현대백화점' TO '길동사거리';
```

[QG-400] 2003-2-7 17:10 현대백화점으로부터 반경 2km 이내에 존재했던 차량과 차량의 위치는?

```
SELECT ID, TRAJECTORY
FROM Vehicleshitory
WHERE VALID AT 'Feb 07, 2003 17:00' AND
  BUFFER FROM '현대백화점'OF 2000 m;
```

[QG-500] 현대백화점~길동사거리 구간을 지나는 차량들과 각 차량들의 궤적은?

```
SELECT ID, TRAJECTORY
FROM Vehicleshistory
WHERE SECTION FROM '현대백화점' TO '길동사거리';
```

[QG-510] 서울81바3578의 현대백화점~길동사거리 구간에서의 궤적은?

```
SELECT TRAJECTORY
FROM Vehicleshistory
WHERE ID =' sl81ba3578' AND
  BOUNDARY FROM '현대백화점' TO '길동사거리';
```

[QG-600] 서울81바3578의 2003-2-7 17:05 에서의 추정위치는? (70%)

```
SELECT POSITION
FROM Vehicleshistory
WHERE ID =' sl81ba3578' AND
  VALID AT 'Feb 07, 2003 17:00' AND
  PROBABILITY 70 percent ;
```

[QG-700] 서울81바3578가 현대백화점을 출발한 시간은?

```
SELECT BEGINOF
FROM Vehiclehistory
WHERE ID ='sl81ba3578' AND
  DEPART FROM '현대백화점'
```

[QG-710] 서울81바3578가 길동사거리에 도착한 시간은?

```
SELECT ENDOF
FROM Vehiclehistory
WHERE ID ='sl81ba3578' AND
  ARRIVE AT '현대백화점'
```

이동체 질의 처리 모형

1. 개념 모형

이동체는 형태에 따라서 이동점(moving point: MP)와 이동 영역(moving region: MR)으로 구분된다. 이동 차량은 이동 점으로 표현하며 점에 대한 이동체 데이터는 다음과 같이 데이터 형태를 정의한다(안윤애 등, 2002; Kim 등, 2002).

이동체는 형태에 따라서 이동점(moving point)와 이동 영역(moving region)으로 세분화 된다. 이것은 이동체에 대한 데이터 표현 복잡도를 의미하는데, 차량의 경우는 이동 점에 해당하며 다음과 같이 데이터 형태를 정의한다.

[정의1](이동체) 시간의 변화에 따라 객체의 위치 값만 변화되는 이동체를 말한다. 이동체 MP는 시간 속성, 공간 속성, 일반 속성을 가지며, $MP=\langle T_A, S_A, G_A\rangle$가 된다

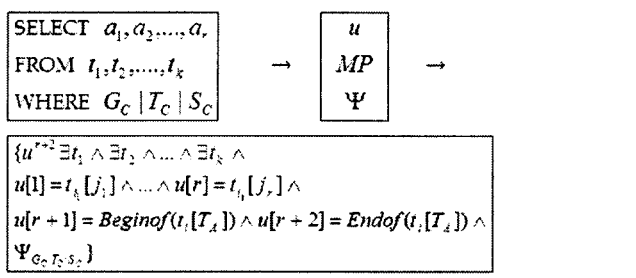
.

[정의2](시간 속성) MP의 시간 속성 $T_A=\langle vt_s, vt_e\rangle$로 구성되며 vt_s는 시작 시간, vt_e는 종료 시간을 나타낸다. 이 때 vt_s와 vt_e는 유효시간(Valid time)의 집합 S_{VT}의 원소가 된다. 유효시간은 실세계에서 발생된 시간을 나타내며, $S_{VT}=\{t_0, t_1, t_2, \cdots, t_k, \cdots, t_{now}\}$이고, 각 원소들은 $t_0<t_1<t_2\cdots<t_k<\cdots<t_{now}$ 의 순서를 가진다. $t_k=t_{k-1}+1$, $t_k=t_0+k$, $k \geq 0$인 정수로 정의된다. t_{now}는 현재 시간을 의미하는 시간 상수이다.

[정의3](공간 속성) MP의 공간 속성 $S_A=\langle x, y\rangle$, $x, y \in R$, R은 실수이다.

[정의4](이동체 데이터베이스) 이동체 데이터베이스를 구성하는 이동체들의 집합은 $S_{MP}=\{MP_0, MP_1, \cdots, MP_n\}$이다. S_{MP}로 구성된 이동체 데이터베이스의 이력 집합은 $H_{MP}=\{H_{MP_0}, H_{MP_1}, \cdots, H_{MP_n}\}$이다. S_{MP}에 속하는 각각의 MP_i에 대한 모든 이력 집합은 $H_{MP_i}=\{mp_{i_0}, mp_{i_1}\cdots, mp_{i_n}\}$이고 $mp_{i_k}=\langle T_A(mp_{i_k}), S_A(mp_{i_k}), G_A(mp_{i_k})\rangle$ 이다. 정의 4에서 mp_{i_k}는 MP_i의 k번째 이력 정보를 의미하고, $T_A(mp_{i_k})$는 MP_i의 k번째 시간 속성, $S_A(mp_{i_k})$는 k번째 공간 속성, $G_A(mp_{i_k})$는 k번째 일반 속성을 의미한다.

[정의5](이동체 질의) 사용자로부터 입력되어 이동체 데이터베이스에 접근하는 이동체 질의는 SQL2를 기반으로 확장된 MOQL 구문 형태로서, $Q(u, MP, \Psi)$와 같이 정의한다. 즉, 이동체 질의(Q) 중에서 SELECT 문은 SELECT절에 기술되는 속성절을 의미하는 u와 연산대상인 테이블을 기술하는 FROM 절에 대응하는 MP 및 시공간 조건을 포함하는 WHERE절에 대응하는 Ψ로 구성된다. 전형적인 이동체 질의(Q)문인 SELECT 문의 예는 다음과 같은 표현과 수식으로 정의된다.



[정의6](이동체 질의 트리) 이동체 질의(Q)는 [정의 5]에서 보인 바와 같이 크게 세 가지 절로 분해되며, 관계대수를 기반으로 확장한 이동체 관계대수로 변환할 수 있다. 즉, SELECT-FROM-WHERE절로 구성된 질의

문은 $(\tilde{\pi}_{a_1,\ldots,a_r}(\tilde{\sigma}_{a_C \gamma_C s_C}(t_1 \tilde{\otimes} t_2 \tilde{\otimes} \ldots \tilde{\otimes} t_k)))$과 같은 이동체 관계대수로 변환되며, 각각의 요소는 다시 Projection node, Selection node, Join node로 매핑된다.

2. 질의 처리 방안

이동체 엔진에 입력되는 이동체 질의는 기본적인 관계연산, 시간 연산, 공간 연산 및 시공간 연산자를 포함하고 있으며, 이들 세부 요소에 대한 개별적으로 구분하여 처리하는 방안을 사용한다.

이동체 질의문이 입력되면 기술된 연산자 유형에 따라서 다음과 같은 세부적인 질의 처리 과정이 진행된다.

- o 이동체 연산이 없는 전형적인 관계형 질의일 경우 부가적인 처리과정 없이 SQL 문을 기존의 상용 데이터베이스관리시스템에서 실행하여 결과를 반환한다.
- o 이동체 연산이 SELECT 절 또는 WEHERE절에 포함되어 있는 경우 다음의 세부 절차를 수행한다.
  - 해당 연산에 필요한 파라미터를 분석하기 위하여 2~3개의 SQL 질의문으로 분해 · 재구성한다.
  - 각각의 재구성된 질의문 중 설정된 질의처리전략을 근거로 임의의 질의처리과정을 수행하여 결과를 반환한다.
  - 반환된 결과는 재구성된 다른 질의문의 처리대상으로 사용한다.
  - 질의 재구성 시 순서는 WEHERE절에서 SELECT절로 진행하며, 최종적인 질의 결과는 SELECT절에서 기술된 이동체 연산을 적용하여 디스플레이한다.
- o 질의처리전략의 설정에 따라서 상기 중간 결과들은 버퍼 영역에서 별도의 과정을 통해 관리한다.

3. 다중 질의 처리안

사용자 또는 이동체 클라이언트로부터 다중의 이동체 질의가 동시에 입력되는 경우에는 어느 하나의 질의에 대한 실행 결과가 다른 질의에서도 일부 또는 전체가 재활용될 수 있다. 특히 이동체 질의유형 중에서 QG-100, QG-600 QG-700 등과 같이 특정한 차량에 대하여 시간 또는 공간 영역상의 특정 시점 또는 지점 검색을 제외한 나머지 시공간 범위 질의는 다중 질의 처리를 통해서 비용을 절감하고 시간을 단축할 수 있다. 예를 들어 [QG-300]의 질의 표현에서 보여진 바와 같은 "VALID FROM 'Feb 07, 2003 17:00' TO NOW AND BOUNDARY FROM '현대백화점' TO '길동사거리'" 구문은 질의 실행 결과로서 산출된 테이블에는 질의에서 기술한 차량ID이외의 다른 차량ID도 포함하게 된다.

따라서 동일하거나 유사한 시간 및 공간 조건을 내포한 또 다른 [QG-300] 질의에 대한 처리 비용과 시간은 이동체 데이터베이스에 대한 접근을 필요로 하지 않아 매우 적게 된다. 하지만 이동체 엔진 내에서는 이를 지원하기 위한 버퍼 관리기가 존재해야 하며, 버퍼 적중률을 높이기 위한 전략과 절차가 부가적으로 요구된다.

시스템 설계 및 구현

1. 이동체 질의 처리기

이동체 질의 처리기는 사용자 질의에 대한 표준화 구문 정의를 지원함으로써 시스템간 호환성을 증대시켜주며, 내부 처리관점에서 정형화된 세부 명세로 분해 및 변환함으로써 그 처리비용을 감소시킬 수 있는 장점을 가진다. 즉, 사용자 질의로부터 임의의 순서로 개별적인 함수/모듈을 호출하는 경우 중복된 함수의 호출과 메모리 사용공간의 불필요한 할당 등과 같은 문제점들을 해결할 수 있다. 이와 같은 특징을 제공하는 이동체 질의 처리기는 데이터베이스 표

```
MOQP_CodeGenerator()
Input : Statement, Roots of tree, Node identifier, Tuple variables
Output : Execution tree
{
   Extract tuple variables from input statement;
   Count a number of tuple variable;
   if (total number of tuple equal to 1) {
      Build ExistingNode and then attach it to tree as a root;
      Build SelectNode and then attach it to ExistingNode as a child;
   }
   else {
      Build CartesianSubtree and then attach it to tree as a root;
      Build SelectNode and then attach it to ExistingNode as a child;
   }
   Build TemporalNode and then attach it to SelectNode as a child;
   Build SpatialNode and then attach it to TemporalNode as a child;
   Build ProjectNode and then attach it to ValidNode as a child;

   if (type of input statement is equal to Select statement)
     Build DisplayNode and then attach it to ProjectNode as a child;
   else
     Build StoreNode and then attach it to ProjectNode as a child;
}
```

FIGURE 2. Moving object execution tree generate algorithm

준 질의어인 SQL을 기반으로 시공간 연산구문을 추가/확장한 MOQL을 입력받아 다음과 같은 처리과정을 수행한다.

1) 이동체 파싱 및 의미 분석

MOQL로 작성된 사용자 질의가 입력되면 토큰단위로 분해한 후 구문분석을 통해 문법의 적합성을 검사한다. 또한 의미분석을 통해 질의타입을 결정하고 NOW와 같은 추상적인 값을 질의 시점의 시간 값과 같은 실제적인 값으로 치환한다. WHERE절에 두 개 이상의 구가 기술된 경우 동등 수식을 통해 불필요한 구문을 삭제한다.

2) 이동체 실행 트리

입력된 MOQL질의문은 SELECT-FROM-WHERE 형식을 가지는데 각각의 절들은 내부적으로 DISPLAY노드, PROJECT 노드, SELECT 노드, SPATIAL 노드, 그리고 TEMPORAL 노드로 변환된다. 기본적인 이동체 질의 트리는 PROJECT 노드, SELECT 노드, SPATIAL 노드, 그리고 TEMPORAL 노드 구조를 가지며, 세부적인 질의 구문에 따라서 각 노드의 위치가 변경된다. 이동체 질의 트리상의 각 노드의 순서 조정은 실제 기술된 조건 값과 내부 처리 버퍼의 인스턴스에 의해 결정된다.

생성된 이동체 실행 트리의 루트 노드로부터 잎 노드를 구성하는 각 노드에 대하여 처리 영역의구분(메모리/디스크)과 대상이 되는 이동체 개수를 추정하여 비용을 측정하고 동등 수식 등의 질의 최적화 기법을 적용하여 최종적인 이동체 실행 트리를 생성한다.

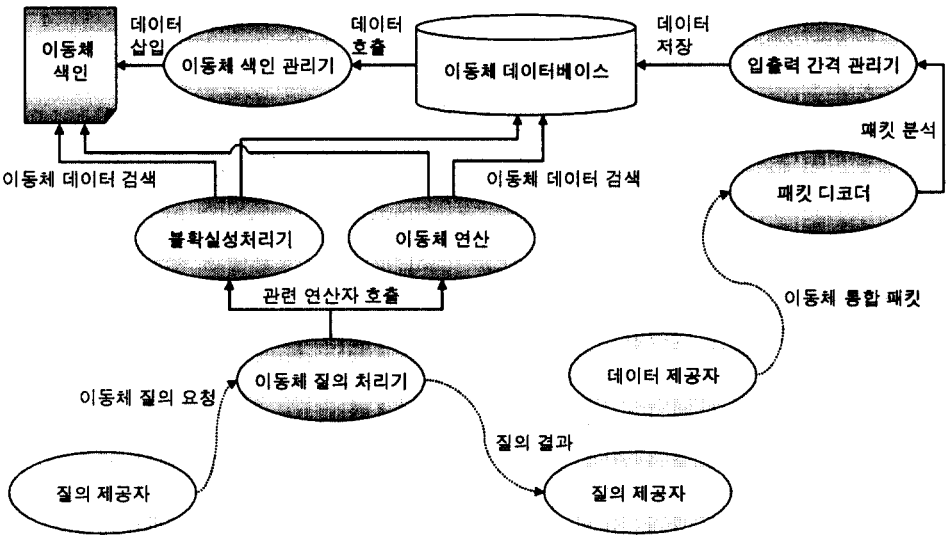


FIGURE 3. DFD in MOQPs

3) 이동체 실행 트리 실행

생성된 이동체 실행 트리의 루트 노드로부터 잎 노드 방향으로 진행하면서 초기화-갱신-전파 단계 등의 재귀적인 형태의 세부 절차를 수행하며, 각 노드의 결과는 다음 노드로 전달된다.

각 노드의 초기화 단계에서는 해당 노드에서 사용될 연산자의 종류와 연산 대상인 이동체 데이터에 대한 포인터를 결정한다. 갱신 단계에서는 지정된 이동체 연산자를 처리 영역에서 실행하고 그 결과를 버퍼영역에 저장한다. 전파 단계에서는 포인터 등을 포함하는 정보를 다음 노드의 초기화 단계에게 전달함으로써 해당 노드에서 처리되어 생성된 결과를 다음 노드에게 전달한다. 이동체 실행 트리를 대상으로 입력된 이동체 질의에 대응하는 이동체 연산이 실행되면, 그 최종 결과와 질의 정보는 이동체 버퍼에 저장되어 관리된다. 이때 입력 질의에서 기술된 시간 조건과 공간 조건으로 구성된 이동체 실행 트리 및 그 결과에 대한 재사용을 제공하여 질의 처리비용의 계산과 성능 향상을 기대할 수 있다.

2. 환경 및 실행화면

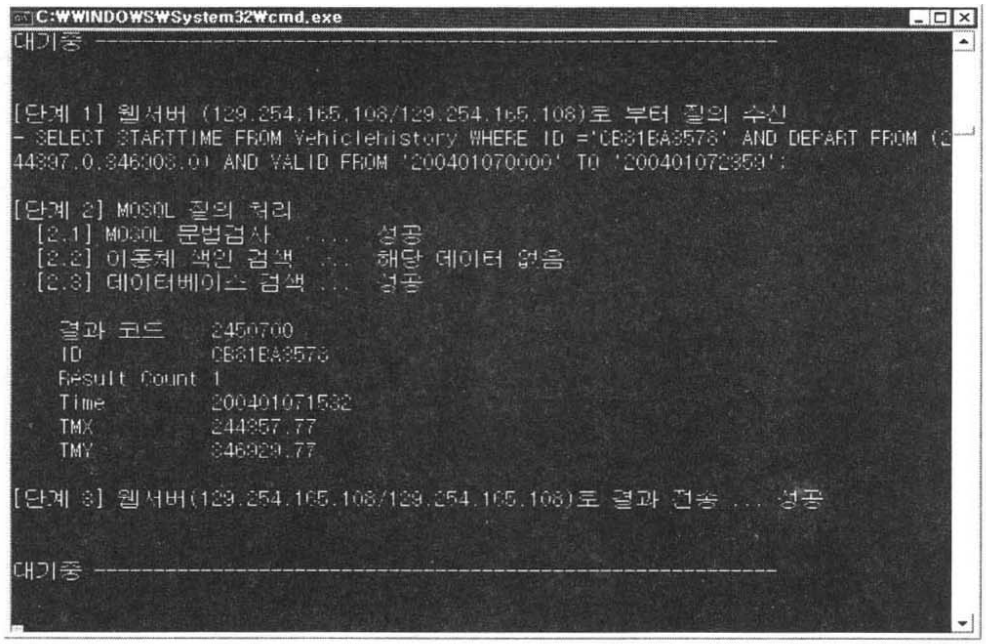
이동체 질의처리기는 MS Windows XP에서 Java로 작성되었으며 JDK 1.4.2를 기반으로 작동된다.

이동체 질의처리기는 이동체 관리 시스템을 구성하는 서브 시스템으로서 상위계층으로부터 입력되어 그림 3에서 보여진 바와 같이 이동체 색인과 이동체 불확실성 처리루틴을 포함하는 일련의 처리가 진행되며 결과인 이동체 위치/궤적 및 전자지도 정보는 계층간 정의된 패킷/스트림 형식으로 반환한다.

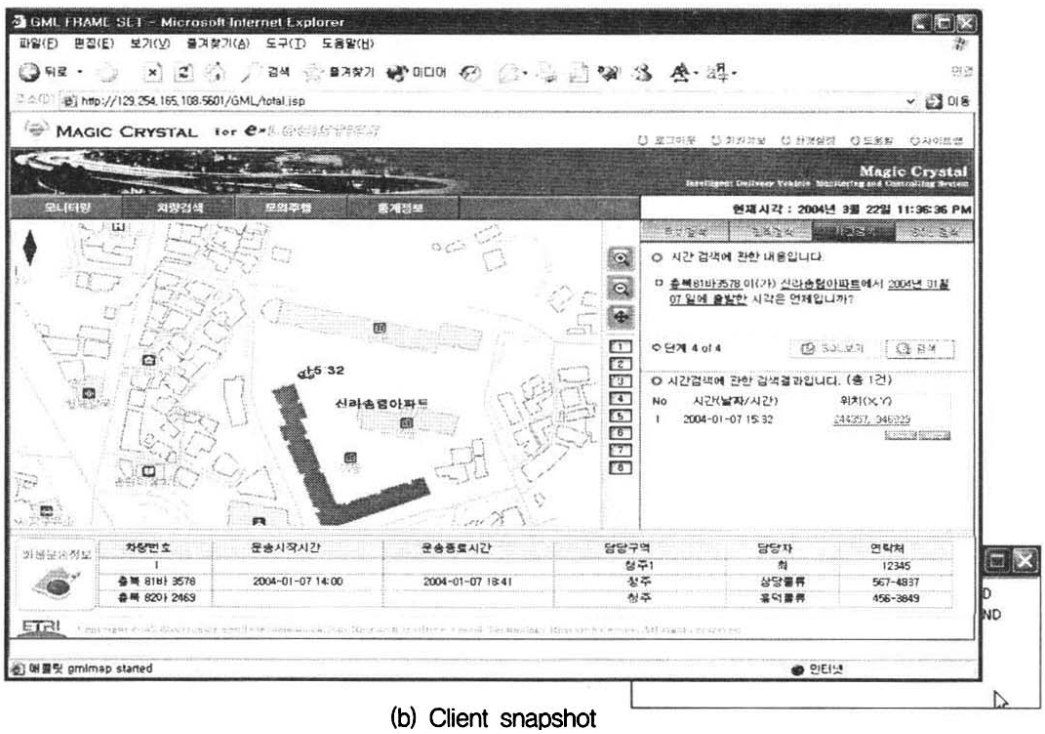
그림 4는 구현된 이동체 관리 시스템 서버의 질의처리기와 클라이언트의 실행화면을 보여준다.

3. 고찰

전통적인 차량관제시스템에서는 차량의 위치 검색과 같은 단순한 형태의 질의만을 지원하며, 개별 응용중심의 메뉴접근방식으로 기능 변경시 개발된 연산자의 활용도가 저하되기 때문에 물류에서의 전문가적 관리를 위한 기능이 미흡하였다.



(a) MOQP shapshot



(b) Client snapshot

FIGURE 4. MOEngine runtime snapshot

본 논문에서는 사용자에게 친숙한 표준 데이터베이스 질의어인 SQL을 기반으로 이동체 위치 및 궤적 검색기능을 제공하는 MOQL 및 처리기를 제시하였다. 그림 4에서 보여진 바와 같이 물류환경에서 요구되는 복잡한 차량 위치 질의를 마법사 형식으로 입력받아 SQL기반의 질의로 변환하여 시스템에서 처리하거나 직접 SQL기반의 질의를 입력받아 처리함으로써 기존의 시스템과의 연동에도 편리함을 알 수 있다. 실험에서는 제안된 질의처리기가 초당 3개 클라이언트로부터 동시에 20여 개의 MOQL 질의를 입력받아 1분 내에 처리 후 반환하는 성능을 보였으며, 결과적으로 초당 1개의 질의처리 성능을 보인다. 이러한 결과는 입력된 질의에 대한 처리과정에서 이동체 위치 데이터의 수집과 저장 및 색인, GML변환 및 프리젠테이션 등과 같은 처리과정과 직접적인 관련이 있다.

결 론

인터넷을 기반으로 광역화된 물류체계를 의미하는 e-로지스틱스 환경에서 효율적인 차량관제를 위해서는 차량의 위치에 관련된 복잡한 유형의 질의와 연산의 지원이 필수적이다. 이 논문에서는 물류응용을 기반으로 기존의 데이터베이스 표준 질의어인 SQL을 확장한 MOQL을 정의하고, 이를 처리하는 처리기의 설계를 설명하였다. 세부적으로 지금까지 개발되어온 이동체 엔진의 세부 구성요소를 정리하고 질의처리기의 구조와 처리 기법 및 구현사항을 설명하였다. 향후 연구에서는 본 논문에서 기술된 이동체 질의를 대상으로 물류 차량관제에 특정화된 기능을 지속적으로 추가할 예정이다. KAGIS

참고문헌

김동호, 이현아, 이혜진, 김진석. 2003. 통합 이동체 위치 데이터 인터페이스 모델. 정보처리학회 추계학술대회논문집 10(2):631-634.

안윤애, 김동호, 류근호. 2002. 차량 위치 추적을 위한 이동 객체 관리 시스템의 설계. 정보처리학회논문지 9-D(5):827-836.

이현아, 이혜진, 김동호, 김진석. 2003. 이동체 관리 시스템을 위한 이동체 질의어 설계. 정보과학회 추계학술대회 논문집 30(2-2):148-150.

이혜진, 이현아, 김동호, 김진석. 2003. GML기반 통합 맵서버 설계 및 구현. 한국지리정보학회지 6(4):71-84.

정원일, 배해영. 2004. GML 기반의 이동객체 데이터모델 및 질의어 명세. 정보처리학회논문지 11-D(1):39-50.

R. Guting, M.. Bohlen, M. Erwig, C. Jensen, N. Lorentzos, M. Schneider and M. Vazirgiannis. 2000. A Foundation for Representing and Querying Moving Objects. ACM Transactions on Database Systems 25(1):1-42.

S. Grumbach, P. Rigaux, M. Scholl and L. Segoufin. 1999. The Design and Implementation of DEDALE.

D. H. Kim, H. A. Lee, J. S. Kim, Y. A. Ahn and K. H. Ryu. 2002. Moving Objects Relational Model and Design for e-Logistics Applications. Proceedings of International Conference on Information Technology & Applications ICITA-2002.

S. S. Park, Y. A. Ahn and K. H. Ryu. 2001. "Moving Objects Spatiotemporal Reasoning Model for Battlefield Analysis. Proceedings of Military, Government and Aerospace Simulation part of ASTC'01. pp.108-113.

O. Wolfson, P. Sistla, B. Xu, J. Zhou, S. Chamberlain, N. Rishe and Y. Yesha. 1999. Tracking Moving Objects Using Database Technology in DOMINO. Proceeding of The 4th Workshop on Next Generation Information Technologies and Systems. pp. 112-119. KAGIS