차세대 U-City통합센터 U-Service 개발에 대한 연구

권창희*, 이주상**

A Study for Development the U-City Multiplex Center's U-Service Technology

Chang-Hee Kwon*, Joo-Sang Lee**

요 약 본 논문은 유비쿼터스 환경의 도시통합관제를 위한 실시간 데이터 처리 방안을 제시한다. 도시내 다양한 정보수집센서로부터 발생하는 데이터와 이벤트에 대해 분석하고 가공하여, 실시간으로 처리하는 것이 도시통합관제의 핵심이다. 본 논문은 다양하게 발생하는 이벤트 정보를 분석하고, 상황에 맞는 조치를 취하기 위한 판단모듈에 Rete 알고리즘을 이용한 추론엔진을 적용한다. 마지막으로, 본 논문에서 제안한 방법의 효율성을 평가하기 위해 시뮬레이터를 구성하여 성능을 테스트한다.

주제어 : 유비쿼터스, 도시통합관제, 추론엔진, 실시간데이터처리, 미들웨어

Abstract The ultimate purpose of cities is 'the enhancement of quality of life in city'. In order to achieve this, it is required to optimize the Ontology-driven ubiquitous services included in the U-City and UIS of regions and communities and so on. Almost U-Service's contents are related to spatial or temporal extent. So it is important to design Spatio-temporal event schema for efficient access to U-City. There is a rapid change into an urban society in the shape of Ontology-driven ubiquitous services in which a content on a region or an incident is newly reconstructed through various ideas with the influence of the ubiquitous environment.

Key Words : Ubiquitous, Integration Control, Inference Engine, Real-Time Data Processor, Middleware

1. 서 론

지능화된 도시의 새로운 패러다임인 U-City는 건설 기반에 첨단 정보통신 인프라와 U-IT 서비스를 도시공간에 융복합하여 도시의 제반 기능을 혁신시킬 수 있는 21세기 정보통신 융합 도시를 의미한다.

U-City는 지금까지 도시의 각각 구성요소를 대상으로 추진해 왔던 정보화의 개념을 확대시켜, 도시 전체를 대상으로 시설물에서부터 교통, 행정, 지불, 가정, 의료 등 전반에 첨단 IT기술을 적용함으로써 각 시스템이 도시통합관제센터를 통해 유기적으로 연결되고 시민들에게 서비스가 제공되는 지능화된 도시를 말한다. 이미 어느 정도 구축되어 있는 전자정부, 지리정보시스템(GIS), 지능형교통시스템(ITS), 시설물관리시스템 등 기존 시스템에다 전자태그(RFID), IPv6, 유비쿼터스센서네트워크(USN), 광대역통합망(BcN) 등 최신 IT기술을 결합함으로써 시민중심 컨버전스 서비스가 구현되는 것이다.[1]

U-City는 도시민들이 원하는 첨단 정보통신 인프라를 도시 구현 초기 단계에 반영함으로써 소요시간 단축, 투자비 감소 및 유비쿼터스 정보서비스의 제공에 의한 도시가치의 증대 등을 주요 목표로 한다.

우리나라의 U-City 추진은 정보통신산업 및 도시건설과 관련된 산업들과의 컨버전스에 의한 도시의 새로운 가치창출을 통해 국가경쟁력 향상이라는 신도시 건설의 새로운 국가 비전을 제시할 수 있다.

본 연구논문은 한세대학교 교내연구 과제로 수행한 논문임.
*한세대학교 IT학부 / 유시티IT융합 도시정책학과 조교수(교신저자)
**이에스이(주) 부설연구소 연구소장
논문접수: 2012년 8월 10일, 1차 수정을 거쳐, 심사완료: 2012년 8월 22일

U-City는 전자정부와 광대역통신망, GIS, RFID, IPv6 등이 기반 인프라가 되고, 그 위에 복지·교육· 환경·안전·교통·레저 등 다양한 서비스가 유비쿼터스 기술을 통해 구현될 것인데, 특히 도시 전체를 모니터링하고, 다양한 매체를 통해 정보서비스를 제공하기 위한 도시통합관제센터가 U-City의 핵심으로 대두되고 있다.[2]

U-City는 '도시통합관제센터'에서 도시의 모든 기능을 통합 관리하게 된다. 교통정보는 물론이고 교통사고가 났을 때 신속하게 경찰, 병원, 구조대, 등이 출동할 수 있도록 자동화되고, 도시의 지하시설과 지상시설 등의 제반관리도 통합 관리할 수 있다.

도시통합관제센터는 U-City내 통신망, 교통망, 시설물 등으로부터 현장정보를 수신하고 이를 종합적으로 분석, 이를 거주민이나 관련 기관에 실시간으로 제공해야 한다. 이를 위해서는 다양한 매체를 통합하고, 매체간에 발생하는 방대한 양의 데이터를 실시간으로 가공·처리할 수 있는 실시간 데이터 처리 미들웨어의 필요성이 대두되고 있다.[3]

하지만, 현재 도시통합관제센터의 구축시에 이기종 단말장치간의 정보 교환을 위한 인터페이스 표준안이 없어서 개발자들간에 인터페이스 협의를 위해 소요되는 시간이 길고, 이로 인해 시스템 구축 기간이 늘어나는 실정이다. 또한, 방대한 데이터 처리를 위해 모든 데이터는 DBMS에 저장하고, DBMS를 통해서 공유하여 사용하므로 과다한 DB트랜잭션이 발생하여, 시스템의 안정화가 어려운 실정이다.

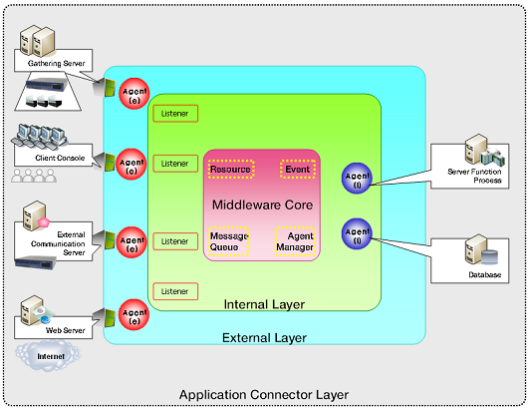
실시간 데이터 처리 미들웨어란 일반적으로 도시통합관제를 위한 다양한 디지털 매체들 간에 데이터를 주고받을 수 있는 통로를 제공함과 동시에 데이터를 실시간으로 분석하여 가공하고 처리할 수 있는 서비스 처리 로직을 제공함으로써 지능화된 커뮤니케이션과 실시간 데이터 처리가 가능하도록 하는 도시통합관제의 핵심이다.

본 논문에서는 수많은 이기종 매체간의 안정적인 통신방식을 지원하고, 이때 발생하는 이벤트와 데이터를 실시간으로 가공·처리하는 모듈에 Rete 알고리즘을 이용한 추론엔진을 적용하여, 도시통합관제를 위한 실시간정보서비스를 제공하는 실시간 데이터 처리 미들웨어를 제시한다.

본 논문의 구성은 다음과 같다. 2장에서는 도시통합관제를 위한 실시간 데이터 처리 미들웨어에 대한 전체 구조를 소개하였고, 3장에서는 실시간 데이터 처리 미들웨어 설계 및 구현 방안에 대해 소개하였으며, 4장에서는 본 논문에서 활용하고자 하는 추론엔진을 이용한 상황 인지에 관하여 소개하였으며, 5장에서는 이러한 실시간 데이터 처리 미들웨어 아키텍처를 바탕으로 실제 실험을 통하여 실험 결과를 고찰하였다. 끝으로 6장에서는 결론으로 실시간 데이터 처리 미들웨어의 필요성 및 성능에 관한 검토와 향후 연구 방향에 대해서 제시하였다.

2. 실시간 데이터 처리 미들웨어 전체 구조

본 논문은 국내외에서 진행 중인 도시통합관제의 표준화 내용을 최대한 수용하면서 통합관제에서의 이기종 어플리케이션 장치간 통합 인터페이스를 지원하는 Connector 개발과 관제 시 발생하는 대규모 데이터와 이벤트에 대한 실시간 처리를 수행하는 미들웨어 CORE를 구현하는 것이다. 또한, 개발자들이 손쉽게 개발할 수 있도록 Rule 기반의 아키텍처를 구현하는 것이다. 그림1은 본 논문에서 구현한 실시간 데이터 처리 미들웨어의 계층 구성도이다.



[그림 1] 미들웨어 계층 구성도

실시간 데이터 처리 미들웨어는 Core를 중심으로 Application Connector Layer, External Layer, Internal Layer 3가지 계층으로 구성된다.

Application Connector Layer는 일반적인 응용프로그

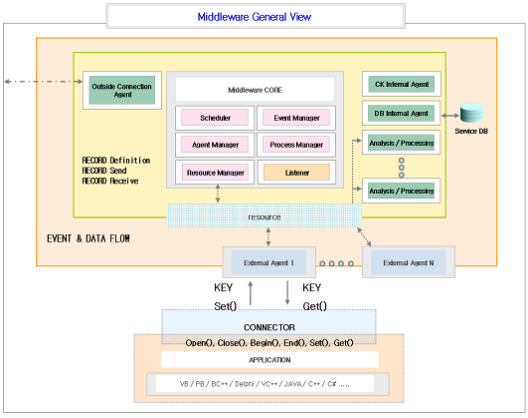
램을 개발하는 환경을 지원한다. 시스템과의 종속성을 배제한 자유로운 개발과 응용을 보장하면서 미들웨어와의 연계를 목적으로 한다. 응용프로그램에 따라 간단한 절차만으로 미들웨어와의 연계를 보장받을 수 있다.

External Layer는 외부의 요구를 받아서 수행하고 외부와의 통신 안정성과 업무를 명시할 수 있는 작업에이전트 기능을 수행한다. Connector를 통한 외부와의 인터페이스를 보장하며, 리스너(Listener)를 통하여 내부 계층으로의 연계를 수행하는 외부연계 에이전트이다.

Internal Layer는 미들웨어 코어를 중심으로 내부의 핵심 모듈로써 에이전트의 생성과 소멸 등의 처리가 이루어진다. 또한, 미들웨어 내부와 외부와의 인터페이스를 위한 핵심적인 역할을 수행한다. 구성은 리스너(Listener)와 내부 에이전트와 특수한 목적으로 수행되는 HSA(Half System Agent)와 시스템의 특성을 결정짓는 SA(System Agent)로 구성된다.

미들웨어 코어는 자원관리, 이벤트관리, 메시지관리, 프로세스관리, 에이전트 관리 등 미들웨어내 모든 에이전트의 생성부터 소멸까지의 라이프사이클을 관리하며 시스템의 모든 모듈간의 통신 및 자원분배 등 전체적인 기능을 유지관리한다.

[그림 2]는 실시간 데이터 처리 미들웨어의 전체 구성도이다.



[그림 2] 미들웨어 전체 구성도

모든 Application 서버의 OS(Windows, Linux, Unix 등)와 개발언어(C++, VB, PB, Delphi 등)에 적용할 수 있는 다양한 환경의 Connector를 통해 데이터를 통합할 수 있어야 하고, 통합된 데이터를 미들웨어 내부에서 사용할 수 있도록 표준화된 포맷으로 변경하여야 한다. 미들웨어 CORE에서는 대규모 데이터 요청에 대해 자동으로 부하조절과 리소스 관리를 통한 실시간 데이터 공유가 되어야 한다. 또한, 병렬 메시징 구조의 멀티 프로세스 구현으로 대량의 트랜잭션 처리 기능과 스케줄링, Broadcast, 다중세션관리, 세션간 Load Balancing 기능이 지원되어야 한다.

통합관제를 위한 미들웨어는 서비스 추가에 대한 유연한 확장성을 가지고 있어야 한다. 신규 서비스가 추가 될 때마다 내부 로직을 변경해야 하는 작업을 하지 않도록 아키텍처를 설계해야 한다. 본 논문에서는 유연한 확장구조를 가질 수 있도록 관제 업무 처리 서비스 간의 연동 로직과 입출력 전문처리를 프로그램에서 분리하고, Rule & Parameter 기반으로 미들웨어에서 서비스 로직을 구현할 수 있도록 설계하였다.

본 논문에서는 미들웨어의 프로세스 상태에 대해 실시간으로 모니터링 할 수 있는 기능과 개발자들이 손쉽게 개발할 수 있도록 환경설정파일 및 Debug, 모듈 Maker 기능과 서비스 모듈에 대해 손쉽게 테스트할 수 있는 시뮬레이션 기능을 구현하였다.

3. 실시간 데이터 처리 미들웨어 설계 및 구현 방안

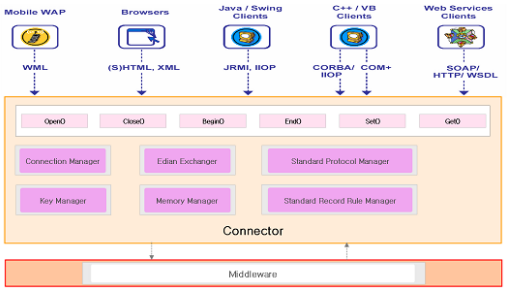
통합 관제 환경에서 실시간으로 데이터를 처리하기 위한 미들웨어를 설계하기 위해서는 미들웨어가 가져야 할 기능에 대한 분석이 먼저 이루어져야 한다.

먼저, 실시간 데이터 처리 미들웨어는 통합 관제 환경에서 이기종 어플리케이션 장치 간 통합 인터페이스를 지원하는 Connector 기능이 있어야 한다. 도시에서 다양한 센서로 부터 올라오는 정보 데이터를 수집하는 여러 장치들 간의 통합된 인터페이스를 위해 하나의 채널을 구성해야 한다. 또한, 관제를 위한 미들웨어는 실시간으로 데이터를 처리해야 하므로, 오류 또는 장애에 의해 일부 모듈이 정지하거나, DBMS가 Down되어도 미들웨어 CORE는 정상적으로 동작되어야 한다. 그리고, 대용량 데이터의 멀티캐스팅을 위한 프로세스 기능과 멀티캐스트를 통해 다양한 데이터의 송수신이 동시에 이루어져야 한다. 즉 여러 종류의 데이터를 공유메모리를 사용하여 동시에 송수신 할 수 있어야 한다. 이외에도 보안을 위한 기능과 신규 서비스 로직이 유연하

게 추가될 수 있는 기능이 요구된다.[4]

3.1 Connector

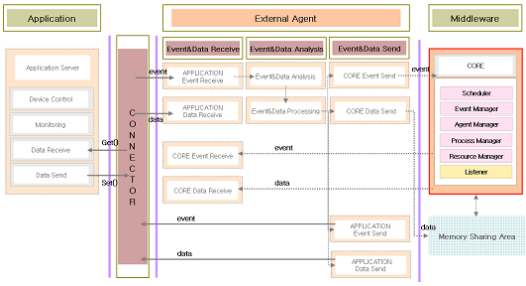
미들웨어와 이기종 어플리케이션의 통신을 위해 어플리케이션의 OS 및 개발언어의 종류에 상관없이 적용될 수 있으며, 기 개발된 어플리케이션들의 통합이 용이하도록 표준화된 인터페이스를 지원한다. 또한, 표준화된 통신방식을 위해 표준함수를 제공하고, 통신상의 오류제어와 흐름제어를 지원하여 보안유지 및 통신안정성을 확보할 수 있다. Connector의 구성도는 다음 [그림 3]에서 보는 바와 같다.



[그림 3] Connector 구성도

3.2 External Agent

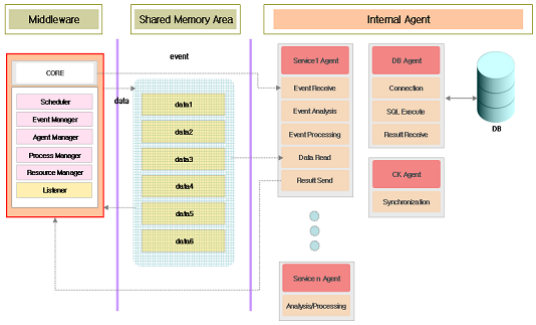
어플리케이션 서버와 미들웨어 사이에 이벤트와 데이터를 연동하는 역할을 수행하며, 이벤트&데이터를 분석하여 어플리케이션 서버로 리턴하거나 미들웨어 내부로 전송한다. 또한, 어플리케이션 서버의 상태를 체크하여 미들웨어 CORE에서 어플리케이션 서버를 관리할 수 있는 기능을 지원하고, 어플리케이션 서버와의 연결이 끊기면 자동적으로 소멸된다. External Agent 기능 흐름도는 다음 [그림 4]에서 보는 바와 같다.



[그림 4] External Agent 기능 흐름도

3.3 Internal Agent

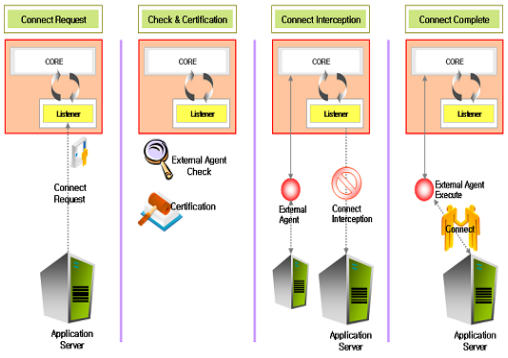
하나의 서비스 로직을 수행할 수 있는 모듈이다. 각각의 내부 에이전트는 미들웨어 Core로부터 독립적으로 실행되며, Core로부터 이벤트를 받고, 해당 이벤트를 분석하여 처리한다. 내부 에이전트는 모든 데이터를 공유메모리에서 읽어서 처리한다. 내부 에이전트는 비즈니스 서비스 로직을 담당하는 서비스 에이전트와 내부 시스템 Clock을 담당하는 CK 에이전트와 DB와의 연동을 위한 DB 에이전트와 외부의 다른 미들웨어와의 연계를 위한 외부연계 에이전트로 구성된다. 비즈니스 서비스 로직은 업무 목적과 특성에 따라 유연하게 확장할 수 있다. 미들웨어 내부에서 실질적으로 데이터와 이벤트를 분석, 판단, 처리하는 역할을 한다. Internal Agent 기능 흐름도는 다음 [그림 5]에서 보는 바와 같다.



[그림 5] Internal Agent 기능 흐름도

3.4 Listener

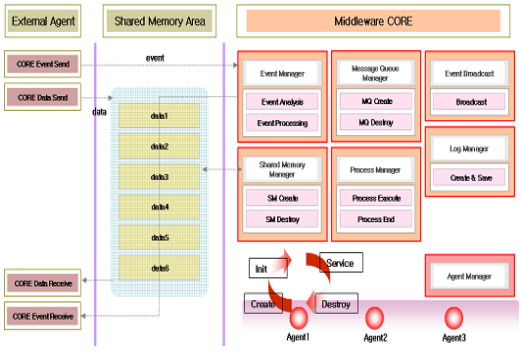
어플리케이션 서버에서 미들웨어와 연결할 때 최초로 만나는 모듈로써, 어플리케이션 서버로부터 접속요청을 받는다. 리스너는 현재 접속하고자 하는 어플리케이션 서버가 인증된 것인지 확인절차를 수행하고, 이미 External Agent가 실행되어 연결되어 있는지 확인하여, 이미 연결되어 있거나, 인증되지 않은 것이라면 연결을 차단한다. 즉, 어플리케이션 서버가 인증된 것이고, 연결되어 있지 않다면 External Agent를 실행시키고 연결하여, 미들웨어와의 연계를 할 수 있도록 지원한다. Listener 기능 흐름도는 다음 [그림 6]에서 보는 바와 같다.



[그림 6] Listener 기능 흐름도

3.5 Core

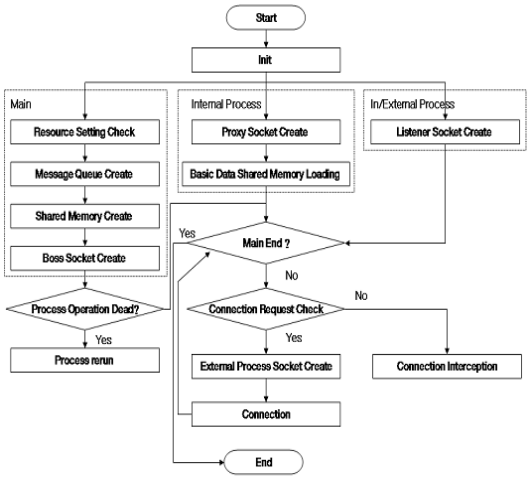
미들웨어 코어는 미들웨어의 핵심 모듈로써, 미들웨어 내부의 모든 이벤트, 메시지, 데이터, 메모리 자원을 총체적으로 제어·관리하며, 에이전트의 생성, 생성 수 초기화, 서비스 실행, 소멸에 관한 모든 권한을 가지면서 에이전트의 생명주기를 관리한다. 또한 생성된 에이전트간에 연동을 주관하고, 모든 프로세스를 관리한다. 미들웨어 코어는 특정 상황에 대한 이벤트를 분석·처리하여 그 결과 지시에 대한 이벤트를 내부의 모든 에이전트에게 동시에 전파하고, 미들웨어 내부의 데이터는 공유메모리영역을 통하여 코어와 모든 에이전트가 동시에 사용하여, 특정 이벤트가 발생했을 경우, 해당 이벤트와 관련된 데이터를 동시에 읽는다. 본 논문에서는 코어의 이벤트 분석을 위해 신경회로망을 적용하여 실시간으로 상황인지를 하도록 구현하였다. 미들웨어 코어의 기능 흐름도는 다음 [그림 7]에서 보는 바와 같다.



[그림 7] Core 기능 흐름도

3.6 미들웨어 동작 순서도

본 연구에서는 server의 역할을 하고 있는 미들웨어를 다중 네트워크 연결이 지원되도록 구현하여 여러 클라이언트와 연동할 수 있도록 하였다. 다음의 그림 8의 순서도는 이러한 서버측 구현과정을 나타내고 있다. 그림 8에서 보는 바와 같이 미들웨어는 리스너와 외부에이전트를 통해 클라이언트에서 접속 요청을 할 때마다, 먼저 리스너가 접속요청을 접수하고 이미 접속되어 있는지, 아니면 허가된 접속인지를 검사하여 인증이 되면, 코어에 알려서 클라이언트와의 통신을 담당하는 외부에이전트를 생성하여 클라이언트와 통신 연결을 한다. 그리고, 여러 개의 클라이언트가 미들웨어와 각각 개별적으로 통신할 수 있도록 클라이언트마다 각기 다른 외부에이전트를 생성시켰다.[5][6]



[그림 8] 미들웨어 동작 순서도

4. 미들웨어 상황인지 모듈 구현

4.1 상황인지 영향변수

본 논문은 기존 도시에서 발생하던 상황정보 자료를 토대로 통합관제 상황인지를 위한 시나리오를 구성하고, 모든 상황에서 발생할 수 있는 이벤트 정보를 코드로 정의하였다. 상황인지를 위한 연동코드는 표 1과 같다.

〈표 1〉 상황인지 이벤트 코드

Name	Code	Description
서비스 ID	S01	방범/방재 서비스 ID
	S02	도로/교통 서비스 ID
	S03	환경 감시 서비스 ID
	S04	공공 시설 서비스 ID
	S05	U-GIS 서비스 ID
이벤트 코드	EC0	침수 이벤트
	EC1	누전 이벤트
	EC2	미세먼지 발생 코드
	EC3	돌발상황 이벤트 발생
	EC4	비상도움 발생
	EC5	화재 감시 이벤트 코드
이벤트 레벨	EC01	Level 1 - 반경 1m 미만 침수
	EC02	Level 2 - 반경 1m 이상 침수
	EC11	Level 1 - 반경 1m 미만 침수 지역 누전
	EC12	Level 2 - 반경 1m 이상 침수 지역 누전
	EC13	Level 3 - 우천시 누전
	EC20	Level 1 - 먼지
	EC21	Level 2 - 황사
	EC22	Level 3 - 유독성 가스
	EC30	Level 1 - 접촉사고, 시위
	EC31	Level 2 - 차량 3대 미만 추돌 사고
	EC32	Level 3 - 대형사고
	EC40	Level 1- 위험
	EC41	Level 2 - 폭력 및 인상 사고
	EC50	Level 1 - 소규모 화재
	EC51	Level 2 - 중간규모 화재
	EC52	Level 3 - 대규모 화재(주요시설, 인구 밀집 지역, 위험물 저장소)
	EC01	침수 경고
	EC02	침수지역 정보
	EC11	누전 발생 경고
	EC12	누전 지역 정보
	EC13	누전 상태 정보
	EC20	먼지 발생 정보
	EC21	먼지 발생 지역 정보
	EC22	먼지 확산 정보
	EC30	돌발상황 정보
	EC31	교통소통 정보
	EC32	BIS 정보
	EC40	비상도움 경고
	EC41	비상도움 위치 정보
	EC50	화재발생 경고
	EC51	화재발생 위치정보
	EC52	화재 규모 정보
신호 상태 코드	0	비정상
	1	녹색 직진
	2	주황 좌회전
	3	빨강 녹색 직좌
	4	주황
	5	빨강

4.2 추론엔진을 이용한 상황인지 모형

추론엔진은 Rete알고리즘과 Leaps알고리즘의 장점을 결합한 Drools 엔진을 이용하여 U-City 이벤트 분석을 수행할 수 있도록 구현하였다. Rule Engine의 핵심은 추론 엔진(inference engine)이다. 추론 엔진의 역할은 Fact와 Rule을 패턴 매칭을 사용하여 분석한 뒤, condition이 true인 것으로 판명되는 Rule의 Action들을 수행하도록 한다. 이러한 패턴 매칭 작업은 객체 지향 개념을 사용하여 확장된 RETE 알고리즘에 의해 수행된다.

추론엔진의 룰 기반 매칭을 위한 입력변수는 도시통합관제를 위한 각 서비스를 구분하는 서비스ID, 발생한 상황에 대한 이벤트 정보를 구분하기 위한 이벤트 코드, 상황의 심각도를 구분하는 이벤트 레벨, 상황에 대한 내역을 구분하기 위한 이벤트 메시지 코드, 상황이 발생한 지역내의 교통 신호 상태를 알려주는 신호상태 코드가 사용되었다.

다양한 이벤트에 대한 분석을 위해 AND, OR, ANY, NOT, SEQUENCE, APERIODIC, PERIODIC 등 7가지의 연산을 수행할 수 있는 룰 분석 기능을 구현하였다.

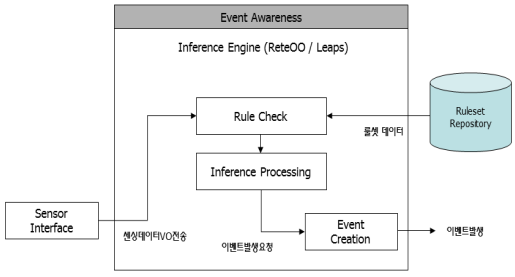
5. 실험 및 결과

5.1 실험환경

본 연구에서 실시간 데이터 처리 미들웨어를 구현하기 위해 사용한 개발PC의 운영체제는 Window7환경이고, 미들웨어 개발을 위해 JAVA언어를 사용하였고, 개발툴은 이클립스3.7.2, WAS는 Tomcat6, DBMS는 오라클10g, 개발서버 OS는 unix, 개발서버 장비는 IBM x3650M3를 사용하였다.

5.2 실험 및 결과

본 연구에서 실시간 데이터 처리 미들웨어의 성능테스트를 위해 가상의 성능부하 시험모듈을 구성하여 테스트를 실시하였다. 시험은 일정시간 동안 트랜잭션을 발생시켜 응답시간 및 서버 자원을 모니터링하여 수행하였다. 평상시 및 부하시험은 60분간 실시하였고, Endurance 시험은 24시간 부하를 지속적으로 발생시켰다.



[그림 9] 상황인지를 위한 추론엔진의 구조

1) 시험내용 및 항목

① 평상시 시험

► 시험항목수 : 17개
► 시험데이터 : 1set
► 사용자 수 : 시험이 가능한 모듈에 대한 Call수를 산정하여 평상시 TPS산정(0.24TPS)
► 부하발생방법 : 부하발생을 위한 성능부하시험 모듈 사용
► 시험방법 : TPS를 기준으로 60분간 지속적으로 트랜잭션 발생

② 시스템 부하 시험

► 시험항목수 : 17개
► 시험데이터 : 1set
► 사용자 수 : 평상시 부하의 5배 기준으로 TPS산정(1.16TPS)
► 부하발생방법 : 부하발생을 위한 성능부하시험 모듈 사용
► 시험방법 : TPS를 기준으로 60분간 지속적으로 트랜잭션 발생

③ Endurance 시험

► 시험항목수 : 17개
► 시험데이터 : 1set
► 사용자 수 : 부하시험과 동일
► 부하발생방법 : 부하시험과 동일
► 시험방법 : 피크타임시 TPS를 기준으로 24시간 지속적으로 트랜잭션 발생

④ 성능시험 판정기준

► 평상시시험 : 목표값(3초 이내)을 만족하고 Fail수가 1% 이내
► 부하시험 : 목표값(3초 이내)을 만족하고 Fail수가 1% 이내
► Endurance 시험 : Memory 누수 등 특이사항 없이 Transaction Fail수가 1%이내

〈표 2〉 실험 결과

테스트 항목	목표 응답 시간 (초)	데이터 발생건수	평균 응답 시간	목표 달성 건수	목표 달성율
ESM 이벤트 발생 정보	3	298	0.51	298	100%
EMS 장애발생 정보	3	298	0.49	298	100%
신호검지기 상태정보	3	101	0.44	101	100%
신호제어기 상태정보	3	101	0.41	101	100%
통신이상여부	3	5	0.83	5	100%
누수예상 측정지점 알람 발생 정보	3	31	0.54	31	100%
유압 측정 정보	3	60	0.64	60	100%
유량 측정 정보	3	60	0.6	60	100%
VDS 검지기 상태정보	3	101	0.51	101	100%
VDS 제어기 상태정보	3	101	0.51	101	100%
VMS 상태정보	3	169	0.45	169	100%
혼잡상황 정보	3	101	0.5	101	100%
돌발상황 감지/종료	3	101	0.51	101	100%
교통 소통정보	3	101	0.58	101	100%
시설물 고장 정보	3	101	0.42	101	100%
돌발상황 요약 정보	3	101	0.45	101	100%
CCTV 통신상태 수신 정보	3	61	0.48	61	100%
합계		1891	0.44	1891	100%

6. 결론 및 향후과제

본 논문에서는 수많은 이기종 매체간의 안정적인 통신방식을 지원하고, 이때 발생하는 이벤트와 데이터를 실시간으로 가공·처리하여 도시통합관제를 위한 실시간 정보서비스를 제공하는 U-City 통합관제를 위한 실시간 데이터 처리 미들웨어를 Rete 알고리즘을 활용한 추론엔진을 적용하여 구현하였다. 또한, 구현한 미들웨어를 기반으로 도시내에서 발생하는 다양한 서비스 정보를 분석하여, 이벤트정보와 상태정보 등을 발생시킬 수 있는 시뮬레이터와 운영자가 정보를 모니터링할 수 있는 통합 운영자 접속환경을 구축하여 그 성능을 테스트하였다.

Rete 알고리즘을 적용한 상황인지 모듈은 이벤트 발생시 해당 이벤트를 분석하여, 미들웨어내 내부 에이전트들에게 상황에 맞는 메시지를 제공하였다.

이는 데이터베이스에 의존하지 않는 상황인지 로직이므로, 실시간으로 발생하는 데이터와 이벤트를 효율적으로 처리할 수 있음을 보여주었다.

향후에는 도시통합관제를 위한 서비스간 비즈니스 프로세스의 명확한 정의와 운영 시나리오 정립에 따른 시스템 보완 및 SOA구조에 맞는 미들웨어에 대한 연구가 진행되어야 한다.

참 고 문 헌

[1] 정부만, 이상훈, 이계원, 박용철, 김재원, 박진석, 이창훈, "한국형 u-City모델 제안," 한국전산원, 2005.

[2] 박용민, "U-City 기반기술 상세 내역", RFID-Tech, 3월호, pp.112-119, 2006.

[3] 한국과학기술정보연구원, "유비쿼터스 미들웨어 기술동향", 한국과학기술정보연구원 보고서, 2004.

[4] 이응주, 윤희용, 조위덕, "유비쿼터스 환경을 위한 코바 기반 미들웨어 비교 연구", 한국지능정보시스템학회 2004 추계학술대회, pp.162-168, 2004.

[5] 데이비드 R. 부트노프, 강철민, "POXIS(포직스) 쓰레드를 이용한 프로그래밍, 인포북. 2005.

[6] W.Richard Stevens, Bill Fenner, Andrew M. Rudoff, "Nnix Network Programming1(Third Edition)", 교보문고, 2005.

[7] 최재원, 김성호, 조준한, 김원철, "인공신경망을 적용한 신호교차로 교통사고심각도 예측에 관한 연구", 대한교통학회지, 제22권, 3호, pp.127-135, 2004.

[8] Jae Hyun Lee "A Study on the Biometric Recognition Algorithm and System Implementation using Artificial Intelligence" 한국해양대학교 박사학위 논문, 2002.

[9] 지연상, 최완규, 이성주, "입출력 데이터 클러스트링에 의한 퍼지 교통 제어기의 설계", 한국퍼지및지능시스템학회지, 제11권, 3호, pp.241-245, 2001.

[10] 홍유식, 최명복, "인터넷을 이용한 교통상황예보", 한국퍼지및지능시스템학회지, 제12권, 3호, pp.300-109, 2003.

[11] 조재훈, 김동화, 오성권, "면역 알고리즘의 개선된 클론선택에 의한 퍼지 뉴로 네트워크와 교통경로선택으로의 응용", 한국퍼지및지능시스템학회지, 제14권 4호, pp.402-410, 2004.

[11] 김은영, 오동열, "분산 유비쿼터스 환경을 위한 상황 인식 미들웨어의 설계 및 구현", 한국컴퓨터정보학회지, 제11권, 5,43호 pp.105-114. 2006.

[12] 정헌만, "유비쿼터스 컴퓨팅 환경에서의 온톨로지 기반 상황 인식 미들웨어", 한국컴퓨터정보학회지, 제14권, 1호, pp.165-173, 2006.

[13] 심춘보, 신용원, "유비쿼터스 컴퓨팅 환경에서 상황인식을 지원하는 컨텍스트 미들웨어 개발", 한국지능정보시스템 학회논문지, 제11권, 1호 pp.53.63, 2005.

[14] 이선명, "테이블탑 컴퓨팅 환경을 위한 미들웨어 설계 및 구현", 포항공과대학교 석사학위논문, 2005.

권 창 희

- 1991년 : 건국대학교 졸업.(공학사)
- 1998년 : 동경도립대학교 도시과학연구과 졸업.(공학석사)
- 2003년 : 동경도립대학교 도시과학연구과 졸업.(공학박사)
- 현재 : 한세대학교 IT학부 교수, 사)한국U-City학회 회장

- 관심분야 : U-City, 도시통합관제, GIS
- E-mail : kwonch@hansei.ac.kr

이 주 상

- 1999년 : 경상대학교 정보통신공학과 졸업.(공학사)
- 2001년 : 한국해양대학교 전자통신공학과 졸업.(공학석사)
- 2008년 : 한국해양대학교 전자통신공학과 졸업.(공학박사)
- 현재 : 이에스이(주) 부설연구소 연구소장

- 관심분야 : U-City, 도시통합관제, 지능형 영상분석
- E-mail : greensory@diycad.co.kr