# 상황인식 처리를 위한 미들웨어 및 컨텍스트 서버의 설계 및 구현

김용기°, 김영국, 심춘보*, 장재우, 김정기**, 박승민**
전북대학교 컴퓨터공학과, 부산가톨릭대학교 컴퓨터정보공학부*, 한국전자통신연구원**
{kykim°, uksky, jwchang}@dblab.chonbuk.ac.kr, cbsim@cup.ac.kr*, {jkk, smpark}@etri.re.kr**

# Design and Implementation of a Middleware and Context Server for Handling Context-Awareness

Yong-Ki Kim° Young-Gug Kim, Choon-Bo Sim*, Jae-Woo Chang, Jeong-Ki Kim**, Seung-Min Park**
Dept. of Computer Eng., Chonbuk National University, Catholic University of Pusan*,
Electronics and Telecommunications Research Institute**

## 요 약

사용자를 중심으로 하는 주변환경과 사용자간 혹은 사용자와 장치간의 상호 운용성을 지능적, 자동적으로 선택하여 지원해 줌으로써 사용자로 하여금 정보 획득 및 실행을 보다 용이하게 하는 상황인식 기술은 유비쿼터스 컴퓨팅 환경에서 가장 중요한 핵심 기술이다. 따라서, 본 논문에서는 유비쿼터스 컴퓨팅을 위한 상황인식 기반의 미들웨어 및 컨텍스트 서버를 설계 및 구현한다. 먼저 미들웨어는 블루투스(Bluetooth) 무선 통신 기술을 이용하여 이동성을 지닌 이동 노드를 발견하고, 컨텍스트 서버에 등록하여 해당 컨텍스트에 적합한 실행 모듈을 서비스하는 기능을 담당한다. 컨텍스트 서버는 주어진 원격 객체의 정보와 사용자의 현재 상태 물리적 환경의 상태 등의 상황 정보를 효율적으로 데이터베이스 서버에 저장하고 관리할 수 있는 기능을 수행한다. 마지막으로 응용 시스템을 통해 제안하는 미들웨어와 컨텍스트 서버의 유용성을 보인다.

## 1. 서론

유비쿼터스(Ubiquitous)란 '어느 곳에서나 존재한다' 라는 의미로 유비쿼터스 환경은 사용자가 거부감을 느끼지 않고 언제 어디서나 존재하는 컴퓨터를 자연스럽고 편리하게 이용할 수 있는 환경을 말한다. 즉, 고성능, 초소형의 컴퓨터가 도처에 편재되어 센싱과 트래킹을 통해 장소나 시간에 따라 그 상황이 변화하는 지능적인 정보 서비스를 제공받을 수 있는 환경을 의미한다. 유비쿼터스 컴퓨팅 기술은 우리가 일 외의 대부분의 여가 시간을 보내는 홈 환경, 이동성, 자가 발전, 고정된 상호 작용 공간 등의 특성을 고려한 자동차에 기반을 둔 텔레매틱스, 여러 사람과 정보 공유 및 협업이 필요한 사무실 등 우리 일상생활 곳곳에서 매우 유용하게 활용될 수 있다.

이러한 유비쿼터스 컴퓨팅 환경에서 사용자가 필요로 하는 서비스를 제공하기 위해서는 일상생활 곳곳에 편재된 센서 및 컴퓨터들이 수집한 각종 상황정보를 효과적으로 상호 공유하여 사용자 및 주변 환경의 컨텍스트를 알아내는 상황인식 처리 기술이 요구된다. 상황인식 처리 기술은 사용자를 중심으로 하는 주변 환경과 사용자간 혹은 사용자와 장치간의 상호 운용성을 지능적, 자동적으로 선택하여 지원해 줌으로써, 사용자로 하여금 정보 획득 및 실행을 보다 용이하도록 지원한다. 아울러 유비쿼터스 컴퓨팅 환경에서 중요한 핵심 기술중에 하나이다. 따라서 본 논문에서는 유비쿼터스 컴퓨팅을 위한 상황인식 기반의 미들웨어 및 컨텍스트 서버를 설계 및 구현한다. 먼저 미들웨어는 블루투스[1] 무선 통신 기술을 이용하여 이동성을 지닌 이동 노드를 발견하고, 컨텍스트 서버에 등록하며 해당 컨텍스트에 적합한 실행 모듈을 서비스하는 기능을 담당한다. 또한 컨텍스트 서버는 주어진 원격 객체의 정보와 사용자의 현재 상태, 물리적인 환경의 상태, 컴퓨팅 자원의 상태 등과 같은 다양한 상황 정보를 효율적으로 데이터베이스 서버에 저장하고 관리할 수 있는 기능을 수행한다. 마지막으로 제안하는 미들웨어와 컨텍스트 서버를 이용한 상황인식 응용 시스템을 소개한다.

본 논문의 구성은 다음과 같다. 2장에서는 상황인식 처리 기술과 관련된 기존의 연구들을 살펴보고 3장에서는 상황인식 기반 미들웨어 및 컨텍스트 서버의 설계 및 구현에 대해서 설명한다. 4장에서는 제안하는 미들웨어 및 컨텍스트 서버의 구현 환경 및 테스트 환경에 대해서 소개한다. 마지막으로 5장에서는 결론 및 향후 연구를 제시한다.

## 2. 관련 연구

유비쿼터스 컴퓨팅을 위한 상황인식 처리 기술과 관련된 기존의 연구 두 가지를 소개하면 다음과 같다.

먼저 첫째, 애리조나 주립대학의 연구[2]는 유비쿼터스 컴퓨팅 환경에서 상황인식을 위해 각각의 장치를 통해 끊임없이 상태를 파악하여 그 상태에 따라 미리 정해진 적절한 조치(action)를 취하는 미들웨어를 제안하였다. 아울러 각각의 상황을 정의하기 위해 CA-IDL(Context-enabled Interface Definition Language)이라고 하는 상황 정의 언어를 제공하고 있으며, 각 상황은 다음과 같은 Context Tuple를 이용하여 정의한다.

Context Tuple : $<a_1, a_2, ..., a_n, t_m>$

여기서, n은 상황 정의를 위한 속성의 수를 의미하며, $a_i$는 i번째 속성의 값을 나타낸다. 그리고 $t_m$은 tuple이 생성된 시간을 의미한다. 예를 들면, Context Tuple의 속성이 각각 위치(x,y), 방향(d), 속도(s)라고 가정하면 그에 대한 Context Tuple은 <(x,y), d, s, t>로 정의할 수 있다.

둘째, IRISA/INRIA의 연구[3]는 이동성을 지닌 사용자에 대해서 사용자의 주변 환경 즉 상황인식 정보를 토대로 그에 적합한 최적의 서비스를 제공하기 위해 먼저 Contextual Object(CO)를 정의하고 이에 기반을 둔 하부 구조를 제안하였다. CO는 다음과 같이 정의한다.

CO(id) : < Variant $V_x$
Attribute $A_i$ : value,
Attribute $A_j$ : value,
... >

예를 들어, 웹 문서의 경우 <$V_1$ Location:$L_1$, Language:French, Browser:with frame>, <$V_2$ Location:$L_2$, Language:English, Browser:without frame>라는 CO가 있다면 사용자의 위치나 언어, 브라우저 타입에 따라 상황에 적합한 해당 Variant를 선택하여 그에 알맞은 웹 문서를 사용자에게 출력하게 된다. 아울러 전체적인 구조는 서버 측과 클라이언트 측으로 구성되며, 서버 측에서는 CO를 저장 및 검색할 수 있는 역할을 담당하며, 클라이언트 측에서의 요청이 있을 경우 CO를 전송한다. 아울러 CO의 Object

variant를 저장하고 있다가 클라이언트 측에서 Object variant의 요청을 받으면 그에 따른 응답을 한다.

## 3. 상황인식 처리를 위한 미들웨어 및 컨텍스트 서버

본 장에서는 상황인식 처리 기술을 위한 미들웨어 및 컨텍스트 서버가 포함된 전체적인 시스템에 대해서 기술한다. 전체적인 시스템 구조는 컨텍스트 데이터베이스를 관리하는 컨텍스트 서버와 일정영역의 이동노드를 담당하는 고정노드(미들웨어), 그리고 임베디드 이동 단말인 이동노드로 구성된다. 서버와 고정노드 사이는 TCP/IP를 통한 유선 네트워크로 구성되어 있고, 고정노드와 이동노드 사이에는 무선 통신인 블루투스를 이용하여 서로 데이터를 교환한다. 전체적인 구조는 그림 1과 같다.

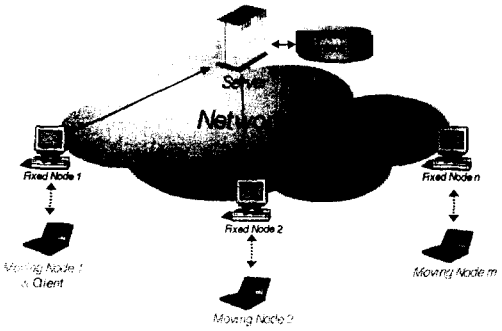


그림 1. 상황인식 처리 기술을 위한 전체적인 시스템 구조

각각의 요소별 컴포넌트 모듈을 살펴보면, 고정노드(미들웨어)는 블루투스 장치를 사용하는 환경에서 상황인식 처리 기술을 위한 원격 객체 발견, 등록, 실행을 하는 모듈과 상황정의 스크립트 및 처리기로 구성되어 있으며, 아울러 원격 객체 정보와 컨텍스트 정보를 객체 데이터베이스와 컨텍스트 데이터베이스에 저장하고 검색할 수 있는 컨텍스트 서버로 구성된다. 이는 그림 2와 같다. 이동노드는 원격 객체(컨텍스트 객체)로서 고정노드와 블루투스 통신을 통하여 데이터를 교환하며, 미리 정의된 상황에 따라서 미들웨어와 상호작용을 통해 내장된 프로그램이 실행되거나 지정된 처리를 수행한다.

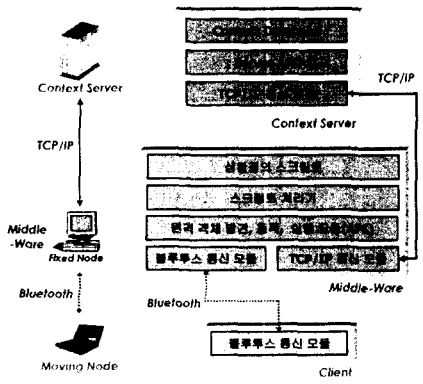


그림 2. 각각의 요소별 컴포넌트 모듈 구성 형태

### 3.1 미들웨어(Middleware)

본 논문에서 제안하는 상황인식을 위한 미들웨어는 IRISA/INRIA[3]의 미들웨어 모델을 참조하여 설계한 것으로 상황정의 스크립트 처리기를 부가적으로 추가하여 범용성을 높였다. 각각의 구성 요소를 살펴보면, 그림 3과 같이 3단계의 계층으로 구성되며, 이는 탐지 및 모니터링 계층, 상황인식 계층, 응용계층이다.

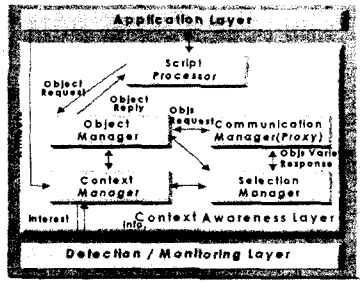


그림 3. 상황인식 처리를 위한 미들웨어

탐지 및 모니터링 계층은 원격 객체의 위치정보나 응용 프로그램과 상호작용에 따른 이벤트, 그리고 CPU 사용량, 메모리의 사용량과 같은 컴퓨팅 자원이나 네트워크 상태를 모니터링 할 수 있는 기능을 담당하며, 이 계층에서는 응용 시스템에 따라서 탐지 모듈의 유연한 형태를 가진다. 상황인식(context awareness) 계층은 유비쿼터스 컴퓨팅을 위한 상황인식 처리 기술의 핵심 계층으로서 미들웨어 역할을 담당한다. 이 계층은 크게 스크립트 처리기, 원격 객체 관리자, 컨텍스트 관리자, 컨텍스트 선택 관리자, 통신 관리자의 5가지의 관리자로 구성되고, 스크립트 처리기는 응용 프로그램의 상황인식 정의 스크립트의 각 내용을 분석하여 규약에 명시된 역할을 수행하도록 한다. 원격 객체 관리자는 각 응용 프로그램에서 현재 사용하고 있는 원격 객체(컨텍스트 객체)에 포함되어 있는 모든 정보를 위한 데이터 구조를 관리한다. 컨텍스트 관리자는 사용자의 취향, 이동 단말 기기의 성능, 현재 위치정보 등을 포함하고 있는 주어진 환경이나 상황에 대한 컨텍스트 정보를 관리한다. 컨텍스트 선택 관리자는 응용 프로그램을 통해 사용자에게 전송될 조건에 부합되는 컨텍스트 정보를 선별한다. 통신(프락시) 관리자는 컨텍스트 서버와 통신을 담당하며, 예상하지 못한 네트워크의 장애로 인한 문제에 대비한다. 응용 계층은 하부의 상황인식 계층인 미들웨어와 독립적으로 동작하면서 미들웨어의 응용 프로그래밍 인터페이스(API)를 통해 상황인식 처리 기술에 기반을 둔 다양한 응용 프로그램을 개발하기 위한 기능을 수행한다.

### 3.2 컨텍스트 서버(Context Server)

상황인식 처리 기술을 위해서는 원격 객체에 대한 정보와 원격 객체로부터 파생되고 추출할 수 있는 상황인식 정보, 즉 컨텍스트를 서버에 저장하고 검색 할 수 있는 컨텍스트 서버가 필요하다. 본 절에서는 컨텍스트를 효율적으로 저장 및 검색하기 위한 컨텍스트 서버의 기능 및 컨텍스트 서버를 구축하기 위한 하부 데이터베이스 관리 시스템에 대해서 기술한다.

상황인식 처리 기술에 적합한 컨텍스트와 컨텍스트 객체를 저장 및 관리할 수 있는 컨텍스트 서버를 처음부터 개발하는 것은 매우 어려우며 또한 많은 시간이 걸리는 작업이기 때문에, 기존 데이터베이스 관리시스템을 이용하여 서버를 구축하는 방법이 바람직하다. 따라서 본 논문에서는 이를 위해 MySQL DBMS[4]를 사용한다. 한편, 컨텍스트 서버는 미들웨어에서 전달되는 패킷의 내용을 분석하고 패킷의 내용을 수행하며, 이를 위해서는 이미 규정된 약속을 가지고 패킷의 내용을 분석하고 또한 패킷을 주고받을 수 있는 네트워크 관리자가 필요하다. 이를 통해서 패킷의 내용을 분석한 후 그 패킷 내용이 컨텍스트 DB에 저장해야 할 컨텍스트인지, 아니면 컨텍스트 객체인지 구별한 후에 이에 해당하는 데이터베이스에 저장 및 검색한다. 그림 4는 전체적인 컨텍스트 서버의 구조를 도식화한 것으로 Communication Manager(CM), Packet Analyzer Manager(PAM), Context/Object Manager(COM), mySQL Query Manager(SQM)와 같이 4개의 관리자를 구성된다. 각각의 다음과 같은 기능을 담당한다.

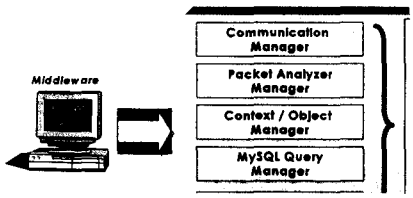


그림 4. 컨텍스트 서버의 구조

(1) CM은 서버와 미들웨어간의 통신을 담당한다. 미들웨어에서 전달되어진 패킷의 내용을 분석하기 위해 해당 패킷을 PAM에게 전달하고, 서버에서 작성된 결과 패킷을 다시 미들웨어에게 전송한다.

(2) PAM은 CM으로부터 전달되어진 패킷을 분석한 후 현재 어떤 상황인지를 분석한다. 이 분석을 통해 COM에 있는 알맞은 해당 함수(메서드)를 호출하도록 지시하는 역할을 담당한다.

(3) COM은 PAM으로부터 호출되는 데, COM은 PAM으로부터 전달받은 내용을 SQL문으로 변환하고, 변환된 결과를 토대로 SQM은 해당 SQL문을 실행함으로써 최종 결과를 데이타베이스에 반영하도록 하는 역할을 담당한다.

(4) SQM은 COM으로부터 받은 SQL문을 데이타베이스에 반영하기 위한 작업을 수행하고 그에 따른 결과값을 CM을 통해 다시 미들웨어로 전송한다.

그림 5는 실제로 구현된 인터페이스(API)들의 구조를 보여주고 있으며 표 1은 각 인터페이스들과 모듈을 나타낸다.

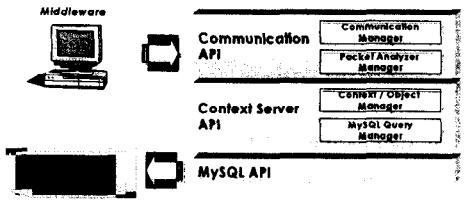


그림 5. 인터페이스(API) 사이의 관계

표 1. 컨텍스트 서버의 인터페이스

| 구성요소 | 역 할 |
|---|---|
| 메인 서버 모듈 | 컨텍스트 서버의 시작 및 TCP/IP 소켓을 통해 통신을 하기 위한 Communication Manager 모듈 |
| 패킷 분석 모듈 | Packet Analyzer Manager 부분으로 미들웨어로부터 받은 패킷을 분석하여 해당 컨텍스트 API를 호출 |
| 컨텍스트 API | 컨텍스트 API로서 DBMS 및 미들웨어와 상호작용을 위한 라이브러리이며 서버측 API에 해당 |
| MySQL API | MySQL Query Manager 부분으로 MySQL(DBMS)과의 통신을 담당하며 해당 결과문을 미들웨어로 전송하는 역할 |
| 파일 수신 모듈 | 컨텍스트 서버가 고정 노드(미들웨어)에게 파일을 전송하는 함수 라이브러리 |
| TCP/IP 소켓 모듈 | 컨텍스트 서버와 TCP/IP 소켓 통신을 위한 소켓 라이브러리 |
| 기타 실행 모듈 | 원격 객체의 상황에 대한 사용자 정의 실행 모듈 |

## 4. 구현 환경 및 테스팅

상황인식 처리를 위한 미들웨어와 컨텍스트 서버를 구현한 구현 환경은 표 2와 같다.

표 2. 구현 환경

| | |
|---|---|
| O.S | Red Hat Linux 7.3(Kernel version 1.4.20) |
| Compiler | GCC 2.95.4, Lex 2.5.4, Yacc 1.28 |
| DBMS | MySQL 3.23.49 |
| Bluetooth Driver | affix 2.0.2[5] |
| System | CPU - 866Mhz 펜티엄3, 메모리 - 64MB |

본 논문에서 제안하는 미들웨어 및 컨텍스트 서버의 유용성을 보이기 위한 응용 시스템의 테스트 환경은 그림 6에서와 같다. 즉, 노트북이나 PDA와 같은 휴대용 단말기를 소유한 사용자(이동 노드)로 하여금 약 60m정도 떨어져 있는 두 연구실(Media 연구실과 DB 연구실)에 고정 노드를 설치하고 두 연구실 사이를 왕복하도록 하였다. 그리고 이동 노드를 소유한 사용자들에 대한 프로파일 정보(음악 파일)를 컨텍스트 서버에 미리 저장해 두었다. 음악은 사용자별로 서로 다르게 정의되어 있으며 같은 사용자에 대해서도 오전, 오후, 야간과 같은 시간대에 따라 다른 곡이 정의되어 있다.

이동 노드가 하나의 고정 노드(Media 연구실)에 접근하였을 때 해당 고정 노드는 이동 노드의 사용자를 감지하고 그 사용자에 대한 프로파일 정보를 컨텍스트 서버로부터 전송받아 음악을 재생하며, 또한 현재의 고정 노드로부터 멀어져감에 따라 블루투스와의 무선 통신이 불가능하게 되면 음악 재생이 멈춘다. 아울러 또 다른 고정 노드(DB 연구실)에 점점 가까워지면 현재의 고정 노드가 이동 노드를 감지하여 같은 사용자에 대한 동일한 음악을 재생하도록 한다.

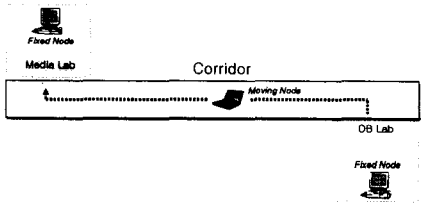


그림 6. 응용 시스템의 테스트 환경

## 5. 결론 및 향후연구

본 논문에서는 유비쿼터스 컴퓨팅을 위한 상황인식 기반의 미들웨어 및 컨텍스트 서버를 설계 및 구현하였다. 이를 위해 미들웨어는 블루투스(Bluetooth) 무선 통신 기술을 이용하여 이동성을 지닌 이동 노드를 발견하고, 컨텍스트 서버에 등록하여 해당 컨텍스트에 적합한 실행 모듈을 서비스하는 기능을 담당한다. 아울러 컨텍스트 서버는 주어진 원격 객체의 정보와 사용자의 현재 상태 물리적 환경의 상태 등의 상황 정보를 효율적으로 데이터베이스 서버에 저장하고 관리할 수 있는 기능을 수행한다. 향후 연구로는 제안하는 미들웨어 및 컨텍스트 서버에 대한 다양한 환경에서의 성능 평가를 수행하는 것이다.

## [참고문헌]

[1] http://www.bluetooth.com/ "Bluetooth Version 1.1 Profile"

[2] Stephen S. Yau, Fariaz Karim, "Context-Sensitive Middleware for Real-Time Software in Ubiquitous Computing Environments," Fourth International Symposium on Object-Oriented Real-Time Distributed Computing, pp. 163-170, 2001.

[3] P. Couderc, A.-M. Kermarrec, "Improving Level of Service for Mobile Users Using Context-Awareness." 18th IEEE Symposium on Reliable Distributed Systems, pp. 24-33, 1999.

[4] 조지 리스 외 3명, MySQL 시스템 관리와 프로그래밍, 한빛미디어, 2002.

[5] http://affix.sourceforge.net/ "Affix : Bluetooth Protocol Stack for Linux"