경량 모바일 미들웨어 시스템 설계

Lightweight Mobile Middleware Systems Design

양승일, 이태규, 박성훈
충북대학교

Yang Seung-Il, Lee Tae-Gyu and Park Sung-Hoon
Chungbuk Univ.

요약

기존의 미들웨어 시스템이 유선 환경을 중심으로 기초로 개발됨에 따라 모바일 무선 이동성 환경을 지원하기 위해서 미들웨어 및 관련 응용 프로그램들이 규모 또는 운용성이 커서 적용되기 어렵다는데 문제가 발생한다. 이러한 문제점을 극복하기 위해서 미들웨어 응용 및 실행 단위가 모바일 환경에 적합하게 최소 및 최적화되어야 하므로 이에 필요한 경량모바일 시스템을 설계 및 제안 한다.

Abstract

A conventional middleware system is optimized for wired environment. Middleware and many applications have to be made in small sized to support for wireless mobile environment. We design a lightweight mobile middleware system which is optimized for mobile environment and middleware applications.

Ⅰ. 서론

최근 IT 환경은 모바일 사용자들을 지원하는 유비쿼터스 환경으로 진입하고 있다. 유비쿼터스 환경에서는 PDA, NOTEBOOK, SMART PHONE 등 많은 모바일 기기들이 사용된다. 이러한 모바일 환경에 맞는 미들웨어시스템에 대한 연구들이 부족한 실정이다. 기존의 미들웨어 시스템이 유선 환경을 중심으로 개발되었으므로 미들웨어 및 관련 응용 프로그램들이 규모 및 운용성이 커서 유비쿼터스 환경을 위한 무선 이동성을 지원하기 어렵다는데 문제점에서 출발하고 있다. 이러한 문제점을 극복하기 위해서 본 연구는 미들웨어 응용 및 실행 단위가 모바일 환경에 적합하게 최소 및 최적화되어야 한다는 데 그 필요성을 인식하고 있다. 이러한 문제점들을 극복한 미들웨어 시스템을 본 연구는 “경량 모바일 미들웨어 시스템 (Lightweight Mobile Middleware Systems)” 또는 "LightMware"라고 한다.

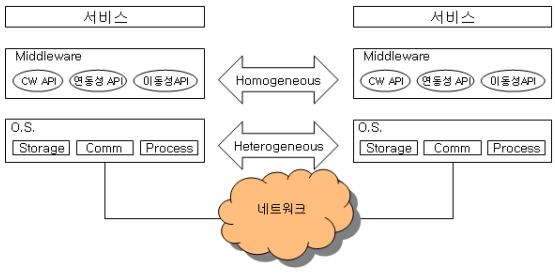
본 경량 모바일 미들웨어 시스템은 모바일 클라이언트의 운영환경에 맞게 미들웨어 및 그 응용 실행 단위를 최소화하는 것이 요구된다. 이러한 목적을 실현하기 위해서는 미들웨어 전체 시스템의 구성을 기능 및 사용자 응용 단위로 나누어서 컴포넌트 (component) 단위로 구성하는 것이 요구된다. 그러므로 본 논문은 이러한 관점을 기반으로 하여 다음과 같이 구성된다. 2장에서는 관련 연구를 알아보고, 3장에서는 경량모바일미들웨어시스템을 설계하고, 4장에서는 결론을 맺는다.

Ⅱ. 관련연구

미들웨어란 "응용 프로그램과 운영체제를 비롯한 기반시스템 사이의 중간 단계에 위치하여 응용시스템 개발 및 운용 서비스를 제공해 주는 소프트웨어 운영 프로세스"를 말한다. 이러한 소프트웨어가 출현한 배경으로는 우선 기존 Client/Server의 2 Tier 환경에 있어서 업무 처리 로직이 Client에 위치하게 되어 Fat Client를 초래함으로써 버전 관리가 어렵고, Client 프로그램

이 많아지게 되면 Server에 추가되는 부하도 동일하게 증가되어 대규모 응용 환경 하의 유비쿼터스 컴퓨팅에 적당하지 않다, 따라서, 비즈니스 로직을 Client에서 떼어내고, Client의 수가 증가하더라도 Server에 미치는 영향을 최소화하고자 하는 측면에서 중간에 Middleware라는 소프트웨어를 활용하게 되었다. 현재 사용하는 EAI, WAS(Web Application Server)도 Web 환경 하에서 운영되는 대표적인 미들웨어라고 볼 수 있습니다[1].

본 연구의 경량모바일미들웨어시스템은 미들웨어 하부 시스템에 대한 은닉성(Hideness) 및 투명성(Transparency) 제공해주는 것이 중요하다. 이렇게 함으로써 하부시스템은 이질적인 시스템 환경이 구축되고 상부시스템에는 동질적인 응용 환경을 구축 가능하게 함으로써 응용 시스템 개발의 효율성 및 생산성을 극대화시키는데 기여할 수 있다[2].



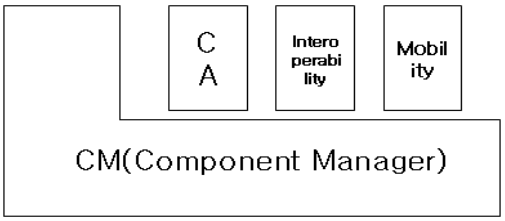
▶▶ 그림 1. 경량 모바일 미들웨어 시스템

Ⅲ. 경량모바일미들웨어 설계

본 경량 모바일 미들웨어 시스템은 미들웨어의 각 요소를 컴포넌트로 구성하고 실행 환경 및 응용 시스템에 따라 동적으로 구성하여 미들웨어 크기를 최소화 하는데 그 설계 목적으로 둔다.

1. 구성요소

아래 그림 2는 4가지 경량모바일미들웨어 기본 요소에 대한 기본 구성을 보여준다.



▶▶ 그림 2. 경량모바일미들웨어 기본 구성

1.1 컴포넌트 매니저 (CM: Component Manager)

전체 Component들을 관리하는 위한 미들웨어 시스템 PLATFORM 및 컴포넌트 관리자 인터페이스를 제공한다. 이를 위해서 컴포넌트 설계(Design), 컴포넌트 추가(Add), 컴포넌트 삭제(Delete), 컴포넌트 수정(편집), 컴포넌트 현황 및 실행 관리, 컴포넌트 로깅(Logging)과 같은 기능들이 요구된다. 이러한 여러 컴포넌트들은 관련성 정도에 따라 그룹단위로 묶여 관리의 효율성을 제공한다.

1.2 상황 인지(CA: Context Awareness)

미들웨어 기반 응용 서비스 자원 및 환경을 감지하여 개발 및 운용 모듈 및 API 인터페이스를 제공한다.

1) 자원 및 환경 관리: CPU, Memory, Disk, Power 등의 하드웨어 자원, OS, DB, F/S(File System), VM 등의 소프트웨어 자원, 유선/무선 종류, 전송속도, 전송 지원 프로토콜, 네트워킹 영역(범위 – LAN, Intranet, Internet) 등의 네트워크 자원의 다양한 환경 인식 및 이벤트 정보를 응용 API와 컴포넌트 매니저 사이에서 공지한다.

2) H/W 자원관리: 단말, 서버, 네트워크 자원 등의 위치 관리, 파워 관리, 생존 관리 서비스를 지원하기 위한 위치 관리 맵(MAP)을 제공하고 자원들 사이의 관계를 트리 구조에 기반 한 자원관리 서비스를 제공한다. 클라이언트 자원 관리는 위치 관리, 파워 관리, 생존 관리를 지원하고, 서버 자원 관리는 CPU 현황 / Memory 현황, I /O 인터페이스 현황을 관리한다. 또한 네트워크 자원 관리는 유선자원과 무선자

원이 있는데 유선 자원 관리는 네트워크의 연결성 관리, 전송속도 관리, 네트워크 범주 등에 관한 관리 서비스를 지원한다. 무선 자원 관리는 무선 단말의 무선 네트워크 접속 시, AP 서버의 컴포넌트 관리자의 승인 절차에 따라 채널을 생성하고 송신자에게는 송신 포트를 구성하고, 수신자에게는 수신 포트를 구성하여 채널 자원을 관리한다.

3) S/W 자원관리 : 미들웨어 COMPONENT 관리, 응용프로그램 COMPONENT 관리, 외부미들웨어 COMPONENT 관리가 있는데 외부미들웨어 COMPONENT는 모바일 미들웨어 응용 서비스 관리를 포함한다. 또한 모바일 웹서비스, 모바일 FTP 서비스, 모바일 Telematics 서비스를 예로 들 수 있다.

(1) 각종 모바일 예약서비스 : 철도예약서비스, 항공예약서비스, 고객관리예약서비스, 데이터베이스 자원 관리(Mysql / MSSQL / ORACLE / MiniSQL / INFORMIX) 등이다.
(2) 시나리오 based API 제공: 현재까지 응용 서비스 및 미들웨어 구성 및 개발 구현 시스템들은 개발자가 모든 자원 또는 환경에 대한 개별 인자를 세부적으로 다루어 주어야하는 복잡하고 어려운 문제점을 안고 있다. 본 연구는 이러한 복잡성 문제점을 극복하는 동시에 개발자 및 사용자에게 용이한 개발 및 응용 인터페이스를 지원하기 위해서 응용 환경 및 사용자 시나리오에 기초한 API 서비스를 실현하고자 한다. 이러한 시나리오 API는 공간 기반 시나리오 API와 시간 기반 시나리오 API를 제공하고자 한다. 공간 기반 시나리오의 예는 공장, 사무실, 외부 거래처 등이 있다. 시간 기반 시나리오의 예로는 오전, 오후, 저녁 등이 있다.

1.3 상호연동성(Interoperability)

경량 모바일 미들웨어는 지원하는 응용시스템들 사이에 데이터(Data) 교환 및 인터페이스(Interface) 교환을 제공하는 상호연동성(Interoperability)을 지원한다. 상호연동성은 LightMware와 동일한 미들웨어를 사용하는 내부 상호연동성과 서로 다른 미들웨어 또는 미들웨어를 사용하지 않는 제3의 응용시스템과 연동을 지원하기 위한 외부 상호 연동성으로 구분된다. 외부 상호연동성은 향후 상호 호환성(Inter-compatability) 모듈로 제공된다.

1) DATA 상호연동성을 제공하는 API로 XML, RSS, ATOM으로 제작하여 제공한다.

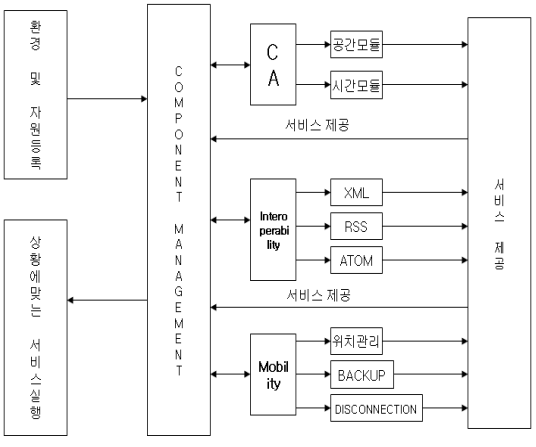
2) Interface 상호연동성을 제공하는 API를 XML, RSS, ATOM으로 제작하여 제공한다.

1.4 시스템 이동성(Mobility)

경량 모바일 미들웨어는 시스템 Mobility, S/W Mobility, Network Mobility등을 지원한다. 시스템 이동성은 이동성이 잦은 단말을 비롯한 이동성이 낮은 서버의 이동성을 지원한다. 소프트웨어 이동성은 응용 소프트웨어 제거 및 추가, 데이터베이스 이동성(Data 이동성 포함) 등을 지원한다. 네트워크 이동성은 기존 네트워크의 제거, 재구성, 추가 등으로 발생하는 동적 네트워크 구성을 지원한다. 이동성 인터페이스 제공하고, 위치관리를 통해 단말의 위치를 관리한다. BACKUP은 응용 서비스의 다양한 롤백(Rollback) 기능을 지원하기 위해 이미 진행된 응용 서비스에 대한 기록을 보전한다. DISCONNECT는 서비스 연결 복원 시, 서비스 복구(Recovery)를 지원하기 위한 단절 연산을 지원한다.

2. 시스템 설계

경량 모바일 시스템의 처리 흐름도는 다음 그림 3과 같다.



▶▶그림 3. 경량 모바일 시스템의 처리 흐름도

1) 주 메뉴 구성

파일 : 미들웨어 프로젝트 관리 데이터를 저장 관리 한다.

자원관리환경 : 하드웨어, 소프트웨어, 네트워크 자원 관리를 지원한다.

Dynamics : 시나리오기반 동적 컴포넌트 관리 서비스를 지원한다.

상호연동 : 미들웨어 상호연동성을 지원한다.

이동성지원 : 시스템, 네트워크, 데이터 이동성 등을 지원한다.

도움말 : LightMware 경량모바일미들웨어시스템 프로젝트 및 구현에 관한 도움말을 제공한다.

2) 구현 인터페이스 뷰(view)

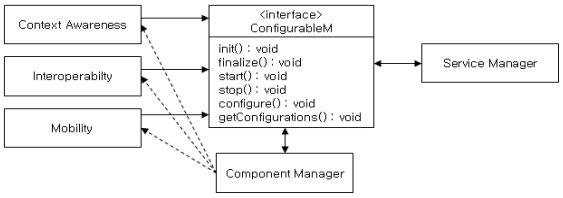
(1) 서버 인터페이스 구현

- CM P/G

관리자가 상황에 맞는 메뉴를 관리하고 필요한 권한을 관리한다. 또한 COMPONENT들을 관리한다.

(2) 클라이언트 인터페이스 구현

- 클라이언트 P/G : 사용자가 등록한 환경 및 자원에 맞는 COMPONENT를 제공해주고 불필요한 COMPONENT는 제거한다.
- 윈도우 클라이언트 응용 인터페이스 개발
- 웹 클라이언트 응용 인터페이스(그림 4) 개발



▶▶ 그림 4. 웹 클라이언트 응용 인터페이스

Ⅳ. 결론 및 향후연구 방향

그동안 많은 미들웨어시스템들이 설계되어 왔지만 모바일 환경에 맞는 미들웨어시스템은 많은 연구가 진행되지 못했다. 지속적으로 발전해가는 모바일 환경에 맞는 미들웨어는 경량이어야 하므로 이를 구현하는 것이 많은 연구과제로 남아 있다.

본 논문에서는 먼저 경량 모바일 환경에 맞는 미들웨어 시스템을 설계하였다. 고려해야할 부분이 상황인지, 상호연동성, 시스템 이동성이다. 즉 상황을 인지하여 서비스를 하되 이기종 또는 서로 다른 시스템 간에 연동이 잘 이루어지고 시스템의 이동성을 충분히 지원해야 하는 것이다. 우리는 이러한 프로토타입으로 공장자동화를 설계하였다. 공장에서 시간과 공간에 따라 달라지는 상황을 인식하여 여기에 적합한 서비스나 스케줄을 보여주는 형태를 설계하였다.

향후 우리는 각 구성요소에 대한 구체적인 설계와 구현에 대한 연구를 진행해야 할 것이다. 유비쿼터스 환경으로 변해가는 현재의 환경에 필요한 표준 및 기술 연구가 필요하다.

Ⅴ. Acknowledgment

본 연구는 교육과학기술부와 한국산업기술재단의 지역혁신인력양성사업으로 수행된 연구결과임.

■ 참 고 문 헌 ■

[1] http://blog.naver.com/tlark12?Redirect=Log&logNo=80065557126

[2] Valerie Issarny, Mauro Caporuscio, Nikolaos Georgantas "A Perspective on the Future of Middleware-based Software Engineering", Future of Software Engineering(FOSE'07), pp. 244-258, 2007.

[3] 이학진, 김성조 "이질적인 홈 네트워크 미들웨어 상호 연동성 지원을 위한 표준 서비스 기반의 통합 구조 설계" 제32회 추계학술발표회 논문집 Vol.32, No.2