The Journal of The Institute of Internet, Broadcasting and Communication (IIBC)
Vol. 16, No. 4, pp.81-87, Aug. 31, 2016. pISSN 2289-0238, eISSN 2289-0246

http://dx.doi.org/10.7236/JIIBC.2016.16.4.81
JIIBC 2016-4-13

유비쿼터스 컴퓨팅 환경에서 원격진단을 위한 스마트 응용의 설계

Design of a Smart Application for Remote Diagnosis in Ubiquitous Computing Environment

오선진*

Sun-Jin Oh*

요 약 최근 스마트 폰과 무선 기반 네트워크 기술의 급속한 발전과 더불어, 이를 이용한 스마트 응용의 개발이 활발하게 이루어지고 있다. 특히 다양한 센서를 이용한 유비쿼터스 센서 네트워크 기술 발달과 시간•장소에 구애 없이 자유롭게 이동하면서 원하는 경우 언제든지 네트워크에 접속하여 시스템의 서비스를 받을 수 있는 유비쿼터스 컴퓨팅 기술이 급속히 발전하면서 첨단 스마트 폰을 이용한 다양한 형태의 스마트 응용들이 속속 개발되어 보급되고 있는 실정이다. 본 연구에서는 이러한 시대적 조류에 발맞춰 유비쿼터스 센서 네트워크를 기반으로 하는 모바일 컴퓨팅 환경에서 첨단 스마트 폰을 이용하여 거리상 멀리 떨어져 있는 지역의 생태계나 사물 및 사람의 실생활 환경, 의료, 건강관리 등 상황정보와 실시간 음성 및 영상을 기반으로 그 지역 환경, 사물 또는 사람의 현재 상황을 원격으로 실시간 진단하고 처리할 수 있는 스마트 응용을 설계하고자 한다.

Abstract With the rapid growth of the up-to-date smartphone and wireless network technologies, huge and various types of smart applications using these technologies are actively developed recently. Especially, multiplex types of smart applications using smartphone are developed and diffused with the rapid development of the ubiquitous sensor network technology using various sensors and the mobile computing technology that enables us to get network services at any time in any places. In this paper, we design a smart application that can accurately diagnose and process the current state of the local environment, objects, and persons remotely based on the context information such as local ecology, circumstances, medical or healthcare records and realtime sound or motion pictures using up-to-date samrtphone technology on the USN based mobile computing environment.

Key Words : Ubiquitous Sensor Network(USN), Remote Diagnosis, Smart Application

Ⅰ. 서 론

인류 문명을 공간 혁명 관점에서 살펴보면, 도시혁명과 산업혁명은 주로 지리적인 제약에 기반한 물리공간을 토대로 전개된 인류 문명의 핵심이라 할 수 있다. 20세기 들어 컴퓨터와 네트워크 기술의 급속한 발전과 더불어 도래한 인터넷 혁명은 IT기술의 발달에 기인한 사이버 전자공간을 창출하였고, 이어 유비쿼터스 혁명으로 발전되어 가고 있으며, 이는 물리공간과 전자공간을 유기적으로 통합한 유비쿼터스 공간으로의 진화를 가능하게 하

*종신회원, 세명대학교 정보통신학부
접수일자 : 2016년 5월 31일, 수정완료 : 2016년 6월 23일
게재확정일자 : 2016년 8월 5일

Received: 31 May, 2016 / Revised: 23 June, 2016
Accepted: 5 August, 2016
*Corresponding Author: sjoh@semyung.ac.kr
Dept. of Computer & Information Science, Semyung University, Korea

고 있다.[1] 한편, 인터넷 혁명이라고 불리는 소위 사이버 전자공간은 인간의 삶과 산업 • 경제적 활동을 수용해 오던 전통적인 물리공간에 필적할 만한 활용성과 실용성을 갖추고 있는 또 하나의 새로운 공간으로 급속히 성장하고 있다.[2] 이러한 전자 공간 중심의 정보화 패러다임에 또 다른 대전환이 나타나기 시작한 것은 1991년 미국 Xerox사 Palo Alto Research Center의 Mark Weiser 연구원의 차세대 컴퓨팅 비전을 제시하기 위해 주장한 유비쿼터스 컴퓨팅 개념이 등장하면서부터이다.[3]

20세기의 정보통신 혁명은 컴퓨터들 간의 연결 그리고 컴퓨터와 사람과의 연결을 통해 사회 전체로 급속히 확산되었으며, 21세기 스마트폰에 의한 기술 융합을 통해 클라우드 컴퓨팅 기술을 촉발시켰다. 즉, 물리공간의 사물들이 인터넷을 통해 접속되면서 사람과 사람을 연결하는 인터넷 공간으로 발전하였고, 21세기에는 사람 뿐만아니라 기계, 사물, 시스템 등 모든 대상을 연결시키는 무한 확장의 개념으로 초연결 사회를 지향하는 초공간을 창출하는 공간 혁명을 가져오고 있다.[1, 4]

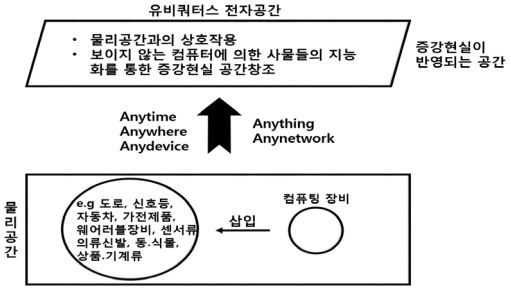

그림 1. 유비쿼터스 컴퓨팅 환경
Fig. 1. Ubiquitous Computing Environment

그림 1은 물리공간과 전자공간이 융합한 모바일 컴퓨팅 환경에서 초공간이라 불리는 유비쿼터스 공간 개념도를 보여준다. 그림에 보인바와 같이 유비쿼터스 공간은 모바일 컴퓨팅 혁명의 결과로 인해 구성된 초연결 공간으로 실세계에 있는 모든 컴퓨팅 장치들 뿐 만아니라 다른 모든 비컴퓨팅 장치들 즉, 사물, 기계, 시스템들이 컴퓨팅 장비를 이식하여 보이지 않으나 모두 네트워크에 연결되어 있으며, 이 사물들은 Anytime, Anywhere, Anydevice 등 시간과 장소에 상관없이 언제든지 필요한 경우 네트워크를 통해 접속이 가능하고, Anything, Anynetwork 등 어떤 사물이거나 어떤 형태의 네트워크든 상관없이 접속하여 서비스를 받을 수 있는 유비쿼터스 공간을 제공하고 있다.[1, 4]

최근에는 Mark Weiser가 주장한 유비쿼터스 컴퓨팅의 개념이 더욱 진화하여 스마트폰, 태블릿 PC, IPTV, 가전, 오디오/비디오 기기 등 우리 주변의 다양한 기기들을 네트워크로 연결하고 컴퓨터로 제어할 수 있는 기술로 발전하였다. 이러한 유비쿼터스 컴퓨팅 환경이 완성되기 위한 필수 기본 조건은 다음과 같다.[5]

1. 네트워크에 연결되지 않은 컴퓨터는 유비쿼터스 컴퓨팅이 아니다. 즉, 항상 네트워크에 접근이 가능해야 한다.
2. 인간화된 인터페이스(Calm Technology)로서 눈에 보이지 않아야 한다. 즉, 내장형 또는 소형 마이크로컴퓨터 칩 형식이 되어야 하며, 지능형 컴퓨팅이 가능해야 한다.
3. 가상공간이 아닌 현실세계의 어디서나 컴퓨터의 사용이 가능해야 한다. 즉, 현실세계의 구체화된 어떠한 장소에서도 컴퓨팅이 가능해야 한다.
4. 인간화된 인터페이스로서 사용자 상황(장소, ID, 장치, 시간, 온도, 조도, 습도, 날씨 등)에 따라 서비스가 변해야 한다.

본 연구에서는 이러한 특징을 가지는 유비쿼터스 컴퓨팅 환경에서 스마트폰이나 태블릿 PC 등의 첨단 단말을 이용하여 원격지에 떨어져 있는 지역의 생태계나 사물 또는 사람의 실생활 환경, 의료, 헬스케어 등 상황정보와 실시간 음성 및 영상을 기반으로 그 지역, 사물, 또는 사람의 현재 상황에 대한 상태를 원격으로 실시간 진단하고 처리할 수 있는 스마트 응용을 설계하고 구현하고자 한다. 본 논문의 구성은 다음과 같다. 2장에서는 관련 연구를 살펴보았고, 3장에서는 USN 기반 모바일 컴퓨팅 환경에서의 원격진단 스마트 응용을 위한 시스템 모델을 서술하였으며, 4장에서는 원격진단을 위한 스마트 응용을 모듈별로 설계하여 주요 알고리즘을 고찰을 하였고, 마지막으로 5장에서 향후 연구내용과 함께 결론을 맺는다.

II. 관련 연구

스마트폰 단말 기술은 최고의 성능과 사양을 갖춘 첨단 기술을 망라하는 방향으로 급속히 발전하고 있어 빠

르고 정확한 정보 처리뿐만 아니라 센서와 같은 다양한 주변장치들과 연동하여 원격으로 실시간 진단하고 처리할 수 있는 고성능 단말들이 속속 등장하고 있다.[6] 최근 발표된 스마트폰 관련 자료에 따르면 스마트폰에 첨단 전자 센서가 10여개나 내장되어 있어 심박수, 지문인식, 위치인식, 속도, 기울기, 기압, 고도, 자기, 온도, 근접도, 조도 등을 실시간 측정할 수 있어 손바닥에 쏙 들어오는 조그만 스마트폰이 이 센서들을 통해 사용자가 지금 어디서 무얼 하는지, 몸 컨디션은 어떤지 등을 속속들이 파악할 수 있게 되었다.[7] 이와 같이 첨단 스마트폰 단말 기술은 무척 빠른 속도로 그 기능이 진화하고 있다.

최근의 스마트 응용들은 다양한 통신 채널들과 결합한 USN이나 블루투스 기반 애드 혹 센서망 응용분야에 이르기까지 매우 다양하며 그 발전 가능성도 무궁무진하다고 할 수 있다.[2] 특히 연산 기능과 통신 기능이 융합한 응용분야나 스마트폰 상의 주변기기와 주위의 센서 모듈들의 연결을 통한 융합 스마트 응용, GPS 등을 이용한 지리적 특징을 고려한 LBS기반 스마트 응용, 개성 중심의 헬스케어나 건강 다이어트, 라이프스타일 등 개인 취향에 맞춘 맞춤형 스마트 응용에 이르기까지 실생활과 관련되는 편이성과 효율성을 강조하는 기능성 스마트 응용들이 주류를 이루고 있다.[8, 9] 또한 스마트 응용 개발을 용이하게 할 수 있도록 도와주는 미들웨어 구현 및 개발에 관한 연구 또한 본격화 되고 있다. 이러한 미들웨어는 비록 스마트 응용 개발 플랫폼 환경이나 휴대 단말의 사양이 다르더라도 약간의 수정 • 보완만으로 스마트 응용을 용이하게 개발할 수 있도록 도와주며, 최근 다양한 개발 플랫폼 환경 하에서의 미들웨어에 관한 연구가 활발하게 이루어지고 있다.[10]

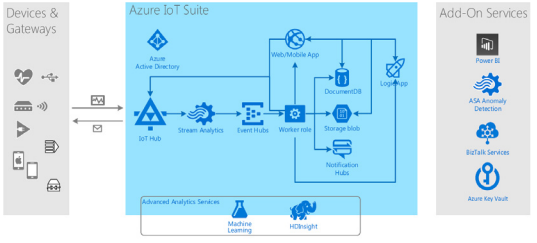

그림 2. MS Azure의 주요 컴포넌트
Fig. 2. Major Components of the MS Azure.

최근 유비쿼터스 컴퓨팅 환경에서 가장 이슈가 되고 있는 스마트 응용분야로는 단연 사물 인터넷을 들 수 있다. 지금 우리가 살고 있는 정보화 사회는 인터넷을 중심으로 주변의 모든 사물들이 서로 연결되어 "초연결 사회"를 구성하며, 이를 기반으로 인간과 사물 그리고 사물과 사물 간 서로 다양한 형태의 데이터 정보를 생산 • 가공 • 처리 • 유통을 통해 기존의 재화의 가치를 높이고 창의적인 생산 방식으로 부를 창출하는 새로운 ICT 기술을 "사물인터넷(IoT: Internet of Things)"이라 한다. 이러한 새로운 패러다임의 변화는 우리들의 기본적인 삶의 모습뿐만 아니라 주변의 환경, 안전, 공공 서비스 등 모든 분야의 모습을 바꾸어 가고 있다.[11, 12, 13] 그림 2는 미국의 MS사가 cloud 기반 IoT 플랫폼으로 제안한 Azure의 주요 컴포넌트 구성을 보여주고 있다. 사물인터넷을 구현하기 위해서는 센서 기술, 상황인식 기술, 칩 디바이스 기술, 무선 기반의 통신 기술, 경량화된 임베디드 네트워크 기술, 자율적 • 지능형 플랫폼 기술, 대용량 데이터를 처리할 수 있는 빅 데이터 기술, 데이터 마이닝 기술, 사용자 중심의 응용 및 웹 서비스 기술 그리고 보안 및 프라이버시 보호 기술 등 다양한 형태의 기술들이 요구된다. 사물인터넷은 다른 디바이스와 연결되는 새로운 장치를 만드는 개념이 아니라 기존에 사용되어 온 아날로그 분야에 ICT 기술을 접목시킴으로써 기존에 없던 새로운 가치를 창출하는 것이다. 따라서 지금까지 존재하는 모든 분야가 사물인터넷 응용 분야라 할 수 있다. 현재 가장 두드러지게 발전하고 있는 대표적인 사물인터넷 응용분야로는 헬스케어와 웰니스, 스마트 홈, 스마트 시티, 물류 • 유통 • 마케팅, 스마트 금융, 스마트 팩토리와 관련된 스마트 응용 분야를 꼽을 수 있다.[14, 15]

Ⅲ. 시스템 모델

본 연구에서는 빠르게 진화하는 휴대용 단말 기술과 더불어 최근 컴퓨팅 트랜드인 유비쿼터스 컴퓨팅 기술을 접목하여 무선 기반의 센서 네트워크를 구성하고, 다양한 기능의 센서를 이용하여 원격지에 떨어진 지역의 환경, 위생, 생태계, 사물, 사람 등을 하나의 유비쿼터스 센서 네트워크로 묶어 실시간으로 그곳의 상황정보를 수집하고 그 지역이나 환경, 사물, 사람에 대한 현재 상태를 원격으로 모니터링하고 진단하며, 필요하면 그 지역의 실시간 음성 및 영상을 받아서 분석하고 처방하여 적절한 형태의 관리와 처리가 이루어 질 수 있도록 하는 유비쿼터스 컴퓨팅 기반 원격 진단을 위한 스마트폰이나 태

블릿 PC와 같은 휴대용 단말을 위한 스마트 응용을 설계하고자 한다. 이 연구를 위해 우선 무선 기반의 유비쿼터스 센서 네트워크를 구축하고, 원격에서 그 지역의 해상도 높은 영상을 실시간으로 전송할 수 있는 네트워크 통신기능을 설계하며, 센서들로부터 실시간으로 센싱되는 데이터를 USN을 통해 수집하여 분석하고 진단하는 알고리즘을 설계한다. 또한, 이렇게 분석 처리된 데이터를 이용하여 현재의 상태를 정확히 진단하고 이에 따른 적절한 처리를 할 수 있는 시스템을 설계한다.

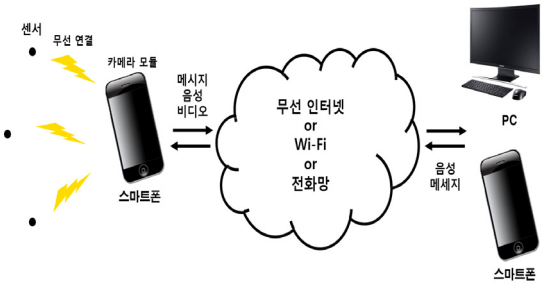

그림 3. 시스템 모델
Fig. 3. System Model.

그림 3은 본 논문에서 제안하는 유비쿼터스 컴퓨팅 환경에서의 원격 진단을 위한 스마트 응용 설계에 대한 시스템 모델을 보여준다. 그림에서 보인바와 같이 제안하는 스마트 응용은 크게 3가지의 주요 모듈로 구성된다. 첫 번째 모듈은 스마트폰에 내장되어 있거나 또는 블루투스와 같은 무선 매체를 통해 연결된 다양한 센서들로부터 실시간으로 지역의 상황정보와 스마트폰의 카메라 모듈을 이용한 실시간 영상정보를 수집하는 모듈로 이 모듈은 원격 진단의 근간이 되는 지역의 주변 상황정보를 실시간으로 수집하고 처리하는 업무를 담당한다. 두 번째 모듈은 이렇게 수집된 상황정보를 보안을 위해 적당한 암호화 과정을 거쳐 인근의 클라우드 저장소에 일시적으로 저장하거나 또는 목적지에 무선 인터넷이나 Wi-fi, 또는 전화망을 통해 전송하는 업무를 담당하게 된다. 이때 전송되는 데이터는 실시간 영상정보뿐만 아니라 스마트폰 사용자의 음성 데이터와 센서에 의해 수집된 다양한 실시간 상황정보를 포함한다. 마지막 세 번째 모듈은 네트워크를 통해 영상, 음성 및 상황정보를 수신하거나 또는 PC나 스마트폰을 통해 서버에 원격으로 접속하여 실시간 이들 정보를 제어하면서 양방향 음성 대화와 영상 정보의 일시정지, 확대, 복사, 재실행 등의 조작을 수행할 수 있도록 하여 원격으로 그 지역의 정확한 상태를 진단하고 적절한 처방을 내릴 수 있도록 하는 환경을 제공한다.

Ⅳ. 스마트 응용 설계

이 장에서는 본 논문에서 제안한 유비쿼터스 컴퓨팅 환경에서의 원격 진단을 위한 스마트 응용을 각 컴포넌트 모듈별로 주요 알고리즘을 설계하고 고찰하였다.

1. 상황정보 수집 모듈 설계

이 모듈은 주로 환경이나 상황정보를 주변에 설치된 주요 센서나 스마트폰에 장착된 센싱 장비와 카메라 모듈 등을 이용하여 관심의 대상이 되는 상황정보를 실시간으로 획득하고 주요 영상과 음성 신호를 수집하여 처리하는 모듈이다. 이 모듈은 주로 블루투스나 Zigbee 등을 이용하여 무선으로 연결된 주변의 센서들이나 스마트폰에 장착된 센싱 장비들로부터 실시간으로 발생되는 무수히 많은 빅 데이터에서 의미를 주는 관심 데이터를 선별하고 시간에 따른 그 변화 추이를 계산하여 처리하는 기능을 수행하고, 이러한 가공된 상황정보를 일시 저장하고 스마트폰의 카메라나 마이크로부터 획득된 관련 영상이나 음성 신호와 함께 상대 목적지에서 조회하고 사용 가능하게 한다.

그림 4는 안드로이드 기반 스마트폰에 장착된 카메라 모듈을 통해 주변 관련 영상을 촬영하여 상대 목적지에서 접근하고 사용할 수 있도록 카메라 모듈을 초기화하고 구동하는 알고리즘의 일부를 보여준다. 그림 5는 주변에 있는 온도, 습도, 조도 및 산소포화도 센서 등을 블루투스를 이용하여 스마트폰과 무선으로 연결하여 이들 센서로부터 발생하는 실시간 상황정보를 획득하기 위해 스마트폰의 블루투스 어댑터를 설정하고 초기화 하고, 주변의 센서장비를 정의하여 무선으로 연결을 시도하는 알고리즘의 일부를 보여준다. 그림 6은 스마트폰의 카메라 모듈을 통해 촬영되어 실시간으로 획득된 영상 정보를 원격지에서 접근하고 제어하여 사용하기 전에 스마트폰의 버퍼 저장소에 일시 저장하고 획득된 영상을 보여주기 위한 프리뷰 알고리즘의 일부를 보여준다.

2. 상황정보 전송모듈 설계

이 모듈은 상황정보 수집 모듈로부터 수집 가공된 데

Camera.java

```
package camera;
// import Android Module
import android.app.Activity;
import android.hardware.Camera;
import android.os.Bundle;
import android.util.Log;
import android.view.SurfaceHolder;
import android.view.SurfaceView;
import android.view.Window;
import android.view.WindowManager;
-중략-
public class Camera extends Activity {
    private String TAG = "Camera";
    // default RTSP command port is 554
    // private int SERVER_PORT = 8080;
    // 변수 설정
    private SurfaceView mCameraPreview;
    private SurfaceHolder previewHolder;
        private Camera camera;
-중략-
    @Override
        public void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            상위 클래스의 onCreate 메소드 호출
            디버그 내용 출력
            requestWindowFeature(Window.FEATURE_NO_TITLE);
                Window win = getWindow();
                프레임의 타이틀 제거한 윈도우 획득
                윈도우 켜진 상태 유지
                상태바와 타이틀바 숨기기
            setContentView(R.layout.cameranativecodecs);
            액티비티 화면을 설정
-중략-
```

그림 4. 스마트폰 카메라 구동 알고리즘
Fig. 4. Smartphone Camera Operation Algorithm.

이터를 상대 목적지로 전송하는 업무를 수행하는 모듈로 실시간으로 수집하고 가공 처리된 센싱 상황정보와 관련 음성 및 영상 정보를 이 모듈에서는 저장할 수 있는 버퍼를 두어 일시적으로 저장한 후, 유무선 통신 전화망을 통해 스마트폰을 이용하여 목적지에 전송하거나 원격으로 접속하여 제어하고 스트리밍하게 된다. 보안을 위하여 전송 전 모든 관련 상황 데이터는 암호화가 필수적이지만 이에 대한 처리와 알고리즘은 그 방법이 다양하고 방대하므로 본 논문에서는 고려하지 않기로 한다. 상황정보와 영상 및 음성에 대한 전송과 스트리밍은 지금까지 통상적으로 사용하는 패킷 전달용 프로토콜인 TCP나 UDP를 사용하는 대신 실시간 전송용 프로토콜인 RTP, RTCP, RTSP 등을 사용하게 되는데 그 주요 이유는 TCP는 재전송 메커니즘이 있어 지연을 유발하고 멀티캐스트를 지원하지 않으며, 혼잡제어 역시 실시간 전송에 적합하지 않기 때문이다. 또한 UDP의 경우 빠른 데이터 전송을 지원하기는 하지만 데이터 그램 방식으로 패킷들이 독립적으로 전송되어 상대 목적지 도착 순서를 보장하기 힘들다. 따라서 실시간 전송을 위한 RTP를 이용하여 비디오나 오디오 패킷을 전송하는데 이 프로토콜은 데이터를 빠르게 전송하는 UDP의 특성을 이용하나 QoS 보장 및 신뢰성 제공을 하지 못하는 단점이 있으며 단방향 통신만을 제공한다. 한편, RTCP는 실시간 전송제어 프로토콜로 RTP의 QoS를 유지하기 위해 함께 사용되며

BluetoothChat.java

```
public class BluetoothChat extends Activity {
private static final String TAG = "BluetoothChat";
private static final boolean D = true;
// BluetoothChat 서비스 핸들러로 부터 보내는 메시지 타입
-중략-
public static final int MESSAGE_DEVICE_NAME = 4;
// BluetoothChat 서비스 핸들러로부터 수신되는 주요 이름
public static final String DEVICE_NAME = "device_name";
// Intent 요청 코드
private static final int REQUEST_CONNECT_DEVICE_SECURE = 1;
private static final int REQUEST_CONNECT_DEVICE_INSECURE = 2;
// 레이아웃 뷰
private ListView mConversationView;
// 연결된 장치의 이름
private String mConnectedDeviceName = null;
// 통신 쓰레드를 위한 Array adapter
private ArrayAdapter<String> mConversationArrayAdapter;
// 발생 메시지를 위한 스트링 버퍼
private StringBuffer mOutStringBuffer;
// 지역 블루투스 어댑터
private BluetoothAdapter mBluetoothAdapter = null;
// 통신 서비스를 위한 멤버 객체
private BluetoothChatService mChatService = null;
    @Override
    public void onCreate(Bundle savedInstanceState) {
    // 지역 블루투스 어댑터 연결
    mBluetoothAdapter = BluetoothAdapter.getDefaultAdapter();
    // 만약 null이면 블루투스 지원안됨
    if (mBluetoothAdapter == null) {
        finish(); return;
}}
-중략-

private void setupChat() {
// 통신 쓰레드를 위한 어레이 어댑터 초기화
mConversationArrayAdapter = new ArrayAdapter<String>(this, R.layout.message);
mConversationView = (ListView) findViewById(R.id.in);
mConversationView.setAdapter(mConversationArrayAdapter);
// 블루투스 연결을 위한 BluetoothChat 서비스 초기화
mChatService = new BluetoothChatService(this, mHandler);
// 발생 메시지를 위한 버퍼 초기화
mOutStringBuffer = new StringBuffer("");
}
-중략-

private void connectDevice(Intent data, boolean secure) {
// 장치의 MAC 어드레스 획득
String address = data.getExtras()
.getString(DeviceListActivity.EXTRA_DEVICE_ADDRESS);
// 블루투스 장치 객체와 연결
BluetoothDevice device = mBluetoothAdapter.getRemoteDevice(address);
// 장치와 연결 시도
mChatService.connect(device, secure);
}
-중략-
```

그림 5. 블루투스로 센서 연결 알고리즘
Fig. 5. Sensor Connection Algorithm with Bluetooth.

데이터 전송을 감시하고 세션 관련 정보를 전송하며, 양방향 통신을 제공한다. 아울러, RTSP는 실시간 스트리밍 프로토콜로 응용계층에서 사 용할 수 있는 프로토콜이며 스트리밍 시스템에서 사용된다. 이 프로토콜은 실제 미디어 스트리밍 데이터를 전송하는 것이 아니라 멀리 떨어진 미디어 서버를 원격으로 접속하고 제어하며 양방향 통신을 지원하는데 현재 네트워크상에서 높은 신뢰성을 보장한다. 본 논문에서는 영상과 음성 정보의 스트리밍을 RTSP를 이용하여 설계하였다. 상황정보와 연산 및 음성 정보에 대한 전송은 각기 다른 채널을 통해 독립적으로 이루어지도록 설계하였으며 각 미디어의 조작 역시 독립적으로 이루어 질 수 있도록 설계하여 상황정보에 대한 모니터링을 자유롭고 편리하게 수행 할 수 있도록 구성하였다.

CameraPreview.java

```
previewHolder.setType(SurfaceHolder.SURFACE_TYPE_PUSH_BUFFERS);
// 비디오 레코더 설정 및 생성
            outPlayer = new RtspVideoRecorder("h263-2000");
            outPlayer = new RtspVideoRecorder("h264");
            outPlayer.open();
      @Override
            public void onResume() {
            // 비디오 재실행 모듈, 비디오 서버 실행 개시
            try {
            // SDF 파일 생성을 위한 비디오 인코더 초기화
            if (streamer == null) {
  streamer = new RtspServer(RtspConstants.SERVER_PORT, rtspVideoEncoder);
            new Thread(streamer).start(); } // 비디오 서버 설정 및 시작
- 중략 -
            super.onResume(); // 상위 클래스의 onResume 메소드 호출 }
      @Override
            public void onPause() { // 비디오 일시정지 모듈
            if (streamer != null) streamer.stop(); // 비디오 서버 정지
            super.onPause(); // 상위 클래스의 onPause 메소드 호출 } // 일시정지
// SurfaceHolder callback 3 단계
SurfaceHolder.Callback surfaceCallback = new SurfaceHolder.Callback() {
// create 상태: startPreview()호출 초기화,PreviewCallback()실행, 쓰레드 영상실행
public void surfaceCreated(SurfaceHolder holder); // surface 생성
// change 상태: 프리뷰 크기 초기화, PreviewDisplay() 설정 후 startPreview() 실행
public void surfaceChanged(SurfaceHolder holder, int format, int w, int h) {
            initializePreview(w, h);// 프리뷰 초기화
            startPreview();          // 프리뷰 시작 }// 프리뷰 크기 초기화 및 설정
// Destroy 상태: 카메라 모듈 해제
public void surfaceDestroyed(SurfaceHolder holder) {
            if (inPreview) { camera.stopPreview(); // 프리뷰 정지 }
            camera.setPreviewCallback(null);
            camera.release(); // 카메라 해제
            camera = null; // captureThread 정지
            outPlayer.stop(); // 비디오 레코더 중지
            inPreview = false; // 동작되지 않는 상태
            cameraConfigured = false; // 동작되지 않는 상태 } // 프리뷰 소멸
}; // SurfaceHolder.Callback
// 카메라와 프리뷰의 가용성 검사
private void initializePreview(int width, int height) {
            if (camera != null && previewHolder.getSurface() != null) {
                  try { // 카메라 프리뷰를 위한 SurfaceView 제공
                  camera.setPreviewDisplay(previewHolder);
                  if (!cameraConfigured) {
                  Camera.Parameters parameters = camera.getParameters();
                  parameters.setPreviewSize(mPreviewWidth, mPreviewHeight);
                              camera.setParameters(parameters);
                              cameraConfigured = true; // 동작 준비 상태 }
                  } } // 프리뷰 초기화
private void startPreview() { //프리뷰 시작
            if (cameraConfigured && camera != null) {
                  // onPreviewFrame() 활성화
                  // camera.setPreviewCallback(cameraPreviewCallback);
                  camera.setPreviewCallback(outPlayer); // captureThread 시작
                              outPlayer.start(); // 비디오 레코더 시작
                              camera.startPreview(); // 프리뷰 시작
                              inPreview = true; // 동작 상태
                  } } // 프리뷰 시작
```

그림 6. 카메라 모듈 영상 프리뷰 알고리즘
Fig. 6. Preview Algorithm for Camera Module Video.

3. 상황정보 수신 및 조작 모듈 설계

이 모듈은 상황정보 전송 모듈로부터 전송되는 상황 데이터와 관련 영상 및 음성 데이터에 대한 독립적인 수신과 처리를 실시간으로 수행하게 된다. 우선 수신되는 상황 정보에 대한 처리는 기존에 수신된 상황정보에 대한 변화량을 기준으로 그 변화 추이를 알기 쉽게 인식할 수 있어 주변 환경에 대한 신속하고 정확한 진단이 이루어 질 수 있도록 설계하였고, 관련 영상이나 음성 데이터의 조작과 처리는 실시간 스트리밍 방식으로 이루어지게 하면서 버퍼에 저장 관리 할 수 있도록 설계하였다. 이렇게 함으로써 실시간으로 진행되고 있는 영상과 음성 데이터에 대한 일시정지, 화상 복사 및 확대, 과거 영상 재생 및 되감기 기능 등의 간단한 조작이 가능하게 설계하여 상대 목적지에서 원격의 지역이나 상황에 대한 진단

MediaPlayer.java

```
@Override
    protected void onStart() {
        super.onStart();          // 상위 클래스의 onStart 메소드 호출
        if(fragment != null){
            Button testButton = fragment.getButton();
            testButton.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    // LocalService에 바인딩
                    if(mMmsPlayerService != null){
                        mMmsPlayerService.start();
                        Log.d(TAG, mMmsPlayerService.testFunc());
                    }          // 버튼 클릭 시, mMmSPlayerServie 시작
                }  });  }
    }
-중략-
@Override
    protected void onStop() {
        super.onStop();          // 상위 클래스의 onStop 메소드 호출
        // Unbind from the service
        doUnbindService();       // 액티비티를 서비스에 해제
    }
-중략-
// 서비스 바인딩을 위해 callbacks 정의하여 bindService()에 전달
    private ServiceConnection mConnection = new ServiceConnection() {
@Override
   public void onServiceConnected(ComponentName className,IBinder service) {
    // LocalService에 바인딩하고 IBinder를 전달하여 LocalService 인스턴스 획득.
     MmsPlayerService.LocalBinder binder = (MmsPlayerService.LocalBinder)
service;
        mMmsPlayerService = binder.getService();
        mIsBound = true;
        }  // 서비스 연결
@Override
    public void onServiceDisconnected(ComponentName arg0) {
        mIsBound = false;
        }  // 서비스가 연결되지 않음
    };
    private void doBindService() {
        // 서비스와 연결 설정.
        Intent intent = new Intent(this, MmsPlayerService.class);
        bindService(intent, mConnection, Context.BIND_AUTO_CREATE);
        mIsBound = true;
    }       // 서비스에 연결_doBindService 메소드 정의

    private void doUnbindService() {
        if (mIsBound) {
            // 기존 연결 끊음.
            unbindService(mConnection);
            mIsBound = false;
        }
    }       // 서비스에 연결 끊음_doUnbindService 메소드 정의
}
```

그림 7. 수신 영상 조작 알고리즘
Fig. 7. Received Video Operation Algorithm.

의 정확도를 높일 수 있게 섬세한 원격 조작이 가능하도록 하였다. 그림 7은 원격지로부터 보내오는 실시간 스트리밍 영상 정보를 제어하여 재생하고 버퍼에 이 영상 정보를 저장하여 일시정지, 되감기, 정지화상 복사와 확대 등 간단한 영상 조작을 할 수 있는 기능을 수행하는 알고리즘의 일부를 보여준다.

V. 결 론

20세기의 정보통신 혁명은 컴퓨터들 간의 연결 그리고 컴퓨터와 사람과의 연결을 통해 사회 전체로 급속히 확산되었으며, 21세기 스마트폰에 의한 기술 융합을 통해 유비쿼터스 컴퓨팅 기술을 촉발시켰다. 최근에는 컴퓨터간 뿐만 아니라 사람과 컴퓨터 나아가 기계, 사물, 시스템 등 모든 사물을 네트워크로 연결시키는 무한 확장

의 개념으로 모든 사물들을 연결시키는 초연결 사회를 통해 유비쿼터스 컴퓨팅 혁명을 실현해 가고 있다. 특히 다양한 센서를 이용한 유비쿼터스 센서 네트워크 기술의 발달과 시간이나 장소에 구애 없이 자유롭게 이동하면서 원하는 경우 언제든지 네트워크에 접속하여 시스템의 서비스를 받을 수 있는 유비쿼터스 컴퓨팅 기술이 급속히 발전하면서 첨단 스마트 폰을 이용한 다양한 형태의 스마트 응용들이 속속 개발되어 보급되고 있는 실정이다.

본 논문에서는 이러한 특징을 가지는 유비쿼터스 컴퓨팅 환경에서 스마트폰이나 태블릿 PC 등의 휴대용 단말을 이용하여 원격지에 떨어져 있는 지역의 생태계나 사물 또는 사람의 실생활 환경, 의료, 헬스케어 등 상황정보와 실시간 음성 및 영상을 기반으로 그 지역, 사물, 또는 사람의 현재 상황에 대한 상태를 원격으로 실시간 진단하고 처리할 수 있는 스마트 응용을 설계하였다. 향후 연구과제로는 본 논문에서 제안한 스마트 응용에 대한 구현과 개인정보의 보안 등을 고려한 암호화에 관한 것이다.

References

[1] S. O. Yang, S. S. Kim, K. S. Jung, Introduction to Ubiquitous Computing, Sangrung Pub. Co., pp. 572, 2012.
[2] John, A., Adamic, L., Davis, M., Nack, F., Shamma, D. A., and Seligmann, D. D, "The future of online social interactions: what to expect in 2020", Proceedings of the 17th International Conference on WWW, 2008.
[3] Mark Weiser, "Hot Topic : Ubiquitous Computing", IEEE Computer, pp. 71 - 72, October, 1993.
[4] K. S. Jung, Introduction to Ubiquitous Computing, Hanbit Media, pp. 668, 2010.
[5] Xerox PARC Mark Weiser, "Computer Science Challenges for the Next 10 Years", http://sandbox.xerox.com/weiser/10years/sld001.htm.
[6] http://www.androidpub.com/1305
[7] http://inside.chosun.com/site/data/html_dir/2014/04/15/2014041500805.html
[8] J. Stark, "Building Android Apps with HTML, CSS, and JavaScript", O'Reilly Media Inc., 2010.
[9] S. J. Oh, "A Study on Characteristics of Smart Phone Camera Module for Measuring a Shooting Object", The Journal of the Institute of Internet, Broadcasting and Communication(JIIBC), Vol. 12, No. 5, pp. 99 - pp. 105, 2012.
[10] S. J. Oh, "Design of a Middleware for Android-based Smartphone Applications", The Journal of the Institute of Internet, Broadcasting and Communication(JIIBC), Vol., 12, No. 2, pp. 111 - pp. 118, 2012.
[11] H. Y. Kim, Internet of Things : Concept, Implementation, and Business, HongReung Science Pub. Co., 2014.
[12] D. Lake, A. Rayes, M. Morrow, "The Internet of Things", The Internet Protocol Journal, vol. 15, no. 3, September 2012.
[13] J. Gubbi, R. Buyya, S. Marusic, M. Palaniswami, "Internet of Things(IoT): A vision, architectural elements, and future directions", Journal on Future Generation Computer Systems, vol. 29, no. 7, pp. 1645-1660, 2013.
[14] P. Pahlevani, M. Hundeboll, M. Pedersen, D. Lucani, H. Charf, F. Fitzek, H. Bagheri, M. Katz, "Novel Concepts for Device-to-Device Communication Using Network Coding", IEEE Communications Magazine, pp. 32-39, 2014.
[15] L. Atzori, A. Lera, G. Morabito, "The Internet of Things: A Servey", Computer Networks, vol. 54, no. 15, pp. 2788-2805, 2010.

저자 소개

오 선 진(종신회원)

- 제 6권 제2호 참조
- 현재 : 세명대학교 정보통신학부 교수

<주관심분야 : 스마트 응용, IoT, 빅데이터, 모바일컴퓨팅, USN 등>

※ 이 저작물은 2015학년도 세명대학교 연구년 지원에 의한 연구임.