JDCS
디지털콘텐츠학회논문지
Journal of Digital Contents Society
Vol. 21, No. 11, pp. 1921-1926, Nov. 2020

Check for updates

휴대용 생체정보측정 장치를 적용한 개인건강정보관리 프로그램 개발

임 병 두[1] · 천 민 우[2*]
[1]동신대학교 대학원 전기전자공학과 석사과정
[2*]동신대학교 보건행정학과 교수

Development of a Personal Health Information Management Program Using Mobile Biometric Information Measuring Device

Byeong-Doo Yim[1] · Min-Woo Cheon[2*]

[1]Master's Course, Department of Electron Electronic Enginering, Graduate School, Dongshin University, Naju, Korea
[2*]Professor, Department of Health Administration, Dongshin University, Naju Korea

[요 약]

인터넷 및 스마트 기기 발달로 인하여 개인이 직접 건강 및 질병에 대한 정보를 검색하고 활용하는 비율이 높아지고 있다. 고령자 및 만성질환자의 증가로 사회적 부담이 높아지고 있어 개인 건강 관리의 중요성이 강조되고 있다. 본 연구에서는 휴대용 생체정보관리 기기를 통해서 수집된 정보를 PHR에 적용할 수 있도록 App.과 서버 프로그램을 개발하였다. App.은 휴대용 생체정보관리 기기를 통해 수집된 심박, 산소포화도, 체온과 개인이 입력한 건강정보를 쉽게 볼 수 있도록 하여 개인 건강 관리가 수월하도록 하였다. 또한, App.으로 수집된 정보는 서버에 자동 저장되도록 구성하여 실시간 혹은 특정 기간 동안의 개인 건강을 분석할 수 있도록 구성하였다.

[Abstract]

With the advancement of Internet and smart devices, an increasing number of people are retrieving and utilizing information on health and diseases on their own. In particular, the increase in the number of elderly and chronic patients has increased the social burden of healthcare costs; hence, the importance of personal health management is being emphasized. This study developed an application and a server program to apply data, which are collected from mobile biometric readers, to personal health records. The application facilitated personal health management by providing convenient displays not only for pulses, blood oxygen levels, and body temperatures collected through a portable biometric reader but also for other health information that is manually entered by each user. In addition, the server was configured to store data collected by the application automatically so that an individual's health status could be analyzed in real time or for a certain period.

색인어 : 개인건강정보, 생체정보, App, 심박, 산소포화도

Key word : Personal health information, Biometric information, Application, Heart rate, Oxygen saturation

http://dx.doi.org/10.9728/dcs.2020.21.11.1921

This is an Open Access article distributed under the terms of the Creative Commons Attribution Non-CommercialLicense(http://creativecommons.org/licenses/by-nc/3.0/) which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.

Received 06 November 2020; **Revised** 19 November 2020
Accepted 19 November 2020

***Corresponding Author; Min-Woo Cheon**

Tel: +82-61-330-3212
E-mail: mwcheon@dsu.ac.kr

Ⅰ. Introduction

The advancement of Internet and smart devices has facilitated information retrieval and use. Accordingly, there has been a rapid growth in services that enable individuals to search for and utilize information on health and diseases directly. In particular, the increase in the number of elderly and chronic patients has increased the burden of healthcare costs on individuals and society. Hence, u-health is being actively studied as an alternative system for personal health management and preventive healthcare for the elderly and those suffering from chronic diseases [1~2]. To improve quality of life, personal health record (PHR) services are required that enable individuals to collect and manage various health and medical information and to maintain health records. A PHR service is a personal health information management service that enables users to confirm their own personal information and medical records, when necessary, and enter such information directly as well [3–5]. In a survey conducted in South Korea on PHR services, 59.8% of the respondents felt that they would choose to use PHR services and 27.8% of the respondents said “Yes” to the question on whether they wanted to use a paid PHR service [6]. PHRs are constructed on the following four types of bases: paper, computer, Internet, and mobile device. Depending on the types of electronic medical recording systems, PHRs are also classified into stand-alone, EMR(Electronic Medical Records)-tethered, and interconnected PHRs [7]. A PHR can be considered an intersection of healthcare information technology and consumer participation in health and well-being. In the promotion of preventive medicine or medical regimens against so-called “modern diseases,” PHRs are being studied as a solution to the healthcare crisis associated with the ever-growing increase in healthcare costs and the demand for better medical and preventive services [8].

This study developed an application and a server program to enable individuals to manage their personal health information conveniently. Data collected from each mobile biometric reader are automatically stored in the server, and users can enter their personal health information using the application. Section 2 shows the app and server configuration diagram, and Section 3 shows the app and server development details.

Ⅱ. App and Server Configuration

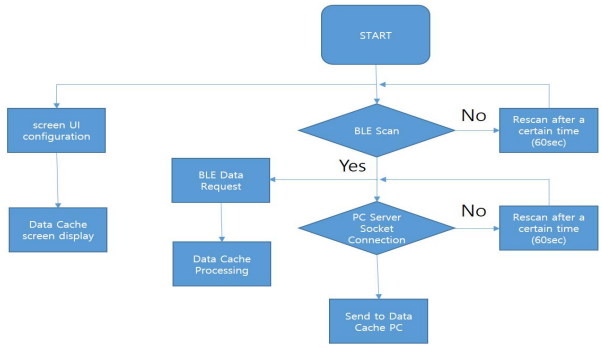


그림 1. 개인건강정보관리를 위한 App.의 구성도
Fig. 1. Flow diagram of the application for personal helth information management

Figure 1 presents a flow diagram of the application for personal health information management. An XBee was applied for communication between a sensor board, which measured a user’s pulses, blood oxygen levels, and body temperatures, and the main controller of a body heat keeper, which collected the measurement data. An XBee module has many advantages such as low power consumption, wide device layout, and short latency. Furthermore, as a high-level communication protocol that uses small and low-power digital radios based on IEEE 802.15.4-2003, XBee is the standard for personal wireless local area networks. Moreover, as an XBee is easy to implement and includes the operating/sleep modes, it requires low installation and maintenance costs and ensures stability. As a very small protocol stack is used, an XBee is suitable for transmitting and receiving a small amount of data and has a high node density per network.

Figure 2 presents a flow diagram of the server that stores, enters, and manages the collected data. Bluetooth communication was used for the mainboard and the application that collected measurement data. For the Bluetooth communication, Bluetooth Classic, which uses a master–slave relationship between devices, was adopted. The Bluetooth low energy (BLE) communication protocol was adopted to solve the issue of the battery draining quickly while being connected in the Bluetooth Classic communication.

The application was implemented in Java and Kotlin, and the server was implemented in C# and WPF(WebSphere Partition Facility). The application was developed using Android Studio, and the server was established using Visual Studio.

The purpose of designing and developing the server was to input and confirm the PHRs. A UI(User Interface) was declared for logger application, and the logger was written. Data parsing was conducted based on a data conversion (App -> PC) protocol.

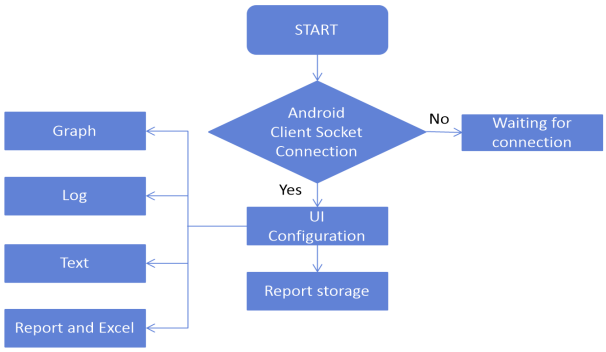


그림 2. 서버 구성도
Fig. 2. Flow diagram of the server

As for reports, a report UI was configured, and report data were processed.

The server socket was connected to port #1553, and then, the transmission, reception, and connection events were registered and processed. A function was designated for the thread processing of events per second, and the log data were processed every 15 s. A box list for displaying the logs was declared, and the rich box style was fixed with a margin of 0 and a line height of 5 so that data could be inserted into the box list.

While a function thread operates for an actual log writing section, data for logs should be written into the rich box. Accordingly, invoke processing was first performed to execute the UI thread. When the corresponding UI thread execution ended, a particular log folder was designated for the input, output, and storage of data. The data were also processed to be displayed in the rich box. Data received in a hex byte array, which is a syntax for processing received data, were converted into a string. Subsequently, the data thus converted were processed again so that they became available to users. Finally, the graphs and data were processed on the screen. A graphic corresponding to the report UI was processed by applying a chart function. The corresponding graph was processed by designating the position and setting the corresponding (legend) value to a target in white font. The data time was comparatively processed to be added to the list, and the data entering the report were processed so that they could follow a routine at a specific time.

As one of the socket programming types, TCP(Transmission Control Protocol) communication is referred to as a stream communication protocol. As TCP communication operates only when both sockets are connected, it is also called a connection-oriented protocol. Because the TCP protocol is reliable, data of the transmitting end arrive at the receiving end in order without loss. In addition, the receiving and transmitting ends should be linked to each other in advance, and data should be exchanged in the connection order. Such a connection-oriented type enables reliable communication in which, once both ends are connected to each other, data are delivered from the transmitting end to the socket of destination in order until the connection is terminated. For a reliable TCP connection, it is necessary to understand the method of using libraries and the operating sequence in TCP programming accurately. As Java provides relevant classes in the java.net package to facilitate TCP programming, those classes of the package need to be used appropriately.

The TCP sockets were processed as a group to block Nagle's algorithm from executing. The corresponding time-out was set to 5 s for initializing the channel. The routine for data processing was changed by processing encode/decode in Korean. Java introduced the concept of socket to network communications. A socket represents the endpoints of a network and provides a reading/writing interface irrespective of how data are actually transmitted. As the network and transport layers are encapsulated, a program can be developed without considering the two layers. Bill Joy developed the socket programming at the University of California, Berkeley. Java integrated the socket as an essential library for portability and cross platform network programming. Socket and server socket classes are provided to support TCP of the TCP/IP(Internet Protocol address) layers. The client attempts a connection to the TCP server by creating a socket object, and the server listens to the TCP connection by creating a socket server object. In turn, the client and the server become connected to each other. When the input and output streams are created using sockets, the TCP socket provides a byte stream communication between two networks, and the socket classes provide methods for reading bytes and for writing bytes. These two methods enable communication between the client and server.

```
fun setting(ip : String) {
client = NettyTcpClient.Builder()
.setHost(ip)
.setTcpPort(1553)
.setMaxReconnectTimes(100)
.setReconnectIntervalTime(10 * 1000)
 rank: ms
.setSendheartBeat(false)
.setHeartBeatInterval(5 * 1000)
 rank: ms
.setHeartBeatData("I'm is HeartBeatData")

.setIndex(0)

.build()

client?.setListener(this) // TCP receiving end
}
```

그림 3. 데이터가 APP에서 PC로 전송되는 프로토콜
Fig. 3. shows a protocol in which data is transferred from APP to a PC.

Figure 3 shows a protocol in which data is transferred from APP to a PC.

Among the open sources, com.github.mikephil.charting was used to initialize a line chart and draw a UI. The corresponding line chart was processed to have a specific position and a fixed

width/height. UI thread processing was conducted every 2 s using a timer to add data portions to the graphs. An average calculating operation was used to calculate the average data from the data thus added, and the average data obtained were displayed on the screen. Devices have a protocol stack for Bluetooth communication. In general, a protocol, which is a set of rules, needs to be defined for network communications. A protocol stack is a group of protocols that are defined and stacked in layers. When Bluetooth signal packets are transmitted or received, the packets are analyzed or generated through such a protocol stack. Data were processed using a BLE scanner to be executed in the Android Lollipop version or above. The device name and RSSI(Receiver Signal Strength Indicator) were received from the corresponding BLE suite, and the data were processed as received. When the scanned device was correct, the data were transferred to the data processing section. Then, the data were represented by activating the Gatt in BLE. The specific values of BLE data were managed, monitored, and processed in a Gatt server, and the corresponding devices were automatically registered in the Gatt.

Addresses and ports for communication with the server were designated and configured, and various other items, including reconnection, were processed and built as optional items. The socket connection was oriented through the optional items that were built. Figure 4 shows the data parsing processing protocol.

```
void Received(object sender, object target, SocketServer.ServerEventArgs e)
    {
        //int OVERFLOW = 100000000;
        //SocketSeConnectClient client = (SocketSeConnectClient)sender;
        try
        {
            // Transform Hex data into String
            String msg = Converter.HexToString(e.Message);

            // Apply UI Threads
            Application.Current.Dispatcher.BeginInvoke(new Action(() =>
            {
                // Set First Start and End in Log Box
                var                range_Hex                =                new
TextRange(_this.LogBox_Hex.Document.ContentStart,
_this.LogBox_Hex.Document.ContentEnd);
                var                range_Str                =                new
TextRange(_this.LogBox_Str.Document.ContentStart,
_this.LogBox_Str.Document.ContentEnd);
                // Initalize when data length exceeds 600
                if                (range_Hex.Text.Length                >                600)
_this.LogBox_Hex.Document.Blocks.Clear();
                if                (range_Str.Text.Length                >                600)
_this.LogBox_Str.Document.Blocks.Clear();

                // Process log data raw value in log box (based on time value)
                _this.LogBox_Hex.AppendText(String.Format("REV[{0}]  :  {1}\r",
DateTime.Now.ToString("HH:mm:ss"), Converter.HexByteToIntString(e.Message)));
                _this.LogBox_Hex.ScrollToEnd();

                _this.LogBox_Str.AppendText(String.Format("REV[{0}]  :  {1}\r",
DateTime.Now.ToString("HH:mm:ss"), msg));
                _this.LogBox_Str.ScrollToEnd();
            }));

            Message m = new Message(msg);
            List<Segment> segList = m.Segments();

            // Parse this message
            bool isParsed = false;
            try
            {
                // Parsing messages
                isParsed = m.ParseMessage();
            }
            catch (Exception ee)
            {
                Console.WriteLine("[Parse] ERROR: {0} / {1} / {2}", ee.Source,
ee.Message, ee.StackTrace);
            }

            // Process if there are no more than 17 OBX data counters among
            parsing values
            if (m.Segments("OBX").Count < 17) return;

            // Importing data
            lot.Id = m.GetValue("PID.2.1");
            lot.Name = m.GetValue("PID.4.1") + m.GetValue("PID.4.2");
            lot.Sex = m.GetValue("PID.7");

            // Age Date
            DateTime dt = DateTime.ParseExact(m.GetValue("PID.8"), "yyyyMMdd",

            // value processing for age
            int age = (DateTime.Now.Year - dt.Year) + 1;
            if (age < 0) age = 0;
            lot.Age = age;

            // Other data processing (height, weight, pulse... )
            lot.Height                =
System.Convert.ToInt32(m.Segments("OBX")[0].Fields(5).Value);
            lot.Weight                =
System.Convert.ToInt32(m.Segments("OBX")[1].Fields(5).Value);
            lot.Pulse                 =
System.Convert.ToInt32(m.Segments("OBX")[2].Fields(5).Value);
            lot.Blood_Oxygen          =
System.Convert.ToInt32(m.Segments("OBX")[3].Fields(5).Value);

            lot.Body_Temperature      =
System.Convert.ToDouble(m.Segments("OBX")[7].Fields(5).Value);
            lot.In_Circulation_Temp   =
System.Convert.ToDouble(m.Segments("OBX")[8].Fields(5).Value);
            lot.Out_Circulation_Temp  =
```

그림 4. data parsing 처리
Fig. 4. data parsing processing

For data transfer, the data were encapsulated, and various information items, such as the corresponding name and medicine information, were designated and managed. The data were processed to be reanalyzed by a protocol designated by the PC. Finally, the data were delivered to a socket that was already connected.

Ⅲ. App. & Server Details of development

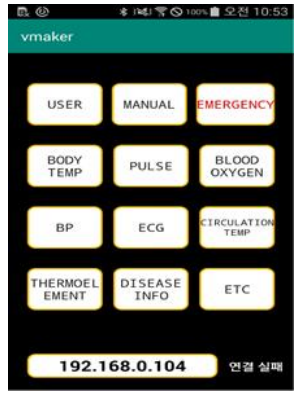
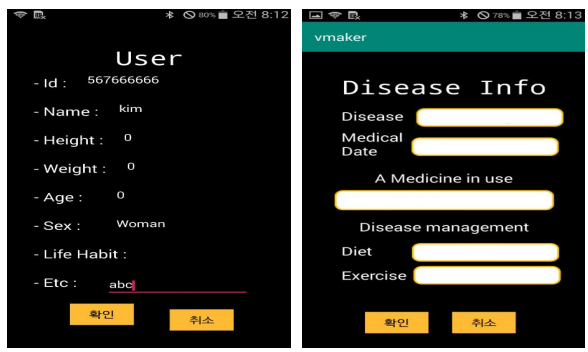


그림 5. 개인건강기록 App.
Fig. 5. PHR application

Figure 5 depicts the application for recording personal health information that was designed and developed in this study. The application was designed such that biometric data measured using a wearable device could be sent to the main controller through an XBee module, and the corresponding data could be delivered from the main controller to the application through Bluetooth. In addition, the design of the application also allowed each user to enter and manage personal health information directly.

The application was designed to input the basic personal information and other clinical information of users and to activate emergency stoppage of a body heat keeper. The application could also display the measurement data of body temperature, pulse, blood oxygen, etc.

Figure 6 presents examples of data that can be input using the application. Figure 6 (a) depicts the items of user information on the page of User. Basic user information such as ID, name, height, weight, and age can be entered. Figure 6 (b) depicts the items entered by the user on the page of Disease Info. This page is designed to enable the input of items such as the user's diseases, the medical date, medicines in use, and exercise.



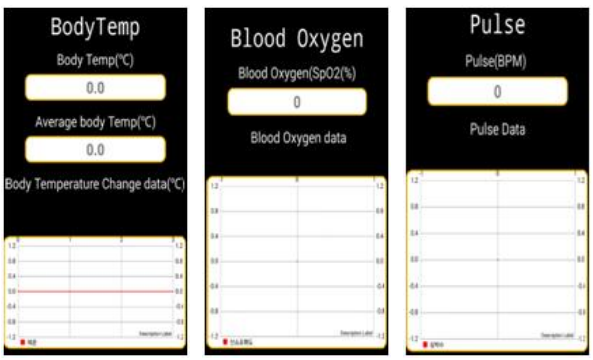
(a) (b)

그림 6. App. 구성 (a)User, (b) Disease Info
Fig. 6. Configuration of the application (a) User, (b) Disease Info

Figure 7 presents examples of the data that are measured by the wearable device and are sent to the application. Figure 7 (a) depicts the information on the page of Body Temp. The current and average temperatures of the user are displayed using numerical values and the graph.

Figure 7 (b) depicts the information displayed on the page of Blood Oxygen. The blood oxygen level of the user is shown in real time. Figure 7 (c) presents information corresponding to the user's pulse. In other words, the heart rate of the user is displayed using numerical values and the graph in real time.

Figure 8 depicts the server program that stores and manages personal health information. When data are delivered from the application to the server, the patient information, body temperatures, blood oxygen levels, and pulses can be checked on the PC. The server was also designed to let each patient enter and directly check their own information. In particular, the server was designed to display data clearly by date on graphs clearly.



(a) (b) (c)

그림 7. App. 구성 (a)체온, (b)산소포화도, (c)심박
Fig. 7. Configuration of the application (a) Body Temp (b) Blood Oxygen and (c) Pulse

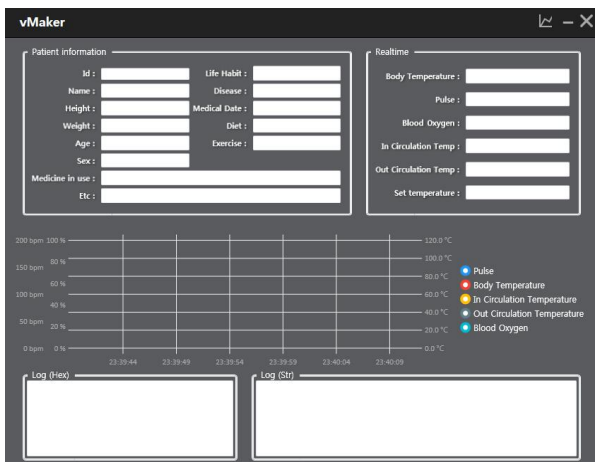


그림 8. PC 구성
Fig. 8. PC configuration

Ⅳ. Conclusion

This study designed and developed an application and a server for collecting, analyzing, storing, and managing personal health data that are either measured through a wearable device or manually entered by each user.

The application and the server were configured as follows. Pulses, blood oxygen levels, body temperatures, and other data, which are measured through a wearable device in real time, were sent to the application via Bluetooth communication. Next, the data were transmitted to and stored in the server through a smart device in which the application was installed. The design of the application also enabled each user to enter their personal information and other disease and medicine information. Considering the system linkage with medical institutions, the disease and medicine information were designed to be entered both into the application and the server. If more users wearing mobile biometric readers can manage their personal information by themselves with the help of such a health information management program, it is expected that medical institutions could provide faster and more accurate diagnostic and treatment services.

감사의 글

This paper is that revised and supplemented the research results supported by the Small and Medium Business Promotion Agency (S2759360) in 2019.

참고문헌

[1] Yoo B S, Kim D H, Cho K R, Kim S H, Oh S J, Cho J S, "u-DailyCare : Design of a Health Management System for Chronic Illness Patients," Communications of the Korean Institute of Information Scientists and Engineers, Vol. 38, No. 1, pp. 146-149, 2011.

[2] Kim D H, Kim S H, Cho K R, Cho J s, "u-DailyCare : Design and Implementation of prevention and Management Sevice for chronic and Adult disease," Communications of the Korean Institute of Information Scientists and Engineers, Vol. 39, No. 1, pp. 197-199, 2012.

[3] Park Y M, Oh Y H, "A Study on The Integration of Healthcare Information Systems based on SOA for PHR

services," Journal of the Institute of Electronics Engineers of Korea, pp. 29-35, 2011.
[4] Park Y M, "A Study on Medical Information System based on P2P for PHR Service," Journal of Advanced Information Technology and Convergence, Vol. 11, No. 12, PP. 123~132, 2013.
[5] Lee M K, Hwang H J, "Design of Secure Personal Health Record Management Systems," Journal of Advanced Information Technology and Convergence, Vol. 13, No. 8, pp. 71-80, 2015.
[6] Kim S Y, Kim H R, Bae J B, Kim Y, "The Consumers' perceptions and reguirements for personal health records in Korea," JKorSoc MedInformatics, Vol. 15, PP. 273~284, 2009.
[7] Heo Y, Yang J S, Park K H, Cha S J, Choi D J, Hwang K H, "PD issue report: personal health record (PHR) service technology and industry trends," Korea Evaluation Institute of Industrial Technology, 11(4):71, 91-4, 2013.
[8] Miller H D, Yasnoff W A, Burde H A, "Personal health records: the essential missing element in 21st century healthcare," J Healthc Inf Manag, 47:166-7, 2009.

임병두(Byeong-Doo Lim)

1995년 : 동신대학교 전자공학과 졸업(공학사)

1995년～현 재: 보성군청
2019년～현 재: 동신대학교 전기전자공학과 석사과정
※관심분야 : 디지털콘텐츠 시스템, 콘텐츠 응용 및 기타 융합기술, 융합콘텐츠, 의료영상, 의공학, 3D 모델링 등

천민우(Min-Woo Cheon)

2003년 : 동신대학교 대학원(공학석사)
2006년 : 동신대학교 대학원(공학박사-전기전자공학)
2008년 : 조선대학교 대학원(의학박사-외과학)

2006년～2008년: ㈜바이오아테코
2008년～2010년: ㈜에코레이
2010년～현 재: 동신대학교 보건행정학과 교수
※관심분야 : 디지털콘텐츠 시스템, 콘텐츠 응용 및 기타 융합기술, 융합콘텐츠, 의료영상, 의공학, 3D 모델링 등