A Service-Oriented Agent Architecture to Support Telecardiology Services on Demand

Ing-Yi Chen* Chao-Chi Huang

Department of Computer Science and Information Engineering, National Taipei University of Technology, Taipei, Taiwan, 106, ROC

Received 6 Apr 2005; Accepted 28 Jan 2005

Abstract

The system provides a solution for healthcare services management. The Service-Oriented Agent Architecture described in this paper is implemented on the Open Services Gateway Initiative (OSGi) framework. It enables healthcare service providers to bridge the gap between various devices’ application environment and to deploy services onto the runtime of end transmission gateway via health care service center. Four subsystems have been successfully implemented in the proposed Telecardiology System: Device Connection Interface, Data Transmission Interface, Service Management Center, and Diagnosis Subsystem. The results of this research can offer better service management framework for large scale of homecare service devices.

Keywords: Service oriented Architecture, Agent, Telecardiology

Introduction

Telemedicine, targeted at improving the quality of life, has made significant advancement during recent years. There has been a considerable effort to investigate issues of using telemedicine devices with communication gateways to promote better life for chronically ill patients [1-8]. For instance, some personal computer (PC) based Electrocardiogram (ECG) telemonitoring systems had been discussed [2-3]. Besides using PC as communication gateways, m-Health [4] system also allows the acquisition and transmission of biomedical data from patient to a remote data center. Although telemedicine devices are gaining growing popularity, two questions left to be answered: how to provide the same interface for a variety of applications in health care device, and how to deal with different control models while building a large scale monitoring system. Instead of using diverse agent, it seems logical to provide a direct link to the service center for existing monitor devices by using on a common agent system.

In this study, we have proposed a unit runtime of telemedicine agents to permit the services to be managed remotely. Such system is derived from a system that originally lacks a consistent application model supporting telemedicine application development. A solution of centralized on demand management system with upgrade services was developed.

This paper is organized as follows. In section 2, relevant previous works dealing with healthcare system are reviewed.

* Corresponding author: Ing-Yi Chen
Tel: +886-2-27712171 ext. 4235; Fax: +886-2-87732945
E-mail: ichen@ntut.edu.tw

In Section 3, the design choices of the system are illustrated and briefly discussed. Later, the detailed architecture and implementation of the Telemedicine System is presented in Section 4, and the result of the study is reported in Section 5. Finally, the conclusions and future works are discussed in Section 6 and Section 7.

Literature Review

In recent years, the development of the m-Health system has led a new way to provide access to data center via wireless communication. For m-Health solution, research groups have been working on methods to transmit physiological data to health care center. An investigation [5] presented the implementation of a GSM based cardiac system for real-time recent communication of physiological data. The authors evaluated the feasibility of WAP solution, consisting of web based health-care agent and central station, to provide out-of-hospital follow-up of cardiac patients. A study [6] investigated a 3G based tele-trauma system for pre-hospital trauma care. This study described a system that assists health-care centers by providing media transformations and data prioritization throughout the pre-hospital procedure. The other system focused on the different aspects of a telemedicine application using Bluetooth based mobile telephone as a biomedical signal transmission gateway for remote examination and medical diagnosis [7]. A research presented a PDA-based physiological monitoring system used in an m-Health application for patient transport [8].

Various functions within those researches are developed.
for realizing healthcare system with several communication platforms. Instead of discussing how to improve functions for various telecommunications equipment; we propose a solution for numerous healthcare service providers to reduce cost of management environment. For healthcare service providers, the solution can also help them to be able to consistently and systematically provision, configure, and manage their services.

**OSGi**

The mission of the Open Service Gateway initiative (OSGi) [9-12] is to provide an open service platform based on Java-based lightweight container for dynamic software components. The OSGi run time enables health care services from multiple service providers to be deployed on a unit application runtime. In addition, it supports a wide range of devices. As a result, service agent can be installed on PDA, PC and Home Gateway web-enabled devices. Under OSGi architecture, application is encapsulated into a bundle which is a java archive file that contains lists of packages and service prerequisites and the OSGi framework automatically loads and runs bundles. Such features enable the mechanism by which healthcare service can be automatically detected and loaded into the health care edge platform.

**Service oriented Agent in Networked Devices**

The Figure 1 illustrates how the Service-Oriented Agent Architecture which works in the Telecardiology system. From a technical point of view, this study focuses on the design of an electrocardiogram health care system that meets the needs of operation support systems for networked health care devices. The service oriented infrastructure is the heart of the Telecardiology system as it provides an integration facility between a set of heterogeneous remote autonomous devices and medicine applications. They enable the seamless integration of usage data from devices with ECG data in real-time, leading to faster reaction times, proactive customer services and improved operational efficiency.

In this study, we apply OSGi on three transmission agent: personal computer based system, home gateway based system, and PDA based system [13]. PDA with wireless connectivity also provides the possibility for automatic updates, synchronization, and database access.

**JMX & OSGi offer remotely manageable health care service**

Multiple forms of devices are now connected to Internet and have gained increased embedded intelligence. This evolution raised operation issues for the management functions involved in the service life-cycle: deployment, configuration, monitoring, maintenance and reconfiguration as necessary.

The OSGi remote management concepts can be implemented by Java Management Extensions (JMX) [14] technology. JMX is a specification that describes an extensible architecture, APIs, and a set of distributed services for network management using the Java programming language. The architecture of JMX aims to scale from the management of few devices to thousands of manageable endpoints [15].

The three main levels of the JMX architecture are Instrumentation, Agent, and Distributed Services. At

![Figure 1. Service Oriented Agent Architecture.](image1)

![Figure 2. Manageable health care service architecture.](image2)

**Instrumentation level**, the manageable healthcare services are made accessible through JMX-specified interfaces. This is done by creating Java objects, called Mbeans. Healthcare services are manageable through these objects. At Agent level, a JMX agent is a software component that exposes a set of standardized agent services to remote management components and directly controls the JMX-manageable resources through their MBean interfaces. In fact, MBeans are managed inside a JMX agent through an MBean server that can load and unload MBeans dynamically. At Distributed Services level, the aim is to specify the interfaces that provide JMX Manager Components. JMX Managers can access agents to manage the health care bundle exposed by the agents.

The health care system architecture includes an Update Manager, which takes advantage of the OSGi deployment capabilities. The Update Manager assists the administrator in installing new healthcare application remotely.

**OSGi agent with JMX management services**

The goal of OSGi management services is to enable the on demand management of service platforms. It means that applications must be able to define on demand in their own management model. To achieve this, we provide management services to enable bundles to define their own management models. In Figure 2, the ECG agent provides an agent service to register MBeans in the MBean server. In this way, the management agent can register an MgtService service in the OSGi service registry at start up time.
To demonstrate how those components work cohesively, a system has been built to transmit the ECG data. At the push of a send button, the user's biomedical information is immediately sent to a healthcare center. The center accepts the data, parses the data, and sends the doctor a message to inform him or her new tasks. Then the doctor make a judgment to see if the user needed additional treat, and messages the users to let them know the diagnostic result. Through the middleware used on the server and the client, the system can remotely upgrade a client device, and troubleshoot and fix any problem immediately if necessary. The usefulness of Service-Oriented agent architecture combined with the OSGi middleware has thus been implemented.

**System Architecture**

The Telecardiology System designed not only provides a way for prophylaxis but also proposes a solution capable of detecting arrhythmia attack for patients receiving rehabilitation care at home. Concept and benefits of processing data in a corporate environment are first introduced in the following sections. Then a description of the OSGi agent and system modules involved in the Telecardiology System is given to provide a framework for discussing common directory definitions, the particularities of implementation, and the role that a corporate directory can play in an integrated environment. Finally, the architecture of the Telecardiology framework is discussed. Figure 3 depicts the architecture of the Service-Oriented-Agent based Telecardiology System.

**ECG Agents**

We used a single-lead interpretive electrocardiogram device to collect, store and forward vital signs data. The device provides an USB interface to connect to home gateway and person computer. Using the OSGi programming model, we also designed an ECGBundle for ECG device attached to the three transmission Device (PC, Home Gateway, and Bluetooth enabled PDA).

**ECG Agents on Home Gateway & PC**

The ECG monitor is based on PL2303 serial-USB adapter capable of converting RS-232 signal to USB signal. In order to control the adapter, an open-source communication API called RXTXcomm which is a Java communications API driver for Linux to access the serial port has been applied. This significant module provides the infrastructure for obtaining electrocardiogram data.

An ECGDeviceBundle was created to implement Bundle-Activator class as the entry point to the bundle. The bundle creates and opens an ECGConn Thread service to automatically track ComConn.java which connects a serial cross link. After ECGDeviceBundle gets ECG data, it stores those records and passes them to service center by invoking two functions that provide logging service and delivering service.

**Bluetooth based ECG Agent**

In the Telecardiology system, a Bluetooth based ECG Agent has also been developed. When Bluetooth devices connect, they form a Piconet connection [16]. This connection is a dynamic network, where ECG monitors act as master while the PDA device is slave. In the system, the JSR-82 Java API has been used for Bluetooth development. Figure 4 shows the Bluetooth based ECG Agent Architecture.

In PDA, a Bluetooth client which implements Discovery Listener interface has been developed. This interface allows an application to specify an event listener that will respond to inquiry-related events. This interface is also used for service searching. The method deviceDiscovered() is called each time a device is found during an inquiry. When the inquiry is completed or canceled, the inquiryCompleted() method will be called. For device discovery, an ECG agent application can

---

![Figure 3. Service-Oriented-Agent based Telecardiology System.](image-url)
obtain a list of devices using startInquiry() function which is provided by class DiscoveryAgent. It can search for services on a Bluetooth server device. The startInquiry() requires the PDA agent application to specify a listener which is notified when ECG monitor devices are found from a real inquiry.

A Bluetooth service is an application acting as a server that provides assistance to client devices via Bluetooth communications. After registering a service, the server application waits for a client application to initiate contact with the server for later service access. The ECG device application offering a service based on the Serial Port Profile (SPP). Its connection model emulates RS-232 control and data signals over the Bluetooth base band, providing transport capabilities for upper level ECG services. Whenever the server application calls acceptAndOpen(), the client application and the server application establish a Bluetooth connection to conduct the ECG data. The ECG Agent enables remote management of the Telecardiology platform through the JMX Agent provided by the platform.

**ECG Data Receiver**

With Web Services, system interoperability is achieved with XML [17-18]. On the client side, we deploy Web Services clients on mobile device and gateway device. MSCSenderBundle is an implementation of the J2ME Web Services Specification (JSR 172). The JSR 172 specification defines APIs for Web services running in the J2ME environment. In addition to the ECG Data Transmission, the MSCSenderBundle bundle offers the ability to export SOAP services that have been registered in the OSGi framework service registry. In order to export a service object as a SOAP service, the only thing needed is to set a property named EcgWSSoap_Stub on the registered service. In a Web Services Provider System, all services communicate with one another in XML in multiple platforms can be integrated between requester and provider. In this way, we use Web Services technologies as the receiver router for ECG Data Receiver.

**Organization Management**

As the number of accessible devices grows, the amount of data also increases. Information describing the various doctors, nurses, patients, and other resources related to telemedicine is collected into a special database. If all of this information could be maintained and accessed in a consistent and controlled manner, it can provide a focal point for integrating a distributed environment into a consistent and seamless system. Lightweight Directory Access Protocol (LDAP) [19] allows a system administrator to centrally define and manage users. An administrator can easily maintain the structure of the organization in large device environments.

Additionally, LDAP build in lots of objects (organization, and user etc.) for storing various data. An object class contains a list of attributes, such as a telephone number or address. Each object class has different attributes. In terms of programming model, every object class needs a group of insert, update, delete, and query function. Hence, we always need to duplicate the basic but redundant function again. The challenge is how to improve programming effort with the LDAP directory.

An application program interface (API) which defines a common language for accessing different kind of LDAP object has been developed to provide applications with rapid access to directory data in large distributed environments. The design focus is to use the same interface to handle different objects and its data structure, an idea derived from the object oriented design model where several objects can be inherited from a basic class. If attribute name and attribute value can be packaged into same container, it is possible to develop a common interface to access functions of LDAP. Hence, within the common API development environment, developers can access LDAP directories through the same Interface. With the inclusion of the common LDAP interface, LDAP Application can change object without the need to insert functional changes. Such scheme significantly lowers the overhead and cost involved in managing LDAP access functions for different LDAP entries.

**Database Transaction**

Transactions are all-or-nothing operations that must be completed in total or not at all. For example, when ECG signals get transmitted into the system, they must be inserted into four tables using an exception-handling mechanism. If only half of this transaction completes, the system will not consistent. EJB (Enterprise Java Bean) [20] and Hibernate, which is an open source object-relation (OR) mapping framework, are two example frameworks supporting transaction handling for Java platform. In our system, we integrate EJB programming model and Hibernate API framework to create the persistence layer storing measured data into four related tables in a single transaction.

![Figure 4. Bluetooth based ECG Agent.](image-url)
User Interface

For doctors, the diagnosis subsystem is the main subsystem for "consulting service" and "diagnosis service" that provides a convenient way of communication through Internet between doctors, patients and medical specialists. At implementation level, the diagnosis subsystem is based on Struts framework which is a Model-View-Controller (MVC) implementation that derives its advantages from Servlet and JSP tags. MVC is a design pattern to help applications to decouple interface from application logic and data. Using this solution, programmer can develop web diagnosis subsystem supporting component-based development for reuse and consistency. However, what the user really cares is an easy-to-use web system, not state-of-the-art technology. The navigation menu bar on the web based diagnosis system is thus built with the usability in mind. The bar integrates the most related information. If a user gets interested in each item, he can immediately get the information form the item's hyperlink. Moreover, each page can easily go back to previous pages.

Result

The Telecardiology system presented in this study is a service-oriented medicine agent architecture whose mission is to improve health care access, quality, efficiency and management. This system has been designed and implemented. The agent unit, which includes a vital-sign signals acquisition module, can acquire ECG data and forward ECG data to medical service center. The kernel of ECC agent is based on OSGi allowing health care service provider to deploy unique environment to the patient’s monitoring device for remote management of the ECG services. As an ECG agent sends biological signal to medical service center, the system invoke a parser to acquire detail ECG data which is then inserted into appropriate tables.

In aspect of data persistence based on the LDAP technology, a system for enhancing ECG data query performance in large scale environment has been developed. In this study, instead of only using relational database as the data repository, we propose the architecture, which combines the advantage of LDAP and relational database to meet the challenge of the larger scale of ECG data. Whenever an application needs to query a list page which contains users’ ECG data, the system first looks up ECG IDs by the user ID in the directory through the user’s DN. After collecting ECG IDs, the system uses those ECG IDs to point to the relevant information stored in specific tables in the relational database.

The Figure 5 shows the response time of a query list related task using LDAP technology. This test result is based on a web application environment running on the IBM WebSphere Application Server 5.1 platform, IBM Tivoli Directory Server 5.1, and IBM DB2 Universal Database 8.1. The WebSphere Application Server platform was chosen for its security features, Java enabled capability, and rich web application framework library. The Scenario is that a user logins to the system, and then views 100 ECG record belonging to the user. In the condition, there are 5000 ECG records in database; each record size is 13KB, and average page size is 15KB. Using the relational database with LDAP technology, the average time to complete a test scenario is 5.94 sec (standard derivation is 0.48) comparing to 7.55 sec (standard derivation is 0.65) with SQL query commands. In this given example, this is nearly 20% time gap. We get the result from 40 trials. Because the relation between users and their ECG records is maintained in LDAP, we can use the hierarchical tree to acquire all indexes of user’s ECG records in a faster manner. Therefore, the system can use a SQL command (EX: “Select * from EcgData as ecg where ecg.UserSn in (1001, 2043, 4152)” ) to get their related information. Conversely, if we only use relational database SQL commands, the system needs to compare user names in ECG measured history in several times. Results from the data show that even the database enables user IDs as an index to enhance search performance, the performance of using LDAP technology is better than the performance of simply using SQL command to query ECG records.

After ECG data get transmitted to medical service center, the diagnosis system provides a graphic display of the electrical activity of the heart. Figure 6 demonstrates the main diagnosis view of the care unit. Both navigation menu bar and care information can be found at the top. ECG waveforms are shown in the middle. The left area is reserved for doctor advice.
Figure 6. View of the diagnosis system.

The system recorded user’s ECG data continuously and put all necessary tasks into one comprehensive application window. Once the doctor logs in, he or she can start the diagnosis right away.

Discussion

Aspects of the future works in this study are discussed as follow:

Security

The safeguarding of the integrity and confidentiality of the transmitted ECG message, in other words, the secure communication between home gateway and service center, is still an issue. Web Service Security is now supported in the J2ME and WCE environments. It provides a set of mechanisms that can be used to secure SOAP message exchanges using message integrity with XML digital signatures, message confidentiality with XML encryption and decryption, and message authentication. WS-security will be integrated into future development to enhance the quality of service of the system.

Reliability

To guarantee that every ECG data can be correctly received at the hospital unit in near real-time, the issue of how to integrate message oriented middleware to provide an asynchronous messaging service that guarantees diagnosis messages delivery needs to be dealt with in further investigation.

Conclusion

This research has successfully demonstrated the potential of Service-Oriented Agent Architecture in telemedicine using an OSGi based Telecardiology system. The Agent unit attached to the transmission device provides a unit runtime environment to transmit ECG data to a health care center. The receiver unit accepts signals via Web Services channel and then parses ECG data for later diagnosis. The system is designed to transmit ECG signals to a hospital server via the OSGi services, which allows services to be shared and corporate with other services. The system is expected to become a reference model as a convenient means of health care service management.

Reference

Dependability of Service Oriented Architectures for Pervasive Computing”, *IEEE WORDS*, January 2003, Mexico.


