Robot Middleware for Android-based device for reusability of Robot SW Component *

DongUk Yu and Hong Seong Park, *Member, IEEE*

Abstract— **Nowadays there have been several billions of Android-based smart phones and lots of tries of applying to the robot domain. This paper suggest the middleware, called AndMid4Comp, operating in Android-based smart phones, which reuses the existing C++-robot SW components open in the OPRoS community. The middleware manages the C++ components and Java components developed by Android developers according to specified properties such as period and aperiodic. Note that operations of C++-based programs are very limited on Android. This paper verifies the proposed middleware via an examples of mobile manipulation robot.**

I. Introduction

Nowadays Android is one of the most important operating systems used in the smart devices such as smart phones [1]. So lots of persons are familiar with Android-based smart devices and then have been trying to apply to various application domains. One of those application domains is a robot application. The Android-based smart device such as smart phone has the high computing power, various sensors, necessary for a robot, and a large community. Especially its computing power is similar to that of PC. So it can be better solution to use the smart device as a robot controller. In addition, there are many open software codes in open robot communities, which users or developers are worthy using. But these types of software are operating in Linux or Windows environment [1,2,3].

But there have been some problems in using an Android-based smart device as a robot controller, which are as follows: reusability of robot software component and control of various HW devices. There have been many researches on the Android-based robot application, which are largely classified into the robot controller [1,4,5] and the monitor (or the master device or teaching pendant for teleoperation) [6,7].

The Android-based robot controller has mainly been used in applications such as entertainment and education [4,5]. Especially it has some problems in developing a commercial robot such as a guiding robot, a mobile servant robot, and a mobile platform with a manipulator. The problems occur due to following constraint:

- In Android-based programming, Java language is basically used. C++ language can also be used but has some difficulty to access HW devices.

*Resrach supported by MoTIE and KEIT.

DongUk Yu and Hong Seong Park is with the Department of Electronics and Telecommunication Engineering, College of IT, Kangwon National University, 1, GangwondaehagGil, Chuncheon, Gangwon-Do, 24341, Republic of Korea (corresponding author to provide phone: 82-33-250-6346; e-mail: hspark@ kangwon.ac.kr).

In other words, because Android developers use Java language in developing of robot SW components, they don't utilize robot SW components written in C/C++ language, which are open in source code. They should develop robot SW components such as navigation, obstacle avoidance, and manipulation by themselves. But it is difficult task for them because most of them can't generally be robot experts. So ROS[1] provides Android developers with the method to utilize the existing SW modules. But additional codes are necessary for the C++ SW to apply to the Android devices In other words the C++ Robot SW module by itself isn't used. So the problem remains still while only Android-based smart devices are used.

So Android developers need a new method, which reuses robot SW component open in the community such as ROS and OPRoS[2] and the SW components are operating in the Android-based devices. This method help the Android developers create various robots and develop various robot services.

So this paper proposes the middleware for an Android smart phone, over which robot SW components in the OPRoS community is reused and interacts with Java SW components developed by Android developers. The proposed middleware is the extension of OPRoS middleware for Android phones. This paper verifies the proposed middleware via its implementation and demonstration of a mobile manipulation robot.

This paper is organized as follows: in the next section the architecture of the proposed middleware is described briefly. Section 3 suggests the implementation of a mobile manipulation robot consisting of Java components and existing OPRoS components. The demonstration and some results are provided from its implementation. Finally some conclusions are given.

II. ARCHITECTURE OF MIDDLEWARE FOR ROBOT SW COMPONENTS IN ANDROID-BASED DEVICE

The proposed middleware is designed to get following goals and its architecture is shown in Fig.1.;

- Existing OPRoS components are executing, which are programmed in C++ language.
- New SW components developed by Android developers are interacting with the OPRoS components. Note that the new components is programmed in Java language.
- The middleware executes two types of components according to the specified periods.

- The Java/C++ components transmits data or executes services each other with the Java/C++ components via 3 types of port such as data port, service port, and event part.

In fact, Fig.1 shows an Android application including middleware, which is called AndMid4Comp (Android-based Middleware for components). So AndMidComp consists of a GUI process and a middleware process for management and execution control of C++ components and Java components. Note that C++ components are existing open components and Java components are made using the Android development tool.

The GUI process provides 4 types of user interfaces (UI), which are Terminal UI, Package UI and Component UI. Terminal UI monitors the status of the middleware process and controls the execution of a robot application consisting of some components. Package UI manages packages and then download or upload them, where the package is an application software consisting of components and a profile including components' properties. Component UI manages components, which are elements of a package, and then upload or download them.

Middleware process consists of a package manager, a C++ engine for C++ component, and an app. Manager for Java components. The package manager manages packages and components stored in a package repository of an Android device, which is shown in Fig. 2. The C++ engine is an important part for execution control of C++ components consisting of a component scheduler, an I/O manager, a package deployment manager, and a node manager. The app. manager manages Java component-based containers. Java component-based container shown in Fig. 3 is made by adding port handlers and a life cycle managers to Java components and is operating as an independent process. The Java component exchanges data and services via port handlers among Java/C++ components and controls the execution of itself such as periodic service.

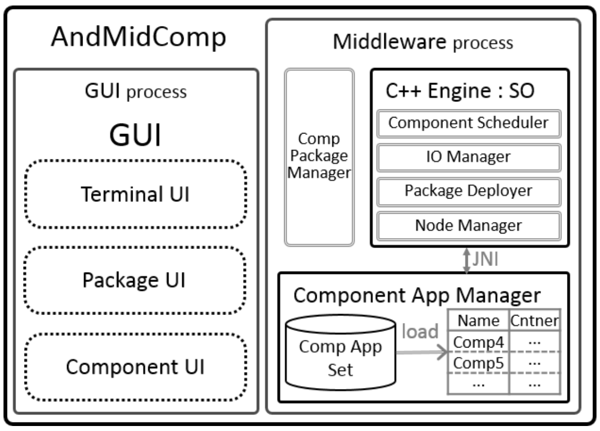

Fig.1 Structure of Android-based Middleware for components

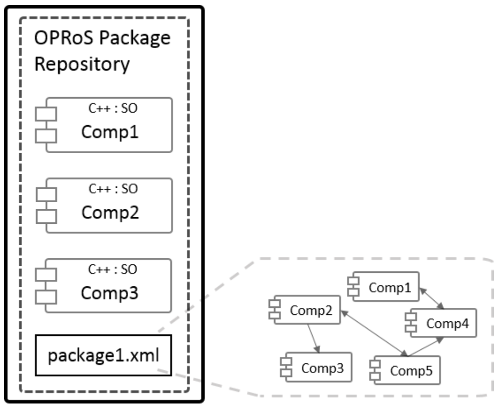

Fig. 2 contents of Package Repository

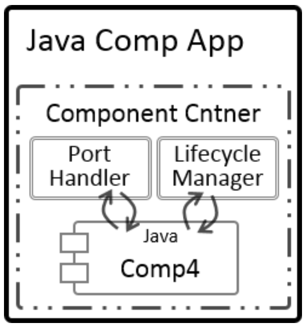

Fig. 3 Structure of container for Java component

III. EVALUATION

To evaluate the proposed AndMid4Comp, the AndMid4Comp is implemented on a mobile manipulator with Android smart Phone shown in Fig. 4. For control of the mobile manipulator, 3 C++ components and 5 Java components are used as shown in Fig. 5. The 2 C++ components are used without any modification of existing OPRoS components. MainComp is modified because MainComp is a coordination component for a mobile part and a manipulator part. Note that the binary code of the component itself are directly used if the source code is properly compiled according to a CPU type.

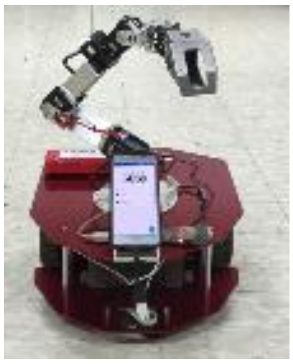

Fig. 4 Android-based Mobile Manipulator

The 3 C++ components used in the Mobile Manipulator in Fig. 4 are as follows;

KinematicsComp : compute the kinematics

MainComp : control and coordinate a manipulator and a mobile platform using service ports

ExpressionComp : decide on emotional expressions representing on display of Smart phone, which is based on IMU. The decision result is transmitted to 2 Java components called RobotFaceComp via RobotfaceService por and TTSComp via TTSService port.

The 5 Java components used in the Mobile Manipulator in Fig. 4 are as follows;

ManipulatorComp : control the manipulator consisting of 4 motors(Dynamicxel) via USB.

MobilityComp : control the wheel controller via USB, which is embedded in the mobile platform called 'Stella B1'

IMUComp : provide the IMU sensor data of Smart phone to other components via data port called IMUData.

TTSComp : provide TTS service via service port called TTSService. Note that the TTS service is based on TTS function provided by Android smart phone.

RobotFaceComp : represent emotional expressions of a robot on the screen of the Android smart phone.

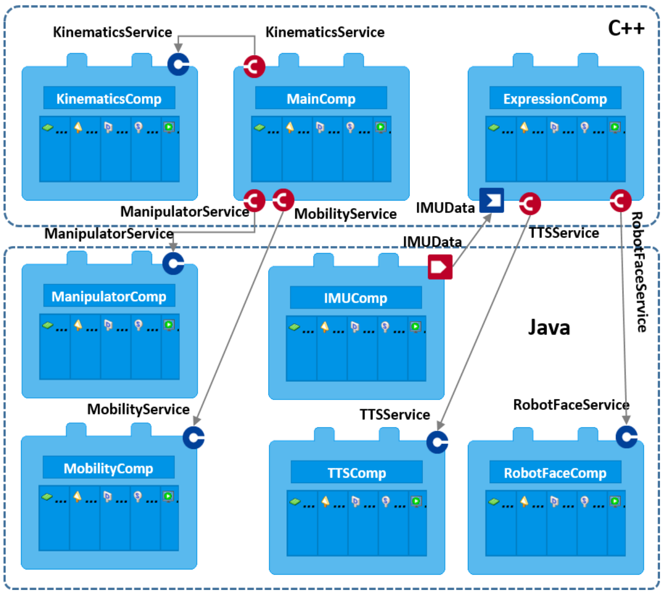

Fig. 5 component configuration for Android-based Mobile Manipulator

These components shown in Fig.5 are implemented and downloaded to the Android smart phone. Then the downloaded package can be shown using the Terminal UI and Package UI. An Android user starts the downloaded package using Terminal UI. Then the robot shown in Fig. 4 is operating like Fig. 6. Fig. 6 shows movement and manipulation of the robot and at the same time emotional expression on the screen of the phone.

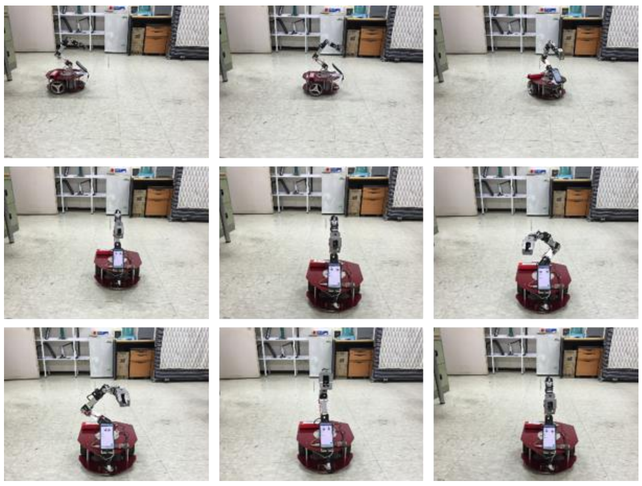

Fig. 6 Operation Sequence of Android-based Mobile Manipulator

It can be known form Fig.6 that the proposed AndMid4Comp is well working. In other words, it can be shown that Android users conveniently make and control an Android-based robot using existing C++ components if the suggest model of this paper is used in the Android-based smart device.

IV. CONCLUSIONS

This paper proposed the middleware for an Android smart phone, called AndMid4Comp, over which robot SW components in the OPRoS community is reused and interacts with Java SW components developed by Android developers. The proposed middleware is the extension of OPRoS middleware for Android phones. This paper evaluated the proposed middleware via its implementation and demonstration of a mobile manipulation robot.

REFERENCES

[1] ROS, www.ros.org
[2] OPRoS, www.opros.or.kr or www.ropros.org
[3] openRTM, www.openRTM.org
[4] K. Ono and H. Ogawa," Personal Robot Using Android Smartphone," Int. workshop on Innovations in Information and Communication Science and Technology, Sep. 2014
[5] S. Jeong, K. D. Santos, and et al, "Designing a Socially Assistive Robot for Pediatric Care," IDC 2015, Jun. 2015.
[6] J. C. Yepes, J. J. Yepes, J. R. Martınez, and V. Z. Perez, "Implementation of an Android based teleoperation application for Controlling a KUKA-KR6 robot by using sensor fusion," 2013 PAHCE, May 2013.
[7] C. Mateo, A. Brunete, E. Gambao, M. Hernando, "Hammer: An Android Based Application for End-User Industrial Robot Programming," IEEE/ASME 10th Int. Conf. on Mechatronic and Embedded Systems and Applications (MESA), Sep. 2014