Preliminary Design of Mobile Visual Programming Apps for Internet of Things Applications based on Raspberry Pi 3 Platform

Adnan Rachmat Anom Besari, Iwan Kurnianto Wobowo, Sritrusta Sukaridhoto, Ricky Setiawan, Muh. Rifqi Rizqullah

EEPIS Robotics Research Center (ER2C), Politeknik Elektronika Negeri Surabaya
anom@pens.ac.id, eone@pens.ac.id, dhoto@pens.ac.id,
rickysetiawan@ce.student.pens.ac.id, rifqimr@ce.student.pens.ac.id

***Abstract*— Learning about sensor technology and actuator early is important as a step towards knowing and introducing of advanced technologies based on Internet of Things (IoT). The difficulties are how to learn sensor technology and move the actuator with accessing General Purpose Input Output (GPIO) of Raspberry Pi 3 Platforms using programming language syntax which often confusing and difficult to understand. To help people learning IoT by using Raspberry Pi 3 with an interesting Android apps, we believe that this learning module can integrate about the ease and attractiveness of IoT System Editor based on Android apps. This research create a mobile programming apps based on Android which people can build IoT project easily with GUI without program and middleware based on Raspberry Pi to connect between apps and hardware with especially task to manage data communication, data flow, and device driver. Hopefully new developer can develop the IoT application easily by using Android mobile visual programming that combined with Raspberry Pi 3 platform.**

Keywords*— Visual Programming, IoT, Middleware, RaspberryPi, Android.

I. Introduction

Technological advancements that continue until now have influenced various aspects of human life. One of the most popular technological developments is the Internet of Thing (IoT). The development in IoT has not only penetrated the professional or industrial sectors, but development in educational world has also increased with the development boards that have specifications with the aim to be a learning module. Apart from the various backgrounds of the developers, IoT developers still culminate in one goal that is to help human life. Interest in studying IoT is quite high, but there is not much software that provides learning media to build IoT. Beginners who want to learn about IoT sometimes feel difficult to learn the basic understanding of programming, because one of them is quite difficult to understand programming grammar.

In building IoT system, it also required hardware or module with the fund is not cheap, moreover trying with trial and error. IoT infrastructure is generally built with Raspberry Pi. Raspberry Pi is one of the SBC (Single Board Computer) which this size is same with a credit card, has the ability like a computer but there are 40-pin GPIO like a microcontroller, and the price is relatively cheap. Based on several considerations including support from forums on the development of sustainable research, support of sensor modules as well. Compatible actuators and high popularity in building IoT infrastructure, we choose Raspberry Pi as the SBC used in this paper. Reflecting on the things about beginners who want to learn about IoT, then needed an application that systematically able to help and explain to process learn an understanding of programming logic in IoT devices. Departing from here, we want to develop Raspberry Pi that can be built in a modular and controlled by beginner through the GUI application on Raspbian OS. Figure 1 show the paradigm of IoT device that controlled by an apps.

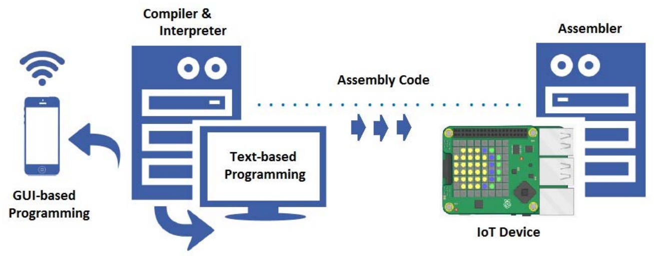


Fig 1. Paradigm of IoT devices controlled by an apps.

This research focuses on the development of middleware on the educational robot and creates apps based on android so that later expected the creation of a protocol to access the sensor and control the actuator with the command from the apps. In addition to this software is also used to coordinate embedded system applications so that the system not only can run well but also can be timely and efficient. Middleware is a programming layer that connects high-level programming with low-level programming on Raspberry Pi. The use of this layer is based on the process of translating commands provided by the user through high-level programming which is then forwarded by the middleware to access the sensor or actuator. Middleware can also be interpreted as a protocol that translates commands between high-level programming with Raspberry Pi. This development aims to support the maximum hardware performance.

II. PREVIOUS WORKS

The Node-RED is a flow-based programming tool, original developed by IBM's Emerging Technology Services team and now a part of the JS Foundation [1]. Node-RED is a programming application to build IoT system with web-based display with software platform using Node-JS. The Node-RED application makes it easy for users to create IoT systems with drag and drop component blocks that represent from IoT devices. Each component block means a node, where each node can connect to each other, transmit and receive data. Node-RED has 9 major component nodes :

- **Input** (inject, catch, status, link, mqtt, http, websocket, tcp, udp, Watson IoT, serial)
- **Output** (debug, link, mqtt, http response, websocket, tcp, udp, Watson IoT, play audio, serial)
- **Function** (function, template, delay, trigger, comment, http request, tcp request, switch, change, range, split, join, csv, html, json, xml, yaml, rbe, random, smooth)
- **Social** (e-mail, twitter)
- **Storage** (tail, file)
- **Analysis** (sentiment)
- **Advanced** (watch, feedparse, exec)
- **Raspberry Pi** (rpi gpio, rpi mouse, rpi keyboard, ledborg, sense HAT)
- **Networks** (ping)

Nakamura et. al. conducted a research about how to design and implement the middleware system for IoT devices toward real-time flow processing [3]. This research created a middleware design for Information Flow of Things (IFoT), which is a framework for processing, analyzing, and merging data in real-time and scaled based on data processing distributions between IoT devices The basic concept of IFoT has several parameters :

- Distributed processing of tasks,
- On-demand flow distribution,
- Online information processing,
- Various data streams integration.

In IFoT, there are several layers. Here is the function of each layer on IFoT :

a. *Task Allocation Mechanism* : Task allocation function is the layer to divide the work for the two classes below, namely the receipe class and task assignment class. Receipe class is useful for receiving data from apps and then divide into multiple jobs in parallel. Task assignment classes are useful for distributing jobs from receipe classes to separate work for the IFoT module

b. *Flow Analysis Function* : This layer consists of learning classes, judging classes, and managing classes. Learning class is a class for IFoT to analyze data from sensors sequentially, then make updates for more accurate data. Judging class is a class for analyzing streaming data from built models. While managing class is to manage data distribution process.

c. *Flow Distribution Function* : This layer consists of publish class, broker class, and subscribe class. Publish class is useful for maintaining the stability of published data. The class is in the data delivery layer. The subscribe class is in the data receiving layer, functioning the same as the publish class that is to maintain data stability. Broker class is useful for organizing data that is distributed according to their respective topics.

d. *Sensor / actuator Integration Function* : this layer serves to set the integration and communication of the sensor class and actuator class. This layer also defines how data from sensors and actuators can be accepted or read by IFoT. The example of the communication protocol used is MQTT / Message Queue Telemetry Transport

Implementation of IFoT prototype is with limited function: flow distribution function and flow analysis function. The first function is flow distribution function built with Mosquitto is an application that using the MQTT communication protocol. For flow analysis function built using Jubatus which has more capability in terms of online machine learning. Each operation between multiple modules is governed by an app. The application is built using OpenRTM- aist.

Ballesteros et. al. conducted a research about management tool that can controls electronic devices through a Raspberry Pi under IoT model based on Android OS [5]. This paper is registered in the development of an application to organize and control electronic tools, infrastructure design, GUI, accuracy, integration with communication systems and performance tests that have been described. Applications are designed to resolve issues in control and management for the creation of the Internet of Things (IoT). The development of mobile applications, databases, electronic designs and computer networks is used for communication functions so raspberry pi can be well controlled.

Lamine and Abid, conducted a research about remote control for domestic equipment from an Android Apps based on Raspberry Pi [6]. This paper shows a remote control using the android app used to control Raspberry Pi card. Stages used in this study is to prepare the equipment and then develop the Android system. In addition to this paper indicated the difference in code that has been developed to run communication between the remote user, web server, raspberry pi and also a series of components. The android app installed on the smartphone connects with the web server and raspberry so it can communicate. The results of this study indicate that the research is successful.

III. SYSTEM DESIGN

In this paper there are two main parts to build middleware in Raspberry Pi on IoT section, which is first on the middleware side between GUI application in Raspberry with Raspberry itself [7]. This first side includes translation of commands sent from GUI apps to be understood by Raspberry through middleware, ie the existence of the standard or the main parameters of the command notation sent. The role of middleware here is to prepare the function call provided on the second side of the device driver [8]. This means that middleware also includes access to device drivers that are about sensor readings through serial communication, I2C, and SPI and actuator access that is in the form of servo motors and DC motors [9]. In addition, middleware has its own parameters to translate commands received from GUI applications. On the first side of the middleware as well as on the second side of the device driver are both based Python programming language which is the standard programming language on Raspberry Pi. The second side is the device driver where there are functions to read the sensor through serial / I2C / SPI and run the actuator that is DC motor and servo motor [10]. In the middleware, there is also a communication protocol using socket programming with the base TCP / UDP to receive commands and send data to the cloud [11]. The function of this communication protocol is also useful for receiving commands and sending data to Android apps whose devices can connect directly through Raspberry's access point. Figure 2 show the middleware architecture design.

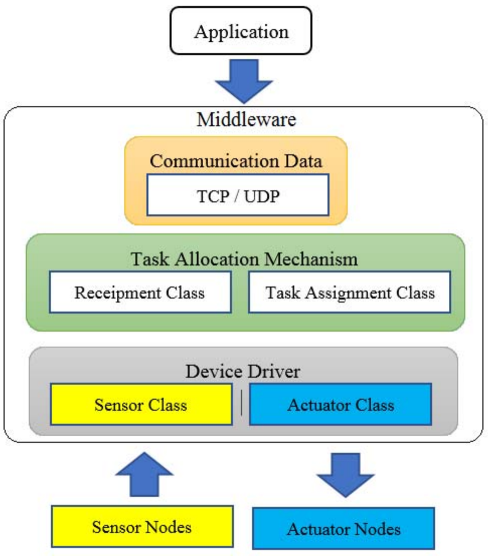


Fig 2. Middleware architecture design.

We create application on 2 devices : for Android phone and for Raspberry desktop. Designed middleware has multiple layers that handle specific tasks, and each layer is connected. Data instruction set which comes through the Android apps or Raspbian apss with Communication Data layer, and will be stored to the Receipment Class in Task Allocation Mechanism and translated into a one pack of tasks in Task Assignment Class to be run by the Raspberry Pi. After the task is given, the process is continued by giving instructions to Device Driver layer. once the command from the apps is received, the next process is the command passed to the Device Driver layer to access the sensor as well as move the actuator. In this case, the user can provide simple instructions to the device driver to access the sensor or move a simple actuator. The system workflow and features has been shown in Figure 3.

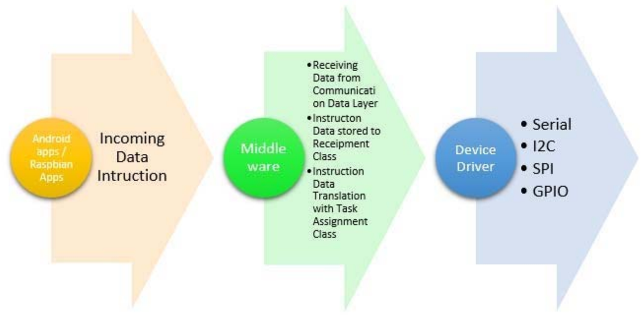

Android apps / Raspbian Apps
Incoming Data Intruction
Middle ware
•Receiving Data from Communication Data Layer
•Instructon Data stored to Receipment Class
•Instruction Data Translation with Task Assignment Class
Device Driver
• Serial
• I2C
• SPI
• GPIO

Fig 3. System workflow and features.

There is a mechanism on Android apps or Raspbian apps is a data instruction set that will be sent via TCP / UDP to the middleware and then compiled. The compiled instruction set data will be translated by the middleware through the Task Allocation Mechanism layer, then divided into the Receiver class and Task Assignment class. Each Data Instruction Set that is sent to the middleware has special parameters to be translated by the middleware. Combination of Data Instruction set from Basic Receieps can be expand to Advance Receieps. However the number of program notation code can have different numbers. For example :

Table 1. Example of controlled devices with Android apps / Raspbian apps

Basic Recipes	
LED	Button
Controlled LED with Button	Controlled Camera with Button
LED Board	LED Bar Graph
Traffic Lights	Stop Motion
Full Color LED	Shutdown
Motion Sensor	Distance Sensor
Light Sensor	Control Motor
Advance Recipes	
Smart Lamp	Multi-room alert
Reaction Game	Music Box
Robot Legs	Robot Vision

In the Task Allocation Mechanism, layer is divided into two, namely Receiver Class and Task Assignment Class. This layer is responsible for translating a series of command notations from the GUI into a series of tasks to complete. This command notation has its own parameters of acceptable commands. In the receiver layer class, it is tasked to parse data from a series of received command notations. The results of parsing the data will be routed to the task assignment class layer to be divided into device driver blocks.

Each data information sent in the form of a collection of function codes and parameters. Each data information has the same structure that has the function code and parameters, but the number of notes code program can have different amounts. On this layer, there are 2 models of Receipment Class, namely *basic receipment* and *advanced receipment*. Here is the format of basic receipment. Data Instruction Set for *basic receipment* is :

```
<Function code> <parameter-1> <parameter-2>...<parameter-n>\
```

Data Instruction Set for *advanced receipment* is :

```
<Function code1> <parameter-1> <parameter-2>.<parameter-n>
<<NotationBreak>>
<Function code1> <parameter-1> <parameter-2>.<parameter-n>
```

Function Code and parameters was build on Raspberry's basic GPIO function. In this middleware, we give additional features or Advanced Data Set to this Raspberry Pi. The implementation of code above for example when the motor with PWM=255 and the duration = 100 seconds. The compiler will generates code below:

- Function Code	: GPIO pin 25
- First parameter	: PWM 255
- Second parameter	: Time 100
- The program code is	: 25,255,100.

Below is the table that show of command notation to access Serial Communication :

Table 2. Command for Serial Communication

Function Code 0xFF	Action Serial 0x01	Baudrate	Time
		300	0-100
		600	0-100
		1200	0-100
		2400	0-100
		4800	0-100
		9600	0-100
		14400	0-100
		19200	0-100
		28800	0-100
		31250	0-100
		38400	0-100
		57600	0-100
		115200	0-100

Below is the table that show of command notation to access SPI Communication :

Table 3. Command for SPI Communication

Function Code 0xFF	Action SPI 0x02	Time
		0-100
		0-100

Below is the table that show of command notation to access I2C Communication :

Table 4. Command for I2C Communication

Function Code 0xFF	Action I2C 0x03	Time
		0-100
		0-100

Below is the table that show of command notation to access PWM :

Table 5. Command for PWM access

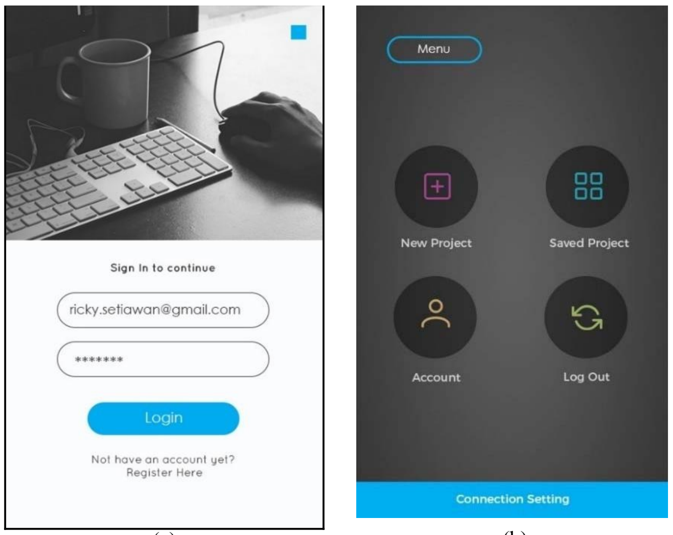
Function Code 0xFF	Action PWM 0x04	PWM	Time
		0-255	0-100
		0-255	0-100

Below is the table that show of command notation to access GPIO :

Table 5. Command for GPIO access

Function Code 0xFF	Action GPIO 0x05	GPIO	Time
		BCM 5	0-100
		BCM 6	0-100
		BCM 16	0-100
		BCM 17	0-100
		BCM 22	0-100
		BCM 23	0-100
		BCM 24	0-100
		BCM 25	0-100
		BCM 26	0-100
		BCM 27	0-100

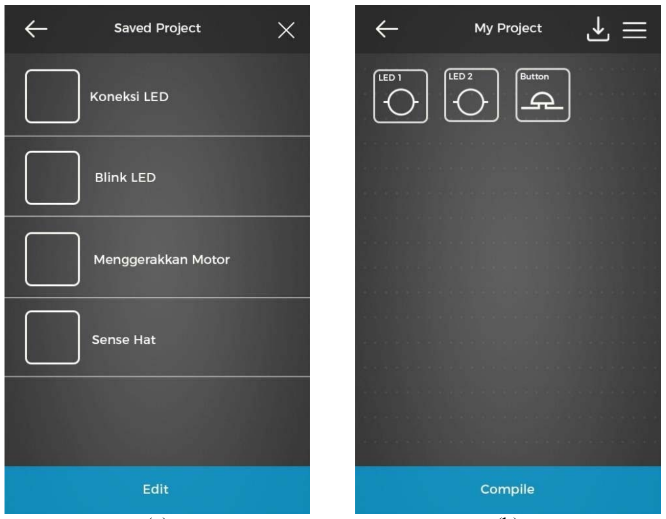
Each interface is a small programming language, the interface describes a set of objects and operations that can be used to manipulate objects. Therefore the design of the application presented using mockup using a simple display.



(a) (b)

Fig 4. *Mockup Application* (a) Login (b) Menu

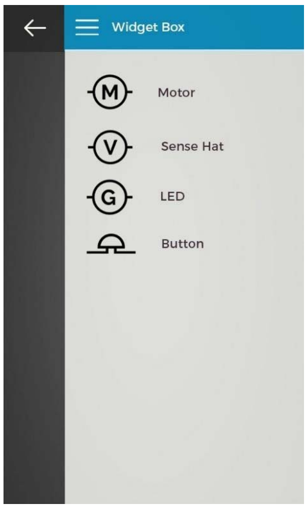
In the first mockup login is a display function used to authenticate accounts that have been created in accordance with the database data to determine which device to be set. If not already registered then must do the registration phase first to set the configuration of account and tool. After the login view and successful, then the next is a menu display that provides four kinds of options ie open the project is stored, create a new project, account configuration and log out. In addition on the main page is also available connection settings that will manage the connectivity between applications and tools to be controlled.

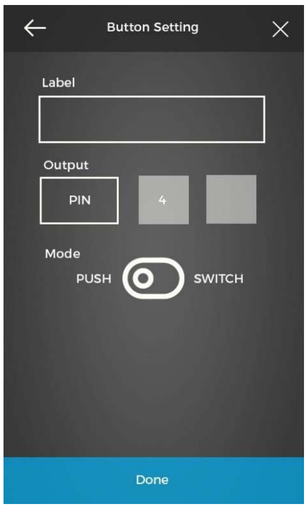


(a) (b)

Fig 5. *Mockup Application* (a) Saved Project (b) Create New Project

If the project menu is stored in the menu pressed, then the next will appear the list of old projects that have been made to be viewed or rearranged with the edit button. In the next menu that makes a new project will be displayed a blank page that will be filled by widgets to set an IoT project in accordance with what we want. The next step after the widget is installed is to save or run. To save it can be done by pressing the click button on the save icon. If all widgets have been installed and configured, then the next step is to run it with a compile button that is below the display.





(a) (b)
Fig 6. *Mockup Application* (a) Widget Box (b) Widget Setting

To select a widget to be included in the project creation can be done by pressing the widget box icon. If icon is pressed then we can choose widgets according to what we want to make project. As for the configuration steps can be seen in the picture widget settings where the user can set the mode or connection of widgets. The visual mobile programming has several contribution and benefit. However, there are several points to note are:

- The application is a pack for IoT project and elements. Visual design is made simpler so every user can easily understand. Operation of the hardware system needs to be tailored to the level of IoT, because not all user have the same basic electronics.
- The programming language is still considered too difficult by the user, so it requires a programming medium corresponding to the students language.
- Node-RED is a node.js based application, using the JavaScript platform as the basic of node.js. This platform is typically used for event-driven data, non-blocking I / O access, real-time applications that can run on cross platforms and distributed devices. Devices that work with Node-RED must have libraries from node.js or modules that match so that these applications can be used smoothly.
- On the system that we created, it needs library and python platform, openRTM-aist and openRTM editor. The component can work in I / O model and can run in real-time. Even when Raspberry is turned on from the beginning , all systems can run directly in our settings.

IV. CONCLUSION

This paper is a preliminary design of the Mobile Visual Programming Apps for IOT devices based on the Raspberry Pi 3 platform. As an IoT device integrated with the sensor / actuator, the device is expected to be easily used modularly with advanced features in Raspberry Pi 3. In this system include with an interactive mobile development system editor that allows students with minimal programming knowledge to use this application easily. This application has a feature that is flexible variable so it can be easy to change the parameters of the sensor / actuator. Future work is, we will build applications that can be multi-platform development programming apps, which where separated with Raspberry Pi 3. This application allows to build IoT systems and devices with a wide coverage. The application programming level is divided into 3 levels, beginner, intermediate, and expert. So for further development, it is hoped not only students who can use this application, but can include a wider scope to play with visual programming for IoT devices. This research also aims to open access with government and private sector to build synergy in building IoT devices. In the final development stage, it is expected that electronics training can be a learning curriculum for elementary, junior, and high school.

ACKNOWLEDGMENT

This research was supported by Local Research Grant on Politeknik Elektronika Negeri Surabaya (PENS) 2017..

REFERENCES

[1] Node-RED, https://nodered.org/about/. Last accessed: August 2017.
[3] Y. Nakamura, H. Suwa, Y. Arakawa, H. Yamaguchi, K. Yasumoto, “Design and Implementation of Middleware for IoT Devices toward Real-Time Flow Processing", IEEE 36th International Conference on Distributed Computing System Workshops, pp.162-167, 2016.
[4] MQTT, http://mqtt.org/. Last accessed: August 2017.
[5] E.A.D. Ballesteros, C.E.C. Calderon, Y.C. Calderon, E.G. Strauss, “Android Management Tool, That Controls Electronic Devices Through a Raspberry Pi Under IoT Model", IEEE 10th Computing Combian Conference, vol 2, pp.237-244, 2015.
[6] H. Lamine, H. Abid, “Remote Control of a Domestic Equipment From an Android Application Based on Raspberry Pi Card”, IEEE 15th International Conference on Science and Techniques of Automatic Control and Computer Engineering (STA), pp.903-908, 2014.
[7] J.K. Amirudin, A.R.A. Besari, F. Ardila. “Perangkat Lunak Pemrograman Berbasis GUI pada Prototipe Robot Edukasi”, Jurnal PA Politeknik Elektronika Negeri Surabaya, 2013.
[8] N.P.H. Pratama, A.R.A. Besari, F. Ardila. “Perangkat Lunak Pemrograman Berbasis GUI pada Simulasi Robot Edukasi”, Jurnal PA Politeknik Elektronika Negeri Surabaya, 2013.
[9] H. Muhammad, A.R.A. Besari, E.H. Binugroho. “Pengembangan Sistem Middleware pada Robot Edukasi”, Jurnal PA Politeknik Elektronika Negeri Surabaya, 2015.
[10] L. Khanif, A.R.A. Besari, S. Sukaridhoto. “Pengembangan Perangkat Lunak Pemrograman Visual Berbasis Android pada Robot Edukasi”, Jurnal PA Politeknik Elektronika Negeri Surabaya, 2015.
[11] Besari, A.R.A, Sukaridhoto, S., Wibowo, I.K., Berlian, M.H., Akbar, M.A.W., Yohanie, F.P.Y., Bayu, K.I.A., "Preliminary design of interactive visual mobile programming on educational robot ADROIT V1", Conference: 2016 International Electronics Symposium (IES), pp.449-503, 2016.