IOP Conf. Series: Materials Science and Engineering **190** (2017) 012009 doi:10.1088/1757-899X/190/1/012009

# Lightweight UDP Pervasive Protocol in Smart Home Environment Based on Labview

**Wijaya Kurniawan[1], Mochammad Hannats Hanafi Ichsan[2], Sabriansyah Rizqika Akbar[3] and Issa Arwani[4]**

Computer Engineering, Computer Science Faculty, Brawijaya University, Indonesia

[1]wjaykurnia@ub.ac.id, [2]hanas.hanafi@ub.ac.id, [3]sabrian@ub.ac.id and [4]issa.arwani@ub.ac.id

**Abstract**. TCP (Transmission Control Protocol) technology in a reliable environment was not a problem, but not in an environment where the entire Smart Home network connected locally. Currently employing pervasive protocols using TCP technology, when data transmission is sent, it would be slower because they have to perform handshaking process in advance and could not broadcast the data. On smart home environment, it does not need large size and complex data transmission between monitoring site and monitoring center required in Smart home strain monitoring system. UDP (User Datagram Protocol) technology is quick and simple on data transmission process. UDP can broadcast messages because the UDP did not require handshaking and with more efficient memory usage. LabVIEW is a programming language software for processing and visualization of data in the field of data acquisition. This paper proposes to examine Pervasive UDP protocol implementations in smart home environment based on LabVIEW. UDP coded in LabVIEW and experiments were performed on a PC and can work properly.

**Keywords.** UDP, Pervasive Protocol, Smart Home, LabVIEW

## 1. Introduction

Internet nodes consist of not only computers or mobile devices but also with everyday appliances that are used such as household electronic devices that have connected to the Internet. Based on a report from the National Intelligence Council realm of IOT (Internet of Things) became one of the six technologies with the greatest risk of the potential impact until 2025 in the United States [1]. The main objective was internet nodes connect different devices inside communication of several different objects [2]. The object has a device, operating system and programming language that works independently. Household devices over time becomes devices that that has an internet device nodes and has a sensor that connected. This combination of technologies from household devices, computers, sensors and computer networks on IOT also emphasize the growing communication technology with ubiquitous [3]. The technology has the properties of low cost, pervasive connectivity and make "all" the wireless device connected and has four models consisting of a connectivity device to device, device to the cloud, the device to the gateway, and the back end of data sharing [4].

Pervasive computing was often referred as ubiquitous field of research that brings revolutionary paradigm for computing models. Pervasive computing offers a large area for computation of size, mobility and usability [5]. Ideally, pervasive model should be able to recognize and accommodate

each connected device [6]. Suppose that a device which has a sensor that automatically collects information, transfer and take their own decisions can be categorized as pervasive devices. In a computer network, pervasive protocol was able to recognize a device and could provide communication services automatically [7].

TCP was a protocol that can be directed, could mean transmit data via the specified route. This would result in adding density of traffic on the network that is used so that the process will take longer [8]. In the smart home environment is not required sending large amounts of data [9]. UDP has an advantage compared with no use of sequence and acknowledgment field as was done by TCP and the most obvious advantage was extra bytes fewer [10]. In addition, UDP does not need to wait for the reception or storing data in memory until the data received [11]. This means, the UDP application is not slowed down by the admissions process and memory be released early.

LabVIEW was an acronym for Laboratory Virtual Instrumentation Engineering Workbench, computer software for processing and visualization of data in the field of data acquisition [12], control instrumentation and automation industry first developed by the company National Instruments. A previous study [13] built the same concept but using three devices, the agent, and the agent directory user agent that uses Salutation Discovery Protocol and Service Location Protocol for pervasive process. Another study [14] proposes for wireless sensor network environment on smart home using LabVIEW with HTTP-CoAP.

In the smart home environment, it does not need a large size and complex data transmission between monitoring sites and monitoring center is needed in the Smarthome tension monitoring system. Therefore, researchers wanted to examine how technology utilizes LabVIEW to configure the UDP protocol is pervasive in Smart Home Environment. In this research we discuss about how the design and configuration of UDP is pervasive in LabVIEW and the successful implementation of the UDP protocol in LabVIEW.

## 2. System Design

In this section, we design the pervasive protocol on the domain host and client, each using a pc. The design of pervasive protocol on the host and client is indicated in **Figure 1**. As the initial conditions, the hosts were in Listen condition. Then the client introduces himself on the host to provide such acknowledgment Client Name, its IP and Service on the host. In this process host indirectly receive the client, but to check duplication, whether the client is already there or not, even if the client is already on another host. The host then sends an ACK, Host Name, IP and Host Header which is then stored by the Client.

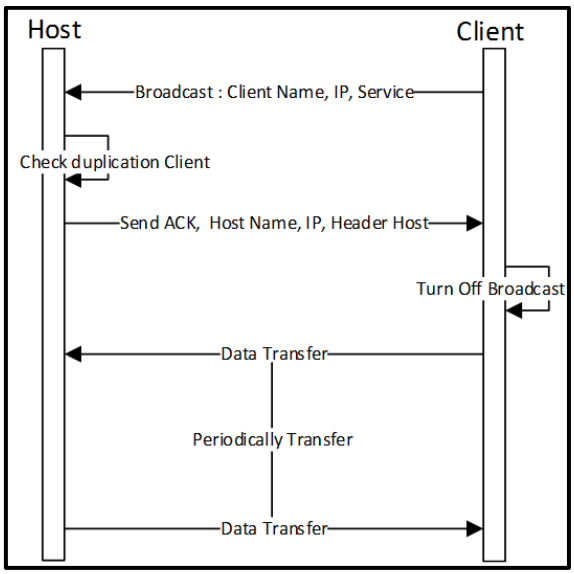


**Figure 1.** System design

Once the Client recognize the host, the next process was turning off the broadcast. Furthermore, Client and Host recognize each other and was able to transmit data. The service that can be delivered was sensor data transmission services, the provision of services control at client were boolean or numerically. Then it would be designed how to make or how to design a protocol that will be used on the host and the client.

### *2.1. Protocol Design on The Host*

In the Host we designed in detail about what is done by the host from beginning. The details about what would be done by the hosts can be seen in **Figure 2**.

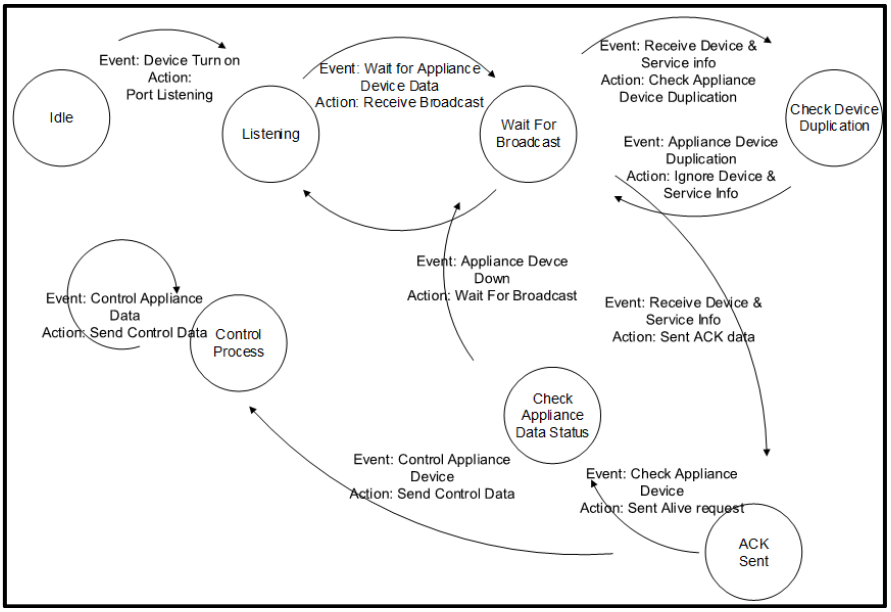


**Figure 2.** Host state machine diagram

Conditions on State Machine Diagrams in **Figure 2** each would be explained as follows:

1. Idle : The first condition to the host when the host first time run was idle, this condition states that the host does not perform any process but at idle only has the acting only listen.
2. Listening : In this condition, the host waiting for clients who are in the host area. Clients who process the data broadcast would be heard hosts on this process. This process would run continuously and this process explained in Wait For Broadcast Process.
3. Wait For Broadcast : This condition states that the hosts just waiting to receive broadcast data. If the host does not receive the data broadcast, the host simply repeating the process of listening and wait for broadcast. But if the host receives broadcast data then will proceed to the Check Device Duplication process.
4. Check Device Duplication : In the broadcast process performed by the client, the host receives a client name and client ip then this process will be checked whether the previous client had been in communication with the host or not. If you've ever it will be continued in the control process, but if it is not going to continue the process of ACK sent and check the status of the data appliance.
5. ACK Sent : After the hosts knew no data were duplicated, the host would send an ACK, in the sense that the host is ready to receive and transmit data. ACK sent by the host was to make requests about what services are owned by the client so that the host knew about anything that can be controlled by the host.
6. Check Appliance Data Status : After replying to client services that are owned, the host process Check Status Data Appliance to return at idle condition to wait if there is another client who wants to join the host..

7. Control Process : However, a client that has previously been identified will be stored and exchanged data. In this process service owned by the client will be able to take total control of the host. In the sense of total control over suppose lit LED, the data transmission gyroscope sensor or services that are owned by the client.

*2.2. Protocol Design on The Client*

After designing Host, will be designed Client. In this design will do the state machine design, the details about what will be done by the client, from the beginning until client being recognized by the Client Host well as in **Figure 3**.

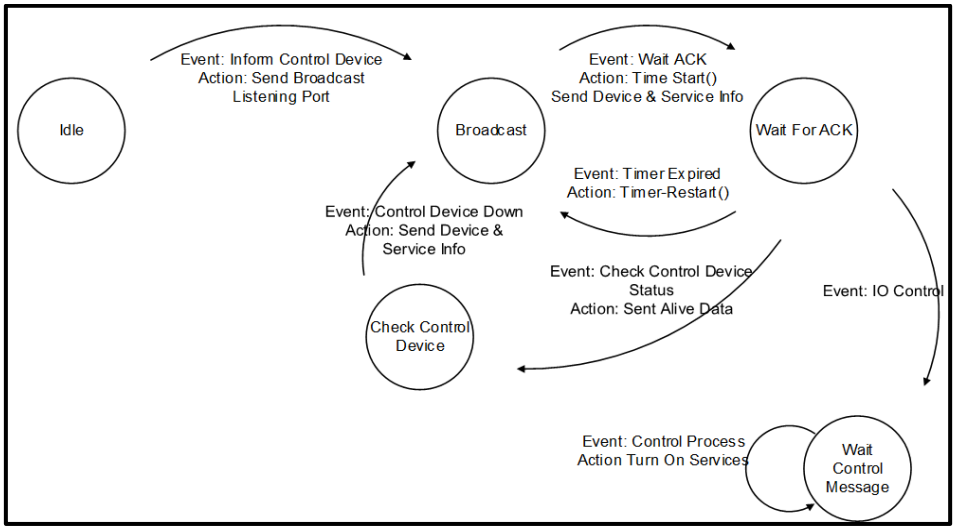


**Figure 3.** Client state machine diagram

Each condition in client state machine in **Figure 3** will explained as follows:

1. Idle : this condition is the initial condition of the client, the client sends the data broadcast was used to spread the data to find the host.
2. Broadcast : in this condition, data was broadcasted until the hosts detected on the "listen" condition. Once the data received by the host and the host sends an ACK, the condition of the client was no longer broadcasted the data, but to wait and count time until certain conditions were described in the Wait For ACK.
3. Wait For ACK : in this condition allows three further conditions. The first possibility when ACK was received from the host, it means the client and the host can communicate well and continued in "wait control message". The second condition of the host ACK waiting time is too long exceeded the specified time, so that in these conditions the client would have to broadcast back on the "broadcast" process. The third condition that if two conditions have been performed was entered in the process “check control device”.
4. Wait Control Message : in these conditions was a continuation of the conditions of the host "control process", the host and client were in condition to communicate. Client waits for commands from the host to operate in accordance the data transmitted from the host.
5. Check Control Device : This condition was the worst conditions, where the client does not receive the ACK sent by the host at the "ACK Sent", and this condition is also re-broadcast after data broadcasted where both did not work well at the time who determined. Client will check whether the package sent was correct or not. This process is repeated to the "broadcast" to the client find a host.

## 3. Implementation and Result

At this session discussed about the implementation on the client side and on the host side, each of which will be implemented in accordance with a design that have been made in the previous session. As well as the results will show the results of the processing system so that conclusions can be drawn.

### *2.3. Host and Client Implementation*

Each state in the state machine implemented on a "case structure" on LabVIEW, each case illustrates the structure of each state on the design in Figure 2 that implemented as shown in **Figure 4**(a) for the host and Figure 4(b) for the client.

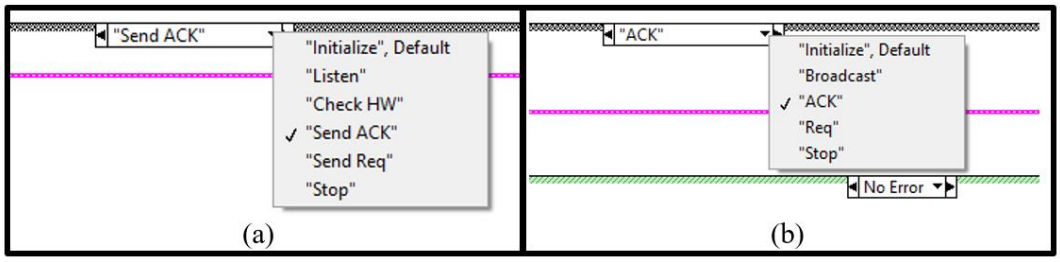


**Figure 4.** (a) Host case implementation. (b) Client case implementation.

Each process has a "case structure" itself, for one implementation of the delivery ACK as shown in **Figure 5**. In Figure ACK delivery process was implemented on the client. In case "Send ACK" There are some case for handling the "error" or "no error", as well as for handling if any of these conditions occurs. Data sent in the form of a "bundle" which contains the "client ID", "client service", "IP hardware" and "host name". "Bundle" in the sense of all that data into one package in which the packet sent to the client.

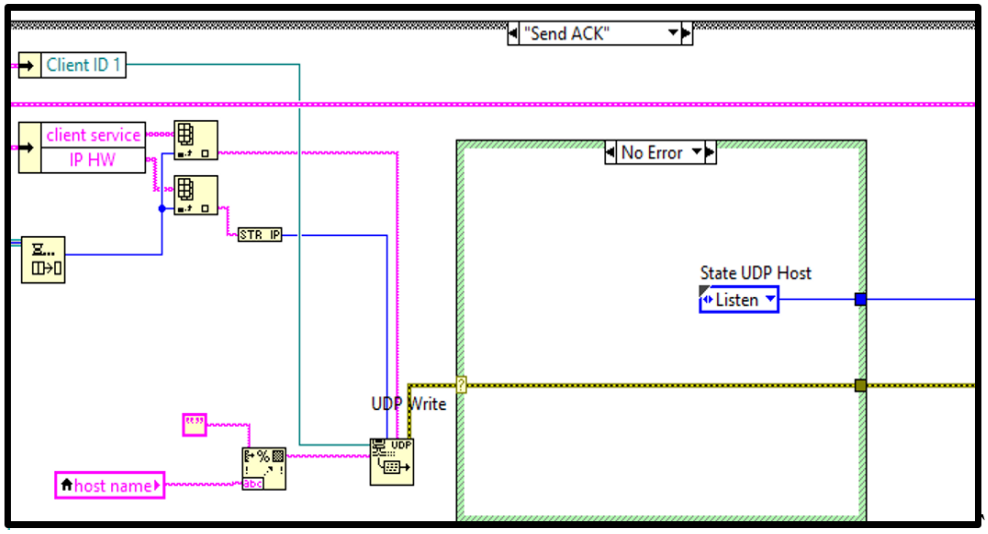


**Figure 5.** Host send ack implementation

Implementation sending ACK on the host will be received by ACK state on the client side. For the implementation of the client ACK in **Figure 6**. This figure shows that the client receives the data bundle that will be processed further.

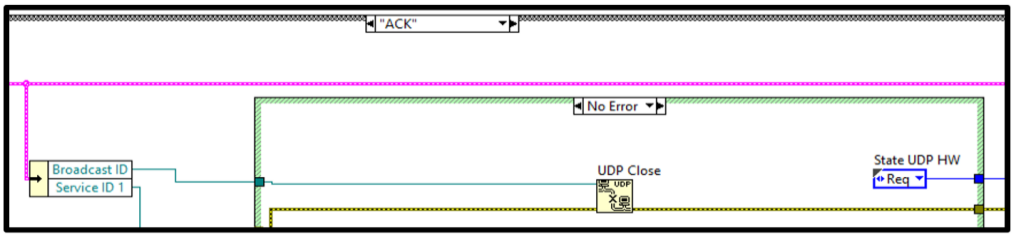


**Figure 6.** ACK client implementation

Just like on the host side, on the client side is also provided for handling the case of two conditions, the condition of "error" and "no error". Client receives an ACK to be processed In later steps. Implementation on both sides of the host and client successfully and completed for further testing processes that get results.

### *2.4. Result*

After encoding, the result on each side of both the client side and the host side. For results from each side of the interface is in **Figure 7**.

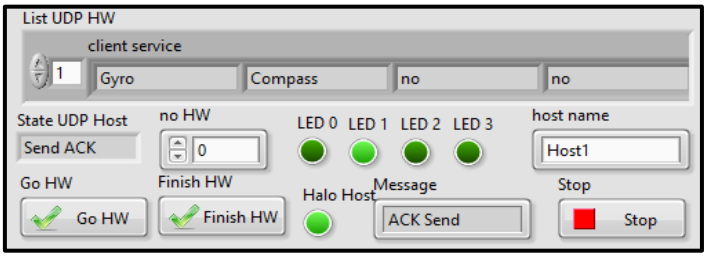


**Figure 7.** Host interface

After running the system, in **Figure 7** there is a menu "UDP Host State" which claims the system is running in the state "Send ACK" as well if it is in another state then the column will declare it is running on the state. List UDP HW is a list of clients who are interacting with the host. Function "no hw" when running the simulation it.

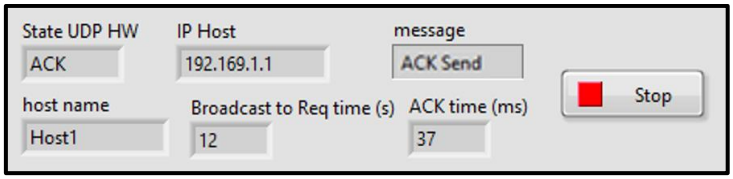


**Figure 8.** Client interface

For the client on **Figure 8**, it only displays "State UDP HW" that functions the same as the host, then the client can figure out "Host IP" and could exchange messages via the "message" to the host. In this trial Broadcast to produce as much as 12s Time Req. In terms of the host broadcast time to find a client takes the 12s, while the time required for delivery of ACK takes 37ms.

## 4. Conclusion and Future Works

We have built this system with mild or as a prototype or usually should called lightweight UDP in LabVIEW. Client can relate to Host with both pervasive. In this research service discovery was performed by the host can identified the client properly so that said construction of this study is successful. Host can control any services which provided by client. This research prototype was undertaken on two PCs with the assumption that the PC are a client and a host on a home device. So for future works this system can be implemented on an embedded system which has a high complexity

with heterogeneous devices that can communicate well. We hope to implement this systems in Arduino, Xbee etc, or using National Instrument hardware such as MyRio etc.

**Acknowledgments**
Special thanks to the Laboratory of Computer System and Robotics Computer Engineering team, Department, Faculty of Computer Science University of Brawijaya. This research to support our research ini "Penelitian Unggulan 2016" scheme in BPP Program, Faculty of Computer Science, Brawijaya University.

## 5. References

[1] Atzori, L., Iera, A. and Morabito, G., "The Internet of Things: A survey. pp. 2787-2805," *Elsevier Computer Networks,* pp. 2787-2805, 2010.

[2] Vermesan, O. and Friess, P., Internet of Things - Global Technological and Societal Trends, Aalborg, Denmark: River Publisher, 2011.

[3] Munir, S.A., Ren, B., Jiao, W., Wang, B., Xie, D. and Jian, M., "Mobile Wireless Sensor Network: Architecture and Enabling Technologies for Ubiquitous Computing," in *International Conference on, Advanced Information Networking and Applications Workshops*, Niagara Falls, Ontario, Canada, 2007.

[4] Rose, K., Eldridge, S. and Chapin, L., "The Internet of Things: An Overview Understanding the Issues and Challenges of a More Connected World," *The Internet Society (ISOC),* 2015.

[5] Dourish, P. and Bell, G., Divining A Digital Future - Mess and Mythology in Ubiquitous Computing, London, England: The MIT Press, 2011.

[6] Banavar, G., Beck, J., Gluzberg, E., Munson, J., Sussman, J. and Zukowski, D., "Challenges: an application model for pervasive computing," in *MobiCom '00 Proceedings of the 6th annual international conference on Mobile computing and networking*, New York, USA, 2000.

[7] Saha, D. and Mukherjee, A.,, "Pervasive computing: a paradigm for the 21st century," *Computer,* vol. 36, pp. 25-31, 2003.

[8] M. Welzl, Network Congestion Control - Managing Internet Traffic, Chichester, England: John Wiley & Sons, Ltd, 2005.

[9] Petrovic, D., Shah, R.C., Ramchandran, K. and Rabaey, J., "Data funneling: routing with aggregation and compression for wireless sensor networks," in *Sensor Network Protocols and Applications, Proceedings of the First IEEE*, 2003.

[10] Snoeren, A.C. and Balakrishnan, H., "An End-to-End Approach to Host Mobility," in *MobiCom* '00 *Proceedings of the 6th annual international conference on Mobile Computing and Networking.*, New York, USA, 2000.

[11] Byers, W.J., Luby, M., Mitzenmacher, M. and Rege, A., "A Digital Fountain Approach to Reliable Distribution of Bulk Data," in *SIGCOMM*, New York, USA, 1998.

[12] Jimenez, F.J. and Frutos, D.J., "Virtual Instrument for Measurement, Processing Data, and Visualization of Vibration Patterns of Piezoelectric Devices," *Elsevier,* vol. 27, no. 6, pp. 653-663, 2005.

[13] Bhatti, N., Dhomeja, L.D. and Malkani, Y.A., "Service Discovery Protocols in Pervasive Computing: A Review," in *Multi-Topic Conference (INMIC), IEEE 17th International*, 2014.

[14] Serna, M.A., Sreenan, J.C. and Fedor, S., "A Visual Programming Framework for Wireless Sensor Networks in Smart Home Application," in *Sensor Networks and Information Processing (ISSNIP)*, Singapore, 2015.