Demonstration of the Plug and Play of Smart Sensor Nodes

Ashwin Juvva, and D. Gurkan
Dept. of Engineering Technology
University of Houston
Houston, TX 77204-4020
akjuvva@mail.uh.edu, dgurkan@uh.edu

Suman Gumudavelli, and Ray Wang
Mobitrum Corporation
Silver Spring,MD
20910 sgumudav@mail.uh.edu,
ray_wang@mobitrum.com

***Abstract* – Sensors and actuators are used in wide range of applications in industry for monitoring, control and automation. Plug and play is a term used to describe the characteristics of a device specification which facilitates the discovery of a hardware component in the network. Management of the smart sensor nodes, similar to the IEEE 1451 Network Capable Application Processor (NCAP), in a network through plug and play reduces the tedious job of monitoring them manually. The objective of this paper is to extend on our previous work of communication between multiple nodes to show the plug and play capability. The demonstration uses IEEE 1451.1 definitions and architecture from a guideline and perspective point of view. If a universal on-the-wire format has been accepted, this work will be exactly applicable to IEEE 1451 approach.**

Keywords- NCAP, IEEE 1451,Plug and Play, sensor networks, smart sensor node, ISHM.

I. INTRODUCTION

In the world of network communications it would be difficult to imagine the monitoring of the physical connections and communication problems of any networking component manually as it is time consuming and a very tedious process. Plug-and-play capability is a natural goal in achieving a unified network interface with automatic loading of drivers of any component as soon as it is connected to a network. The plug and play capability of the smart sensor nodes allows to inform the client node immediately when a new server node joins the network and similarly notify the client NCAP when already existing NCAP node leaves the network. The model of 1451 compatible node is shown in Fig.1 as outlined in the newly-released 1451.0. Here, the archived standard, IEEE 1451.1 (referred to as Dot1 from here on), has been placed on the application layer for functional exchange of information and data among the NCAPs. In this respect, NCAPs can be referred to as smart sensor nodes transferring sensor data or acting on the commands issued to actuators. In addition, per definition of NCAPs, they can also have embedded algorithms for better decision fusion and control [1].

II. IEEE 1451.1

IEEE 1451.1 is used for communication between NCAPs or an NCAP and other nodes in the network for distributed measurement and control applications. This standard defines interfaces for connecting NCAPs to control networks through a common object model which defines Transducer Blocks, Function Blocks, and NCAP Blocks. Its purpose is to provide a network-neutral application model that will reduce the transducer interfacing efforts to different kinds of control networks [2].

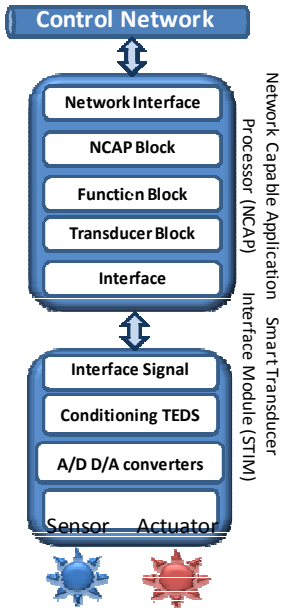

Figure 1. Model of IEEE 1451 Suite of Standards in a Distributed System of Sensors and Actuators.

Dot1 part of the 1451 suite of standards has been archived for further compatibility studies towards IEEE 1451.0-related specifications. Dot0 has placed 1451.1-related functions into the application layer as shown in Fig. 2. Even though DOT1 is archived it would still be a good reference to showcase the functionality of 1451 based communication model.

This work is funded by Mobitrum Corp.

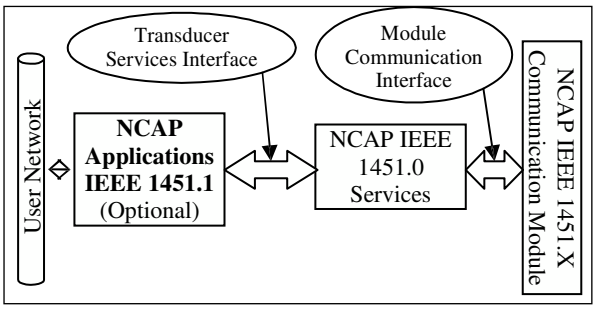

Figure 2. Dot0 and Dot1 Conceptual Relationships.

III. PLUG AND PLAY ON A SENSOR NETWORK

A network becomes dynamic when all the nodes in the network have the plug and play capability. Dynamic network evolution achieves network performance improvement and cost reduction. Through the plug and play capability of smart sensor nodes we achieve greater network flexibility and efficiency. The figure 3 shows the Plug and Play of the smart sensor nodes. The newly joined server node in the network sends self identification messages to the client and the server leaving the network is identified by the keep-alive messages. Although the concept of keep-alive messages and self-identification of nodes are well-studied in computer networking areas, a sensor network's plug and play capability has not been well defined. This paper and demonstration's goal will be to raise awareness on the requirements and expectations from plug and play capabilities on sensor networks.

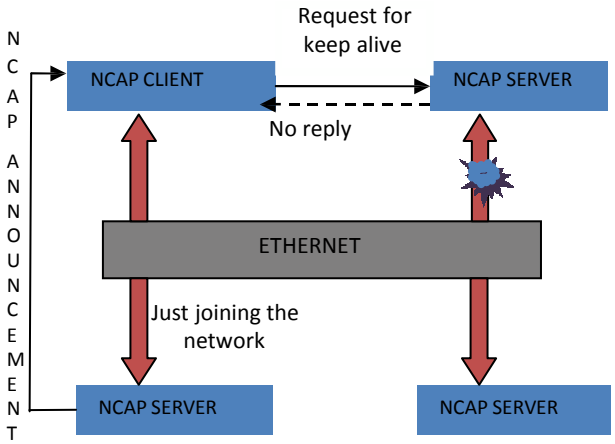

Figure 3.Plug and Play of NCAPs in a network.

There are two types of network changes during the plug and play of nodes: a new node joining the network and a node leaving the network.

A. *Discovery of a new NCAP in the network:*

A newly added server node to the network can be discovered by the client node. The discovery process can be done in two ways, (i) the client NCAP polls the server NCAPs for the keep-alive messages. The advantage of this implementation: when the server is switched on the client comes to know about the newly added server node after getting the keep alive messages from the server nodes. The disadvantage of this implementation is the network delay in knowing the newly added node into the network. (ii) The second way of discovering a newly added server NCAP in a network is the server NCAP sending the self identification messages to the client node through the PSK_NCAPBLOCK-_ANNOUNCEMENT publication key. Once the client node identifies the newly added server node, it displays a message consisting of the IP address of the newly added node and the list of server nodes maintained by it. The advantage of this implementation: the list of server nodes at the client side is updated at a much faster rate than the first case. The second method is implemented in our demonstration.

Figure 4 shows the procedure followed when a new server node is added to the network. The client updates the list of server nodes maintained by it as soon as it gets the PSK_NCAPBLOCK_ANNOUNCEMENT and the list of server object properties once it gets PSK_NETWORK_-VISIBLE_SERVER_OBJECT_PROPERTIES from the newly added server node.

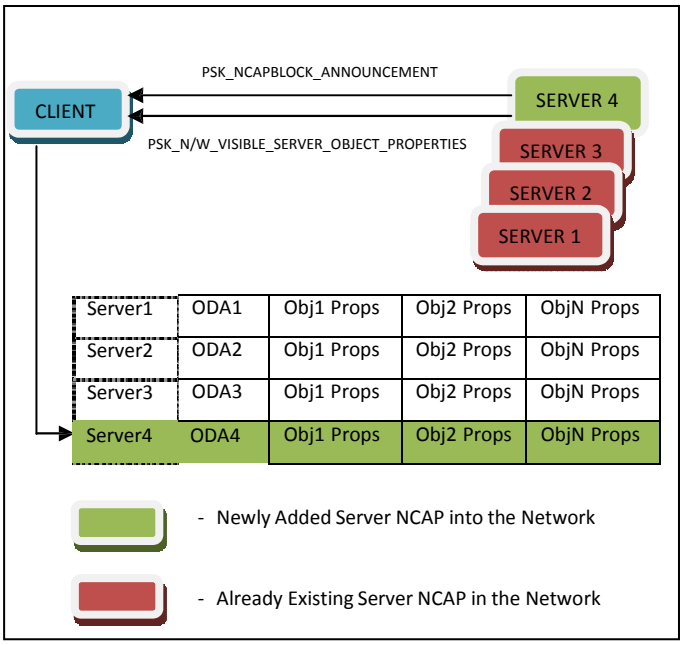

Figure 4. New Server NCAP discovery in a network .

B. *An NCAP leaving the network:*

A server node leaving the network can be identified by the client node using the keep-alive messages.

A periodic request for keep-alive messages is sent from the client node to the server nodes. The publication key for sending the request for keep alive messages from the client is PSK_REQUEST_NCAP_BLOCK_ANNOUNCEMENT and the keep alive messages from the servers has the publication key PSK_NCAP_BLOCK_ANNOUNCEMENT.

These publication keys are the same as the one's used initially for discovering the server NCAPs in the network.

The client node maintains the list of server nodes in the network in a table. After getting the keep alive messages from the server nodes, the client node checks the table to find out which server node is out of the network and updates the table accordingly.

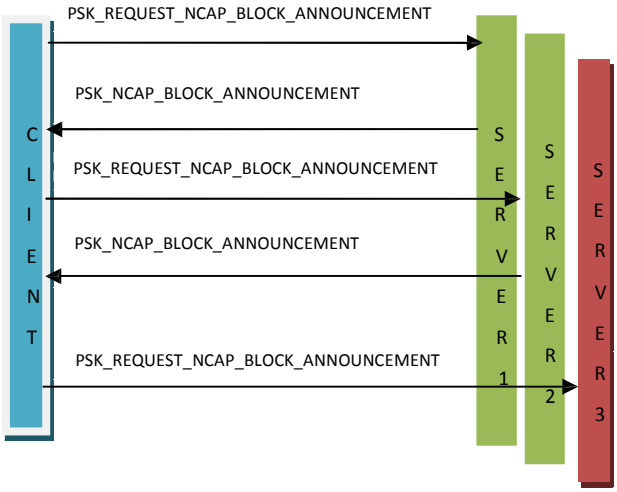

Figure 5. Server NCAP identified while exiting from the network.

Fig 5. shows the procedure followed when an already existing server node leaves the network. The table entry of the node is deleted from the list of servers maintained by the client. Fig 6. shows the variation of the number of nodes maintained by the client in the table as the server nodes enter and leave the network.

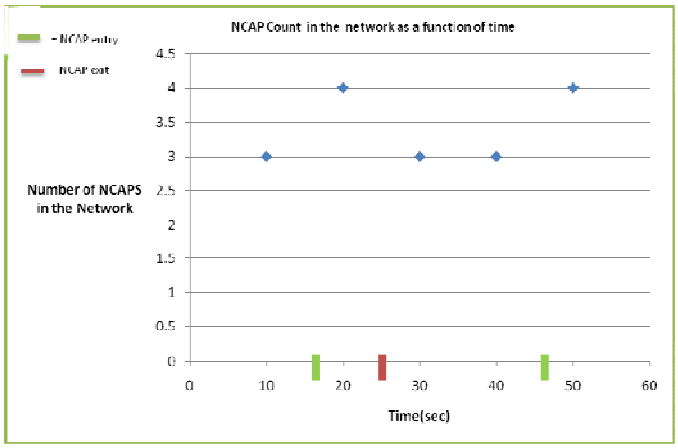

Figure 6. NCAP count in the network as a function of time

IV. TESTBED

NCAPs are developed in C++ environment and they implement one transducer block, one function block, and one NCAP block. The NCAP node is manufactured by Mobitrum: MOBEE NET™ – Wireless Intelligent Sensor. This NCAP provides the following capabilities: support for publish-subscribe communication, support for client-server communication, marshal and demarshal the messages according to the on-the-wire format publicly available on request and on ISSNet web site. Client node can collect the data from multiple Server nodes and segregate the data received according to object dispatch address of the server.

The testbed consists of a network of a client NCAP and nine server nodes. Data is composed of the readings from the real transducers with their timestamps from the NASA Stennis Space Center (SSC). The client collects the sensor data from each server node. Server one is programmed to send the first sensor data from the data file, server two sends the second sensor readings and so on. The data has been provided to this experiment from actual measurements performed on the demo test stand for rocket engine testing at the NASA-SSC. The control screen for this setup is displayed in Fig. 7.

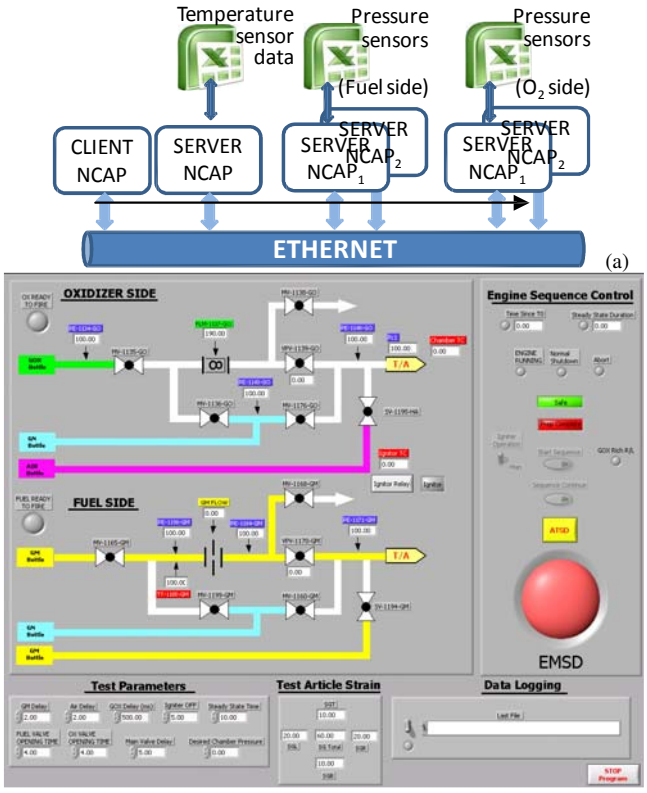

Figure 7. (a) Network with server and client NCAPs on the same Ethernet. (b) Control screen for demo rocket engine test stand at NASA-SSC.

V. SOFTWARE ARCHITECTURE

The NCAP software is built with three layers. The base layer is the Adaptive Communication Environment (ACE) which takes care of TCP and UDP communication. It provides software interface for application to use the TCP and UDP communication. The layer above the ACE is the abstract IEEE 1451.1 C++ reference implementation of abstract transducer information model. It implements most of the base classes. This implementation is not complete and can be considered as a lighter version of complete implementation [6].

The topmost layer is the actual application layer which implements and instantiates the NCAP with NCAP block, function block, transducer block and other ports required.

The number of nodes communicating were limited to six (in the demonstration at [1]). In this demonstration the number of NCAPs communicating has been extended to a total of ten. The publication keys for the communication between the client and the server nodes are already defined in IEEE 1451.1 with each key having a numerical value.

VI. RESULTS

The results are composed of the snapshots of messages displayed when/if a node enters the network and is detected by the client node:

NO OF NCAPS PRESENTLY IN THE OBJECT DISPATCH ADDRESS TABLE: 6

WITH IP ADDRESSES:

0. 192.168.2.203

1. 192.168.2.205

2. 192.168.2.207

3. 192.168.2.206

4. 192.168.2.204

5. 192.168.2.208

RECEIVED 'NEW PSK_NCAPBLOCK_ANNOUNCE-MENT' AND APPENDED TO THE OBJET DISPATCH ADDRESS (ODA) TABLE

NO OF NCAPS PRESENTLY IN THE OBJECT DISPATCH ADDRESS TABLE: 7

WITH IP ADDRESSES:

0. 192.168.2.205

1. 192.168.2.204

2. 192.168.2.207

3. 192.168.2.206

4. 192.168.2.208

5. 192.168.2.203

6. 192.168.2.209

VII. CONCLUSION

This demonstration of fully functional working model would encourage the use of the standard in the industry. The health monitoring of the IEEE 1451 nodes and sensor networks in general through the plug and play of smart sensor nodes would help standards and control network design approaches better prepared for future applications.

VIII. FUTURE WORK

The present work is to extend the previous work in terms of the number of NCAPs communicating in the network using 1451 compatible format. This work demonstrates the discovery of a node entering the network and leaving the network. This paper presentation will include a demonstration of the plug and play of smart sensor nodes during the conference.

ACKNOWLEDGMENTS

The authors would like to thank Fernando Figueroa of NASA – Stennis Space Center, Integrated System Health Management Division, for his helpful discussions.

REFERENCES

[1] Suman Gumudavelli, D. Gurkan, and Ray Wang, "Emulated Network of IEEE 1451 Application with Multiple Smart Sensor Reports," IEEE Sensor Applications Symposium, 2009.

[2] IEEE 1451.1 Standard for a Smart Transducer Interface for Sensors and Actuators—Network Capable Application Processor (NCAP) Information Model. 1999.

[3] Anshul Singla, H. Ma, R. Franzl, H. Liu, D. Gurkan, D. Benhaddou, X. Yuan, J. Morris, M. Turowski, and F. Figueroa, "Design of a Test Suite for NCAP-to-NCAP Communication based on IEEE 1451," IEEE Sensor Applications Symposium 2008.

[4] Richard Franzl, Jonathan A. Morris, and D. Gurkan, "Implementation of IEEE 1451.1 Conformance/Functionality Testing using LabView," IEEE Sensor Applications Symposium 2008.

[5] IEEE Instruments and measurement magazine, April 2008.

[6] Schneeman, R., "Implementing a Standards-based Distributed Measurement and Control Application on the Internet", Sensor Integration Group, NIST, June 1999.