USING SNMP FOR REMOTE MEASUREMENT AND AUTOMATION

Nikolay Rumenov Kakanakov, Elena Dimitrova Kostadinova, Grisha Valentinov Spasov

Department of Computer Systems and Technologies, Technical University of Sofia, branch Plovdiv, 61 "St. Petersburg" Blvd., Plovdiv 4000, Bulgaria, phone: +359 32 659758, www: http://net-lab.tu-plovdiv.bg/ , e-mail: {kakanak, gvs}@tu-plovdiv.bg, ellikostadinova@yahoo.com

An example implementation of SNMP agent for Beck IPC@Chip platform is presented. The agent software is written in C/C++ and is based on the SNMP framework provided by DMH Software. Test-bed network and configuration are built in "Distributed Systems and Networking, Virtual Laboratory" in Technical University, branch Plovdiv. The presented experimental results provide a base data for evaluation of the leaks and performance of the example implementation. The experiments are monitored and analyzed using specially built MS .NET application. The results are stored in XML files for further analyses. The examined metrics are request/response delay and jitter.

Keywords: Distributed Embedded Systems, Remote Monitoring and Control.

1. INTRODUCTION

The increasing invasion of Internet in all areas and the decreasing prizes of computation hardware led to the development of the remote monitoring and control of all kind of devices – in medicine, at home, at office, in industry, etc. For interoperation of these devices and for remote interfacing with the user a communication technology should be chosen. There are a number of candidates for this role and there is a need for evaluation and estimation of the candidates. Among these candidates is the application layer protocol, embedded in the standard TCP/IP stack – the Simple Network Management Protocol (SNMP). It has been developed for remote management of network devices but it has a good adaptation for distributed automation. In this paper an example implementation of SNMP for embedded controller is presented and an examination of time delay and jitter of the implementation is made [2].

1.1 SNMP Architecture

Simple Network Management Protocol (SNMP) [1] is a request-reply protocol, which operates on application layer of the TCP/IP protocol stack and uses UDP as a transport mechanism. The network administrators use SNMP to monitor and control network-attached devices, and to get information about their status and characteristics. Besides, SNMP allows finding network problems [7]. SNMP is studied (examined) in two aspects: a protocol for exchange of management information, and a database for the managed data. Three versions of SNMP exist: SNMP version 1 (SNMPv1), which is chosen in this paper, SNMPv2 and SNMPv3 (its standardization is pending) [2].

An SNMP-managed network consists of three basic components – managed

devices, agents, and management stations [7]:

- Managed devices – managed devices (called network elements) are hardware devices such as routers, switches, servers, etc., that are connected to networks. Each one of the managed devices must contain an SNMP agent.
- Agents – an agent is a software module that is running in a managed device and process incoming get-or-set requests. Also, the agents collect and store information for the managed devices (for example, the number of received error packets) and make this information available by SNMP.
- Management stations – these devices execute applications that monitor and control the managed devices and provide the interface between the human network manager and the management system. The management station issues requests for management operations, and then, passes received responses (from the agents) to the users; it receives traps from agents as well.

1.2 Management Information Base (MIB)

Each management station and each agent in an SNMP-managed network maintains a local database, known as Management Information Base (MIB). The MIB objects are organized hierarchically as a tree, the levels of which are assigned by different organizations. A part of this hierarchy is shown on figure 1. The top-level MIB object IDs belong to three main standard organizations: CCITT, ISO, and joint ISO/CCITT, but the basic part of the current MIB activity is dedicated to the Internet community and, how is shown, is a part of the ISO branch. The current standardized in Internet MIB, named MIB-II, is a part of the "mgmt" branch and contains a number of standard objects grouped in different categories ("system", "interfaces", etc.). Also, the tree is extensible, because the vendors can define their own object IDs as a portion of "experimental" (for not yet standardized objects) and "private" branches [5, 7].

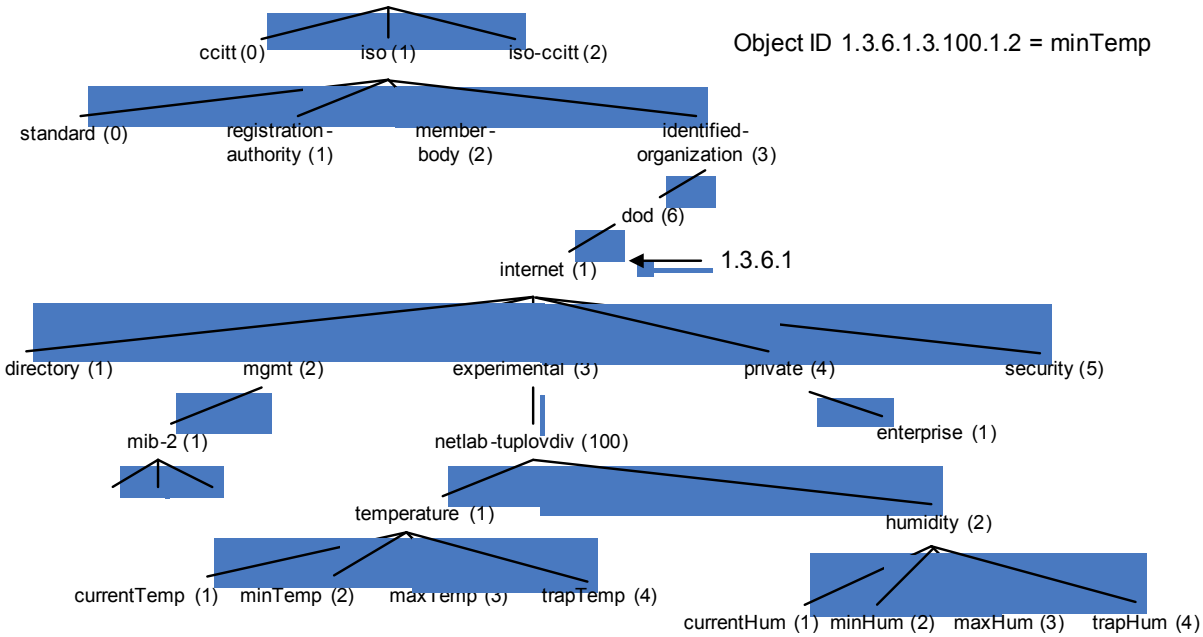

Figure 1: A part of the MIB object identifiers tree – the root is unnamed

Different computers use different ways for data representation, which make

difficult the capability of SNMP to exchange information between managed devices. For that reason, SNMP use a special subset of Abstract Syntax Notation One (ASN.1) [6], called Structure of Management Information (SMI), which defines both the packets exchanged by SNMP and the structure of managed objects [4, 7].

The management station and the agent use MIB and a relatively small set of commands to communicate: Get, GetNext, Set, GetResponse and Trap. The management station sends a Get or GetNext command to retrieve the value of one or more managed objects from an agent, and a Set command to set the value of a managed object within the agent. By GetResponse the agent returns an answer to the above three commands. The last command – Trap – is the only initiated by the agent; it sends a Trap to the management station when a specific event occurs [7].

1.3 SNMP packet format

The SNMP is a packet oriented protocol with two different types of packets (in SNMPv1). The first is used for exchange of management information and contain two basic parts: a packet header and a protocol data unit (PDU), shown on figure 2. The header contains two fields: version and community name, the PDU – the command to be performed ("Get", "Set", and so on) and the operands (which are object instances participating in the operation) [7].

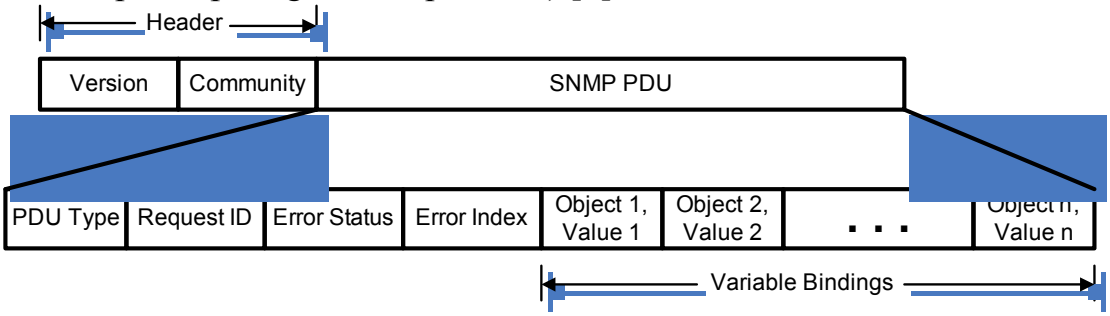

Figure 2: Format of SNMPv1 Get, GetNext, GetResponse, and Set PDUs

The second type of packet, used in SNMPv1, is for the traps (illustrated in figure 3) and consists of the following fields [7]:

- Enterprise – shows the type of managed object generating the trap;
- Agent address – shows the address of the managed object generating the trap;
- Generic trap type – shows the type of the generated trap;
- Specific trap code - shows a specific trap code;
- Time stamp – shows the time elapsed between the last (re)initialization of the network entity and the generation of the trap;
- Variable bindings - associate object instances with their current values.

PDU Type	Enterprise	Agent Address	Generic Trap Type	Specific Trap Code	Time Stamp	Variable Bindings

Figure 3: Format of the SNMPv1 Trap PDU

2. Example Implementation

In this section an example implementation of the SNMP agent is presented. The platform used for testing is IPC@Chip [10] SC12 Embedded Web Controller and the experiments are carried out in "Laboratory of Distributed Systems and Computer

Networks" in Technical University of Sofia, branch Plovdiv [11]. The implemented SNMP agent, based on an example SNMP proxy agent [9], possesses the following capabilities: it can process incoming Get requests for the current temperature and humidity (measured by sensor attached to the controller); incoming Set requests for setting an admissible interval (minimum and maximum) for the read temperature and humidity, and sends a Trap if the sensor readings are out of this range.

To ensure this functionality, two new groups of managed objects are added to the MIB tree's "experimental" branch at first, both shown in figure 2. The objects are described in two ways: in a MIB file using standard rules in [4], and in a .c file. The host system uses the MIB file to display the readable name of the object and to interpret its value when requested. The second file is in the agent and consists the whole objects information as described in the MIB file – object syntax, object status, etc. Hence, the agent uses this file to determine whether the requested object is managed and changeable. By means of these two files, our new objects are made accessible for the agent as well as for the host.

Next, some custom functions are implemented to the agent: Get, returning the values of the new objects, and Set, allowing a change on these values. A Trap generating function is also added to the agent. When starts, the agent checks repeatedly the trap condition and sends a trap if needed. For testing the custom Get functions, temperature and humidity measurement is implemented. There is parallel task started on the controller that queries the sensor and writes down the values in memory. The SNMP task, on request, reads the values form the memory. The access to that memory area is controlled by a semaphore.

SC12 @CHIP-RTOS version 1.10B [10] with SNMP MIB support and large memory model is used. The agent is compiled with the Borland C/C++ 5.02 Compiler and downloaded to the IPC@CHIP via FTP. Some fields in the chip initialization file (i.e., chip.ini) need to be modified – READ- and WRITECOMMUNITY, both set to "public". Read-community string is like a password and allows (or denies when is "private") getting information from the agent. Write-community string, respectively, protects the managed objects to be set to a specified value. On these conditions the agent is able to reply to our Get and Set requests, and notify with Traps when the trap event occurs.

3. EXPERIMENTAL RESULTS

The implemented SNMP agent is tested with custom build MS.NET application, based on ActiveSocket library (ActiveExperts [8]). The time delays are measured using QueryPerformanceTimer, built in .NET platform and providing accuracy of 0.001ms. The test-bed scenario includes a single SNMP agent, running on IPC@Chip platform. There are static SNMP objects that can be get or set – maxTemp and minTemp, maxHum and minHum, and two SNMP objects that has dynamically changed values – currentTemp and currentHum. These two objects present the temperature and humidity measured on the controller in real time. The management

station is a Intel P4 PC with 1GB of RAM, running MS Windows XP SP2, with MS .NET Framework.

The experiments are carried out 255 times for each command and the average delay is calculated, together with the standard deviation and jitter. The subsequent SNMP requests are sent in relatively big intervals to isolate the interference between them. The delay times for different commands are shown in Table1 and their jitter in figure 4.

Table 1: Delays for different SNMP commands

	GetStatic	Set	GetTemp	GetHum
min	46,02	43,71	45,87	46,29
average	52,99	52,02	55,06	53,67
deviation	0,76	0,97	0,98	0,82
deviation %	1,43	1,87	1,78	1,52

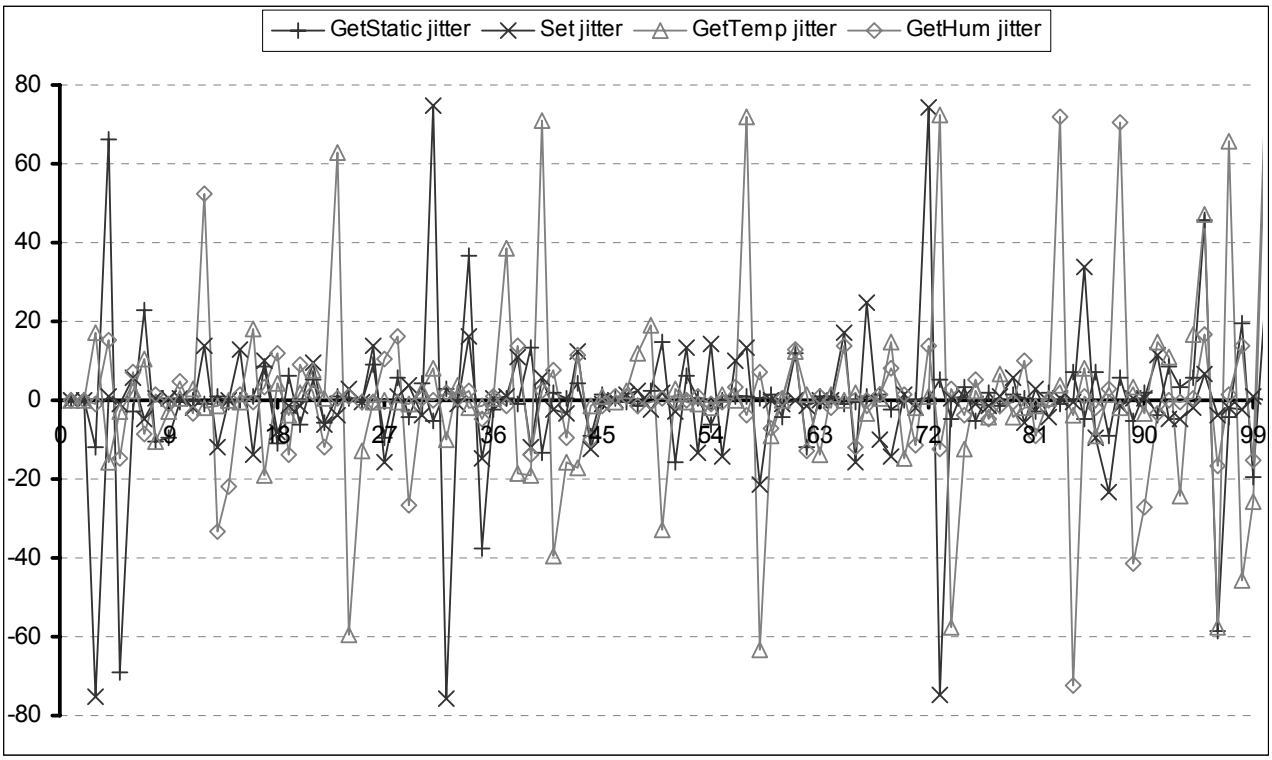

Figure 4: Jitter for different SNMP commands for IPC@Chip platform.

The delay is measured on the Application-Level and on the Packet-Level [3]. The proposed results are that measured on the Application-Level because they provide data for more complex examination – both network and management station delay. The data collected on the Packet-Level is in close relation to the network parameters and is used to calculate the Round-Trip Times of the network [3]. The measured Round-Trip Time, using packet analyzer is about 43ms. It is very stable and has no significant jitter because the experiments are carried out in local area switched network.

4. CONCLUSIONS

The calculated results show that the SNMP can be applied to some of the fields of distributed control, like home automation and remote parameter monitoring because the delays are relatively small. The results also show that there are not significant differences in Get and Set commands and between getting of static or dynamic values. The last is true when applying different tasks for temperature and humidity measurement and for SNMP. Thus, separating the sensor driver and SNMP task leads to better performance and reduced delay in measurement.

The calculated deviation is about 1 to 2% which makes this approach suitable for soft real-time and non-real time applications. Nevertheless, the calculated jitter for getting temperature and humidity is more deviated from zero than jitter for the other commands. It can be explained by the delay for waiting for the semaphore to access shared memory for exchange of measured temperature and humidity between tasks.

5. ACKNOWLEDGEMENT

The presented work is supported by National Science Fund of Bulgaria project – "**VU-966/2005**", entitled "Web Services and Data Integration in Distributed Automation and Information Systems in Internet Environment", under the contract "**VU-MI-108/2005**", and contract **992 NI-17 R&DS** of TU Sofia.

6. REFERENCES

[1] Case, J., M. Fedor, M. Schoffstall, and J. Davin, "A Simple Network Management Protocol", RFC 1157, Performance Systems International, Performance Systems International, and the MIT Laboratory for Computer Science, May 1990.

[2] Hong Liu, Xu Shao, Lingshan Kong, Wei Ding, "Integration of Web into an embedded SNMP-based management environment," 5th IEEE International Conference on High Speed Networks and Multimedia Communications, 2002, pp.320-324.

[3] Laurent Andrey, A. Lahmadi1, O. Festor, "On Delays in Management Frameworks: Metrics, Models and Analysis," In 17th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management - DSOM 2006, Springer.

[4] McCloghrie K., and M. Rose, "Structure and Identification of Management Information for TCP/IP-based Internets", RFC 1155, Performance Systems International and Hughes LAN Systems, May 1990.

[5] McCloghrie K., and M. Rose, "Management Information Base for Network Management of TCP/IP-based Internets: MIB-II", RFC 1213, Performance Systems International and Hughes LAN Systems, March 1991.

[6] International Telecommunication Union, "Information technology - Abstract Syntax Notation One (ASN.1): Specification of basic notation", International Standard ISO/IEC 8824-1, July 2002.

[7] Cisco Documentation, available on we: http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/, July 2007.

[8] ActiveSocket Network Communication Toolkit – website: http://www.activexperts.com/files/activsocket/manual.htm.

[9] DMH Software, The Internet Management Company – Website: http://www.dmhsoftware.com/.

[10] Beck IPC GmbH, IPC@Chip Homepage - http://www.beck-ipc.com/ipc/.

[11] Distributed Systems and Networking, Virtual Labopratory - website: http://net-lab.tu-plovdiv.bg/.