Multiplatform Management of a Hard Real-Time Ethernet Switch

Aleksander Pleszko, João Paulo Barraca
DETI/IT - University of Aveiro
Aveiro, Portugal
{alek, jpbarraca}@ua.pt

Joaquim Ferreira, Pedro Gonçalves
ESTGA/IT - University of Aveiro
Aveiro, Portugal
{jjcf, pasg}@ua.pt

***Abstract*— The Hard Real-Time Ethernet Switch (HaRTES) allows using the same network to handle multiple traffic flows, without compromising the performance of real-time applications. Furthermore, it also provides flexible and on-line scheduling techniques with admission control capabilities, thus real-time communications flows can be added, removed and updated online with strict temporal isolation. However, HaRTES lacked a standard management interface to configure its parameters and view its status. This paper describes a multiprotocol (SNMP and NETCONF) management interface design, and implementation. It also presents a preliminary validation of the two management technologies.**

***Keywords*— Embedded Systems, SNMP, NETCONF, Ethernet Switch, Quality of Service, Flexible Time-Triggered, Real-Time communications**

I. INTRODUCTION

Networked Embedded Systems (NES) are widely disseminated in many application domains ranging from industrial automation to building automation and vehicular system. Some application domains exhibit strict timeliness, predictability and precedence constraints. In these cases, special-purpose real-time communication networks, known as fieldbuses, must be used to achieve the desired properties.

One network technology that became widely used in these systems is Ethernet, however, it was not originally developed to meet the requirements of NES, namely in what concerns key aspects such as predictability, timeliness and reliability. There are currently available technologies, that enable mechanisms of guaranteed Quality of Service (QoS), such as MPLS [1] and RSVP [2] especially combined with IntServ [3] and DiffServ [4] models, however they only provide statistical guarantees.

Over the last decade, several Ethernet-based protocols have been developed, e.g.: Ethernet-Powerlink, Profinet, EtherCAT and Ethernet/IP, which take advantage of some of Ethernet's appealing attributes, e.g. large bandwidth, cheap silicon and high availability, while removing or reducing the sources of non-determinism arising from its MAC protocol and/or from the current switched architecture. At the Ethernet level, 802.1Qat allows the implementation of the DiffServ model through its priority field, while the Stream Reservation Protocol (SRP) [5], which was recently proposed as the standard for end-to-end reservations at the Ethernet level, follows the IntServ model.

However, the timeliness guarantees provided by those protocols are essentially static, based on pre-run-time analysis. On-line admission control is not generally available and neither is on-line adaptation of the communication requirements according to effective needs or to a quality-of-service (QoS) adaptation policy. This has motivated the development of a new generation of Ethernet switches, Hard Real-Time Ethernet Switch (HaRTES) [6], able to provide timeliness guarantees, efficient bandwidth usage and support for operational flexibility as required by dynamic real-time distributed embedded systems. The new switching platform was build upon recent work on the FTT (Flexible Time-Triggered) communication paradigm to develop Ethernet switches with enhanced transmission control, traffic scheduling, service differentiation, transparent integration of non-real-time nodes and improved error confinement mechanisms, particularly with respect to temporal misbehaviors. For the purpose of preforming QoS reservations, SRP support was added.

However, currently there is no standardized interface for remote HaRTES management, i.e., parameterizing and monitoring the switch behavior. Therefore, efficient and deterministic management tools are required to take advantage of the properties of the HaRTES switch.

This paper describes a multiplatform management interface design, and implementation and evaluation for a HaRTES switch and presents a preliminary validation of the two management approaches, from the perspective of management latency and variance.

The rest of the paper is organized as follows. Section 2 presents related work referring to management of network system. In Section 3 The Real-Time Ethernet Switch and its features i.e. QoS technologies are described. Section 4 presents the two proposed management technologies applied to HaRTES switch. Section 5 describes the implementation of the switch management interface, while Section 6 presents results regarding the latency of two well known management solutions. Finally, Section 7 concludes the paper and proposes some future lines of work.

978-1-4673-4941-3/12/$31.00 ©2012 IEEE

II. RELATED WORK

SNMP is widely adopted by most of the real-time Ethernet (RTE) equipment supporting several RTE protocols, e.g., TT Ethernet (TTE), Ethernet Powerlink, Profinet, Industrial Ethernet, etc.

The TTE A664 Pro Switch [7] has a built-in management module for network monitoring (SNMP v1). The TTE A664 Pro Switch supports secure network management and allows data loading and querying of health and status information. Weidmuller IE-SWxx-M Industrial Ethernet switches [8] can also be managed via SNMP. The IE-SWxx-M switches support traps for the link-up, link-down, confirmation error, cold restart and warm restart functions. Profinet [9], also uses SNMP for maintaining and monitoring network devices.

Ethernet Powerlink adopts a proprietary protocol for the network management derived from CANopen, based on Process Data Objects (PDO), Service Data Objects (SDO), and Network Management (NMT) Objects. According to [10] Ethernet Powerlink routers are managed by Powerlink SDO and optionally by the SNMP v3.

A common property of all previous mentioned RTE protocols is that they do not allow dynamic reconfiguration with real-time guarantees. They are fully static systems in which all operating conditions are completely defined at pre-runtime In these RTE protocols system reconfiguration involves stopping the system, apply the modifications and restart it. Since monitoring is not a time-critical activity and maintenance is performed offline, SNMP is well suited for these tasks in the case of the above mentioned RTE protocols. NETCONF, a newer technology, is other valid alternative, however and to the best of our knowledge, it is not supported by current RTE equipment.

For the specific case of FTT Ethernet networks based on the HaRTES switch and supporting timely operational flexibility, it is necessary to assess the performance of network management technologies. Notice that, both SNMP and NETCONF do not provide real-time guarantees, however this is no impairment for HaRTES, since modifications of the communications requirements are not made directly by the management services. Online requests to modify communication requirements are processed by the admission control and, if accepted, their timeliness is secured by real-time scheduling.

III. FTT-ENABLED SWITCH

The FTT-enabled Ethernet switch was described in a number of recent publications that address its main concepts and features [11] and that discuss different design options, e.g. regarding the integration of the FTT master [12] and of traffic scheduling servers [13]. HaRTES is an FTT-enabled switch based on the Flexible Time-Triggered (FTT) paradigm with the FTT master included inside the switch (Master Module in Figure 1).

HaRTES is designed for micro-segmented networks and follow the Flexible Time-Triggered (FTT) paradigm, supporting two main traffic classes, real-time and non-real-time, the former being further divided in synchronous and asynchronous. The synchronous traffic is time-triggered and scheduled on-line. This class of traffic is polled by the switch using a master/multi-slave transmission control technique, according to which a single poll is broadcast once every so-called Elementary Cycle (EC) of fixed duration, using a specific message called Trigger Message (TM). Nodes decode the TM and transmit immediately the scheduled messages and the switch takes care of their serialization. The scheduling is done in a way that all messages referred in a TM fit in the respective EC. The FTT master holds information about the nature of the data exchanges regarding the type of addressing (unicast, multicast and broadcast) and which end nodes are involved. With this information, the master computes which messages follow disjoint paths and build schedules that exploit this parallelism, increasing the aggregated throughput [13].

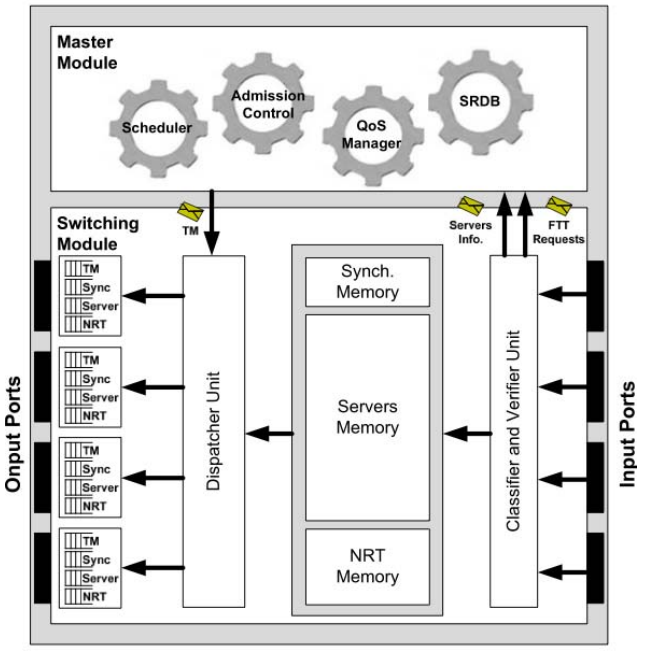

Figure 1 - Functional architecture of the server-based FTT-enabled Ethernet switch [11].

The asynchronous and non-real-time traffic, on the other hand, is sent to the switch without any transmission control thus arriving at any instant. The switch confines this traffic within the associated servers, thus making sure that it does not interfere with the synchronous one and also making sure that asynchronous real-time traffic is handled with servers that have higher priority than those used for non-real-time traffic. One very important feature of this kind of traffic is the absence of need for transmission control in the end nodes that boosts the flexibility of the system since it is now possible to connect any kind of existing Ethernet node without modification, e.g., general purpose workstations with general purpose operating systems and network drivers, being its traffic automatically confined by the switch.

Figure 1 presents the functional architecture of the switch to support server-based scheduling. It follows closely the one in [12], which does not support server-based scheduling, with

two main modules, the Switching Module and the Master Module. The traffic arrives via the input ports in the former module and is submitted to the Classifier and Verifier Unit that classifies and validates the received massages. The data messages are forwarded directly to the memory unit while FTT control messages, e.g., negotiation messages, are transferred to the Master Module. The memory is divided in three independent zones, each one for each traffic class, namely synchronous, server and non-real time. The other main block inside the Switching Module is the Dispatcher Unit that handles the output queues per traffic type and, according to the scheduling per- formed by the master and conveyed in the Trigger Message, transmits the selected messages from the memory directly. The Master Module executes a complex set of operations, namely the admission control, QoS manager, scheduler and it also implements a System Requirements Database to store the information related to the traffic management [14].

IV. MANAGEMENT TECHNOLOGIES

Over the last two decades IETF produced several network management standards notably Simple Network Management Protocol (SNMP) [15], Common Open Policy Service (COPS) [16], and more recently NETCONF [17]. SNMP technology, despite some well-known issues, became the de facto management technology for IP networks. During the last years IETF dedicated efforts to standardize its successor, the NETCONF technology, aiming to solve some shortcomings of SNMP, while adding improvements mainly related with security and encoding efficiency [18], while offering a more simplified development process.

SNMP follows a client/server model in which the managed device, called SNMP agent, assumes the server role, while the SNMP Network Management Systems (NMS) assumes the client role. The roles between SNMP elements are inverted in event reporting scenarios where the SNMP agent notifies the SNMP manager by means of unsolicited notification (TRAP) operations. SNMP agent stores status and configuration information (such as various types of statistics). Management is done through querying and modifying the appropriate variables in the managed device, sending SNMP operations (*Get, GetNext, GetBulk*, and *Set*) to the SNMP agent. Protocol operations are encoded in a very efficient manner and transported over User Datagram Protocol (UDP), resulting in a lightweight message transport specially conceived for overloaded networks.

The SNMP protocol is widely used today in network management as well as in the area of equipment management, mainly as a monitoring tool. Despite the fact that the status of SNMPv1 and SNMPv2 is historic and only SNMPv3 is a full standard, SNMPv3 is not much used in network management [24], and SNMPv2c is dominant.

SNMP information is stored in a repository named Management Information Base (MIB). Each object in the database has assigned a name, a value, a type, a description containing detailed information needed to correct implementations, and a set of operations you can perform on the object (read/write value). Objects are stored as leaves in the tree-like structure. An object can only be identified when the full path to it is provided.

Network Configuration Protocol (NETCONF) [17], is a network management protocol that provides mechanisms to install, manipulate, and delete the configuration of network devices as well as its monitoring. NETCONF design followed a layered approach composed of four layers: a transport layer that implements the information transport, a RPC layer that implements the XML-RPC remote procedure call, an operations layer that implements NETCONF operations and a content layer containing the configuration data.

Although NETCONF [19] has been designed to be independent of the data modeling language, the IETF recommends using an Extensible Markup Language (XML). YANG is data modeling language specially designed to handle the configuration data as well as the protocol messages. The language that was conceived to ease the human's interpretation, follows a C like syntax and has an XML equivalent named YANG Independent Notation (YIN) intended for the machine reasoning.

NETCONF base protocol defines nine operations. Additionally RFC 5717 defines a partial lock mechanism for NETCONF Remote Procedure Calls (RPC), allowing partial configuration blocks, defined by means of a filtering XPath expression. NETCONF elements may additionally define new operations during the capability exchange process, which, if common to both peers, can then be used during the session.

NETCONF management information can be carried by a set of security enabled transport protocols such as SSH [20], SOAP [21], BEEP [22], or TLS [23]. In order to promote interoperability SSH support is mandatory. However, management elements are free to negotiate an alternative secure communication protocol.

V. HARTES MANAGEMENT INTERFACE

Due to the modular nature of the switch architecture, integrating management functions to the existing system implied the design and instantiation of an additional module. This module exposes internal state of the switch, by operating over information existing at the different modules, to the external world. A custom communication interface was developed for this purpose. It provides access to the internal state variable through a very efficient Request/Response model based on the existence of well-known attributes. Managing a given attribute (e.g. getting its value) implies issuing a command stating the attribute to retrieve. Currently, it is only supported manipulation of attributes, lacking support for the generation of events. While this interface aims to provide an efficient way of communicating with the internal components of the HaRTES switch, it lacks standardization. Therefore, an architecture was devised to interface the HaRTES switch with standard management tools. As depicted in Figure 2, at the core of this architecture relies the Management Server component. It acts as a bridge between the communication socket it provides and the HaRTES

modules. The communication protocol used in this socket is kept private. However, wrappers allow integration of additional existing management solutions.

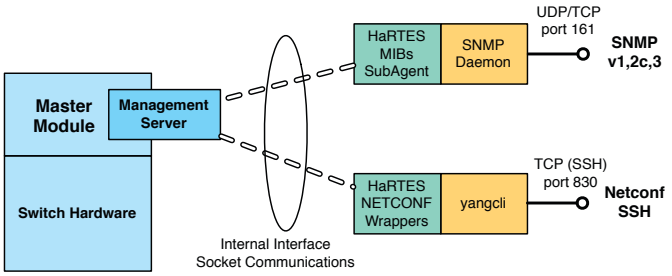

Figure 2 – General architecture of the management components.

Currently, two wrappers were developed, implementing three different MIBs, through two management solutions. The management solutions are SNMP and NETCONF. The MIBs are the IF-MIB [26], SRP MIB [27] and a custom developed MIB, the HaRTES MIB. The parameters of the three MIBs are available through both SNMP and NETCONF as the wrappers integrate directly with the management software of both solutions. In the case of SNMP, the wrapper integrates as a subagent of the SNMP daemon, while in the case of NETCONF, it integrates as a subagent of the `yangcli` tool.

Figure 3 depicts the simple MIB created for the purpose of providing more information regarding the internal operation of the HaRTES switch. For the purpose of analyzing QoS reservations, the SRP MIB will be used.

```
ATNOG-HARTES-MIB generated by evoking: snmptranslate -
M+. -mATNOG-HARTES-MIB -Tp -IR hartes
+--hartes(123321)
   |
   +--hartesObjects(1)
      |
      +-- -R-- Integer32 eCycleDuration(1)
      +-- -R-- Integer32 sasWindowDuration(2)
      |
      +--gwTable(3)
      |  |
      |  +--gwEntry(1)
      |     |  Index: turnaroundTime, switchingDelay,
      |     |         transmissionTime
      |     |
      |     +-- -R-- Integer32 turnaroundTime(1)
      |     |        Range: 1..2147483647
      |     +-- -R-- Integer32 switchingDelay(2)
      |     |        Range: 1..2147483647
      |     +-- -R-- Integer32 transmissionTime(3)
      |              Range: 1..2147483647
      |
      +-- -R-- Integer32 idleTime(4)
      +-- -R-- Integer32 qosType(5)
      +-- -R-- Integer32 maxPeriodicMsg(6)
      +-- -R-- Integer32 maxAPeriodicMsg(7)
      +-- -R-- Integer32 mtu(8)
      +-- -R-- Integer32 maxPckSize(9)
      +-- -R-- Integer32 bRate(10)
            +-- -R-- Integer32 interfaces(11)
```

Figure 3 – Internal state HaRTES MIB.

The process of querying the HaRTES switch takes several steps. It starts when the management protocol triggers a query in the local tool (`yangcli` or `snmpd`). If the attribute queried is present in the MIBs supported, the request is translated to an internal query, and then dispatched to the Management Server. This component will then validate the request, locate the module where the information is stored, and then provide with the answer requested. It should be made clear that the existence of this socket-based interface was devised due to several reasons. First it better isolates operation of the wrappers and the Management Server. This aims at reducing the impact of management operations to the Master Module of the HaRTES switch. Isolation also reduces the number of exploits through the management interface, as if required, firewall characteristics can be added to the Management Server. The second characteristic of this interface is that it allows reuse of the same management interface to several different management solutions. In this case the same interface can be easily used with both SNMP and NETCONF, while relying on components requiring a small number of modifications from the components auto-generated by the support tools.

VI. Evaluation and Results

We evaluated the prototype management interface to determine its suitability for the task of managing a distributed system with real time constrains. It is not expected for SNMP and NETCONF to be real time aware. However, the inherent latency and variations observed when using these management methods will have impact on the remaining systems of a distributed management infrastructure. We also aim at increase the comprehension of the latency penalty of both solutions, in comparison with our protocol agnostic interface.

Our experimental setup consisted of a Pentium(R) Dual-Core CPU T4400 @ 2.20GHz and 2GB of RAM, running Ubuntu 12.04 LTS 32 bits. Software modules included SNMP version 5.7.1, NETCONF version 2.2-3 and FTT-Server version 2.5.3-1. Clients were launched as separate processes and the FTT-Server custom module was on a separate thread.

Evaluation focused in triggering a series of management commands as fast possible, while a passive monitoring application was capturing all communications being made. This approached allowed us to monitor management traffic without causing much interference to the management process. Values of jitter and communication latency experienced by the client application were afterwards extracted from the log. All analysis was done offline after the experiments completed. We aimed at minimizing external delay factors. Therefore, no network was actually used and all communication was done to the local host. If an Ethernet connection were to be considered, we would observe a fixed amount of latency due to transmission buffers, reception buffers, as well as transmission to the medium and other existing queues. Also, jitter would be added in non-easily deterministic way [25].

Our work focused in the development of a protocol agnostic management interface, enabling the HaRTES switch to be managed both through SNMP (v2c and v3) and through NETCONF. Therefore, we evaluated the performance constrains of both approaches. In order to have a clear idea about the overhead introduced by each protocol, and the absolute minimum bounds we can expect to experience given

the current hardware, the internal, protocol agnostic, communication delay was also measured.

Each management operation, for each protocol, was repeated 150 times. Afterwards, the top 10 values (roughly 6.67%) with higher difference from the average of the entire set of experiments were discarded. The remaining 140 values were considered for analysis, and consider a 95% confidence interval.

As a reference for the analysis, we measured the internal latency from the time the request was received, until the switch management software was queried and a result was provided to the management protocol. The latency values obtained are depicted in Figure 4. Internal queries were always lower than 250us, and presented an average of 156us +/- 5.54us, with a standard deviation of 33.5us. In order to calculate the penalty of SNMP or NETCONF, this average value should be deducted. While observing some jitter in internal communications, 95% of the results stayed within 7.10% of the average value. Because we are dealing with microseconds, and given the hardware available, we expect this jitter to have minimum impact in management.

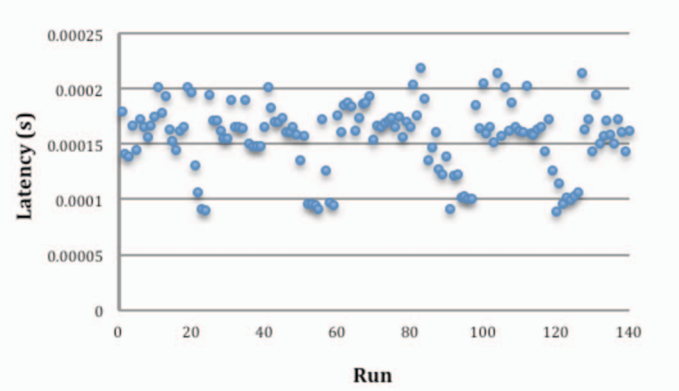

Figure 4 - Internal communication latency independently of the management approach.

SNMP was evaluated using its two most used varieties: SNMP v2c and SNMP v3. All management actions focused in getting a single value. As described before, SNMP considers UDP communications and small individual transactions following a request/response approach with two packets for each transaction. All information required is contained in the request, and the reply also takes a single UDP packet. SNMP v3 follows the same approach but introduces changes to messages in order to add support for shared key authentication. Still, each management action is composed by a simple exchange of messages through UDP. TCP can also be used for SNMP v3, but this way of operation was not included in this work.

Figure 5 depicts the values obtained for both SNMP v2c and SNMP v3. Due to the slightly higher complexity of SNMP v3, the management latency was a little higher than SNMP v2c. The first presented an average query latency of 1.4ms +/- 0.041ms, with a standard deviation of 92.1us, while SNMP v2c presented an average query latency of 0.9ms +/- 0.034ms, with a standard deviation of 207.5us. It is interesting to observe that SNMP v2c presents lower average latency, but also less error. In part, the smaller number of packets of each transaction when using SNMP v2c can explain this behavior. If the internal latency values are subtracted, SNMP v2c presents a latency increase of 577%, while SNMP v3 presents an increase of 910%.

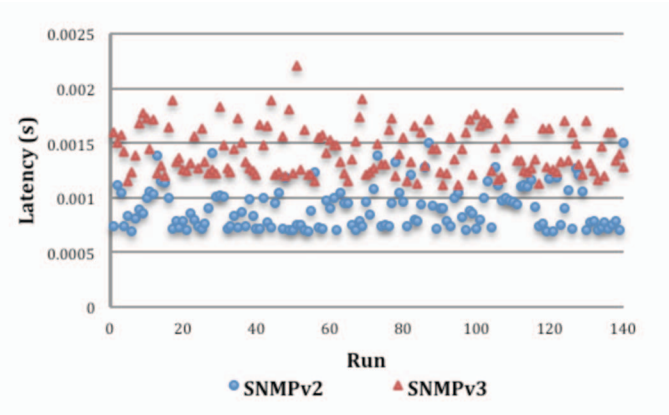

Figure 5 - Scatter chart of SNMPv2 and SNMPv3

The same methodology was followed in order to evaluate NETCONF. Many transport methods are supported by the latest standard. As SSH is mandatory, and the one most commonly available, we focused in evaluating NETCONF over SSH. Because SSH is a secure protocol supporting both peer authentication and private communications, the overhead produced is expected to be much higher. Figure 6 depicts the latency observed for the best 140 queries. As shown, NETCONF introduces much higher latency into management functions, averaging at 583*ms* +/- 15*ms,* with a standard deviation of 92.1*ms*. The added overhead is so high that the values obtained for internal latency are three orders of magnitude lower, and therefore negligible. The error margin itself is two orders of magnitude higher, which reflects the high overhead of this management protocol.

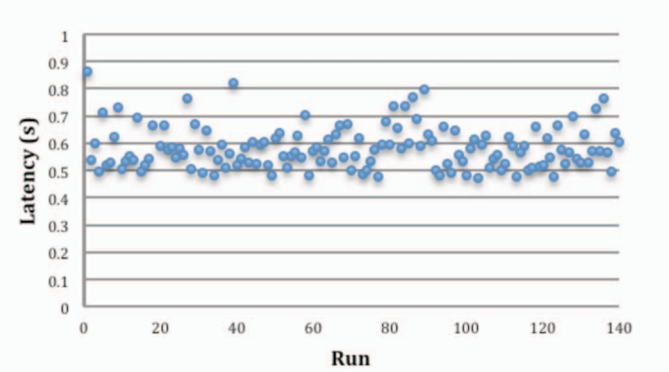

Figure 6 - Scatter chart of NETCONF latency

From the evaluation of these three management approaches, it is clear that the use of any standard compliant management protocol will introduce very high latency and variability in processes. The lightest solution is SNMP v2c, followed by SNMP v3. However, the term lightest can be very misleading as it increases latency by a factor of 5. NETCONF falls in a completely different bucket, increasing management latency by a factor of almost 4000. It should be taken in consideration the advantages of authenticating communication peers and secure the management process, which NETCONF supports. However, because best practices state that management traffic

should be transported in networks completely isolated from clients (e.g. dedicated VLANs), these advantages become less clear.

However, when considering real-time systems, with dependencies in remote managed systems, lower latency will result in higher scalability for the system, as well as higher levels of determinism. In scenarios where critical timings must be observed, such as automotive or industrial scenarios, SNMP still proves to be the best management approach if standards are to be respected. Custom developed solutions show to provide higher performance, at the cost of interoperability and eventually future evolution.

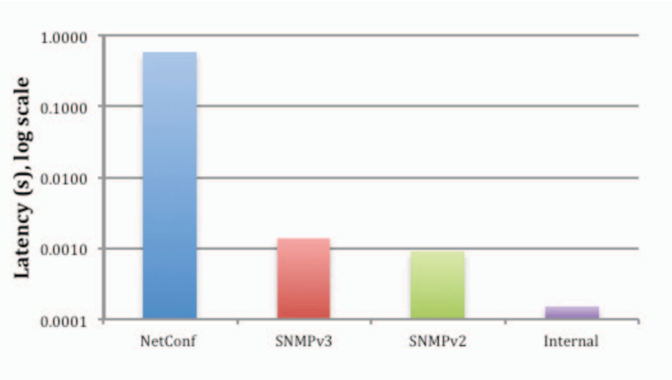

Figure 7 - Latency of the different management methods in comparison with internal latency

VII. CONCLUSIONS

Networked embedded systems, relying on Ethernet-based communication infrastructures, are becoming increasingly popular in many application domains with strict timeliness and dependability requirements. Real-time Ethernet-based protocols, developed over the last decade, fail to provide flexibility in terms of online adaption of the communication requirements. The Hard Real-Time Ethernet Switch, developed within the scope of the Flexible Time-Triggered Switched Ethernet protocol allows using the same network to handle multiple traffic flows, with strict temporal isolation, so that real-time flows can coexist with non real-time ones.

The HaRTES switch lacked a standard management interface to configure its parameters and monitor its status. This paper described the design process of a multiplatform (SNMP and NETCONF) management interface to help monitor the switch performance and configure its parameters. Experimental results shown that smaller delays introduced by SNMP protocol make it better to monitor and configure the HaRTES switch.

Future work will include the implementation of asynchronous notification operations, as well as full management support, to enable online configuration of the switch. We are also considering a more detailed analysis of SNMP and NETCONF adequacy for hard real-time applications.

REFERENCES

[1] E. C. Rosen, A. Viswanathan, and R. Callon, Multiprotocol Label Switching Architecture, RFC 3031, 2001.

[2] B. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification, RFC 2205, 1997.

[3] B. Braden, D. Clark, and S. Shenker, Integrated Services in the Internet Architecture: an Overview, RFC 1633, 1994.

[4] Y. Bernet, R. Yavatkar, P. Ford, F. Baker, L. Zhang, M. Speer, B. Braden, B. Davie, E. Felstaine, and J. Wroclawski, A Framework for Integrated Services Operation over Diffserv Networks, RFC 2998, 2000.

[5] "IEEE Standard for Local and Metropolitan Area Networks---Virtual Bridged Local Area Networks Amendment 14: Stream Reservation Protocol (SRP)," IEEE Std 802.1Qat-2010 (Revision of IEEE Std 802.1Q-2005), vol., no., pp.1-119, Sept. 30

[6] R. Santos, "Enhanced Ethernet Switching Technology for Adaptable Hard Real-Time Applications", PhD Thesis, Department of Electronics Telecommunications and Informatics Uninersity of Aveiro, 2011.

[7] http://www.tttech.com/products/ttethernet/flight-and-rugged-hardware/switch-a664-pro-24/ - Accessed June 29, 2012.

[8] http://www.weidmuller.ru/news/pi_ie_en.pdf - Accessed June 27, 2012.

[9] PROFIBUS International: PROFINET Specification, Profinet IO Application Layer Service Definition

[10] EPSG Draft Standard 301, Ethernet POWERLINK Communication Profile Specification Version 1.1.0, EPSG 2008.

[11] R. Santos, R. Marau, A. Oliveira, P. Pedreiras, and L. Almeida, "Designing a costumized Ethernet switch for safe hard real-time communication", Factory Communication Systems, 2008. WFCS 2008. IEEE International Workshop on, 2008, pp. 169-177.

[12] R. Santos, R. Marau, A. Vieira, P. Pedreiras, A. Oliveira, and L. Almeida, "A synthesizable ethernet switch with enhanced real-time features", Industrial Electronics, 2009. IECON '09. 35th Annual Conference of IEEE, 2009, pp. 2817-2824.

[13] R. Santos, A. Vieira, P. Pedreiras, A. Oliveira, L. Almeida, R. Marau, and T. Nolte, "Flexible, efficient and robust real-time communication with server-based Ethernet Switching", Factory Communication Systems (WFCS), 2010 8th IEEE International Workshop on, 2010, pp. 131-140.

[14] R. Santos, A. Vieira, R. Marau, P. Pedreiras, A. Oliveira, L. Almeida, and T. Nolte, "Implementing Server-Based Communication within Ethernet Switches", 2nd Workshop on Compositional Theory and Technology for Real-Time Embedded Systems (CRTS'09), Washington, USA, 2009.

[15] J. Case, M. Fedor, M. Schoffstall, and J. Davin, Simple Network Management Protocol (SNMP), RFC 1157, 1990.

[16] D. Durham, J. Boyle, R. Cohen, S. Herzog, R. Rajan, and A. Sastry, The COPS (Common Open Policy Service) Protocol, RFC 2748, 2000.

[17] R. Enns, NETCONF Configuration Protocol, RFC 4741, 2006.

[18] P. Gonçalves, J. L. Oliveira, and R. Aguiar, "A study of encoding overhead in network management protocols", International Journal of Network Management, pp. n/a-n/a, 6 Feb 2012 2012.

[19] M. Bjorklund, YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF), RFC 6020, 2008.

[20] M. Wasserman and T. Goddard, Using the NETCONF Configuration Protocol over Secure Shell (SSH), RFC 4742, 2006.

[21] T. Goddard, Using NETCONF over the Simple Object Access Protocol (SOAP), RFC 4743, 2006.

[22] E. Lear and K. Crozier, Using the NETCONF Protocol over the Blocks Extensible Exchange Protocol (BEEP), RFC 4744, 2006.

[23] M. Badra, NETCONF over Transport Layer Security (TLS), RFC 5539, 2006.

[24] J. Schönwälder, A. Pras, M. Harvan, J. Schippers, and R. van de Meent, "SNMP Traffic Analysis: Approaches, Tools, and First Results", Integrated Network Management, 2007. IM '07. 10th IFIP/IEEE International Symposium on, 2007, pp. 323-332.

[25] G. Cena, I. C. Bertolotti, and A. Valenzano, "Experimental analysis of latencies in Ethernet communications", Factory Communication Systems, 2006 IEEE International Workshop on, 2006, pp. 303-312.

[26] Flick, J., "IEEE 802.12 Interface MIB", RFC 2020, 1996.

[27] MIB for support of 802.1Qat Stream Reservation Protocol, (SRP) in 802.1Q Bridges, part of IEEE Std 802.1Q-2011.