J Netw Syst Manage
DOI 10.1007/s10922-013-9272-6

Performance Analysis and Evaluation of an Ontology-Based Heterogeneous Multi-tier Network Management System

Lisa Frye · Zhongliang Liang · Liang Cheng

Received: 14 July 2011 / Revised: 19 February 2013 / Accepted: 20 February 2013

Abstract Many of today's networks are Heterogeneous Multi-tier Networks, consisting of wired, Ad-hoc, and wireless sensor nodes. Each node type requires a management system and method to retrieve management data from the nodes. Management of the Heterogeneous Multi-tier Network can be daunting, requiring multiple management systems and manual data aggregation and analysis. A network management system to properly manage such a network was developed to automate some of the data aggregation and analysis. Ontology was utilized as the foundation of the Heterogeneous Multi-tier Network Management System to provide automation of management tasks. Analysis of the developed management system is presented to evaluate its response time and scalability. The analytical model was also utilized to perform bottleneck analysis. Results from the Network Management System deployed in a test environment supported the theoretical results.

Keywords Computer network management · Heterogeneous networks · Multi-tier networks · Analytical model · Performance analysis

L. Frye (✉)
Kutztown University, OM250, P.O. Box 730, Kutztown, PA 19530, USA
e-mail: frye@kutztown.edu

Z. Liang
150 Cambridgepark Dr, Cambridge, MA 02140, USA
e-mail: Zhongliang.liang@hp.com

L. Cheng
Lehigh University, 19 Memorial Drive West, Bethlehem, PA 18015, USA
e-mail: cheng@cse.lehigh.edu

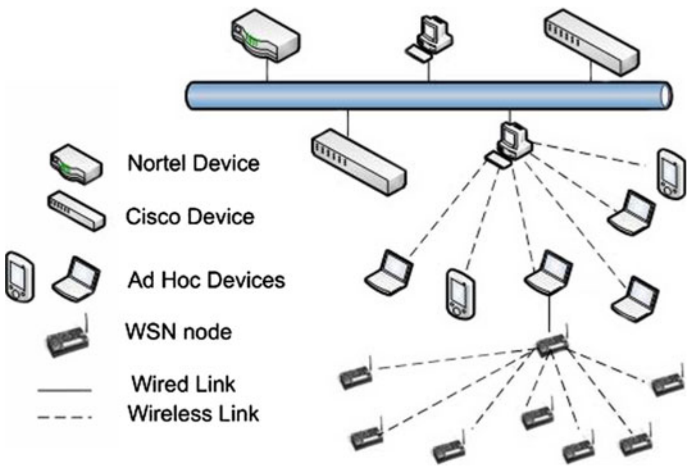

Fig. 1 A heterogeneous multi-tier network (HMN)

1 Introduction

Network management is a critical aspect of all deployed networks. One component of network management that is instrumental in many other management components is topology management. This management function is used to find the currently deployed nodes in the network, their information, and the network layout. Topology management is just one component of most Network Management Systems (NMSs) that are used to manage networks.

The standard network management protocol for wired networks has become Simple Network Management Protocol (SNMP) [1]. This standardization has resulted in a common communication protocol to request management data from nodes. There was hope that all management information for nodes would become common. This information is represented in Management Information Bases (MIBs). Some MIBs are common among all wired nodes, but the proprietary nature of the network nodes developed by different manufacturers has prevented the SNMP MIBs from becoming entirely standard. Each manufacturer has developed their own MIBs to represent data in their devices. This leads to one aspect of heterogeneity that NMSs should address.

Another aspect of heterogeneity in a network is the possible tiers that may be deployed in the network. A network may consist of wired nodes, Ad-hoc nodes, and wireless sensor network nodes. Combing these possible tiers in a network, along with nodes from various manufacturers creates a Heterogeneous Multi-tier Network (HMN) (see Fig. 1).

Response time is an important network characteristic. Network management traffic should not negatively impact response time. When new network management software is installed on the network, performance should be analyzed to ensure it does not adversely affect network performance. If the network management software does adversely affect network performance, the software must be fine-tuned, either by modifying parameters or changing the software implementation configuration. The performance of network management software can be analyzed using a theoretical model.

1.1 Problems of Heterogeneous Multi-tier Network Management

Network Management Systems have been primarily developed to transfer data to a central management station through the use of protocols, such as SNMP for wired nodes, ANMP (Ad-hoc Network Management Protocol) [2] for Ad-hoc nodes, and SNMS [3] (a Sensor Network Management System) or sNMP (sensor Network Management Protocols) [4] for wireless sensor network nodes. When the data arrives at the management station, it is often categorized by the NMS (e.g. all alarms are grouped together, all discovered devices are grouped together, etc.), but the data analysis is typically the responsibility of the network manager (a human).

As the size of the network increases, the volume of network management data that requires manual analysis also increases. Due to the demanding jobs of network managers and the volume of data analysis required, this task is often left undone. If data analysis is not completed, problems in the network may not be discovered until they are a critical issue, possibly causing downtime. Automation, or even partial-automation, of the data analysis required would assist the network manager by reducing the amount of manual data analysis required.

Human data collection and analysis introduces a greater possibility for error. Even trained network managers often miss steps, enter data incorrectly, misinterpret data or results, do tasks differently, etc. These human characteristics often lead to poor data analysis and incorrect results. Applying any level of automation to network management tasks introduces a higher level of consistency in the management of the networks. This leads to improved network performance and reliability.

The heterogeneous nature of HMNs adds yet another layer of complexity to the network management problem. For each network type deployed, wired, Ad-hoc Network (AHN) and Wireless Sensor Network (WSN), there will be a NMS for that sub-network. Even within the same network type, such as wired, devices from different manufacturers may require separate NMSs due to the proprietary nature of each manufacturer's devices. This requires the network manager to consult multiple NMSs just to identify the data that requires a manual analysis, which may be a laborious task. The challenge is to provide a unified view to a network consisting of heterogeneous nodes, described in different ways, using a single NMS. To date this has not been addressed.

1.2 Ontology

Ontology [5] is an area of research that can assist in the development of a single system for easier management of a HMN. The primary benefits of ontology are interoperability and inference. Interoperability offers a method to share domain knowledge, overcoming differences in different terminology for the same concept or mearning for the same term. Interoperability is a goal for a NMS for HMN due to the different terminology used in each tier or network device type.

Utilizing ontology for the development of the Heterogeneous Multi-tier Network Management System (HMNMS) allowed for different terms in each domain (Cisco, Nortel, Ad-hoc nodes, and wireless sensor nodes) to be mapped into one term. The same concept in each domain can be represented by one term in the HMNMS. This allows the network manager to query for a characteristic of any network node using one term instead of a different term for each device type.

1.3 Network Performance

Network performance is a critical aspect of any network. Adding additional traffic to a network, such as that from a NMS, may affect the network performance. The addition of a NMS should provide benefit to the network manager but not become a burden to the operation of the network. To ensure this, it is imperative that a component of the NMS does not become a bottleneck in the network performance.

A NMS must be scalable as the network size increases. As additional nodes are added to the network, the NMS must adapt and not adversely affect network performance. It is also important that the response time of the NMS continue to be acceptable for management of a large network.

A theoretical performance analysis was completed to determine the effect the developed NMS has on network capacity. This analysis examined the scalability of the NMS in regards to the number of supported nodes and its response time. To the best of our knowledge, this is the first time a theoretical performance analysis has been proposed for assessing the performance effects of the management system for a HMN.

1.4 Contribution

The contribution of this work is a description of the ontology subsystem of the state-of-the-art, ontology-based Heterogeneous Multi-tier Network Management System (HMNMS). Another contribution is a theoretical performance analysis of the HMNMS demonstrating implementation decisions that will prevent bottlenecks in the network. The system evaluation analyzes possible bottlenecks in the system as well as system capacity. An experimental validation for the HMNMS was conducted. The developed performance analysis model can be used to conduct a performance analysis of a NMS for HMN.

The remainder of the paper is organized as follows. Section 2 discusses related work. The developed NMS for HMN is described in Sect. 3. Section 4 describes the agent and MIB developed for Ad-hoc network management. The system performance analysis is described in Sect. 5. The implementation of the HMNMS is explained in Sect. 6. Section 7 provides conclusions and future work.

2 Related Work

2.1 Network Domain Ontology

The utilization of ontology in network domains has seen extensive research in recent years. There has not been a NMS developed for a HMN; however, some related

Table 1 Comparison of network ontology work

Work	Wired tier	AHN tier	WSN tier	Goal
Moraes [6]	X			Performance management
Orwat [7]		X		Security management
O'Donoghue [8]			X	Configuration management
OntoSensor [9, 10]			X	Trend discovery in sensor measurements

work has contributed to the network management domain. There has been research in the use of ontology for various aspects of network management. The majority of this work is for one-tier networks, not HMNs, in a specific area of management. The following works highlight some of the applications of using ontology in specific areas of network management. Table 1 summarizes the related work presented here.

Moraes, Sampaio, Monteiro, and Portnoi [6] developed an ontology, MonONTO, that can be used primarily in performance management, including quality of service and monitoring. MonONTO was used with an expert system that could determine application performance based on previous performance. The previous performance is learned from the network and fed to the knowledge base. The knowledge base contained ontology instances about advanced network applications, application users, and network monitoring. These instances were used to determine the most likely network performance in a given situation. This work was for wired networks in determining application performance in a specific network environment.

Another possibility for ontology in network management is the area of security for Mobile Ad-hoc Networks (MANETs). Orwat, Levin, and Irvine [7] developed the MANET Distributed Functions Ontology (MDFO) assist with this task. This system was used to augment the decision-making process for MANET performance and security. One part of MDFO is the translator, which will convert information collected from the network into ontology semantics. The database, populated with MANET device information, is queried when a MANET function is necessary. Relevant attribute values are then sent to the decision making process. This work was only for a homogeneous, single-tier network (an AHN).

Configuration management often requires much human interaction and ontology can assist with this task. Cleary, Danev and O'Donoghue [8] developed an ontology technique to assist with the configuration of wireless networks. The ontologies in this system are used to validate user configurations and suggest possible configurations. The new application interacts with a traditional network management system via an XML representation of the configuration data. The application reads the XML data and uses it to create ontology instances. The reasoner validates the consistency and integrity of user configurations against the knowledge base and suggests possible configurations. The new configuration is converted to XML and fed back to the NMS for deployment to the network. The engine uses rules to create new knowledge from existing knowledge, to create agreements between concepts, and to check consistency. The new approach reduces the amount of human interaction needed for configuration tasks. This ontology was developed for use in configuration of WSNs.

Table 2 Comparison of analytical models

Work	Pros	Cons
Ismail and Zin [12]	Heterogeneous network Measured propagation and transmission delays Actual results verified simulated results	Only a single-tier network Only 1 environment tested (1 university)
Hedayati et al. [13]	Heterogeneous network Studied congestions rates Actual results verified simulated results	Only a single-tier network Only 1 environment tested (1 university)
Nishida [14]	Bottleneck analysis and optimal resource allocation Only a single-tier network	Only evaluated in a SONET application Mathematical model for very specific scenario

OntoSensor [9, 10] is a middle-level ontology designed for a heterogeneous sensor network prototype environment. The WSN's base station includes an OntoSensor ontology. Information about the sensors, including the data acquisition boards, sensing elements, and processors, is included in the repository. The repository responds to Ad-hoc queries to assist in trend discovery in the measurements. This was another work conducted for use with WSNs.

Each cited study utilized ontology in the network management domain. The works all utilized ontology to assist with a specific area of network management for a homogeneous single-tier network. The work presented in this paper utilizes ontology for the network management of a Heterogeneous Multi-tier Network.

2.2 Analytical Models for Networks

There has been research conducted on the development of an analytical model for the performance analysis of heterogeneous networks. Each of the referenced studies used the M/M/1 queuing theory model [11] on a heterogeneous network.

The related works evaluated various performance metrics for single-tier networks. Many networks deployed today are multi-tier, consisting of wired and Ad-hoc nodes. The analytical model presented in this paper fills this research gap by incorporating multiple network tiers. This research required further development for the application to multi-tier networks. It was necessary to incorporate gateways into the model, both for Ad-hoc networks and Wireless Sensor Networks. A comparison of these models is summarized in Table 2.

Ismail and Zin [12] developed a simulation model based on the M/M/1 queuing theory model. The model developed was used to analyze the performance behaviors of a live network. Specifically, it measured the performance of a heterogeneous network over a Wide Area Network (WAN) in a higher education institute. The environment consisted of a Local Area Network (LAN) at the institute's main campus connecting to a branch campus via a WAN. The lessons learned from the live heterogeneous environment were translated into a logical model. The goal was to study the performance of the services offered to the branch campus via the WAN.

The model was used to find the total size of various packet services of all the clients in the heterogeneous environment, Trafik_Heter. The model was:

$$Trafik_{Heter} = \frac{\mu_{Jumlah}}{\frac{J_{LAN}}{v} + \left(\frac{\mu_{Jumlah}}{C_{LAN}}(n+1)\right)} + \frac{\mu_{Jumlah}}{\frac{J_{WAN}}{v} + \left(\frac{\mu_{Jumlah}}{C_{WAN}}(n+1)\right)} \quad (1)$$

where μ_{Jumlah} is the total size of packet services requests by clients, J_{LAN} is the LAN distance, J_{WAN} is the WAN distance, v is the speed of light, C_{LAN} is the LAN bandwidth, C_{WAN} is the WAN bandwidth, and n is the total nodes in the two networks (LAN and WAN). The propagation and transmissions delays were measured for remote data transfers. The simulation model returned results within one second of the actual values. These results verified that the simulation model could be used to estimate data transfer times in a heterogeneous WAN environment. The simulation model was developed following the assumptions that there would be no packet loss, no jitter in delays, and that there would be sufficient network bandwidth.

Hedayati et al. [13] proposed a similar approach, following the M/M/1 queuing theory model, to monitor network traffic. Hedayati, et al developed a model to simulate and monitor the traffic of a heterogeneous LAN environment. Similar to the simulation study of Ismail and Zen, this testing environment utilized a higher education institute's LAN.

The model developed calculated the instantaneous congestion rate, A_0(t), and the stable congestion rate, A_C. The equation developed for the instantaneous congestion rate was

$$A_0(t) = P_1(t) = \frac{1}{(m+1)(1-e^{-(m+1)t})} \quad (2)$$

where $P_1(t)$ is the arrival probability of the queue length for the router's group at time t and m is the service rate. The following equation was developed to calculate the stable congestion rate

$$A_C = P_{C+1} = 1 - \frac{m}{M} \quad (3)$$

where C is the routers' buffers and

$$M = (1 + m + C_v)A_{C-1} - (1 + (C-1)_v(1 - A_{C-1})A_{C-1} + m. \quad (4)$$

The results of the live network tests were similar to the simulation model, confirming that the simulation model can be used to calculate network throughput and congestion rates for a heterogeneous network. Network throughput is the rate for successful delivery of messages on the network, often in some form of bits per second. The congestion rates are the amount of data on a network that causes delays in packet delivery.

Nishida [14] also proposed a model to analyze network performance, specifically of a NMS, based on the Open Systems Interconnect (OSI) architecture. The end-to-end performance model developed was utilized to conduct bottleneck analysis of the NMS. The end-to-end performance was defined as the accumulation of the processing time of all the system components. The end-to-end performance model combines a performance organizational model, a routing model, and a protocol conversion model.

The performance organizational model specified three types of nodes. A split node receives a message and then may multicast or broadcast the message to multiple nodes. A filter node will perform message filtering, meaning the message will only be forwarded if it meets specified criteria. A path node does not change the number of messages output, it simply forward the message to one node. These application nodes are part of the performance organizational model.

The routing model introduces two additional node types, a routing node and a relay node, characterized by the message processing performed at the node. The routing model includes the configuration and characteristics of these two node types. A routing node makes the decision at the network layer, usually by performing a lookup in a routing table for the message destination. A relay node will make the decision of where to send the message at the link layer.

The protocol conversion model includes any node that converts the message from one protocol to another. The protocol conversion can be performed at any layer of the OSI model. For example, for network management, there may be a need to convert from one network management protocol to another one or between two link-layer protocols.

The end-to-end performance model was translated to a mathematical model to evaluate the performance. During the translation, protocol processing in each node was also considered. This processing depends on the OSI layer to which processing is performed in each node. Also incorporated into the mathematical model were a transmission delay and application processing time for each node.

The model was translated to a mathematical model based on queuing theory [15]. The mathematical model was considered for two scenarios. First, the model was used to perform bottleneck analysis. This is useful in a network to improve performance of an application. When the bottleneck node is identified, it can then be fine-tuned. The second scenarios that utilized the mathematical model for end-to-end performance was to determine the optimum resource allocation, specifically the placement of application nodes and routing or relay nodes. The following formula was developed for these scenarios in a SONET management application with two Data Communication Channels (DCC):

$$T_{ne} + T_{dcc1} + T_{dcc2} + T_{os} < T_{max}. \tag{5}$$

T_{ne} represents the management application processing time in the managed node, T_{dcc} is the transmission delay for a DCC, T_{os} is the application processing time by the operating system, and T_{max} is the largest acceptable delay.

3 Heterogeneous Multi-tier Network Management System

An open issue in the management domain is the management of a HMN. A solution to address this issue, focusing on the topology discovery aspect of network management, was proposed by Frye and Cheng [16]. Their solution was the design and development of an innovative method to perform topology discovery for a

HMN. An example network, consisting of wired network devices from two vendors (Nortel and Cisco), Ad-hoc devices, and WSN nodes, is considered here for the sake of discussion.

In order to perform network management functions in this environment, multiple NMSs are required, one for the Nortel devices and one for the Cisco devices. It is necessary to have two NMSs because of the proprietary nature of the NMSs used by manufacturers. Managing an Ad-hoc and a WSN would require additional NMSs, one for each network type. The work in [16] described the solution proposed by Frye and Cheng that would allow one NMS to manage a HMN, specifically perform topology discovery.

To completely realize a solution to this problem, a new Heterogeneous Multi-tier Network Management System (HMNMS) [16] was developed. The HMNMS consists of three components: an Ontology Subsystem, an Ontology Instances Interface, and a Graphical User Interface (GUI). The Ontology Subsystem includes various ontologies, a knowledge base, and a reasoner. The knowledge base contains the ontologies and the instances in the ontologies that were created by the Ontology Instances Interface. The reasoner used is the FaCT++ [17] reasoner and provides the interface between the knowledge base and the GUI. In [16], Frye and Cheng described the overall design for the proposed HMNMS and presented evaluation results for a multi-tiered test network. The work presented in this paper proposes an analytical model for evaluating their solution. Additional experimental results of deployment of the HMNMS are also described here.

3.1 System Ontologies

The related works presented on network domain ontologies demonstrated a variety of ontology systems that were developed. Some of the benefits of ontology are reusability, reliability, sharability, portability, and interoperability [18]. This allows researchers to easily adapt or extend an existing ontology for new requirements. The ontology developed in this research was not adapted from an existing ontology due to the specialized requirements of the previous research and the four specific device types implemented (Nortel, Cisco, Ad-hoc, and wireless sensor devices) in this work. Each of the previous ontologies developed consisted of only one network tier; the ontologies developed required domain knowledge for three different network tiers, as well as two different device types in the wired tier. The domain experts were able to develop the appropriate ontology from defined domain terminology easier from scratch than trying to transform an existing ontology. Future work will investigate the possibility of adapting an existing ontology for use in the HMNMS or modifying the developed ontology so it can be easily integrated into new research. The performance analysis approach established by this research effort will be applicable for analysis of more complicated NMSs built with merged or existing ontologies defining additional network node types.

Several ontologies [19] were developed for the Ontology Subsystem of the HMNMS. The ontologies represent each tier of the HMN, including wired nodes, Ad-hoc nodes, and sensor nodes. The three tiers managed by the HMNMS consist of four different node types with the wired tier consisting of two different nodes, Cisco

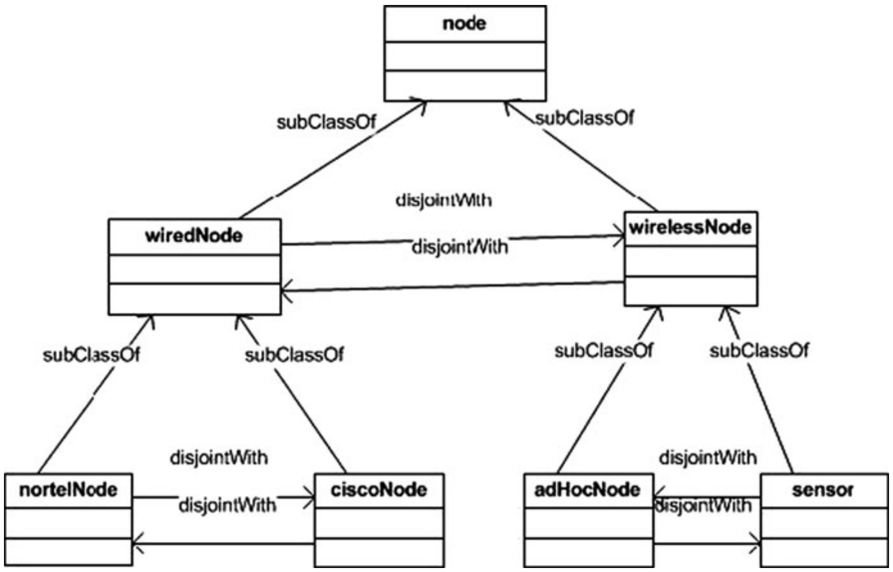

Fig. 2 The ontology class diagram

and Nortel. Despite both being in the same tier, their heterogeneity requires different MIBs, thus also requiring separate ontologies. The classes in the ontologies represent the various device types in the HMN and are the following: *node*, *wired*, *wireless*, *Cisco*, *Nortel*, *Ad-hoc*, and *sensor*. To take advantage of the class inheritance feature, the ontologies were developed in a hierarchical fashion (see Fig. 2). The main class is the *node* class, which contains the properties representative of any device type. Each subclass represents more specific device types.

The importance of the ontology to the HMNMS is the ability of ontology to provide a common vocabulary for the network management domain. This introduces interoperability among the management of the different node types in the HMN, allowing one NMS for the heterogeneous nodes. This interoperability is supported by a mapping ontology that is also part of the Ontology Subsystem. Table 3 lists the terms (properties) in each domain and the corresponding common term that was defined in the mapping ontology.

To illustrate the interoperability feature provided by ontology, consider the address of each node. For the two types of wired nodes (Cisco and Nortel) and Ad-hoc nodes, the address used is the IP address. Although these three nodes all use IP address, each represent this property differently. A sensor node may also use an IP address but many times a single node ID is used, which is the case for this HMN. This requires the ability of the HMNMS to understand that a request for a node's address may be either an IP address or a node ID. The ontology code representing the address property in the ontologies representing the four device types is provided in Fig. 3 (&e represents the node ontology). The mapping ontology is used to instruct the reasoner that the following four properties: (1) Cisco node *sysIPAddr*,

Table 3 Common terms in the ontology

	Cisco domain	Nortel domain	Ad-hoc domain	WSN domain	Common term
System name	*sysName*	*sysName*	*name*	*name*	*name*
System location	*sys Location*	*sys Location*	*location*	(*xcoord ycoord*),	*location*
System description	*sysDesc*	*sysDesc*	*description*	*description*	*description*
Serial number	*chassis Serial Number*	*rcChas Serial Number*	*serial Number*	*serial Number*	*serial Number*
Address	*sysIP Addr*	*rcSys IPAddr*	*ipAddress*	*nodeID*	*address*
Subnet mask	*sysNet Mask*	*sysNet Mask*	*subnet Mask*	N/A	*subnet Mask*
Role (cluster head or member node)	N/A	N/A	*role*	*role*	*role*
Status (alive/active or dead/ inactive)	N/A	N/A	*status*	*status*	*status*
Remaining energy	N/A	N/A	*remaining BatteryLife*	*residual Energy*	*energyLeft*

```
<owl:DatatypeProperty rdf:ID"sysIPAddr">
<rdfs:domain rdf:resource="\&e;node"/>
</owl:DatatypeProperty>
```

(a) Cisco node ontology

```
<owl:DatatypeProperty rdf:ID="rcSysIPAddr">
<rdfs:domain rdf:resource="\&e;node"/>
</owl:DatatypeProperty>
```

(b) Nortel node ontology

```
<owl:DatatypeProperty rdf:ID="ipAddress">
<rdfs:domain rdf:resource="\&e;node"/>
```

(c) Ad-hoc node ontology

```
<owl:DatatypeProperty rdf:ID="nodeID">
<rdfs:domain rdf:resource="\#sensor"/>
</owl:DatatypeProperty>
```

(d) Sensor node ontology

Fig. 3 The ontology code for the node address property

(2) Nortel node *rcSysIPAddr*, (3) Ad-hoc node *ipAddress*, and (4) sensor node *nodeID*, are all the same and can be retrieved through the generic *address* property (see Fig. 4). In Fig. 4, &c represents the Cisco ontology, &d represents the Nortel ontology, &a represents the Ad-hoc ontology, and &b represents the sensor ontology. This allows the GUI to submit a single query for all nodes and their

```
<owl:DatatypeProperty rdf:about="\&c;sysIPAddr">
<rdfs:subPropertyOf rdf:resource="\&e;address"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:about="\&d;rcSysIPAddr">
<rdfs:subPropertyOf rdf:resource="\&e;address"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:about="\&a;ipAddress">
<rdfs:subPropertyOf rdf:resource="\&e;address"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:about="\&b;nodeID">
<rdfs:subPropertyOf rdf:resource="\&e;address"/>
</owl:DatatypeProperty>
```

Fig. 4 The mapping ontology for the address property

address and the result is a node listing with their appropriate address (either an IP address or a node ID).

Another example of the importance of the mapping ontology is the remaining energy in wireless nodes (Ad-hoc and wireless sensor nodes). The remaining energy is a primary concern due to limited energy resources in wireless devices. Without the use of ontology, the network manager would need to query multiple sources, possibly even to the individual node level, to find the energy left in each node. By utilizing the HMNMS, the manager could ask one query to find the remaining energy in each node. The query would consult the mapping ontology to return the results. This would allow the network manager to easily identify nodes that have little energy remaining.

3.2 Ontology Interface

The developed ontologies are loaded into the knowledge base by the reasoner. As the HMNMS runs, the various deployed nodes are also loaded into the knowledge base as instances. This is the task of the Ontology Instances Interface.

To obtain the appropriate data from each node type, a management protocol is used. SNMP is used to obtain the management data for the two types of wired nodes.

ANMP is a protocol developed for management of AHNs, but has not been maintained. To manage the Ad-hoc nodes, a new SNMP agent and MIB were developed, which are explained in Sect. 4. Data is not currently obtained from the sensor nodes due to a lack of standard management protocol. Future work will consist of determining an alternative method to obtain sensor node data.

To create the instances in the knowledge base, the Ontology Instances Interface uses an SNMP Application Programming Interface (API) and requests SNMP data, such as address, name, location, etc. from the deployed devices. Each SNMP response is assigned to the appropriate property for that node type and an instance is added to the knowledge base. All deployed nodes are queried when the HMNMS is initialized. Future work will include the addition of periodical queries for new nodes in the HMN and updated information for existing knowledge base instances representing deployed nodes.

4 Ad-hoc Network Management

In this research, the HMN includes an AHN tier. Although there has been research in the area of management of an AHN, a management protocol or de-facto standard for the management of an AHN does not currently exist. To test the HMN and verify the theoretical performance analysis, it was necessary to retrieve management data from Ad-hoc nodes in the HMN. SNMP was extended to provide management for an AHN [20]. A new MIB was defined and an SNMP agent was developed for this MIB.

The first step towards solving the Ad-hoc network management dilemma involves identifying what services an Ad-hoc network management system should provide. While wireless infrastructure networks and Ad-hoc networks are very similar, the differences are what make Ad-hoc network management important. The key difference between standard wireless architectures and Ad-hoc architectures is that nodes in an Ad-hoc system are inherently mobile. A major purpose of an Ad-hoc network is to enable on-the-fly networking for nodes that are not always in the same location. The mobile nodes in an Ad-hoc network introduce added complexity to power and routing management. Often, nodes that are not stationary will need to run on battery power. Monitoring and conserving battery power is crucial, and is a top priority of Ad-hoc network management.

An additional complexity is also added in Ad-hoc routing. Node movement may cause route changes, signal degradation, and link loss. Although Ad-hoc routing is very important, it was not a focal point of the research done for this paper. Many existing Ad-hoc network management protocols combine routing and network management into a single protocol. While hybrid routing and network management protocols introduce benefits not otherwise available, forcing a relationship between the two is not a perfect solution. The Ad-hoc Network Management Protocol (ANMP) is an example of a hybrid routing and management protocol. The protocol uses network management information to elect special routing nodes known as cluster heads. Doing so allows the routing protocol to choose the nodes that are stronger, and less likely to fail, as key routing entities. At the same time, such a hybrid network management and routing protocol causes an unnecessary coupling. Selecting one single routing protocol to be used for all Ad-hoc networks is a poor decision. Routing protocols may all serve the same purpose, but they all have strengths and weaknesses. In some situations, one protocol may excel, while in another, it may prove inadequate. As a result, Ad-hoc network management systems need to function independent of any routing protocol, while still allowing interaction.

An Ad-hoc network management protocol should enable power information retrieval without imposing a routing protocol. SNMP already provides the entire framework necessary for such a protocol. SNMP is a highly developed and versatile protocol, and it lends itself well to Ad-hoc network management. Along with the newly created Ad-hoc MIB, standard MIBs currently used for wired networks can also be used for Ad-hoc nodes. This includes standard information for nodes, such as name, location, description, serial number, etc. The newly created Ad-hoc MIB and these identified standard MIBs were used to obtain the management data from the Ad-hoc nodes in the HMNMS.

Table 4 Ad-hoc management information

OID	Object name	Possible values
.1	adhocPowerStatus	Acpower(1)
		Dcpower(2)
		Unknown(3)
.2	adhocBatteryPresent	Yes(1)
		No(2)
		Unknown(3)
.3	adhocBatteryLifePercent	Undeterminable(0)
		0–100 %
.4	adhocBatteryLifeSeconds	Undeterminable(0)
		0–?

4.1 Ad-hoc Agent and MIB

The primary purpose of the designed agent was to allow remote queries for power information and to handle unsolicited message generation when exceptional, network impacting events occur. The agent and MIB were developed to respond to queries by the HMNMS to provide data about Ad-hoc nodes. For Ad-hoc nodes, this includes the typical data, such as system name and IP address, as well as information about the remaining energy. The key system power variables that the Ad-hoc agent manages, for the remaining energy, are power source, battery presence, and battery life. Table 4 presents the managed information available for an Ad-hoc node. The prefix for the OID field in the table is 1.3.6.1.4.1.35318.1.1.1.1.

Because not all Linux and Windows platforms provide all of the power information the Ad-hoc agent can support, the values of power status, battery present, and battery life percentage may be unknown in certain situations. Generally speaking, Linux and Windows both provide these values. Linux, however, does not provide support for battery life seconds remaining. When any value cannot be computed, it will be set to an unknown or undeterminable status.

In addition to responding to queries for the above objects, the Ad-hoc agent is also capable of sending unsolicited messages, known as traps, to configured nodes when exceptional events occur. The notifications made available by the Ad-hoc agent are low battery and critical battery. To maintain flexibility, the agent allows low and critical battery thresholds to be configured remotely. The threshold configuration values can be specified in percentages or in seconds. When battery life drops below a given threshold, a notification is generated and further notifications of the same type are temporarily disabled. Notifications are re-armed after the battery level climbs to an SNMP configured value, or to 100 percent battery life remaining. Notifications allow subscribing nodes to receive alerts when a node they subscribed to reaches a low or critical battery life threshold. These alerts enable dynamic network modifications to be used to improve performance, based on power levels.

The management information included in the Ad-hoc MIB provides feedback to the network manager about the battery life of Ad-hoc nodes. This allows the network manager to make better decisions. The feedback would include which Ad-hoc nodes had minimal battery remaining allowing the routing protocol to route around these nodes. For instance, assume that node A had a small amount of battery power remaining, learned from the feedback about node A using the Ad-hoc agent and MIB. If the remaining battery power for specific node was below a specific threshold, then the routing protocol should not use this node for routing. The advanced reasoning would then infer that the routing protocol should not use node A for routing purposes.

4.2 Ad-hoc Agent Interaction

A critical problem with the development and design of the Ad-hoc agent involved interaction between agents. SNMP agents are programmed to answer requests or generate notifications for a specific area of the OID tree. For instance, the sysName object, with OID 1.3.6.1.2.1.1.5, is housed inside of the SNMPv2-MIB MIB created by the Internet Engineering Task Force (IETF). The sysName object is available on almost all SNMP-enabled systems, as a system name is always a useful value. If Cisco releases a new SNMP enabled switch, an agent that responds for sysName requests is necessary, as well as requests for Cisco switch specific information. Because manager nodes prefer to query only one agent for managed information, sysName and proprietary MIB values need to be available within the same agent. The obvious but inefficient solution is to create an entirely new agent for each proprietary device that also responds to all standard queries like that for sysName. But what if the Cisco switch also wants to include SNMP support for an internal circuit they purchase from an outside company? Now the agent must be programmed to respond to those requests as well.

It is easy to see how quickly an agent's requirements can grow unmanageable. To accommodate for this, IETF developed the Agent Extensibility Protocol (AgentX). AgentX defines a system of master agents and sub-agents. Master agents communicate with sub-agents via the AgentX protocol. A master agent provides core SNMP agent functionality, listening on a specific port, accepting incoming SNMP requests, and generating SNMP responses and notifications. Contrary to a standard SNMP agent, a master agent does not have direct access to managed information. Instead, the master agent is responsible for directing incoming requests and outgoing responses or notifications to and from the correct sub-agent. It is the responsibility of the sub-agent to process requests from the master agent and provide responses. It is also the sub-agent's responsibility to generate notifications for exceptional events, and to send these notifications to the master agent.

This hierarchical system allows many sub-agents to run under one port, and also masks the true SNMP protocol from sub-agent implementations. This means that as long as the master agent supports the latest, standard compliant version of SNMP, then all sub-agents inherit this functionality. For this reason, the designed agent for Ad-hoc networks functions as an AgentX sub-agent.

The development of this SNMP agent and MIB for Ad-hoc nodes provided a method to retrieve information from the Ad-hoc nodes to the management station.

This allowed the addition of an AHN tier to the HMN for this research for testing and verification of the HMNMS.

5 System Performance Analysis

There are many issues common to network applications, including NMSs, such as response time, bandwidth capability and connectivity. Two of these issues that were evaluated for the developed NMS are response time and scalability. The network manager must monitor and maintain the network but not impact the user's experience. Therefore, it is imperative to optimize the network's bandwidth, by minimizing the extra traffic introduced by the NMS.

There are many situations that require immediate attention of the network manager. For example, when a fault occurs, particularly a critical fault, it is important for the network manager to immediately receive the alert. This requires the NMS to have a reasonable response time for alarm notification and query responses. The end-to-end performance of the HMNMS was analyzed to evaluate its response time and bottleneck analysis.

To aid in understanding the impact of the HMNMS on the network, a performance analysis was conducted. This analysis will assist the network manager during implementation of the HMNMS. It provides an analytical view of the impact of system design on network capacity. This includes indications of how many network devices can be supported in part of the network. The analytical analysis was conducted for a heterogeneous, two-tier network, consisting of wired and Ad-hoc nodes.

5.1 End-to-End Performance Model

The end-to-end performance of the HMNMS, (T_{NMS}) was defined in [17] as

$$T_{NMS} = T_d + T_{ui} + T_{ont} + T_{int} \tag{6}$$

where T_d is the processing time in the network devices, T_{ui} is the UI processing time, T_{ont} is the processing time of the Ontology Subsystem, and T_{int} is the processing time of the Ontology Instances Interface. The transmission and propagation delays were negligible, as determined by a variety of trial runs. For this reason, they were not considered separately in the end-to-end performance but were combined with the network devices' processing time, which is represented by T_d in Eq. (6). The end-to-end performance was further evaluated by the development of analytical end-to-end delay models for each term. It was assumed that there was no packet loss due to infinite buffers in the analytical model.

The HMNMS has two main systems; the User-Ontology System and the Management-Query System. The User-Ontology System consists of T_{ui} and T_{ont}. This system is utilized when the network manager sends a query to the knowledge base. Invoking this system does not impact the Management-Query System. For instance, if the network manager wants to know the address of all deployed devices (network devices and Ad-hoc nodes), the query is sent to the knowledge base, which is a part of the Ontology Subsystem, and the response is returned to the network manager via a user interface. Since the User-Ontology System is separate from the

Management-Query System, it does not impact the end-to-end performance of the Management-Query System and will not be considered in the queuing analysis for the end-to-end performance. The end-to-end performance becomes:

$$T_{NMS} = T_d + T_{int}. \tag{7}$$

The Management-Query System is the aspect of the HMNMS that was further evaluated using a queuing model. This system was utilized to obtain the management information from the devices. The Ontology Instances Interface sent a query to all the nodes. Each node sent its response back to the Ontology Instances Interface, which then added a new instance or updated an existing instance in the knowledge base. Various implementation tests, which are discussed in Sect. 6 of this paper, revealed that the overhead of the Ontology Instances Interface is negligible.

The NMS was modeled as a packet network with multiple queues. Each device in the network was modeled as an independent queue. For simplicity, packet arrivals were assumed to follow the Poisson process, and all packets (queries and responses) had identical lengths. In addition, it was assumed that all packets had the same priority when processed at each device (this may not be the case in practical systems). In the implementation of this system, each management packet for a certain device generated one response packet. Under the assumption of identical packet length, queries and responses for each device were viewed as an independent packet flow. The average end-to-end delay for each packet flow is the summation of delays in all queues the flow has traversed. Because the system is a request/response application, a flow may traverse a node twice.

The set of all flows in the network is denoted as $\mathbf{P}$. Each flow $p \in \mathbf{P}$ has an arrival rate x_p and may traverse a node i $c_{p,i}$ times, where $c_{p,i} \in \{0, 1, 2\}$ (0 is the case where a flow never traverses node i; 1 represents the case where flow p traverses an end device i and returns; 2 is the case that a flow traverses a node both on its inquiring and responding paths). For any given node i in the network, denote the set of all flows that traverse it as $\mathbf{P}_i$, where $\mathbf{P}_i \subseteq \mathbf{P}$. The total packet arrival rate λ_i for a node i can be written as:

$$\lambda_i = \sum_{p \in \mathbf{P}_i} c_{p,i} x_p. \tag{8}$$

From the conclusion of the Kleinrock Independence Approximation [11] (under this approximation, all queues in the network can be modeled as M/M/1 queue) and the conclusion for M/M/1 queue, the average number of packets in queue i can be expressed as:

$$N_i = \frac{\lambda_i}{\mu_i - \lambda_i}. \tag{9}$$

Here μ_i is the packet processing rate of node i. If the propagation delay is ignored, then Little's theorem [21] can be applied and the average packet delay can be written as:

$$T_i = \frac{N_i}{\lambda_i} = \frac{\frac{\lambda_i}{\mu_i - \lambda_i}}{\lambda_i} = \frac{1}{\mu_i - \lambda_i}. \tag{10}$$

In a network with multi-access, for example, when multiple nodes access the same Ethernet bus or wireless channel, contentions among nodes competing for one channel will also cause a delay. From the conclusion in [22], for a slotted CSMA/CD network, the approximated average packet delay caused by multi-access can be expressed as:

$$W(\lambda) = \frac{\lambda\overline{X^2} + \beta(A + 2\lambda)}{2[1 - \lambda(1 + B\beta)]}. \quad (11)$$

Here λ is the total arrival rate to the bus from nodes, β = τ *C*/*L*, where β is the propagation and detection delay (in packet transmission units) required for all sources to detect an idle channel after a transmission ends, τ is this time in seconds, *C* is the raw channel bit rate, and *L* is the expected number of bits in a data packet. $\overline{X^2}$ is the mean-square of the packet duration, and is calculated by $\overline{X^2} = \sum x^2 Pr.(X = x)$. Under the assumption of identical packet lengths, $\overline{X^2}$ is simply X^2. *A* and *B* are two constants and their values depend on the detailed assumptions of the network (see [23]). This delay *W*(λ) can be added to the delay of any flow going through a gateway with multi-access.

Use I_p to denote the set of all nodes flow *p* has traversed, and J_p to denote the set of all gateways flow *p* has traversed, the total delay of a flow *p* can be expressed generally as:

$$T_{p,total} = \sum_{i \in \mathbf{I}_p} T_i + \sum_{j \in \mathbf{J}_p} W_j(\lambda). \quad (12)$$

The actual expression of the average end-to-end delay depends on the topology and settings of the network. Consider the network in Fig. 1 as an example, the topology can be generalized as in Fig. 5.

The queuing delay caused by the switch/router is:

$$T_{\text{wired}} = \frac{1}{\mu_{\text{wired}} - \lambda_{\text{wired}}} = \frac{1}{\mu_{\text{wired}} - x_{\text{wired}}} \quad (13)$$

where x_{wired} is the data rate of the switch/router query flow. The queuing delay caused by the Ethernet gateway is:

$$T_{\text{eth-gw}} = \frac{1}{\mu_{\text{eth-gw}} - \sum_{\mathbf{P}} 2x_p}. \quad (14)$$

The system is a request/response system, so all flows have traversed the Ethernet gateway twice. Similarly, all flows have traversed the Ontology Instances Interface, the Ontology Subsystem and the User Interface twice, so their expressions for the queuing delay can be easily obtained. This equation gives the expression of the delay in the ideal case; the actual delay model will be slightly different from this depending on the operating system scheduling for forwarding network packets.

In a traditional UNIX system, the scheduler categorizes tasks into five different categories each with a different priority. These five categories are (in decreasing order) [24]:

- Swapper
- Block I/O device control

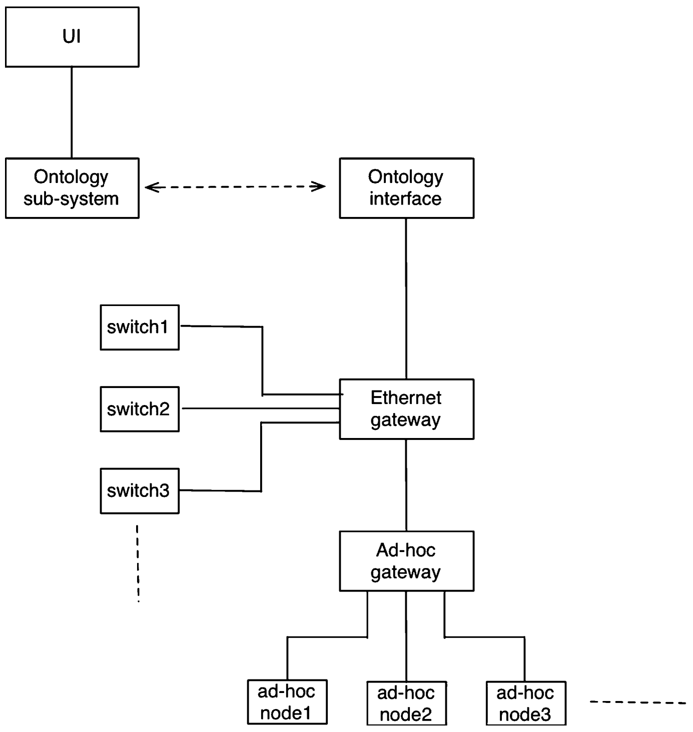

Fig. 5 Generalized Network Topology

- File manipulation
- Character I/O device control
- User processes

This scheduling scheme is intended to provide the highest for I/O operations. Forwarding packets are I/O tasks with higher priorities over user processes, such as a NMS task, which do not involve I/O operations.

The total average end-to-end delay for inquiring a wired node (switch or router) is:

$$\begin{aligned} T_{\text{wiredtotal}} &= T_{\text{wired}} + T'_{\text{eth}-\text{gw}} + T_{\text{int}} + W_{\text{eth}-\text{gw}}(\lambda_{\text{eth}-\text{gw}}) \\ &= \frac{1}{\mu_{\text{wired}} - x_{\text{wired}}} + \frac{1}{\mu'_{\text{eth}-\text{gw}} - \sum_{\mathbf{P}} 2x_p} \\ &\quad + \frac{1}{\mu_{\text{int}} - \sum_{\mathbf{P}} 2x_p} + W_{\text{eth}-\text{gw}}\Big(\sum_{\mathbf{P}} x_p\Big). \end{aligned} \tag{15}$$

$T_{\text{wired}}, T'_{\text{eth}-\text{gw}}, T_{\text{int}}$ are delays caused by wired devices(switch/router), Ethernet gateway, and the ontology interface, respectively. $T'_{\text{eth}-\text{gw}}$ is the delay caused by the Ethernet gateway to forward packets, which is different from $T_{\text{eth-gw}}$, since forwarded packets will send interrupts to the processor and cause additional overhead to the packets being processed in the gateway. An Ad-hoc gateway will behave in the same manner.

Assume there are m_{wired} switches/routers and m_{adhoc} Ad-hoc nodes in the network. The total average end-to-end delay for inquiring an Ad-hoc node is:

$$\begin{aligned} T_{\text{Ad-hoctotal}} &= T_{\text{adhoc}} + T'_{\text{adhoc-gw}} + T'_{\text{eth-gw}} + T_{\text{int}} \\ &\quad + W_{adhoc-gw}(\lambda_{\text{adhoc-gw}}) + W_{eth-gw}(\lambda_{\text{eth-gw}}) \\ &= \frac{1}{\mu_{\text{wired}} - x_{\text{wired}}} + \frac{1}{\mu'_{\text{adhoc-gw}} - 2m_{\text{adhoc}}x_{\text{adhoc}}} \\ &\quad + \frac{1}{\mu'_{\text{eth-gw}} - \sum_{\mathbf{P}} 2x_p} + \frac{1}{\mu_{\text{int}} - \sum_{\mathbf{P}} 2x_p} \\ &\quad + W_{\text{adhoc-gw}}\Big(\sum_{\mathbf{P}} m_{\text{adhoc}}x_{\text{adhoc}}\Big) + W_{\text{eth-gw}}\Big(\sum_{\mathbf{P}} x_p\Big). \end{aligned} \tag{16}$$

In the case illustrated in Fig. 5, $\sum_{\text{P}} 2x_p = 2m_{\text{adhoc}}x_{\text{adhoc}} + 2m_{\text{wired}}x_{\text{wired}}$, since all flows traversed the Ethernet gateway, Ontology Inferences Interface, Ontology Subsystem, and user interface twice.

5.2 System Capacity Analysis

An analysis of the scalability of the Management-Query System was performed. The goal of the theoretical analysis was to provide a qualitative conclusion about the HMNMS. The number of wired or Ad-hoc nodes that can be supported under reasonable query response time was determined. Possible bottleneck nodes were also identified.

From the end-to-end delay expressions in Eqs. (15) and (16), the total delay for a query flow is the summation of delays caused by each device along the path. As a result, the system reaches its capacity when any path device (the two gateways in this case) in the network reaches its capacity. In this case, any query flow traversing this device will have infinite delay.

To keep the system stable, each term on the right hand side of Eq. (16) should not go to infinity. This requires the denominator of each term to be greater than 0. The Ethernet gateway is most likely to be the bottleneck, since it is traversed by all traffic in the network. To keep $T_{eth\text{-}gw}$ finite, the number of wired and Ad-hoc nodes that can be supported must satisfy:

$$\mu_{\text{eth-gw}} - (m_{\text{adhoc}} + m_{\text{wired}})2x_p > 0 \tag{17}$$

which can be written as

$$m_{\text{adhoc}} + m_{\text{wired}} < \frac{\mu_{\text{eth-gw}}}{2x_p}. \tag{18}$$

Similarly, in order to keep the delay caused by the Ad-hoc gateway to be bounded, the number of Ad-hoc nodes should also satisfy:

$$\mu_{\text{adhoc-gw}} - m_{\text{adhoc}}2x_p > 0. \tag{19}$$

This can also be transformed into

$$m_{\text{adhoc}} < \frac{\mu_{\text{adhoc-gw}}}{2x_p}. \tag{20}$$

Equation (18) imposes a constraint on the maximum number of Ad-hoc and wired nodes that can be supported by the Ethernet gateway and Eq. (20) is a constraint on the maximum number of Ad-hoc nodes that can be supported by the Ad-hoc gateway. The capacity of the network is determined by both equations, requiring both equations to be satisfied at the same time. This is a theoretical prediction of the bound. In a practical situation, the constraint may vary due to the operating system scheduling policy and other system dependent parameters (e.g. traffic pattern).

5.3 Analysis Results

The theoretical analysis provided insight to deployment considerations for the HMNMS. Specifically, two deployment parameters were considered in the analysis, the inter-query time and the number of nodes that one Ad-hoc gateway can support.

The inter-query time is the amount of time between queries to the deployed node. This time must be small enough to get accurate information from the nodes for proper management but not too small that the queries inject too much traffic into the network. The theoretical analysis, supported by Eq. (18), shows that the Ethernet gateway will become a bottleneck when the number of queries per second approaches x. To keep the HMNMS from adversely affecting the network performance, each node should not be queried more frequently than every x seconds.

Equation (20) indicates the number of Ad-hoc nodes that can be supported by one Ad-hoc gateway. When that number is reached, an additional Ad-hoc gateway should be added to the network. The analysis indicates that an additional Ad-hoc gateway should be added for every x number of Ad-hoc nodes. These conclusions show that the gateway nodes will be the bottlenecks in the network.

6 Implementation

6.1 Test Network

The test network for this deployment of the HMNMS consisted of two of the three possible tiers, wired and Ad-hoc. Sensor nodes were not part of this test network because of the lack of standard management protocol. The wired tier consisted of both Cisco and Nortel nodes.

The wired nodes were previously configured with SNMP data, which was part of the standard installation of the network devices. This included information such as the IP address, network mask, name, location, etc. There was no additional configuration required for their deployment to the test network.

The Ad-hoc nodes were Linux laptops with an Ad-hoc routing protocol installed. This was required for the laptops to be deployed in an Ad-hoc network. The new

SNMP Ad-hoc agent and MIB required some additional installation and set-up. Because this agent was created as an AgentX subagent, it requires an AgentX master agent in order to function. The AgentX master agent used in the test network was the Net-SNMP distribution [25]. The new Ad-hoc agent was also installed and configured to start at system startup. When the laptops were booted, the AgentX master agent and new Ad-hoc agent were started and ready to answer SNMP requests.

The HMNMS was run on a management station that was part of the test network. The management station was able to access the wired network, the Ad-hoc network, and the developed ontologies. The management station was able to obtain the appropriate data for all deployed devices. A simple query was sent from the management station to the knowledge base requesting the address and description of all deployed nodes in the test HMN. These were properly returned because of the interoperability provided by incorporation of ontology in the HMNMS. A portion of the query results is shown in Fig. 6.

6.2 Test Network Results

The test network was deployed to verify the theoretical analysis with empirical results. The focus of this analysis was the Ad-hoc tier of the network since the Ad-hoc gateway was a focus of the analysis. The management station was directly connected to the Ad-hoc gateway via a network cable in order to eliminate any fluctuations in the Ethernet gateway performance. The Ad-hoc gateway was connected to two laptops, acting as Ad-hoc nodes, with 802.11 g wireless network cards. The management station sent periodic SNMP query packets to the Ad-hoc gateway and two Ad-hoc nodes, since in practice the management station usually sends queries periodically, rather than randomly, to devices.

Figure 7 shows the delay caused by the Ad-hoc gateway and one Ad-hoc node from the theoretical analysis of the test network. The delays caused by multi-access were omitted for simplicity. The processing rate for all devices was assumed to be

```
--> address: 192.168.2.210
-->sysDesc: Cisco Systems Catalyst 1900,V9.00.06

--> address: 192.168.2.150
--> sysDesc: BayStack 450-24T HW:RevL FW:V1.36
SW:v1.3.1.2

--> address: 10.0.0.1
--> description: Linux misty 2.6.28-11-generic
\#42-Ubuntu SMP Fri Apr 17 01:57:59 UTC 2009 i686

--> address: 10.0.0.2
--> description: Linux lucky 2.6.28-11-generic
\#42-Ubuntu SMP Fri Apr 17 01:57:59 UTC 2009 i686
```

Fig. 6 Query results from HMNMS for test network

identical and equal to 1/20 packet per millisecond. The x-axis is the normalized query arrival rate (actual arrival rate divided by processing rate of the device). The y-axis shows the delays caused by the Ad-hoc gateway and the Ad-hoc node respectively.

It can be seen from Fig. 7 that the delay caused by the Ad-hoc gateway increases faster than that caused by the Ad-hoc node. The experimental results below were different from the theoretical results mainly because the query rate in the experiment is constant, which is more realistic. Another reason is that the operating system assigns different priorities to local packets and forwarded packets. These will be explained later. It can be concluded that the Ad-hoc gateway will be the bottleneck of the network, as the delay caused by gateway has a higher order of growth than the Ad-hoc node. When network traffic load is high, the delay caused by the gateway will be significantly higher than the Ad-hoc node. This theoretical conclusion conforms with the experimental results below.

Test results for the test Ad-hoc network are shown in Fig. 8 to 10. The figures illustrate the management query delay for each of the Ad-hoc nodes in the network for a variety of inter-query times. The inter-query time (labeled in the figures' key as inter-arr) is the amount of time between queries sent by the management station. A smaller inter-query time indicates more query packets being sent, which leads to additional network traffic. The inter-query time must be small enough to obtain accurate management information from the nodes but not too small that the management of the network causes too much additional network traffic.

The x-axis in the figures is the query packet index (or packet number), ordered in time sequence, for each device in the network. A total of 100 management queries were sent to each network node. The y-axis is the delay for the management query

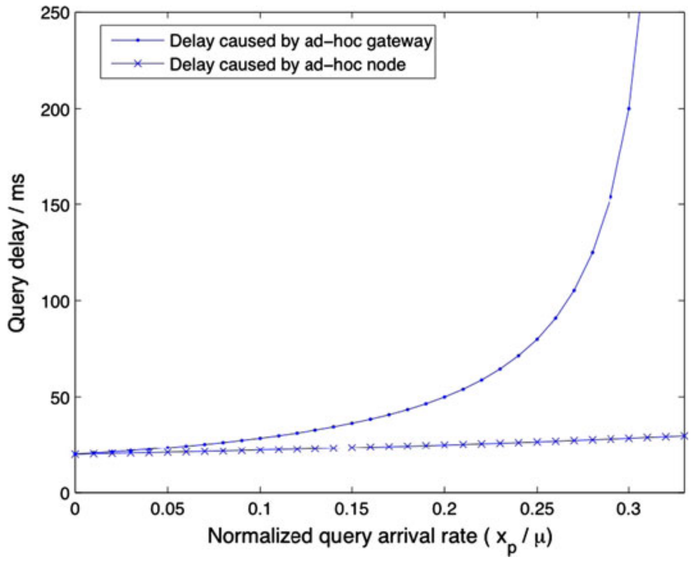

Fig. 7 Delay caused by Ad-hoc gateway and node from theoretical analysis

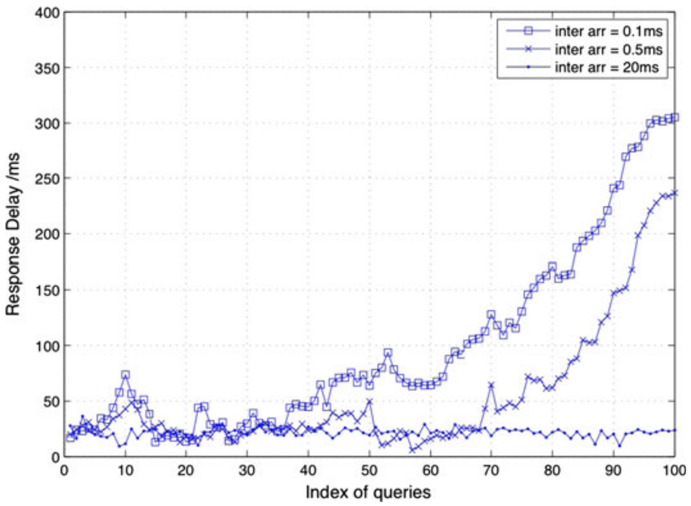

Fig. 8 Query delay at Ad-hoc gateway

and response. Figure 8 shows the query delay for the Ad-hoc gateway when the query packet inter-query time equals 20ms, 0.5ms and 0.1ms respectively. The delay grows as the packet index grows, and decreases as the inter-query time grows. Figures 9 and 10 are query delays at Ad-hoc node 1 and 2, respectively.

The results in the test network are analogous to the theoretical results. In comparing the results of the Ad-hoc gateway in both the theoretical (Fig. 7) and actual results (Fig. 8), it can be seen that they follow a similar growth rate as the number of packets increase. As predicted in the theoretical results, the Ad-hoc gateway (Fig. 8) has a higher rate of growth than the non-gateway nodes (Fig. 9 and 10). It can be concluded that the actual results follow from the theoretical results.

A practical network has a capacity for the number of packets in the network. The first observation is the buffering effect: in the beginning, there were fewer packets in the network, consequently the delay was smaller for packets. As more packets were injected into the network, the buffering effect occurred and the queuing delay increased causing the end-to-end packet delay to increase.

The second observation is that queries for the Ad-hoc gateway had a higher average delay than those for the Ad-hoc nodes. The reason is that each time the network card receives an Ethernet frame which matches the local MAC address or is a link-layer broadcast, it issues an interrupt to the operating system [26]. The network driver for the network card then handles the interrupt, allowing the gateway to immediately forward any packets not destined for the gateway. Forwarding packets are I/O tasks with higher priorities over local MIB checking tasks, which do not involve I/O operations. In addition, interrupts will frequently suspend the current task and seize the processor. These aspects all add to the larger average round trip delays for the SNMP packets destined for the Ad-hoc gateway.

The conclusion can also be drawn that the Ad-hoc gateway will be the bottleneck of the network since as traffic load increases all packets dedicated for the Ad-hoc

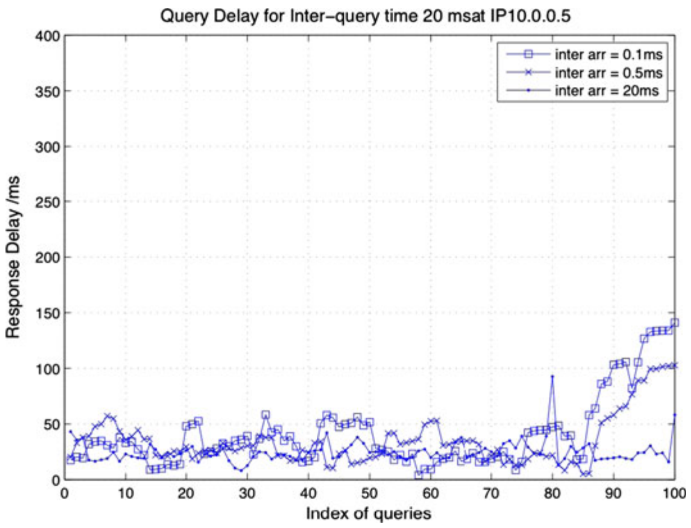

Fig. 9 Query delay at Ad-hoc node 1

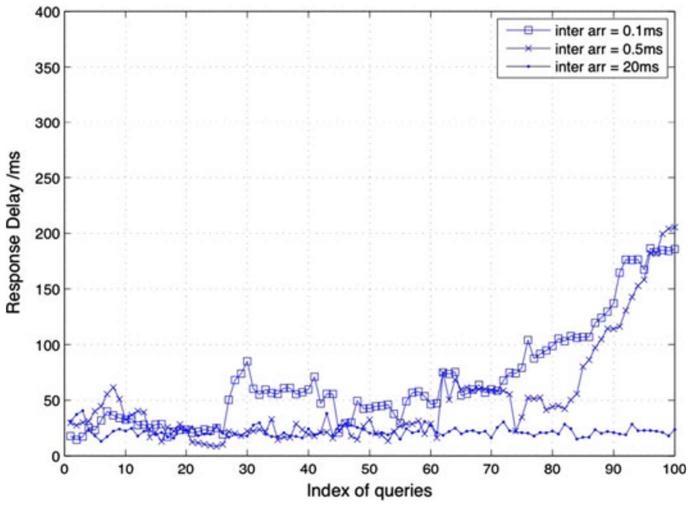

Fig. 10 Query delay at Ad-hoc node 2

network will have to be processed at the gateway and thus increases the work load of the gateway. The delay for packets querying the network gateway will increase faster than other packets and eventually cause packet loss. This conclusion is the same as the conclusion drawn from the theoretical analysis and is verified by the query delay for the Ad-hoc gateway shown in Fig. 8.

6.3 System Deployment

In consideration of the contributions of this work, the system was analyzed in a test environment and deployed in a live environment to obtain a quantitative measurement of the performance and deployment of this solution. The live environments were a corporate network consisting of Cisco devices and a university network consisting of Nortel devices. While these two networks were homogeneous, these deployments provided an opportunity to test the deployment of the HMNMS.

The HMNMS used existing management protocols to obtain node information. This contributed to the ease of deployment for the HMNMS. Deployed network nodes would most likely already support the standard management protocols by default. If a node did not support the standard management protocol, it was easily installed or enabled. Deployed nodes required no additional software to be installed to work with the HMNMS.

The HMNMS system, consisting of the three components, was installed on a single management station. This required the installation of the FaCT++ reasoner and the Ontology Instances Interface. The reasoner required access to the ontologies developed; they could be copied onto the management station or onto a web server that was accessible to the management station.

To demonstrate the ability to deploy the HMNMS, tests were conducted in two live environments, a large manufacturing corporation and a mid-sized university. Each network consisted of thousands of nodes with the management being conducted on wired network devices. In both circumstances, the management station was a pre-configured laptop. Both networks were being monitored by the HMNMS within a matter of minutes. The networks contained no Ad-hoc devices so results were not evaluated with the analytical model. These implementations were conducted to evaluate the ease of deployment of the HMNMS in a live environment. Within minutes topology discovery information was viewed by the network manager via the HMNMS UI.

6.4 Results of System Deployment

The results of the deployment of the HMNMS in a live university environment showed that the HMNMS incurred negligible, acceptable overhead. The results, which are shown in Fig. 11, demonstrated the majority of the response time was the request and response for the SNMP data. This overhead is not unique to the HMNMS as it exists in any NMS that utilizes SNMP. To display network information, any NMS will have to query the devices for the SNMP data. The results of the second live deployment, in a corporate network, were similar to the university network results.

The overhead for the insertions of the instances into the knowledge base was minimal. This time was relatively constant as the number of devices increases. The query response time to obtain the network topology was also minimal. This time increased as the number of devices increased; however, it was minimal compared to the time required to perform a manual analysis of the data. This overhead was

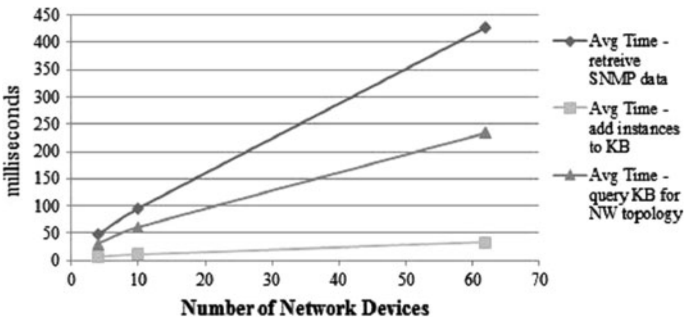

Fig. 11 Results from the HMNMS in live university network

acceptable provided the benefits obtained from utilizing the HMNMS, which automated the data analysis.

7 Conclusions

The topology management of a network is a crucial component of network management. This is a complex task for the network manager due to the amount of human interaction required. Introducing additional node types and tiers into the network further complicates the task. The complexity of management can be reduced by using ontology, which introduces data semantics into the system. A HMNMS using ontology was developed to provide management for a HMN.

A performance analysis of the start-of-the-art ontology-based HMNMS is important for its continual design and deployment. A theoretical analysis was performed to provide an analytical view of the network performance when the HMNMS was added to the network. The analysis concluded that the gateways in the network were the bottlenecks of query flow. A test Ad-hoc network was set up to conduct experiments for query delays. The theoretical conclusion was supported by the experimental results. The HMNMS was deployed in two live networks to verify ease of deployment and use.

Future work for the ontologies developed will be to evaluate the possibility of adapting an existing ontology for use in the HMNMS. If an existing ontology is not identified, then the developed ontologies will be refined, maintained, and extended to make the HMNMS more feasible in a live network. The primary network management focus of this research was topology management. The HMNMS will be extended to include other areas of network management.

The analytical model will be refined and extended. Additional performance evaluations utilizing the analytical model will be considered. This will include dynamic parameters for the model, such as packet size and the number of response packets generated. Consideration of packet loss will also be incorporated and analyzed. Utilizing the analytical model for traffic modeling will be pursued. This will require the introduction of a large amount of complex traffic into the model instead of the constant traffic rate that was used in this work.

Future work will comprise additional tests to substantiate the theoretical results. Conclusions have been drawn about the validity of the analytical model; however, additional tests are required to fully authenticate the analytical model developed. Future work will also include the consideration of more implementation options in the analytical model. These will include routing and mobility in the Ad-hoc network, which are common in most Ad-hoc networks.Additional tests will be conducted in live networks, consisting of both wired and Ad-hoc tiers. Upon completion of these tests, the implementation details and results will be discussed with the network managers to further improve the HMNMS.

Also to be considered in future work is the addition of Wireless Sensor Network (WSN) nodes to the test HMN. This will require a decision on a reliable way to obtain appropriate network information from these sensor nodes. When the WSN tier is added to the HMN, the analytical model will be updated to provide analysis of this tier of the HMN.

Acknowledgments This research was partially supported by the Pennsylvania Department of Community and Economic Development via the Pennsylvania Infrastructure Technology Alliance.

References

1. Case, J., Fedor, M., Schoffstall, M., Davin, J.: (Online). A simple network management protocol (SNMP), RFC 1157. Available at: http://www.ietf.org/rfc/rfc1157.txt (May 1990)
2. Chen, W., Jain, N., Singh, S.: ANMP: Ad-hoc network management protocol. IEEE J. Sel. Areas Commun. **17**, 1506-1531 (1999)
3. Tolle, G., Culler, D.: Design of an application-cooperative management system for wireless sensor networks. In: Proceedubgs of the Second European workshop on wireless sensor networks, pp. 121–132 (2005)
4. Deb, B., Bhatnagar, S., Nath, B.: (Online). Wireless sensor networks management (2005). Available at: http://www.research.rutgers.edu/bdeb/sensor_networks.html
5. Uschold, M., Gruninger, M.: Ontologies: Principles, methods and applications. Knowl. Eng. Rev. **11**(2), 93–136 (1996)
6. Moraes, P.S., Sampaio, L.N., Monteiro, J.A.S., Portnoi, M.: MonONTO: a domain ontology for network monitoring and recommendation for advanced internet applications users. In: Proceedings of the IEEE network operations and management symposium workshops (2008)
7. Orwat, M.E., Levin, T.E., Irvine, C.E.: An ontological approach to secure MANET management. In: Proceedings of the third international conference on availability, reliability and security, pp. 787–794 (2008)
8. Cleary, D., Danev, B., O'Donoghue, D.: Using ontologies to simplify wireless network configuration. In: Proceedings of formal ontologies meet Industry (2005)
9. Goodwin, C., Russomanno, : An ontology- based sensor network prototype rnvironment. IEEE Fifth international conference on information processing in sensor networks (2006)
10. Russomanno, D.J., Kothari, C.R., Thomas, O.A.: Building a sensor ontology: a practical approach leveraging ISO and OGC models. The 2005 international conference on artificial intelligence (2005)
11. Kleinrock, L.: Queueing Systems. Volume 1: Theory. Wiley, London (1975)
12. Ismail, M.N., Zin, A.M.: Development of simulation model in heterogeneous network environment: comparing the accuracy of simulation model for data transfers measurement over wide area network. Int. J. Multimed. Ubiquitous Eng. **5**(4), 43–58 (2010)
13. Hedayati, M., Kamali, S.H., Izadi, A.S.: The monitoring of the network traffic based on queuing theory and simulation in heterogeneous network environment. In: Proceedings of the 2009 international conference on information and multimedia technology (ICIMT '09), pp. 396–402 (2009)

14. Nishida, T.: End-to-end performance modeling for distributed network management systems. In: Proceedings of the tenth annual joint conference of the IEEE computer and communications societies (INFOCOM), pp. 121–129 (1991)
15. Giambene, G.: Queuing Theory and Telecommunications: Networks and Applications. Springer, Berlin (2010)
16. Frye, L., Cheng, L.: A network management system for a heterogeneous multi-tier network. In: Proceedings of IEEE GLOBECOM 2010 (global communications conference, exhibition and industry forum) (2010)
17. Google Code Fact++: [Online]. Available at: http://code.google.com/p/factplusplus/ (2010)
18. Spyns, P., Meersman, R., Jarrar, M.: Data modeling versus ontology engineering. ACM SIGMOD Rec. **31**(4), 12–17 (2002)
19. Frye's Network Management Research [Online]. Available at: http://faculty.kutztown.edu/frye/res/index.html
20. Frye, L., Fox, K.: Extending SNMP for ad hoc network management. In: 27th Annual conference of the pennsylvania computer and information science educators (PACISE) (PACISE '12). Millersville, 30–31 March (2012)
21. Gross, D.: Fundamentals of Queueing Theory. Wiley, London (2009)
22. Bertsekas, D., Gallager, R.: Data Networks, Chap. 4, p 318. Prentice Hall, Upper Saddle River (1987)
23. Lam, S.S.: A carrier sense multiple access protocol for local networks. Comput. Netw. (1976) **4**(1), 21–32, Feb (1980)
24. Stallings, W.: Operating Systems: Internals and Design Principles, 3rd edn. Prentice Hall Engineering/Science/Mathematics, Upper Saddle River (1997)
25. Net-SNMP (Online). Available at: http://net-snmp.sourceforge.net/
26. Barry, P., Crowley, P.: Modern Embedded Computing: Designing Connected Pervasive, Media-Rich Systems, Chap. 7, p. 215. Morgan Kauffman, San Francisco (2012)

Author Biographies

Lisa Frye is an Associate Professor at Kutztown University of Pennsylvania. Her research interests include networks, network security and ontology. She received a Ph.D. in Computer Science from Lehigh University. Her previous industry experience includes Network Server Manager at Kutztown University, Systems Engineer at Electronic Data Systems and System Support Analyst at Unisys Corporation.

Zhongliang Liang is a Software Engineer at Hewlett Packard. He received his Master of Applied Science degree from McMaster University (Hamilton, Canada) in 2010 and a Master of Science degree from Lehigh University (Bethlehem, PA, USA) in 2012.

Liang Cheng is an associate professor of computer science and engineering at Lehigh University. He has been the principal investigator (PI) and a co-PI of fifteen projects funded by the U.S. NSF, DARPA, DOE and other sponsors. He has published in areas of sensing systems, heterogeneous networks, and distributed systems.