A Network Management System Based on Ontology and Slow Intelligence System

F. Colace, M. De Santo
DIEII – Università degli Studi di Salerno, Via Ponte Don Melillo, 1, 84084
Fisciano (Sa) Italy
e-mail: {*fcolace, desanto}@ unisa.it*

Abstract

The last decade has witnessed an intense spread of computer networks that has been further accelerated with the introduction of wireless networks: this growth has increased significantly the problems of network management. Especially in small companies the management of such networks is often complex and faults have significant impacts on their businesses. A possible solution is the adoption of the Simple Network Management Network administrators can manage network performance, find and solve network problems, and plan for network growth by the use of the SNMP. Over the past years much efforts has been given to make more effective the Simple Network Management Protocol and new approaches has been developed. In particular a promising approach involves the use of Ontology. The ontology based network management has recently evolved from a theoretical proposal to a more mature technology and this is the starting point of this paper where a novel approach to the network management based on the use of the Slow Intelligence System methodologies and Ontology based techniques is proposed. The Slow Intelligence System is a general-purpose system characterized by being able to improve performance over time through a process involving various phases as enumeration, propagation, adaptation, elimination and concentration. Therefore, the proposed approach aims to develop a system able to acquire, according to the Simple Network Management Protocol, information from the various hosts that are in the managed networks and apply actions in order to solve problems. To check the feasibility of this approach and its performance an experimental campaign in a real scenario has been designed and the first experimental results in a real scenario are showed.

Keywords: *Ontology – Network Management – Slow Intelligence System*

1. Introduction

Networks and distributed computing systems are becoming increasingly important. This rash spread, however, resulted in increased difficulty in configuring and managing computer networks. The concept of network management is quite articulated. It involves activities such as the identification and management of various devices, monitoring their performance and much more. So efficient and intelligent configuration management techniques are to reach an automatic or semi-automatic configuration for these devices [1]. A solution for this problem can be the adoption of the Simple Network Management Protocol (SNMP). The SNMP is not only a protocol but can be considered as a general framework for the network management. This framework provide the following components [18][19]:

- network management objects known as MIB objects. In fact in the framework management information is represented as a collection of managed objects that together form a virtual information store, known as the Management Information Base (MIB)
- a data definition language known as SMI (Structure of Management Information) that defines the data types, an object model and rules for writing and revising management information
- a protocol SNMP for conveying information and commands between a managing entity and an agent executing on behalf of that entity within a managed network device
- security and administration capabilities

In literature ontology is considered a good way for supporting the network management and many papers deal with ontology based methodologies for network management [2]. Ontology based network management, in fact, has recently evolved from a theoretical proposal to a more mature technology. The main reason of this is in the significant role that ontology plays in the information model harmonization [3]. In the network management there are many management information models and a harmonization is needed. This harmonization can not be made just as a syntactic translation but a semantic translation is needed. Many papers deal with this approach: in [4] ontologies are used to provide this harmonized information model with an approach to map and merge different information model definitions, taking into account their semantics in a common ontology based model. A similar approach is in [5] where the management information is merged from several sources. Other works in recent years have also included related proposals about using ontologies for different aspects of network management. For instance [6] proposed using ontologies for the integration of network management policies. That previously showed is the starting point of this paper. In fact it introduces a novel approach to the network management based on the use of the Slow Intelligence System methodologies [7] and ontology. The Slow Intelligence System is a general-purpose systems characterized by being able to improve performance over time through a process involving enumeration, propagation, adaptation, elimination and concentration phases. This approach works at its best when adopts the ontology for the representation of its knowledge base. So the proposed approach aims to develop a system able to acquire information from the various devices that are in the managed networks and apply solutions in order to solve problems. In particular the proposed system can handle multiple networks and adopt solutions that have proved successful in some other context. By the use of ontologies the system will be able to choose the right action to take when some hosts send alerts. The use of the Slow Intelligence System approach will allow the system to automatically infer the actions to take. In order to test the effectiveness of the proposed approach, it has been applied to various LANs that adopt the SNMP protocol for the network management. This paper is organized as follows. The next section introduces the proposed approach and describes the Slow Intelligence System and the ontology approach. The third section gives more details on the proposed approach and describes its operative workflow while the fourth section shows the experimental results. Finally some conclusions are provided.

2. A Network Management Framework Based on the SIS Approach

In this section we will describe the architecture of the proposed Network Management tool through the description of its main components. In particular it will be showed how the framework works according to the Slow Intelligence System approach and by the use of the ontological formalism for the management of the knowledge base. The architecture of the proposed system is described in figure 1.

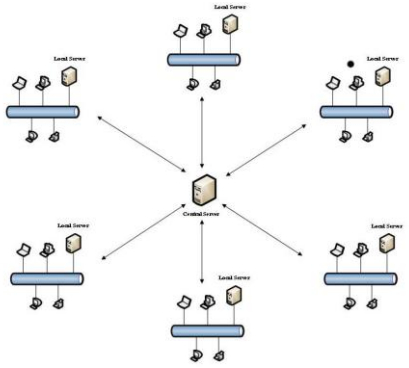

Figure 1 A Possible Working Scenario

Each server manages a computer network and works according to the principles of the Slow Intelligence System.

2.1 The Slow Intelligence System

A Slow Intelligence System is a general-purpose system characterized by being able to improve performance over time [7]. A Slow Intelligence System continuously learns, searches for new solutions and propagates and shares its experience with other peers. It differs from expert systems in that the learning is implicit and not always obvious. A Slow Intelligence System seems to be a slow learner because it analyzes the environmental changes and absorbs that into its knowledge base while maintaining synergy with the environment. Usually a Slow Intelligence System solves problems by trying different solutions, is context-aware to adapt to different situations and to propagate knowledge and may not perform well in the short run but continuously learns to improve its performance over time [7]. A Slow Intelligence System workflow is typically composed by the following phases:

- Enumeration: In problem solving, different solutions are enumerated until the appropriate solution or solutions can be found.

- Propagation: The system is aware of its environment and constantly exchanges information with the environment. Through this constant information exchange, one SIS may propagate information to other (logically or physically adjacent) SISs.

- Adaptation: Solutions are enumerated and adapted to the environment. Sometimes adapted solutions are mutations that transcend enumerated solutions of the past.

- Elimination: Unsuitable solutions are eliminated, so that only suitable solutions are further considered.

- Concentration: Among the suitable solutions left, resources are further concentrated to only one (or at most a few) of the suitable solutions.

The sixth one, on the other hand, is rather unique for a Slow Intelligence System:

- Slow decision cycle(s) to complement quick decision cycle(s): SIS possesses at least two decision cycles. The first one, defined as the quick decision cycle, provides an instantaneous response to the environment. The second one, defined as the slow decision cycle, tries to follow the gradual changes in the environment and analyze the information acquired by experts and past experiences. The two decision cycles enable the SIS to both cope with the environment and meet long-term goals. Sophisticated SIS may possess multiple slow decision cycles and multiple quick decision cycles. Most importantly, actions of slow decision cycle(s) may override actions of quick decision cycle(s), resulting in poorer performance in the short run but better performance in the long run. The structure of a Slow intelligence System by the introduction of the basic building block and advanced building block.

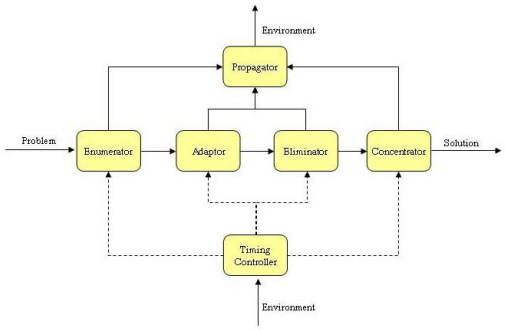

Figure 2 A Basic Building Block BBB

Problem and solution are both functions of time, thus we can represent the time function for problem as x(t)problem, and the time function for solution as y(t)solution. The timing controller is also a time function timing-control(t). For the two-decision-cycle SIS, the basic building block BBB can be expressed as follows:

```
if timing-control(t) == 'slow'
then /* timing-control(t) is 'slow' */
       y(t)solution = gconcentrate (geliminate (gadapt (genumerate(x(t)problem))))
else /* timing-control(t) is not 'slow' */
       y(t)solution = fconcentrate (feliminate (fadapt (fenumerate(x(t)problem))))
```

where $g_{enumerate}$, g_{adapt}, $g_{eliminate}$, $g_{concentrate}$ are the transform functions for enumeration, adaptation, elimination and concentration respectively during slow decision cycles, and $f_{enumerate}$, f_{adapt}, $f_{eliminate}$, $f_{concentrate}$ are the transform functions for enumeration, adaptation, elimination and concentration respectively during quick decision cycles. An Advanced Building Block can be a stand-alone system as shown in Figure 3. The major difference between an ABB and a BBB is the inclusion of a knowledge base, further improving the SIS's problem solving abilities.

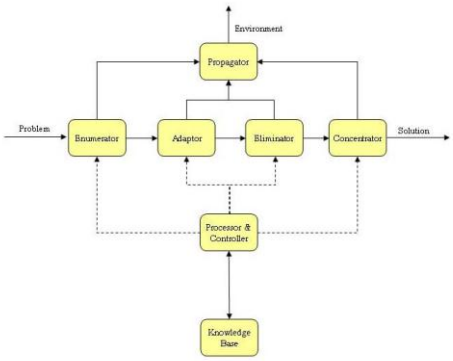

Figure 3 The Advanced Building Block ABB

As showed in figure 3 the advanced building block works using a Knowledge Domain that contains all the information that the ABB needs in order to manage the various problems. Each ABB works with a well defined Knowledge Domain that can change through the interaction with the other peers. An effective way for the representation of the Knowledge Domain is the adoption of the ontology formalism. In the next paragraph more details on the Ontology formalism will be given.

2.2 The Role of the Ontology in a Slow Intelligence System

The definition of ontology is still a challenging task [8]. The term 'ontology' has its origin in the Greek word 'ontos', which means 'being'. So in this sense ontology could be defined as a branch of philosophy dealing with the order and structure of reality. In the 1970s ontology came to be of interest in the computer science field. In particular the artificial intelligence community started to use the concept in order to create a domain of knowledge and establish formal relationships among the items of knowledge in that domain for performing some processes of automated reasoning, especially as a means for establishing explicit formal vocabulary to be shared among applications. The term 'ontology' was first used in the computer science field by Gruber who used the term to refer to an explicit specification of a conceptualization [9]. The use of this term is rapidly growing due to the significant role it plays in information systems, semantic web and knowledge-based systems, where the term 'ontology' refers to "the representation of meaning of terms in vocabularies and the relationships between those terms" [10]. Also this kind of definition is still satisfactory for each field where ontology can be applied and so perhaps a good practical definition would be this: "an ontology is a method of representing items of knowledge (ideas, facts, things) in a way that defines the relationships and classification of concepts within a specified domain of knowledge" [8]. Following this point of view, ontologies are "content theories", since their principal contribution lies in identifying specific classes of objects and the relations that exist in some knowledge domains [11][12]. Ontologies are usually classified into lightweight and heavyweight ontologies [12]. Lightweight ontologies include concepts, simple relationships among concepts (such as specialization is_a) and properties that describe concepts. Heavyweight ontologies add axioms and constraints to lightweight ontologies. Axioms and constraints clarify the intended meaning of the terms gathered in the ontology. Heavyweight and lightweight ontologies can be modelled by the use of different knowledge modelling techniques and they can be implemented in various kinds of languages which are usually divided in two groups: classical and ontology mark-up language [13]. The

ontology mark-up languages, mainly used in the context of semantic web and of which the most important is OWL [10], have their own syntax, their own expressiveness, different knowledge representation paradigms and their own reasoning capabilities provided by different inference engines [13]. It is important to underline how database community as well as the object oriented design community build models using concept, relations and properties but they usually impose less semantic constraints. Ontologies are typically not static entities and so in recent years ontology evolution processes have drawn considerable attention of the researchers. The ontology evolution can be considered as the "timely adaptation of an ontology to the arisen changes and the consistent management of these changes" [14]. This definition suggests that a successful evolution can only be achieved by having both "adaptation" and "change management". In the literature there aren't many approaches capable of handling these two tasks within one framework. Another core aspect of ontology evolution is how to guarantee the consistency of the ontology and the dependent applications [14][15]. In this sense many papers are introducing approaches and methodologies for the ontology evolution management and its change requirement description. In particular frameworks for the management of atomic and complex changes have been introduced [16]. A particular aspect of ontology application is in the analysis and comparison of particular ontologies that could be used to derive information beyond operational data. In this case ontologies could be for management support. This is a very interesting application field but some critical problems remain to be solved. In fact usually the lightweight ontology furnishes a very simple and generic representation of a context and so is not able to well manage a system or supporting users in the interactions with it. In particular if it represents the services and components of a system probably its computational and functional optimization could be not reached. However heavyweight ontology could be very difficult to define and includes some aspects that are not all the time useful and the risk of wasting system's resources to maintain heavyweight ontology is quite high. With the aim to avoid the above described problems, in this paper a lightweight plus ontology is proposed, which can be defined as $O = \{C, A, R_H, R\}$ where C is the concept set, A is the concept attributes set, R_H expresses the hierarchical relationships among the concepts and R is the set of non-hierarchical relations. By the introduction of the non-hierarchical relations, a lightweight plus ontology is more complex and semantically richer than the lightweight ontology, but is not as complex as heavyweight ontology because there are no axioms to consider. The lightweight plus ontology will be the starting point for the representation of the knowledge domain that is involved in the ABB.

3. The Proposed Network Management Approach

The aim of this paper is the design and the implementation of a network management tool based upon the slow intelligence system approach. The system follows the architecture showed in figure 1 and in this paragraph more details on the operative workflow will be given. First of all the local server is described: it has the role to collect the information about the faults that are happening in the network and to solve them according the slow intelligence approach. At this aim each local server needs a knowledge domain represented by the following lightweight plus ontologies:

- O_{SNMP_i} $\{C_{SNMP\_i}, A_{SNMP\_i}, R_{HSNMP\_i}, R_{SNMP\_i}\}$: this ontology defines the entire structure of SNMP protocol events' signals that the local server "i" can manage. This ontology is part of a more general ontology O_{SNMP} developed by the analysis of SNMP standard (RFC 1157) and of the related Structure of Managed Information (RFC 2578)

- $O_{Fault\_i} = \{C_{Fault\_i}, A_{Fault\_i}, R_{HFault\_i}, R_{Fault\_i}\}$: this ontology describes each kind of possible errors that can occur within a LAN. This ontology is part of a more general ontology O_{Fault} developed by network manager experts that express also the relationships with the events that are in the O_{SNMP} ontology

- $O_{Cause\_i} = \{C_{Cause\_i}, A_{Cause\_i}, R_{HCause\_i}, R_{Cause\_i}\}$: this ontology defines the causes of the faults that may occur in a LAN. This ontology is part of a more general ontology O_{Cause} developed by network manager experts that express also the relationships with the faults that are in the O_{Fault} ontology

- $O_{Solution\_i} = \{C_{Solution\_i}, A_{Solution\_i}, R_{HSolution\_i}, R_{Solution\_i}\}$: this ontology defines the solutions that can be taken to recover from fault situations which occurred within a LAN. This ontology is part of a more general ontology $O_{Solution}$ developed by network manager experts that express also the relationships with the faults that are in the O_{Fault} ontology

- $O_{Action\_i} = \{C_{Action\_i}, A_{Action\_i}, H_{Action\_i}, R_{HAction\_i}, R_{Action\_i}\}$: this ontology aims to identify the actions to be taken in order to recover from fault's situations. This ontology is part of a more general ontology O_{Action} developed by network manager experts that express also the relationships with the faults that are in the $O_{Solution}$ ontology

- $O_{Component\_i} = \{C_{Component\_i}, A_{Component\_i}, R_{HComponent\_i}, R_{Component\_i}\}$: this ontology describes the components that are within the LAN "i". This ontology has to be developed by the network administrator of LAN "i" that defined also the relationships among the components and the SNMP events.

- $O_{Environment\_i} = \{C_{Environment\_i}, A_{Environment\_i}, R_{HEnvironment\_i}, R_{Environment\_i}\}$: this ontology describes the operative context where the LAN "i" works. This ontology has to be developed by the network administrator of LAN "i" that defined also the relationships among the environment and the SNMP events.

These ontologies represent the knowledge base of each advanced building block. The local server works as depicted in figure y and it acts like a slow intelligence and follows the following phases:

Enumeration Phase: in this phase the Local server tries to find all the actions that can be adopted in order to solve a fault. In particular the input of this stage is the SNMP event and the outputs are the actions that can be adopted. If the event has been managed in the past, the system adopts the previous actions and passes in the concentration phase. If the SNMP event has not been ever managed the enumeration module adopts the following functions:

$$F_{Enumeration}: E \times O_{SNMP\_i} \times O_{Fault\_i} \times O_{Solution\_i} \times O_{Action\_i} \rightarrow A^N$$

where E is the space of SNMP events and A is the space of the actions. In other words this function accepts as input the SNMP event that could be in the O_{SNMP_i} ontology. In this way it is possible, analyzing the ontologies, to find the actions that can lead to the solution of the fault. In general this function gives more than one actions that can be adopted and each of them has an effectiveness grade established by experts. At this point the system can evolve in the adaptation phase. If the function is not able to find an action the propagation phase has to be invoked.

Propagation Phase: in this phase the local server sends the SNMP event to the central server that tries to calculate the actions by the use of the function:

$F_{Enumeration}$: E x O_{SNMP_CS} x O_{Fault_CS} x $O_{Solution\_CS}$ x O_{Action_CS} -> A^N

If the central server is able to find the actions, it will send them to the local server "i" and it will send also the parts of ontologies that are needed for the event's resolution. In particular the following function will be invoked:

$S_{Propagation}$: E x O_{SNMP_CS} x O_{Fault_CS} x $O_{Solution\_CS}$ x O_{Action_CS} -> O^N

This function sends to the local server "i" the following ontologies

$O'_{SNMP\_CS} \subset O_{SNMP\_CS}$
$O'_{Fault\_CS} \subset O_{Fault\_CS}$
$O'_{Solution\_CS} \subset O_{Solution\_CS}$
$O'_{Action\_CS} \subset O_{Action\_CS}$

these ontologies have to be merged to the ontologies that are in the local server "i" in the following way

$O_{SNMP\_i} = O_{SNMP\_i} \cup O'_{SNMP\_CS}$
$O_{Fault\_i} = O_{Fault\_i} \cup O'_{Fault\_CS}$
$O_{Solution\_i} = O_{Solution\_i} \cup O'_{Solution\_CS}$
$O_{Action\_i} = O_{Action\_i} \cup O'_{Action\_CS}$

If the central server is not able to infer the actions the SNMP event has to be send to the other local servers in order to infer the actions. Also in this case each local server "j" calculates the actions by the use of the following function:

$F_{Enumeration}$: E x O_{SNMP_j} x O_{Fault_j} x $O_{Solution\_j}$ x O_{Action_j} -> A^N

and send to the central server the parts of ontologies that need for the actions' inference by the use of the function:

$S_{Propagation}$: E x O_{SNMP_j} x O_{Fault_j} x $O_{Solution\_j}$ x O_{Action_j} -> O^N

The central server collects the actions and updates its ontologies according to the previous described method. It sends the actions and the ontologies to the local server "i". If both the central server both the various local servers are not able to infer actions, the local server "i" has to send an error signal to the network administrator. The local server "i" collects the actions and the ontologies from the central server and invokes the adaptation phase.

Adaptation Phase: in this phase the inferred actions has to be customized according to the components that are in the LAN and the environment where the LAN works. In particular for each action the following function is invoked:

$A_{Adaptation}$: A x O_{SNMP_j} x O_{Fault_j} -> A

At the end of this phase the system obtains a list of adapted actions. Obviously not all the actions can be adapted and so for each adapted action an improvement for its effectiveness grade is set.

Elimination Phase: the system collects all the actions inferred in the previous phases ranking them according to this function:

$F_{Elimination}$: A^N-> A

This function can be implemented in various ways according to predefined strategy. In this case the adopted approach is the following:

$F_{Elimination} = max_{i=1…N}$ effectiveness grade (A_i)

At the end of this phase the concentration phase can be invoked

Concentration Phase: In this phase the local server "i" adopts the selected action. If this action leads to the problem's resolution and comes from the central server the local server "i" updates its ontologies. If the action does not lead to the problem's resolution a message is send to the network administrator that can decide both to adopt one of the other actions retrieved in the other phases both to solve the fault in a manual way.

Summarizing the operational workflow of the system can be described as follows:

Step 1: a SNMP message as result of a fault generated by a LAN's device is sent to a local server "i"

Step 2: The local server "i" receives the SNMP message. This is the beginning of the enumeration phase.

Step 3: The local server "i" tries to identify the problem through analysis of various ontologies that describe its knowledge base. If the SNMP event was managed in the past the concentration phase can start (step 9). Otherwise the system infers a list of actions that can applied for the resolution of the problem and generates the solutions and the actions that the various hosts in the LAN have to be apply. At this point the adaptation phase can start (step 7). If the local server "i" is not able to infer any actions the request is sent to the local server. In this way the propagation phase (step 4) can start.

Step 4: The central server tries to infer the actions that can solve the faults that the SNMP event sent by local server "i" represents. If it is able to find actions the central server sends them and the ontologies parts that are needed for the event management. In this way the adaptation phase (step) can start. Otherwise the central server sends the SNMP events to the other local servers

Step 5: The various local servers try to infer the actions from the received SNMP event. If actions are retrieved each local server sends them and the parts of ontologies needed for their inference.

Step 6: The central server collects the various answers from the local servers and send them local server "i" and the adaptation phase (step 7) can start. If no answers from local servers are received an empty action is sent to the local server.

Step 7: The local server "i" starts to adapt the actions according to the environment and components LAN's ontologies. After this phase the elimination phase can start. If in this phase no actions have been inferred a message to the network administrator have to be sent.

Step 8: The local server "i" selects the action to apply from the other ones collected in the other phases according to a predefined rule.

Step 9: The local server "i" can apply the action in order to recover the fault situation and update, if needed, its ontologies

4. Experimental Results

In order to test the performance of the proposed system an experimental campaign has been designed. First of all the working scenario has been set: the system has to manage three laboratories during their normal working time. These laboratories are equipped in the following way:

First Laboratory
- **1 Cisco Router Cisco**
- **3 Cisco Catalyst Switches**
- **56 Personal Computers equipped with heterogeneous operative systems and applications**
- **2 Network Printers**

Second Laboratory
- **1 Cisco Router**
- **2 Nortel Switches**
- **40 Personal Computers equipped with heterogeneous operative systems and applications**
- **2 Network Printers**

Third Laboratory
- **1 Cisco Router Cisco**
- **2 Cisco Catalyst Switches**
- **42 Personal Computers equipped with heterogeneous operative systems and applications**
- **1 Network Printers**

In each of these laboratories a local server was settled and the system monitored the three LANs for one week collecting the various SNMP signal and managing the various faults. The local servers have been furnished by the various ontologies and in particular they adopted an O_{SNMP_i} covering about the 60% of concepts of the full O_{SNMP} that can manage about 250 events. Starting from the O_{SNMP_i} the experts has built the others ontologies. The system's performances have been evaluated according various approaches. The first parameter is the following:

$$CA = \frac{SolvedFaults}{Events}$$

The aim of this index is the evaluation of the effectiveness of the system in the resolution of the faults. For the evaluation also the precision and recall parameters has been introduced. The precision is defined in the following way:

$$precision = \frac{TruePositive}{TruePositive + FalsePositive} \qquad recall = \frac{TruePositive}{TruePositive + FalseNegative}$$

These parameters are typically used in information retrieval where a perfect precision score of 1.0 means that every result retrieved by a search was relevant whereas a perfect recall score of 1.0 means that all relevant documents were retrieved by the search. In our case the precision means how many events have been resolved in the correct way (the true positive) respect the number of events that system tried to solve. So in this case a false positive is a fault that the system managed in a wrong way. The recall represents how many events have been resolved in the correct way (the true positive) respect the number of events that system could solve. For the evaluation of the propagation phase the following parameters has been introduced:

$$RCA = \frac{Solved\_Faults\_After\_Central\_Server\_request}{SolvedFaults}$$

In order to evaluate the capacity of the system to share knowledge among the various local servers also the following ontological parameters [17] has been introduced:

- NOC: Number of Concepts in the ontology
- NOL: Number of Leaf Concepts in the ontology
- NONHR: Number of "non-hierarchical" relationships
- NOF: Number of Fanouts
- AF-C: Average Fanout per Class
- MaxDIT: Maximum Depth of Inheritance Tree

All these parameters have been evaluated each 24 hours for each local server "i" and an average value has been expressed for the evaluation of the system. The obtained CA, Precision, Recall and RCA are depicted in the appendix of this paper as the ontological parameters for the full O_{SNMP} ontology and the various O_{SNMP_i} ontologies. The obtained results confirm the effectiveness of the proposed approach. The system allows an effective sharing of knowledge among the various servers as the ontological parameters show. In fact after about the 30% of the managed events each system reaches good results in their management and the local servers improve their knowledge domains. The system shows a very good CA result and more in general after a first training phase achieves very good performances both from the precision both from the recall point of view. It is important to underline that the various local ontologies show a less complex structure of the full ontology and so, in this way, it is easier to manage them.

5. Conclusion

In this paper a novel method for network management has been introduced. This method is based on Ontology and Slow Intelligence System approach. It has been tested in an operative scenario and the first experimental seems to be good. In particular the proposed approach showed how an ontology based interoperability framework can help to improve several tasks in the network management value chain. The proposed approach introduces a powerful way for the improvement of the information model interoperability and allows the introduction of services for the automatic resolution of networks fault. The opportunity to continuosly upgrade the knowledge base allows to continuosly upgrade the capacity of the system to manage new faults. In particular network administrators will not only benefit from more powerful applications, but they can transfer their expert knowledge into the

management applications and in this way automating more and more network management tasks. The future works aim to improve the system by the use of new and effective methodologies for the ontology management and the use of some artificial intelligence approach for the automatic inference of action when the system is not able to find anyone.

References

[1] Hui Xu, Debao Xiao, "A Common Ontology-based Intelligent Configuration Management Model for IP Network Devices", Proceedings of the First International Conference on Innovative Computing, Information and Control, pp. 385-388, 2006

[2] López de Vergara J.E., Guerrero A., Villagrá V.A., Berrocal J., Ontology Based Network Management: Study Cases and Lessons Learned, Computer Science Journal of Network and Systems Management, Volume 17, Number 3, pp. 234-254, 2009

[3] Wong A.K.Y., Ray P., Parameswaran N., Strassner J., Ontology mapping for the interoperability problem in network management, IEEE Journal on Selected Areas in Communications, 23(10), 2058-2068, 2005

[4] López de Vergara J.E., Villagrá V.A., Asensio J.I., Berrocal J., Ontologies: giving semantics to network management models, IEEE Network, Volume 17, Number 3, pp. 15-21

[5] Keeney J., Lewis D., O'Sullivan D., Roelens A., Boran A., Richardson R., Runtime semantic interoperability for gathering ontology-based network context, Proceedings of IEEE/IFIP Network Operations and Management Symposium (NOMS 2006), 56-65,Vancouver, 2006

[6] Van der Meer, S., Jennings B., O'Sullivan D., Lewis D., Agoulmine N., Ontology based policy mobility for pervasive computing, Proceedings 12th Workshop of the HP Open University Association, 2005, 211-224

[7] Shi-Kuo Chang, "A General Framework for Slow Intelligence Systems", International Journal of Software Engineering and Knowledge Engineering, Volume 20, Number 1, February 2010, pp. 1-15.

[8] Jepsen, T., Just What Is an Ontology, Anyway?, IT Professional, vol. 11, no. 5, pp. 22-27, Sep./Oct. 2009

[9] Gruber, T.R, Translation approach to portable ontology specification, Knowledge Acquisition, vol. 5, pp. 199-220, 1993

[10] "OWL Web Ontology Overview", W3C Recommendation, 10 february 2004, http://www.w3.org/TR/2004/REC-owl-features-20040210/

[11] Maedche A., Staab S., Ontology Learning for the Semantic Web, IEEE Intelligent Systems, vol. 16 no. 2, Mar/Apr 2001, Page(s): 72-79

[12] Corcho, O., A Layered Declarative Approach to Ontology Translation with Knowledge Preservation, Vol. 116 Frontiers in Artificial Intelligence and Applications, 2005

[13] Corcho, O., Gómez-Pérez, A., A Layered Model for Building Ontology Translation Systems, International Journal on Semantic Web and Information Systems, 1(2): 22-48, 2005.

[14] Haase, P., Stojanovic, L., Consistent evolution of OWL Ontologies, ESWC, Lecture Notes in Computer Science, vol. 3532, Springer

[15] Yinglin Wang, Xijuan Liu, Rongwei Ye, "Ontology Evolution Issues in Adaptable Information Management Systems," E-Business Engineering, IEEE International Conference on, pp. 753-758, 2008

[16] Zhang, L., Xia, S., Zhou, Y., Xia, Z., User Defined Ontology Change and its Optimization, Control and Decision Conference, 2008. CCDC 2008. Chinese, pp. 3586-3590, 2008

[17] Orme, A.M., Haining, Y., Etzkorn, L.H., Indicating Ontology Data Quality, Stability and Completeness Throughout Ontology Evolution, Journal of Software Maintenance and Evolution: Research and Practice, vol. 19, pp. 49-75, 2007

[18] Stallings, W., SNMP, SNMPv2, SNMPv3, and RMON1 and 2, Addison-Wesley, Reading, MA, 1999

[19] Case, J., Mundy, R., Partain, D., Introduction and Applicability Statements for Internet Standard Management Framework, RFC 3410, 2002, ftp://ftp.rfc-editor.org/in-notes/rfc3410.txt

Appendix

	SNMP_Events_1	SNMP_Events_2	SNMP_Events_3	SNMP_Events_4	SNMP_Events_5	SNMP_Events_6	SNMP_Events_7	Total
Local_Server_1	1423	1562	1233	1321	1401	720	603	8263
Local_Server_2	1378	1492	1541	1647	1230	601	532	8421
Local_Server_3	1401	1351	1321	1678	1345	654	472	8222

	CA_1	CA_2	CA_3	CA_4	CA_5	CA_6	CA_7	Average_Value
Local_Server_1	60,01%	77,40%	90,84%	91,07%	91,29%	95,83%	98,84%	84,12%
Local_Server_2	64,73%	73,93%	84,23%	87,37%	96,50%	97,67%	98,31%	83,47%
Local_Server_3	60,17%	75,94%	87,51%	89,27%	96,21%	97,55%	98,94%	84,19%

	Precision_1	Precision_2	Precision_3	Precision_4	Precision_5	Precision_6	Precision_7	Average_Value
Local_Server_1	72,01%	85,93%	91,58%	95,55%	96,75%	98,29%	99,67%	90,31%
Local_Server_2	72,05%	85,04%	92,78%	94,98%	96,98%	99,83%	99,81%	90,29%
Local_Server_3	73,05%	84,65%	88,11%	97,08%	98,40%	99,69%	99,79%	90,55%

	Recall_1	Recall_2	Recall_3	Recall_4	Recall_5	Recall_6	Recall_7	Average_Value
Local_Server_1	86,61%	93,29%	95,40%	96,94%	99,22%	100,00%	100,00%	95,59%
Local_Server_2	85,03%	91,61%	94,68%	96,25%	97,70%	98,49%	99,81%	94,30%
Local_Server_3	83,38%	89,30%	95,54%	98,23%	98,85%	99,07%	99,36%	94,59%

	RCA_1	RCA_2	RCA_3	RCA_4	RCA_5	RCA_6	RCA_7	Average_Value
Local_Server_1	22,13%	12,90%	8,93%	7,15%	3,36%	3,33%	2,01%	8,76%
Local_Server_2	17,60%	9,16%	5,62%	3,89%	2,36%	1,53%	0,19%	6,05%
Local_Server_3	24,08%	15,89%	10,64%	5,21%	3,17%	2,51%	0,86%	9,07%

Table 1 Obtained Results

SNMP_Full_Ontology	
NOC	378
NOL	250
NONHR	9
NOF	1732
AF_C	4,58
MaxDIT	7

O_{SNMP_i}	NOC_1	NOC_2	NOC_3	NOC_4	NOC_5	NOC_6	NOC_7
Local_Server_1	189	223	278	289	302	315	318
Local_Server_2	176	218	267	281	298	306	311
Local_Server_3	183	197	253	278	293	301	305

O_{SNMP_i}	NOL_1	NOL_2	NOL_3	NOL_4	NOL_5	NOL_6	NOL_7
Local_Server_1	150	172	184	195	200	203	204
Local_Server_2	158	171	181	190	198	202	203
Local_Server_3	143	162	176	184	191	193	194

O_{SNMP_i}	NONHR_1	NONHR_2	NONHR_3	NONHR_4	NONHR_5	NONHR_6	NONHR_7
Local_Server_1	6	7	8	8	8	9	9
Local_Server_2	5	6	7	8	8	9	9
Local_Server_3	7	8	9	9	9	9	9

O_{SNMP_i}	NOF_1	NOF_2	NOF_3	NOF_4	NOF_5	NOF_6	NOF_7
Local_Server_1	754	903	1163	1201	1278	1332	1389
Local_Server_2	730	930	1034	1175	1208	1299	1326
Local_Server_3	734	892	979	1078	1189	1256	1301

O_{SNMP_i}	AF_C_1	AF_C_2	AF_C_3	AF_C_4	AF_C_5	AF_C_6	AF_C_7
Local_Server_1	3,99	4,05	4,18	4,16	4,23	4,23	4,37
Local_Server_2	4,15	4,27	3,87	4,18	4,05	4,25	4,26
Local_Server_3	4,01	4,53	3,87	3,88	4,06	4,17	4,27

O_{SNMP_i}	MaxDIT_1	MaxDIT_2	MaxDIT_3	MaxDIT_4	MaxDIT_5	MaxDIT_6	MaxDIT_7
Local_Server_1	7	7	7	7	7	7	7
Local_Server_2	7	7	7	7	7	7	7
Local_Server_3	7	7	7	7	7	7	7

Table 2 Ontological Parameters for the full OSNMP and the various OSNMP_i