Autonomic Fault Management based on Cognitive Control Loops

[1]Sung-Su Kim, [1]Sin-seok Seo, [2]Joon-Myung Kang, and [1,3]James Won-Ki Hong
[1]Dept. of Computer Science and Engineering, POSTECH, {kiss, sesise, jwkhong}@postech.ac.kr
[2]Department of Electrical and Computer Engineering, University of Toronto, joonmyung.kang@utoronto.ca
[3]Division of IT Convergence and Engineering, POSTECH

Abstract— **This paper presents an efficient fault management approach based on cognitive control loops in order to support autonomic network management for the Future Internet. The cognitive control loops determines urgency of network alarms, processes urgent alarms more quickly, and then infers root causes of the problems based on learning and reasoning. We show that we reduce a number of alarms by correlation and detect alarm priorities using an ontology model based on the policy.**

Index Terms— **Autonomic Fault Management, Cognitive Management, Alarm Correlation, Association Rule Mining**

I. INTRODUCTION

The Internet is a very successful modern technology. Despite that success, fundamental architectural and business problems exist in its design. Incremental patches have been added to solve those problems so far. However, there is a limitation to solve inherent problems incrementally, such as a lack of IP addresses, security, and management problems. There are some approaches for the design of the Future Internet: revolutionary and evolutionary [1] [2]. In this design, management of the Future Internet is one of the important topics. However, we do not have a clear picture of the Future Internet yet and many emerging technologies are investigated for the Future Internet. For example, Content Centric Networking (CCN) is the one of the hot issues and network virtualization and autonomic networking will be the key technologies for the Future Internet [3].

Although a new Internet architecture substitutes the current Internet architecture, a basic paradigm of network management will not be changed. The paradigm is to understand current status of network and take the appropriate actions. In order to understand the network status, we need to monitor network devices, links, and servers. Network administrators suffer from lots of network events and alarms. Enterprise networks generate millions of network alarms per day. In cloud computing or virtualized network environment, there will be more network events and alarms to be analyzed. In addition to physical entities, alarms related to virtualized resources will be generated.

Existing rule based and case based alarm correlation approaches need manually defined rules and cases based on assumption that a managed network is stable. However, there might be a missing dependency between alarms and a manual modification is necessary when a managed network is changed. For example, if a topology of the managed network is changed, some rules related with a topology should be changed manually. Therefore, it is necessary to update a dependency model with learning. Alarms contain information about serious status of network resources, such as link, router, switch, etc. However, this fragmentary information does not tell the impact of a certain problem. Serious and urgent alarms need to be detected and processed more quickly than normal alarms.

We propose an efficient fault management approach based on a cognitive control loop which is a part of the new FOCALE model. The cognitive control loop determines priorities of network alarms, processes alarms with three different control loops, and then infers root causes of the problems based on learning and reasoning. In order to evaluate our approach, we synthetically generate alarms, correlate and analyze them to find root causes. In addition, we propose ontology for determining the priorities of alarms. Urgent cases are treated immediately with specified actions. Otherwise, possible sets of actions are examined and the most appropriate one is selected. In our experiment, 16 different alarms are reduced to four clusters by using learned rules and our clustering algorithm. It means that the effort and time of higher-level network manager s can be reduced.

The organization of this paper is as follows. Section 2 covers related work on a FOCALE autonomic architecture [4] for the future Internet and alarm correlation. Section 3 presents a concept of a cognitive control loop. Section 4 describes a detailed approach for processing network alarms. Section 5 presents a case study to validate our concept and algorithm. Finally, Section 6 presents conclusions and future work.

II. RELATED WORK

In this section, we present a FOCALE autonomic architecture and existing alarm correlation approaches.

A. FOCALE

FOCALE [5] is an autonomic networking architecture. The acronym FOCALE stands for **F**oundation – **O**bservation – **C**ompare – **A**ct – **L**earn – r**E**ason, which describes its novel control loops. Note that other autonomic approaches, such as Since there are at least two fundamentally different operations

978-1-4673-0269-2/12/$31.00 ©2012 IEEE

that the control loop is responsible for – monitoring vs. (re)configuration – this overloads the semantics of the control structure, since these two operations have nothing in common. Indeed, a fault received from one managed entity might not have anything to do with the root cause of the problem; hence, the (re)configuration loop will affect different entities. FOCALE uses the DEN-ng information model [7] and the DENON-ng ontologies [8] to translate disparate sensed data into a common networking *lingua franca*. The DEN-ng information model is currently being standardized in the Autonomic Communications Forum (ACF); its previous versions have already been standardized in the TeleManagement Forum and in the ITU-T. The DEN-ng is used to represent static characteristics and behaviors of entities; the DENON-ng is then used to augment this model with consensual meaning and definitions so that vendor-specific concepts can be mapped into a common terminology. This enables facts extracted from sensor input data to be reasoned about using ontology-based inferencing.

B. *Alarm Correlation Approaches*

There are four alarm correlation approaches, rule-based alarm correlation [9], codebook-based alarm correlation [10], case-based alarm correlation, mining based alarm correlation [11] . However, Rule-based, codebook-based, and case-based approaches are highly dependent on expert knowledge of skilled operators. Especially, it is not easy to reflect dynamically changing network condition such as wireless or overlay environments because rules or dependency models are made manually based on the assumption that network is mostly stable. Mining based alarm correlation is able to detect the cause and effect relationships between alarms [11, 12]. However, it is hard to detect relationships in a short period of time because of its long processing time. Our method used both rule-based and mining based approaches. Efficiency can be taken from the rule-based approach and dynamic changing relationships are detected by mining based approach.

III. Cognitive Control Loops

FOCALE [5] control loops are self-governing, in the system senses changes in itself and its environment, and determines the effect of the changes on the currently active set of business policies. As shown in Fig. 1, the FOCALE control loops [13] operate as follows. Sensor data is retrieved from the managed resource (e.g., a router) and fed to a model-based translation process, which translates vendor- and device-specific data into a normalized form in XML using the DEN-ng information model and ontologies as reference data. This is then analyzed to determine the current state of the managed entity. The current state is compared to the desired state.

In order to strengthen the self-awareness, the new FOCALE cognition model employs a model of human intelligence built using simple processes, which interact according to three layers, called *reactive*, *deliberative*, and *reflective* [14, 15]. The new FOCALE cognition model employs cognitive processes as shown in Fig. 2**.**

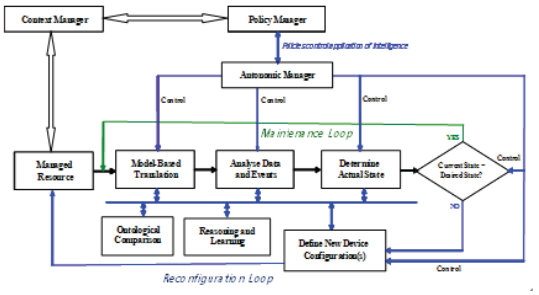


Fig. 1 Simplified Version of the FOCALE Autonomic Architecture

Since all processes use the Finite State Machine (FSM) and reasoner, the system can recognize when an event or a set of events has been encountered before. Such results are stored in short-term memory. This reactive mechanism enables much of the computationally intensive portions of the control loop to be bypassed, producing two "shortcuts" labeled "high priority" and "urgent". The deliberative process is embodied in the set of bold arrows, which take the Observe-Normalize-Compare-Plan-Decide-Act path. This uses long-term memory to store how goals are met on a context-specific basis. The reflective process examines the different conclusions made by the set of deliberative processes being used, and tries to predict the best set of actions that will maximize the goals being addressed by the system. This process uses semantic analysis to understand why a particular context was entered and why a context change accrued to help predict how to more easily and efficiently change contexts in the future. These results are also stored in long-term memory, so that the system better understand contextual changes its reasoning to aid debugging.

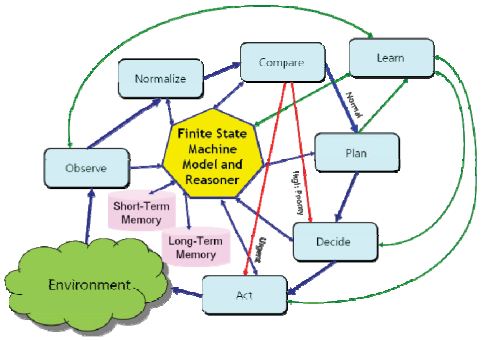


Fig. 2 FOCALE Cognitive Model

IV. Autonomic Fault Management

In this section, we describe how alarms are processed in a cognitive control loop. Cognitive control loops are able to adapt to changing environment with reasoning and learning. In addition, alarms are classified based on their urgency to solve important problems more quickly.

A. *Multiple Control Flows based on the Priorities of Alarms*

The cognitive control loops process network events and alarms. At the same time, relationships between alarms are leaned to adapt to changing environmental conditions. As shown in Fig. 3, multiple control loops are available based on a priority of an alarm.

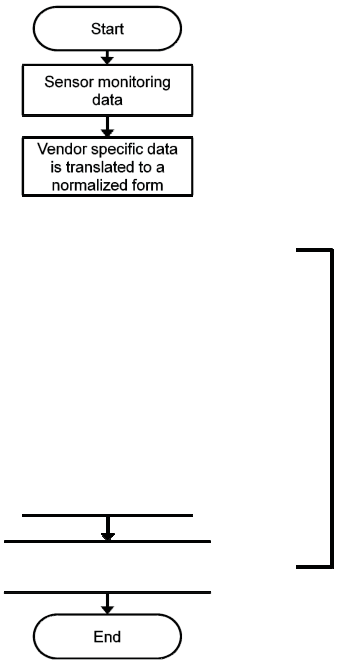


Fig. 3 Multiple Control Flows based on a Priority

These control flows are mapped to the new FOCALE cognition model in Fig. 3. In an observe phase, data is retrieved from the managed resource (e.g., SNMP polling or trap). Vendor specific data is translated to a normalized form based on the DEN-ng information model. Network alarms are filtered and correlated in order to efficiently find root cause alarms. In this phase, a dependency model is used to correlate alarms. At the same time, a normalized data is fed to a learning phase. Changing environment conditions are captured by learning, especially relationships between alarms are detected to update a dependency model. After correlating alarms in a normalize phase, a priority of the alarms is determined by classifying the alarm. The alarm is classified as urgent if this alarm affects serious performance degradation of network resources or services. Alarm priorities are determined based on a policy. If an alarm is urgent, a set of actions is sent to the network devices without passing through plan and decide phases. This is the difference from the previous version of FOCALE control loops. If the current state is a high priority, it skips a plan phase for taking immediate actions. For a low priority alarm, a plan phase takes a high-level behavioral specification from humans, and controls the system behavior in such a way as to satisfy the specifications. It means that a plan phase computes all the possible sets of actions to change the current state to a desired state. A decide phase chooses a set of actions which maximize a goal. Finally, an act phase sends commends for chosen action to target network devices. Model-based translation converts device-neutral actions to device-specific commands.

Network alarms are correlated in the normalize phase of new FOCALE control loops. First, alarm information is extracted to the form of Fig. 4. Typically, a single failure affects to other services and devices. Therefore, if a single failure occurs somewhere in the network, many alarms related to the failure are generated. Once a fault occurs, many identical alarms are generated to notify the fault before it is fixed. Those identical alarms are generalized as shown in Fig. 4. By alarm generalization, the number of alarms is reduced. Generalized alarms are then correlated to reduce the number of alarms and find root cause alarms.

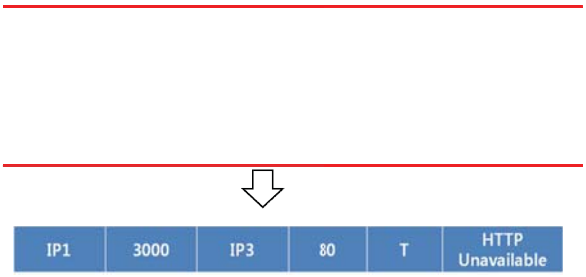


Fig. 4 Example of Alarm Generalization

Fig. 5 An Example of a Dependency Model

Alarm correlation depends on a basic dependency model and association rules detected in the learning process are added. As we mentioned, a learning process learns relationships between alarms. Fig. 5 shows a basic dependency model which is manually defined. It is based on the TCP/IP model. A lower layer problem affects to higher layers. For example, if a server link is down, an IP layer is also unavailable. At first, a manually defined dependency model is used. Additional rules learned from association rule mining are added to the basic dependency model to adapt to changing environment conditions.

Algorithm 1 describes how to make a set of clusters based on the association rules. Initially, alarms are generalized and grouped with a same alarm ID. It means that each alarm is a single cluster by itself in the initial phase. Then, all the association rules are examined one by one and the corresponding clusters are merged. In this way, each cluster contains both alarms and relationship information. Therefore, root cause alarms can be analyzed easily. Based on relationships between alarms belonged to the same cluster, root causes can be inferred.

Algorithm 1. Alarm Clustering

Input: A set of *E* of alarms (a_1, a_2, …)
A set of *R* of association rules (r_1, r_2, ...)
Output: A set of *C* of clusters (c_1, c_2, …)
1: *C*= group the set *E* by an identical alarm ID
2: *n*= count(*R*)
3: **for** *i* = 1 to *n*
4: rule r_i is form of $a_j \rightarrow a_k$
5: find cluster c_l, including alarm a_j
6: find cluster c_m, including alarm a_k
7: merge c_l and c_m into c_l
8: put the association rule r_i into c_l

B. Association Rule Mining

We can use various machine learning techniques for inferences. For efficient alarm correlation, it is extremely important to find relationships between alarms. In this paper, association rule mining is used to find the cause and effect from relationships between alarms.

Table 1. Alarm transaction data sets

TID	Transaction item sets
1	A1, A2, A4, A5
2	A1, A4, A5
3	A2, A3, A4, A5
4	A1, A2, A4,
5	A1, A3, A5

The transaction database is made of the alarm data in the managed network after pretreatment shown in Table 1. Each transaction in a database has a unique transaction ID and contains a subset of the items. A rule is defined as an implication of the form $X \rightarrow Y$, where $X, Y \subseteq I$ and $X \cap Y = \emptyset$. A priori association rule algorithm basically has two steps; the first is finding all frequent item sets in a data set by applying *min_sup* (minimum support threshold).; the second is generating association rules based on the frequent item sets. For any transaction sets for X, the support for the X, *sup(x)*, is defined as a portion of the transactions in the data set which contains the item set in Equation (1). In Table 1, support of {A1} is 4/5 (80%). We assume that the default value of *min_sup* is 10% and the support of {A1} is greater than *min_sup*. Therefore, rules related to A1 should be found.

$$\mathrm{sup(x)} = \frac{\mathrm{count(x)}}{\text{the number of total transaction}} \times 100\ (\%) \qquad (1)$$

Confidence of the rule $X \rightarrow Y$ is defined in Equation (2). In Table 1, $conf(A1 \rightarrow A4)$ is $sup(A1 \cup A4) / sup\ (A1) = 75\%$. Frequent item sets and the minimum confidence constraint are used to form rules.

$$\mathrm{conf(x \rightarrow y)} = \frac{\mathrm{sup\ (x \cup y)}}{\mathrm{sup\ (x)}} \times 100\ (\%) \qquad (2)$$

C. Determination of Alarm Priorities

One of the most important features of the cognitive control loops is that alarms are controlled differently based on their priorities. Urgent alarms can be processed faster than normal alarms. Classifying urgent alarms is dependent on a goal and policy of a network. We defined the ontology based on the DEN-ng information model to make effective semantic representations of network elements, alarms, and their priorities.

Fig. 6 describes the concept of network elements and alarms for determining their state and priorities. An element is a network resource that has its own state, such as CPU utilization, link throughput, etc. An element provides services and notifies its state to a network administrator. A notification can be an alarm or event. An alarm has a destination, source, and type as described in Fig. 4. Alarms are classified into three classes: urgent, high priority, and low priority. An element also provides a service. A service has three classes: gold, silver, and bronze. A gold service is the most important service.

Three alarm classes are defined based a policy of a managed network. For example, it can be defined by a network administrator that if an alarm affects to the Service Level Agreement (SLA) violation of a gold service, it can be classified as urgent. We use Semantic Web Rule Language (SWRL) [18] to make conditional rules into the ontology.

We assume that alarms related to a gold service are urgent and a gold service is provided by the server WS2 in Fig. 7 and alarms related to WS2 are classified as urgent. For example, "WS2 HTTP unavailable", "WS2 IP down", or "WS2 port down" are urgent alarms needed to be fixed as soon as possible. The following SWRL rules are for classifying alarms based on our assumption. These SWRL rules determine alarms as urgent if alarms are about an element that provides a gold service.

- *Alarm(?a) ∧ hasAlarmDest(?a, ?dest) ∧ Element(?dest) ∧ providesService(?dest, ?s) ∧ GoldService(?s) →UrgentAlarm(?a)*
- *Alarm(?a) ∧ hasAlarmSrc(?a, ?dest) ∧ Element(?dest) ∧ providesService(?dest, ?s) ∧ GoldService(?s) →UrgentAlarm(?a)*

Fig. 6 Ontological Model for Alarms and Priority

V. Evaluation and Result

In this section, we describe evaluation and its results for validating our proposed approach.. We implemented the alarm

correlation algorithm in a Java language and used the Weka [16] library for association rule mining. We generated synthetic alarm data sets for the experiment which correlates alarms and finds root causes. Fig. 7 shows the experimental topology composed of 22 nodes with four critical alarms. "IP Down", "Link down", "Port down", and "Router down" occur randomly at designated nodes as shown in Fig 8. A default route from R6 to R1 is through R3-R0-R1. If a link between R3 and R0 is down, N4 cannot connect to WS1 and FS2. We generated two synthetic alarm data sets for training and validation. For example, the link of the router R0 is down from 5 to 15 second and the WS1's port is down from 20 to 30 second. If "WS1 port down" occurs, N1 and N8 generate an alarm "WS1 HTTP Unavailable". We assumed that N1 and N8 periodically poll the states of all the servers in the network.

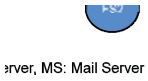


Fig. 7 Experimental Topology

Based on the generated synthetic alarm data set, the cause and effect relationships are detected. Table 2 shows a part of alarms, alarm IDs, and detected association rules. A3➔A4 and A3➔A6 mean that the services on WS1 and FS2 are unavailable if the R0-R3 link becomes down. Based on the rules, when A3, A4, and A6 alarms are generated, A3 is identified as the root cause alarm. There are thousands of alarms in enterprise networks [17] and a large number of rules can be detected.

Table 2. Alarms and Association Rules Detected

Alarm ID	Alarm	Rules
A3	R0-R3 link down	A3➔ A4
A4	N4➔WS1 HTTP unavailable	A3➔A6
A5	N1➔WS1 HTTP unavailable	A8➔A6
A6	N4➔FS2 FTP unavailable	A9➔A6
A7	N4➔WS2 HTTP unavailable	A3➔A7
A8	N1➔FS2 FTP unavailable	
A9	FS2 IP down	

Then, all the critical alarms described in Fig 8 are generated simultaneously. The type and the number of generated alarms are described in Fig. 8. The node N1 generates "WS2 HTTP unavailable" and "FS2 FTP unavailable" alarms. However, in the specific time window, the node N1 generates multiple alarms when the fault is not fixed during the time window. Those alarms are including redundant and similar alarms. For example, the node N1 generates five "WS2 HTTP unavailable" and five "FS2 FTP unavailable" alarms. Those identical alarms are generalized as we explained in Section 4. Therefore, a higher level manager receives a reduced number of alarms. In our experiment, the generalization process reduces 100 alarms to 22 alarms. However, alarms generated by node N6 and N8 are not received because of the failure of R4.

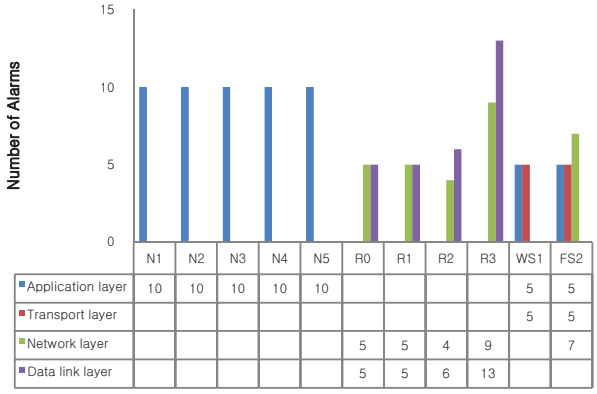


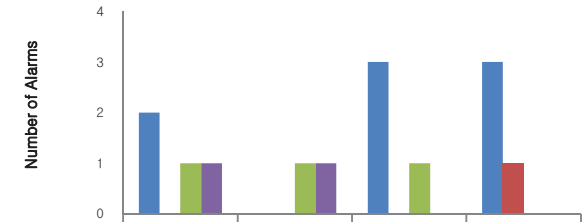
Fig. 8 Synthetically Generated Alarms from Each Device

Fig. 9 shows an output of our clustering algorithm. Even if the total number of alarms is still large, we can find a root cause alarm in each cluster easily. For example, the cluster 1 consists of "R0 link down", "WS1 HTTP unavailable", and "FS2 FTP unavailable" alarms. Based on the association rule in Table 2, we can infer that the root cause alarm is "R0 link down". The cluster 4 consists of "WS1 port down" and "WS1 HTTP unavailable" alarms. Therefore, network administrators only can focus on the root cause alarm. Fig. 9 shows the number of clusters and alarms of each cluster. The root cause alarm of the cluster 1 is "R0-R3 link down". The other alarms caused by "R0-R3 link down" are included in the cluster 1. The cluster 1 includes the other alarms caused by "R0-R3 link down", such as "N4➔WS1 HTTP unavailable", "N4➔WS2 HTTP unavailable" and "N4➔FS2 FTP unavailable". The root cause alarm of the cluster 2 is "R4 router down", and "R4 neighbor loss" and "R4 IP down" are related alarms. The root cause alarms for cluster 3 and 4 are "FS2 IP down" and "WS1 port down" respectively. 14 different alarms are reduced to four clusters.

Based on the ontological model shown in Fig. 6, we made a SWRL rule for determining a priority of an alarm. The alarms in four clusters are examined and classified to determining their priorities by a SWRL rule shown in Section 4. An alarm "N4➔WS2 HTTP unavailable" is determined as an urgent alarm. WS2 provides a gold service, which has the highest priority among the service classes. It means that the problem is significant and should be solved immediately. The root cause of the alarm is "R0-R3 link down" by analyzing from the alarm

cluster 1. Therefore, a set of commands for recovering a link R0-R3 is sent to R0 and R3 without passing through plan and decide phases.

Ontology and SWRL rules enable us to analyze services affected by an alarm as well as the state of network devices. Based on the analysis, we can determine whether the alarm is urgent or not. It enables us to solve an urgent problem quickly. If the cognitive control loop is not used, alarms are processed one by one. Even if an alarm is urgent, it would be treated same as other normal alarms. Then, critical alarms cannot be examined while others are being processed. This is the strength of a cognitive control loop.



	Cluster1	Cluster2	Cluster3	Cluster4
Application layer	2		3	3
Transport layer				1
Network layer	1	1	1	
Data link layer	1	1		

Fig. 9 Clustered Set of Alarms

VI. Conclusions

In this paper, we have proposed an efficient fault management approach based on cognitive control loops. We have shown a case study to validate our concept using the synthetically generated alarm data sets. At first, manually defined dependency model is used. Missing and changing dependencies are detected by a learning phase of the control loops. Ontology and SWRL rules are used to represent the relationships among network resources, services, and alarm priorities. From the evaluation, we have shown that we reduced a number of the alarms, processed the alarms with different orders based on the alarm priority, and found root causes easily by association rules.

Our future work is to evaluate the performance of the control loops. We will show that our approach can process urgent alarms more quickly comparing to the existing control loops.

Acknowledgments

This research was supported by the WCU (World Class University) program through National Research Foundation of Korea funded by the Ministry of Education, Science and Technology (R31-2010-000-10100-0) and the KCC(Korea Communications Commission), Korea, under the "Novel Study on Highly Manageable Network and Service Architecture for New Generation" support program supervised by the KCA(Korea Communications Agency)" (KCA-2011-10921-05003)

References

[1] A. Feldmann, "Internet clean-slate design: what and why?", ACM SIGCOM Computer Communication Review, Vol. 37, Issue 3, Jul. 2007, pp. 59-64.

[2] M. Blumenthal and D. Clark, "Rethinking the design of the Internet: The end to end arguments vs. the brave new world", ACM Transactions on Internet Technology, vol. 1, no. 1, Aug. 2001, pp. 70-109.

[3] V. Jacobson, D.K. Smetters, J.D. Thornton, M.F. Plass, N.H. Briggs, and R.L. Braynard, "Networking Named Content," In CoNEXT '09, Rome, Italy, Dec. 2009

[4] J. Strassner, "Autonomic Networking – Theory and Practice", 20th Network Operations and Management Symposium (NOMS) 2008 Tutorial, Brazil, April 7, 2008.

[5] J. Strassner, N. Agoulmine, and E. Lehtihet, "FOCALE – A Novel Autonomic Networking Architecture", ITSSA Journal, Vol. 3, No. 1, May 2007, pp 64-79.

[6] IBM, "An Architectural Blueprint for Autonomic Computing, v7", http://www-03.ibm.com/autonomic/pdfs/AC%20Blueprint%20White%20Paper%20V7.pdf.

[7] J. Strassner, "Introduction to DEN-ng", Tutorial for FP7 PanLab II Project, 2009.

[8] M. Serrano, J. Serrat, J. Strassner, and M. Ó Foghlú, "Management and Context Integration Based on Ontologies, Behind the Interoperability in Autonomic Communications", extended journal publication of the SIWN International Conference on Complex Open Distributed Systems, Chengdu, China, Vol 1, No. 4, Jul. 2007

[9] D. Banerjee, V. R. Madduri, and M. Srivatsa, "A Framework for Distributed Monitoring and Root Cause Analysis for Large IP Networks," 28th IEEE International Symposium on Reliable Distributed Systems, September 2009, pp.246-255.

[10] White Paper, "Automating Root-Cause Analysis: EMC Ionix Codebook Correlation Technology vs. Rules-based Analysis, Nov. 2009.

[11] Jukic O. and Kunstic M., "Logical Inventory Database Integration into Network Problems Frequency Detection Process," ConTEL 2009, Jun. 2009, pp.361-365.

[12] Risto Vaarandi, "A Data Clustering Algorithm for Mining Patterns from Event Logs," IP Operations and Management (IPOM 2003), Oct. 2003, pp.119-126.

[13] J. Strassner, J.W.K. Hong, S. van der Meer, "The Design of an Autonomic Element for Managing Emerging Networks and Services", International Conference on Ultra Modern Telecommunications (ICUMT 2009), October 12-14, 2009, St. Petersburg, Russia.

[14] M. Minsky, "The Society of Mind", Simon and Schuster, New York, ISBN 0671657135, 1988.

[15] J. Famaey, S. Latre, J. Strassner, and F. De Turck, "A Hierarchical Approach to Autonomic Network Management", Network Operations and Management Symposium Workshops, 2010 IEEE/EFIP, , Osaka, Japan, April 19, 2010.

[16] Ian H. Witten and Eibe Frank, "Data Mining: Practical Machine Learning Tools and Techniques with Java Implementation", Morgan Kaufmann Publishers. ISBN 1-55860-552-5.
[17] X. Chen, Y. Mao, Z. M. Mao, and J. Van Der Merwe, "KnowOps: Towards an Embedded Knowledge Base for Network Management and Operations," In Proceedings of the 11th USENIX conference on Hot topics in management of internet, cloud, and enterprise networks and services (Hot-ICE'11), Berkeley, CA, USA, 7-7.
[18] SWRL: A Semantic Web Rule Language Combining OWL and RuleML,W3C Member Submission 21, May 2004.