A Normative and Self-Organizing Piloting Model for Virtual Network Management

Carolina Valadares, Manoel T. Abreu Netto, and Carlos J. P. de Lucena
Department of Informatics
Pontifícia Universidade Católica do Rio de Janeiro
Rio de Janeiro, Brazil
{cvaladares, mnetto, lucena}@inf.puc-rio.br

***Abstract*—The Internet has become an essential role in the society, serving every day billions of users spread all over the world. It is a complex network that holds an extensive range of services, applications and technologies. Its model, however, makes it difficult to solve structural problems such as management and maintaining. Network virtualization has been proposed to tackle this issue. In this paper, we use the concept of multi-agent system, norms and self-* properties to propose and validate an autonomic self-organizing model for virtual network management. As our proof-of-concept, we show that our system, which is composed of a virtual network of virtual machines capable of self-organizing themselves in a totally decentralized way across a physical infrastructure in order to cope with environment changes, satisfies its main goal of efficiently re-organize itself with no central control.**

Keywords—multi-agent; selg-organazing; virtual network management; norms; autonomic network;

I. INTRODUCTION

The Internet is a complex network that servers billions of users spread all over the world. It caries an extensive range of services, technologies, applications and has also enabled a variety of forms of human interactions and information exchange. Even through its architecture facilitates the deployment of new applications, due to its transparency, its model makes it difficult to solve structural problems such as scalability, management, mobility and security [3]. The Internet is a large-scale network and a trivial approach for its management, which involves human being interference, becomes costly and flawed as its size increases.

Autonomic network virtualization has been pushed forward by its proponents to tackle the Internet ossification problem. It represents a new approach that has recently received substantial attention from academia, whereas it is able to run multiple virtual networks simultaneously on the top of a single physical substrate. We intend to deal with the complexity aggregated to the new concept of virtual network by enabling the self-management behavior. This self-* capability represents a specific area of autonomic computing [11], a term coined by IBM, to deal with such complexity by enabling systems to self-manage themselves. The main key behind autonomic network visualization is, therefore, the building of flexible networks capable of managing themselves in order to deal with external changes and interferences from the environment.

Virtual networks can support simultaneous independents network experiments, services and architectures over a shared substrate network [7][9]. Each virtual network is capable of running its own protocols, routing process, services and management solutions, in a way of totally isolation and independency, although they share the same infrastructure. It is composed of a set of physical and virtual resources, as depicted in Figure 1, in which physical resources (substrate node) consist of devices such as router, access points, and are able to embed many virtual nodes. These virtual nodes are connected together by virtual links, which is also embedded on physical resources. Both virtual node and virtual link belong to a dedicated virtual network, that supports a specific service or protocol [5], in which every substrate and virtual node has a self-organizing piloting agent embedded, responsible for handling local decisions and actions, which characterizes it as a decentralized model.

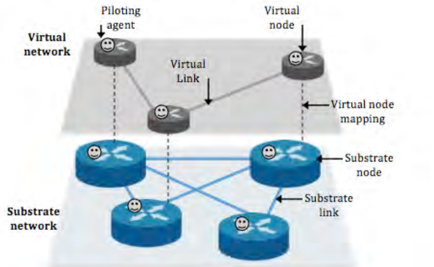

Figure 1 Virtual Network Model

The piloting agent itself is what leads the virtual network to emerge a self-organizing and is in charge of handling local behaviors to enable a proper control and management of the virtual network, its components, and the network flows, in order to maintain the efficient use of substrate resources on network virtualization. It represents the adaptive method running inside of each substrate and virtual node, which is responsible for adapting and managing the network resources in order to meet quality policies and users requirements in case of environment changes. Moreover, it is composed of high-level norms and a self-organizing control loop to retrieve local knowledge to support the decision making on whether to self-organize the substrate network to cope with changes on either traffic loads or resources availability.

As our proof-of-concept, we implemented and validated a piloting system for virtual network management, in which the piloting system acts upon a network failure by either creating a new virtual router capable of handling the traffic demand or migrating an existent virtual router to a distinct host. The main goal of our research is to offer a scalable and robust way to evaluate the effectiveness of our piloting system, also its ability to self-configure its virtual resources on specific scenarios.

The remainder of this paper is structured as follows. Section 2 summarizes the related work on adaptive provisioning. Section 3 brings an overview of the concepts applied in this work. In section 4, we describe the self-organizing piloting model itself, under the multi-agent system (MAS) perspective. We evaluate testbed setup and experimental results in section 5. Finally, section 6 concludes this paper and presents on going and future works.

II. RELATED WORK

The problem of virtual network management can be divided into two main sub-problems. Firstly, there is the Virtual Network Mapping problem [1], which tackles the problem of mapping virtual resources in the physical infrastructure, concerning about the efficient resource mapping while dealing with the simultaneous optimization of the placement of virtual nodes and links on a substrate network. Secondly, assuming that the virtual network has been provisioned, the adaptive maintenance itself comes into play in order to deal with dynamic changes from the variations in the substrates and virtual networks, also related to failures, mobility, migration and maintenance needs. The idea behind the adaptive provisioning is to maintain the original topology and service levels agreements during the virtual network lifetime. The virtual network provisioning involves virtual routers and links management, such as live migration, and virtual router allocation.

Although there are in the literature substantial amount of work dealing with Virtual Network Mapping, from the Network perspective, to the best of our knowledge, there are few studies on adaptive provisioning of instantiated virtual networks to cope with dynamic changes in service demands and resource availability, mainly from the MAS perspective.

We have not dealt with Virtual Network Mapping problem yet, as this paper cover mainly the adaptive provisioning, in which it maintains the virtual network running as efficiently as possible during the virtual network lifetime. In order to solve the virtual network provisioning problem, many approaches have been suggested dealing mostly with (i) virtual node live migration to a distinct host and (ii) virtual link reassignment and setup to preserve the virtual network topology.

For instance, the authors of [13] proposed an autonomic system called Violin, which manages a virtual environment, composed of virtual nodes capable of live migration across a multi-domain physical infrastructure. Moreover, in [14], the authors proposed an adaptive virtual resource provisioning, which brings substrate node agents to cope with failures and severe performance degradation in network virtualization. Furthermore, [15] proposes a distributed self-organizing model to manage the substrate network resources. There also exists approaches dealing with virtual link reassignment, as the system proposed in [16], in which it changes the mapping of virtual links if the load of specific physical links increases more than a certain threshold.

We note that these approaches have treated the virtual network management from a semi-decentralized way, in which the autonomic entities are spread only over substrate nodes. Differently from those highlighted research, the self-organizing model proposed in this paper addresses the management of substrate and virtual resources by taking advantage of the total distribution of the autonomic entities spread all over the network, including virtual networks rather than only substrate nodes.

III. CONCEPTS

Recent research has pointed out network virtualization as a promising technique to deploy future networks that meet current and future users requirements [7][3][10]. The main idea behind network virtualization is of slicing (sharing) physical resources to create multiple virtual networks capable of running its own protocols, services and management solutions. Hence, its main concept relies on the fact that it adjusts the network flow and routes, in an autonomic way, dismissing any kind of central/external control. It aims to maintain the quality of service (QoS) defined in the SLA by controlling the agent's behaviors through norms. Each substrate and virtual node has a piloting agent embedded, which is responsible for capturing local information, reasoning about the collected data by translating simple measurements into significant knowledge, exchanging the acquired knowledge and supporting the instantiation and management of virtual resources. Therefore, it is in charge of managing virtual resources already existent by replacing or migrating overloaded virtual routers, or creating and instantiating new ones.

The main characteristics of the architecture for virtualization, in which a virtual network represents a collection of virtual routers connected together by a set of virtual links to form a virtual topology, which is essentially a subset mapped on the top of the underlying physical network, is regarding to (i) virtual node, and (ii) substrate node. Virtual nodes are hosted on a particular substrate node, in other words, it is a slice of its physical host, comprising CPU, memory RAM, storage capacity, etc. The substrate node, usually composed of physical resources, the resource manager, virtual nodes and virtual links, consists of devices such as router, access points or physical links, and are able to embed many virtual nodes. Further details regarding to virtual network architecture can be found in projects like 4AWARD [4].

A. Network Virtualization Management

Network virtualization management, therefore, involves operations such as instantiating, deleting, monitoring, migrating virtual networks elements and setting its resource-allocation parameters. Such functionalities are what make our piloting system a suitable model for creating and managing multiple virtual networks and, as a consequence, for supporting the pluralist approach for the Future Internet, since it is able to create multiple customized virtual networks at the same time it exhibits a flexible management and a real-time control [17]. An important challenge on network virtualization is the efficient allocation of the physical resources at virtual network mapping and adaptive provisioning stage. To accomplish such efficient use the management of the physical resources should be frequently executed at runtime in order to deal with the variation on the load requests of different users.

B. Multi-Agent System

Recent research has pointed out that providing a distributed self-organized approach for the management of virtual networks is a viable solution to deal with the increase of complexity that network virtualization has brought. We strongly believe that such complexity could be handled by autonomic computing together with the concept of Multi-Agent System (MAS), Norms and Self-* properties.

We propose a virtual network architecture applying the MAS paradigm as a modeling foundation. We have chosen such paradigm mostly because it seems to be particularly suitable to build automatic system, due to some properties of agents, such as autonomy, proactivity, adaptability, cooperating, and mobility. Moreover, the notions of agents and organizations and their decentralized and pro-active nature match well the requirements of large-scale autonomic computing environments. In the other hand, Self-* brings to the piloting model the ability to self-manage its own resources in order to meet polices and user’s requirements.

Accordingly, this paper provides the design and evaluation of a distributed, autonomic and self-organizing system based on MAS and Self-Organizing approaches [13] to ensure distributed negotiation and synchronization between the substrate nodes and virtual resources, so that the virtual and physical nodes are able to handle autonomous and intelligent agents, which exchange messages and cooperate to each other to carry out the distributed virtual network management. We apply such concepts to enable communications between the substrate and virtual agents to gain performance and scalability results of the distributed and autonomic virtual network manager, in order to tackle the virtual network adaptive provisioning challenge.

IV. SELF-ORGANIZING MODEL

Our normative and self-organizing piloting system is based on a distributed algorithm, which embeds an autonomic agent inside every virtual and physical node, disseminated all over the substrate and virtual network. The agents monitor, capture and reason about local information, communicate with each other, cooperating, in order to exchange their local knowledge and decisions feedback, so that each piloting agent turns into an autonomic entity capable of inferring about the global network state and, as a consequence, supporting the core of the self-organizing model to trigger adaptation plans depending on the local knowledge, global inferring and environment condition.

The control loop consists of four main behaviors: collector, analyzer, decision maker and executor, which are executed every so often. Firstly, relevant data from the measurement of availability of resources and network load is acquired by the behavior known as Collector, which is also responsible for storing this local information. Secondly, the Analyzer behavior comes into play in order to translate the measured information and the exchanged knowledge into local knowledge, it also checks if the translated data is in accordance with the quality of service and policies requirements by verifying whether a re-organization of virtual components is required. Afterwards, the Decision Maker thought the knowledge analyzed, might active a self-organization by running an adaptive plan. Such reorganization is activated by the identification of both network overload or lack of resources inside the substrate and virtual node.

The Decision Maker represents the core of our piloting system, since it is responsible for having the virtual network as stable as possible, avoiding any kind of bottleneck, overloaded link, and keeping high levels of quality of Service. It is in charge of translating the analyzed data into an action that might prevent future critical scenarios. The adaptive plans, re-organize the virtual network resources for a more efficient use of them. Such re-organization might be triggered by the detection of an (i) overloaded substrate node, which triggers the replacement of virtual node behavior, responsible for replacing a virtual router by a new one capable of handling the actual demand, and (ii) the identification of lack of physical resources in the virtual node, which causes, in this case, an increasing of the virtual node capacity.

The following components give us a better description of the self-organizing architecture itself:

A. Self-organizing behaviors and the cognition loop

We have designed supporting behaviors, which encompass the ones responsible for the communication task, knowledge sharing, event trigger, and environment sensors and adaptive plans behaviors, which represent the actions taken by the decision-maker. The later has used ontologies for a proper understanding while exchanging actions requests between agents. Such behaviors are divided in three distinct categories: Adaptive plans, Environment sensors and Control loop.

1) Adaptive Plans.

Create Virtual Router: Creating a virtual node can impact the virtual network in two distinct ways. In the first case, if an existent virtual node is running multiple flows from different users request, its piloting agent might trigger an adaptive plan that will tackle the instantiation of a new virtual router to balance the requested flows. From the instantiation on, the flows get balanced between those two virtual nodes and the new virtual router starts to respond some of the requested flows. The second case occurs when it supports the Replace virtual node adaptive plan, explained bellow.

Replace Virtual Router: The replace virtual node plan is triggered in a specific scenario where a virtual router suffers from anomalies and failures such as lack of resource, link overload or when it gets unresponsive. Together with the Create Virtual Router plan, they create a new virtual router, capable of handling the current demand and users requests, and the new virtual router takes place of the failed virtual router. All services and flows must be kept running inside the new virtual router.

Migrate Virtual Router: Migrating virtual nodes across distinct physical hosts is an important functionality of our virtual network manager through the piloting system: It facilitates fault management and load balancing, since we can migrate virtual nodes aiming a better distribution of network load usage. Whether the piloting agent detects a future critical scenario regarding to physical resources, it might trigger the migrate adaptive plan, which is responsible for migrating a running virtual node to a different substrate node, maintaining the same virtual topology and running process.

2) Environment Sensors.

Monitor: Monitors are behaviors coupled to the Collector behavior to handle the different types of data collection; they are responsible for measuring specific information from links and physical resources, which will be later filtered in the Collector actions. The Monitors are composed of (i) devices monitors, which is in charge of monitoring Ethernet and virtual devices, (ii) routers monitors, that monitors the router tables running inside a virtual node and (iii) resources monitor, which will monitor physical and virtual resources such as memory RAM, CPU, IO load, etc.

Informers: Such behaviors are responsible for the communication between autonomic entities- virtual and physical. It sends request of data update, when a virtual node has out dated information related to its neighbors, it caries out knowledge sharing and inform the running events, such as the execution of an adaptive plan.

3) Control Loop

The core of our self-organizing model is composed of an autonomic control loop, in which four behaviors run frequently. Such behaviors come in to action as a machine state, where there exists distinct transitions between the behaviors depending on the state of the piloting agent. The components of the main control loop are:

Collector: The collector is responsible for obtaining information, supervising, monitoring and storing necessary measurement from network links, physical and virtual resources that are of significance to the self-properties of the underlying network. It captures data from both substrate and virtual nodes and also from the neighborhood.

Analyzer: The acquired data, from the Collector, is translated into knowledge by the Analyzer behavior that checks whether they are in accordance with the quality of services and required policies. Accordingly, it verifies the current performance, predicts future critical scenarios and detects events, such as link overload. It is also in charge of activating the decision-making, in case of adaptation need.

Decision Maker: The core of our self-organizing piloting model makes decision according to the knowledge retrieved by the Collector and Analyzer, also from the knowledge exchanged between neighbors nodes. Such decisions depend essentially on the virtual network state, the local knowledge and the norms undertaken. The decision itself is based on the choice of adaptation plans previously designed, such as (i) activating the creation or the delete of a virtual node, (ii) tuning the amount of virtual resources allocated to a specific virtual node, and (iii) migrating a virtual node to a different physical host.

Executor: The executor actually performs the decision previously made by reconfiguring the managed component and communicating with other autonomic managers.

B. Norms

In order to provide a controlled autonomy to the virtual devices, restricting its behaviors to prevent malfunctions and undesirable behaviors, we apply the concept of normative agents. Thus, the proposed model is responsible for adjusting the network flow and routes by controlling the agents' behaviors through norms. Such norms are what makes the piloting model aware of the required polices, quality of services and user's requirements (SLA and QoS).

The normative regulation system is divided into two groups, virtual and physical, in which the piloting agents might (i) abide by the norm, and (ii) violate the norm, it also can restrict access to the network to those agents that violate such norms by delaying their control-loop execution. Such norms are checked every so often through the piloting control loop and it informs the agent's neighbors all norms that have been complied or violated.

C. Self-Organizing Piloting Communication

In order to validate the multi-agent self-organizing piloting system based on a distributed algorithm, which consists of autonomic entities spread all over the virtual network, we first need to evaluate the best mean of communication between the piloting system components. Such evaluation represents an important sub-task of our work, since it supports describing a proper self-organizing model for the context of autonomic virtual network management.

The use of multi-agent communication to represent the piloting system model is essential since our piloting system makes use of this autonomic communication between agents of different substrate nodes to gain advantages over traditional approaches to manage virtual networks. In developing our piloting model and implementing it we have had to address two key issues regarding to the agent communication:

1) *Where do we host the virtual piloting agents?*
2) *How to ensure the reliability in such distribution?*

It has been necessary to simulate different scenarios for different methodologies bearing in mind mainly the efficiency of the piloting prototype. The major advantage the piloting system can offer through the communication approach itself lies in the fact that through the communication between piloting agents from different virtual and physical machines we can ensure that the virtual nodes of the network are in accordance with the rules and policies of the model.

The first issue we addressed by simulating different scenarios where we host the piloting agent under the substrate node and under the virtual node, in which we could be able to automatically reproduce adaptive plans for a given adaptive scenario. The second issue we addressed by defining a suitable set of norms and adaptive plans that incorporates both anomalies from the surrounding environment, and failures of distributed communication cases. By avoiding having to develop extra requirement norms for the case in which the piloting agent is hosted under the virtual node, we reduce the amount of norms, and are able to ensure that the self-organizing model and its adaptive plans are more reliable.

V. Proof of Concept and Initial Experiments

The initial experiments correspond to scenarios where virtual and physical resources, allocated to the virtual network, suffer from anomalies such as substrate/virtual node overload. The piloting system maintains the virtual network topology by selecting new virtual or physical resource to replace or compensate for the affected resource. Two resource failure scenarios are discussed in this paper: (i) virtual node overload, and (ii) substrate node overload. We aim, through the proof of concept, show that MAS and self-* capabilities are feasible approaches to deal with virtual network complexity. Thus, we present adaptive plans to deal with both critical scenarios in order to measure the effectiveness of the proposed piloting system and its ability to self-manage its virtual resources under critical scenarios, by dynamically binding and allocating new resources to maintain the virtual network.

Virtual node overload: When the piloting agent detects that its supported virtual node is about to get overload, it must either request its substrate node it belongs to allocate, at runtime, resources for the virtual node or, if not possible, re-instantiate a new virtual node in the same substrate node to take place of the failed node. The virtual links associated with the affected virtual node should also be reallocated if necessary.

Substrate node overload: When a substrate node, that hosts multiple virtual nodes, fails, gets overloaded or gets unresponsive, all agents hosted on its virtual nodes can detect such failure through keep-alive messages exchanged periodically. Only substrate node agents that belong to the same neighborhood are allowed to collaborate in order to choose alternative hosts where the affected virtual nodes as well as their associated links will be migrated or allocated. Thus, the distributed adaptive migrate router plan is executed for each virtual node hosted inside the substrate node in which the failure was detected.

A. *Experimental Setup*

We carried out preliminary experiments in which the virtual network topology and the initial mapping of the virtual network allocated on the top of the computers A, B and C, are arranged as depicted in Figure 2. The virtual network itself contains two virtual nodes, Va and Vb, hosted inside the substrate nodes A and B respectively. Two sets of flows are running inside the virtual network. Although the simplicity of this setup, it has enabled us to evaluate two different scenarios:

Case 1: For the first scenario we set up a virtual network containing two virtual nodes, Va and Vb, with 256MB of memory RAM and a limited network bandwidth of 5MB/s each. The virtual network runs a data flow associated to the user's request, in which the packets are transmitted over the virtual link starting at the virtual node Va and arriving at the node Vb. The experiment itself consist of generating a large amount of data flow and forwarding them to the virtual node Va, hosted inside the substrate node A, in order to force a network performance degradation. In the meantime, when the virtual node Va is about to dismiss the QoS, due to the overload caused by the traffic generator, its supporting piloting agent detects a possible future failure and acts upon it by creating a new virtual node, capable of handling the current data flow as it has larger network bandwidth, and replacing the affected router with the new one, maintaining the same network configuration and data flow settings. In matter of a few seconds, after a small interference (~3 seconds), the virtual network and the virtual link get stable again, obeying the required norms and, as consequence, the user's request no longer gets fuzzy. This experiment, despite being simple, simulates a scenario in which agents located inside virtual nodes are able to detect high utilization either of the virtual links or virtual resources and decide to update or replace the affected virtual node.

Case 2: The second scenario differs from the first one in that it handles live virtual node migration instead or node replacement. Similar to the previous experiment, we set up a virtual network, maintaining the same topology and capacity, also responding to a user's request. Unlike the first scenario, the purpose of this one is to generate a large amount of data flow and forward the generated packets to the substrate node A, in order to overload the substrate node A instead of the virtual node. After a short while, when the substrate node is

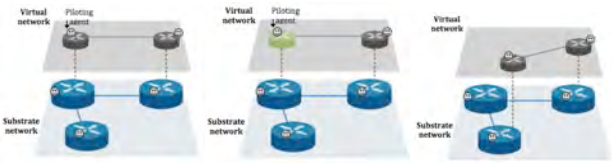

Figure 2. a) The initial Virtual network setup; b) Case1 result; c) Case 2 result.

about to get overloaded due to the traffic generator and, as consequence, next to dismiss the QoS, its supporting piloting agent, in accordance with the required norms, triggers the adaptation plan responsible for the virtual node live migration. The piloting agent responsible for supporting the affected physical machine is then in charge of applying the live migration algorithm on all virtual nodes the affected node hosts. The algorithm itself considers only physical routers from the neighbourhood to support the decision on where to migrate, and the closest one with enough resource availability is the one chosen as destination. In matter of a few seconds, the virtual node is then hosted inside a different substrate node from the neighbourhood, in this case, the substrate node C. This experiment simulates a scenario where agents located inside substrate nodes detect a high utilization of the physical resources and decide to migrate the virtual nodes from affected physical routers.

VI. Conclusion and Future Work

We analyzed the impact and the effectiveness of the self-organizing behavior emerged from our proposed piloting model, in which it is able to control and manage virtual resources in order to address the complexity of an autonomic virtual network management. Through this research we have proposed and validates an autonomic piloting model from the multi-agent system perspective. The experimental results of this paper showed us that it satisfies the model's main goal of automatically reconfigure itself, in order to meet the quality requirements and to improve the network performance whenever it is exposed to a critical scenario.

Through our piloting system, we show that it is possible to design an autonomic virtual network manager by applying MAS approach together with self-* capabilities in order to distribute the responsibility to maintain the virtual network running in accordance with the policies and requirements. Although our current work has focused on piloting system designing, modeling and agent communication, we believe that this general model will certainly support the development of more complex network structure, which will be able to perform live migration of virtual routers supported by agent reputation, virtual link management, normative approach to support the policies and requirements, all those from the MAS perspective. We also highlight that, besides the topics above, virtual link management, process of knowledge acquiring /sharing and live migration, considering agent reputation, are important points that deserve our attention in a near future investigations.

References

[1] J. Nogueira, et. al., "Virtual network mapping into heterogeneous substrate networks", in ISCC 2011, June 2011.

[2] A. Fischer, J. F. Botero, M. Duelli, D. Schlosser, X. Hesselbach, and H. De Meer. "ALEVIN - a framework to develop, compare, and analyze virtual network embedding algorithms", In: Electronic Communications of the EASST, 2011, vol. 37, pp. 1–12.

[3] D. Clark, R. Braden, K. Sollins, J. Wroclawski, D. Katabi, J. Kulik, X. Yang, T. Faber, A. Falk, V. Pingali, M. Handley, and N. Chiappa. "New Arch: Future generation Internet architecture", Technical report, MIT Laboratory for Computer Science and International Computer Science Institute (ICSI), 2011.

[4] "4WARD FP7 project." [Online]. Available: http://www.4ward-project.eu/

[5] I. Fajjari, M. Ayari, G. Pujolle and Hubert Zimmermann. r"Towards an Autonomic Piloting Virtual Network Architecture", In: IFIP International Conference on New Technologies, Mobility and Security - NTMS, IEEE XPlore, 2011, Paris, France.

[6] M. A. Netto, B. S. Neto, E. Cirilo, C. Lucena. "A Self-Organizing and Normative Piloting System". Technical repot, Pontifícia Universidade Católica do Rio de Janeiro, 2013, 05/13.

[7] M. S. Blumenthal and D. D. Clark. "Rethinking the design of the Internet: the end-to-end arguments vs. the brave new world", In: ACM Transactions on Internet Technology, 2001, 1(1):70–109.

[8] N. Egi, A. Greenhalgh, M. Handley, M. Hoerdt, L. Mathy, and T. Schooley. "Evaluating Xen for router virtualization", In: International Workshop on Performance Modeling and Evaluation (PMECT), 2007.

[9] N. Feamster, L. Gao, and J. Rexford. "How to lease the Internet in your spare time", In: SIGCOMM Comput. Commun. Rev., vol. 37, no. 1, pp. 61-64, 2007.

[10] N. Fernandes, M. Moreira, I. Moraes, L. Ferraz, R. Couto, H. Carvalho, M. Campista, L. Costa, and O. Duarte. "Virtual networks: isolation, performance, and trends", In: Annals of Telecommunications, 2011, vol. 66, pp. 339–355.

[11] P. Horn "Autonomic computing: IBM's perspective on the state of information technology, also known as IBM's Autonomic Computing", 2001.

[12] T. Anderson, L. Peterson, S. Shenker, and J. Turner. "Overcoming the Internet impasse through virtualization", In: IEEE Computer Magazine, 2005, vol. 38, no. 4, pp. 34-4.

[13] P. Ruth, J. Rhee, D. Xu, R. Kennell and S. Goasguen. "Autonomic live adaptation of virtual computational environments in a multi-domain infrastructure", In: Proc. IEEE ICAC, 2006, pp.5 -14 .

[14] Ines Houidi, Wajdi Louati, Djamal Zeghlache, Panagiotis Papadimitriou, Laurent Mathy "Adaptive virtual network provisioning", In: Proceedings of the second ACM SIGCOMM workshop on Virtualized infrastructure systems and architectures, 2010, New Delhi, India .

[15] C. Marquezan, L. Granville, G. Nunzi, and M. Brunner. "Distributed autonomic resource management for network virtualization," In: Network Operations and Management Symposium (NOMS) IEEE, 2010, pp. 463-470.

[16] C. Senna, M. Soares, D. Batista, E. Madeira, and N. Fonseca. "Experiments with a self-management system for virtual networks," in II Workshop de Pesquisa Experimental da Internet do Futuro (WPEIF), 2011, Campo Grande.

[17] N. C. Fernandes, M. D. D. Moreira, I. M. Moraes, L. H. G. Ferraz, R. S. Couto, H. E. T. Carvalho, M. E. M. Campista, L. H. M. K. Costa, and O. C. M. B. Duarte, "Virtual networks: Isolation, performance, and trends," To be published in the Annals of Telecommunications, 2010.