Projecting FCAPS to Active Networks

Raouf Boutaba
University of Waterloo
Dept. of Computer Science
rboutaba@bbcr.uwaterloo.ca

Andreas Polyrakis
University of Toronto
Dept. of Computer Science
apolyr@cs.toronto.edu

Abstract – **Active Networks is one of the most promising and discussed trends in the area of Computer Networks. It allows us to program the network nodes to perform advanced operations and computations, and thus, control their behavior. These properties change considerably the scenery in the area of computer networks and, consequently, affect Network Management. Indeed, Active Networks do not only open the way to enhance current management techniques and improve their efficiency, but they also create perspectives to deploy novel ones. This paper attempts to present the impact of Active Networks upon the current Network Management techniques. In order to achieve this, Network Management is examined through the five areas of the FCAPS framework; for each one, the limitations of the current applications and tools are presented, and how these can be overcome by exploiting Active Network properties is discussed. The contribution of this paper is to gather and classify the various ideas found in the literature in this area, combine then and propose some new ones.**

1. INTRODUCTION

The events in the area of computer networks, during the last few years, reveal a significant trend towards open architecture network nodes, whose behavior can be easily controlled, and that can perform advanced computations and tasks. This trend can be verified by a series of facts [1]: emerging technologies and applications that demand advanced computations and perform complex operations; sophisticated protocols that demand access to network resources; and research towards open architecture nodes. Active Networks, a technology that allows flexible and programmable open nodes, seems to be an interesting candidate to satisfy these needs.

Active Networks (AN) [1], [2], [3] is a relative new concept, emerged from the broad DARPA community in 1994-95. In AN, programs can be "injected" into the active devices and affect their behavior and the way they handle data. Active devices no longer simply forward packets; instead, data is manipulated by the installed programs. Packets may be classified and served on per-application or per-user basis. Complex tasks and computations may be performed according to the content of the packets, which may even be altered as they flow inside the network. Hence, AN can be considered active in two ways [2]: First, active devices perform customized operations on the data flowing through them. Second, authorized users/applications can "inject" their own programs to the nodes, customizing the way their data is manipulated. Due to these properties, open-node architecture is achieved, where custom protocols and services can be easily deployed. The network, thus, becomes highly flexible and adaptive to the users and administrators' needs.

Architecturally, AN can be divided into the Discrete (or programmable), and the Integrated (or capsule) approach [1], [4]. The main difference between those two approaches is that in the former, the programs are sent to the active nodes through separate, out-of-band channels, while in the latter, the code is embedded into the data packets. This imposes some differences in the capabilities of the two approaches, which are, however, out of the scope of this document. In both approaches, though, the architectures usually define some basic primitives in the active nodes that provide critical or commonly used functions. Such functions may include packet manipulation, access to the environment of the node, navigation schemes, scheduling, storage and other. Besides, all architectures attempt to address the issue of security and safety. Actually, this is one of the greater concerns in the deployment of AN, since the open architecture of the active nodes makes them very vulnerable to errors or attacks [2], [3], [4]. Several techniques are used in order to ensure security and safety: authentication of the users; safe execution environments and restricted set of operations; inspection of the integrity of the code; restriction to programs downloaded by trusted servers; restricted and authenticated access to resources; etc.

This radical changes that AN introduce can be beneficial for a wide range of applications and tools [2], [4]: Firewalls and proxies can be implemented easily by programs that filter

0-7803-7342-1/01/$17.00 © 2001 IEEE

packets according to their content. Nomadic routers can also be efficiently implemented, by installing code that regenerates the traffic to the users according to their needs (e.g., by encrypting data for users connected to entrusted links, by compressing data for modem users, etc). Multitasking protocols that reserve resources, aggregate data, and perform selective packet dropping can be easily deployed. Multi-path routing is also feasible, by programming nodes to forward packets to different links, according to their content. Several such applications, on several areas, can be envisioned.

Obviously, Network Management (NM) is also affected by AN. This paper attempts an overview of this impact. In order to study this effect, NM is examined through the five areas of the FCAPS framework (Fault, Configuration, Accounting, Performance and Security Management) [5], [6]. For each area, the deficiencies and limitations of current techniques are discussed, and the proposals that may resolve such limitations and enhance them are presented. Note that AN also introduce several new management issues; however, these are not discussed in this document.

2. NETWORK MANAGEMENT

NM deals with monitoring and controlling the network in order to ensure its undisturbed and efficient operation. NM is also concerned with ensuring that the users get the services defined in their Service Level Agreements (SLAs), and with maintaining accounting information for those services.

The monitoring of the network is one of the most crucial tasks for NM, since it provides information on the network status. The collected data can be used to reveal and prevent abnormal and undesirable situations, as well as to configure the network parameters. Several tools and applications for monitoring the network exist. In most cases, the data are collected by polling the devices regularly, nevertheless in some cases the devices themselves may initiate alerts when certain thresholds are exceeded. In order to collect the data, most applications and tools depend on the SNMP protocol. This protocol provides a simple and uniform way to query the network devices. Through SNMP commands, network managers can request values from the Management Information Base (MIB) of the managed device. SNMP protocol also allows managers to set values in the MIB, affecting in this way the behavior of the managed device.

NM may be divided into several functional areas. ISO has distinguished and standardized five major ones: Fault, Congestion, Accounting, Performance and Security Management; this standardization is known as the FCAPS framework [5], [6]:

- **Fault Management** deals with detecting, isolating, fixing and recording errors that occur inside the network.
- **Configuration Management** has to do with maintaining accurate information on the configuration of the network (hardware and software) and controlling parameters that relate to its normal operation.
- **Accounting Management** relates to user management and administration, as well as with accounting and billing for the use of the resources and services.
- **Performance Management** attempts to maximize the network performance. It is strongly related to QoS provisioning and to parameters like resource utilization, delay, jitter and packet loss.
- **Security Management** deals with ensuring security and safety in the network.

3. FCAPS IN ACTIVE NETWORKS

By exploiting AN properties, current NM techniques can be improved and enhanced. This section attempts to give an overview of how the existing limitations can be overcome, either by enhancing current applications and tools, or by deploying novel ones. This analysis takes place with regard to FCAPS. However, before starting analyzing each area individually, the impact of AN upon distributed management is discussed, because this affects all FCAPS areas.

A. Distributed Management

A very important property of AN, as far as NM is concerned, is that it enables the distribution, in various levels, of the management applications and tools. This property has a strong impact upon all the areas of FCAPS. Due to its great importance, most AN Management architectures attempt to address it. Mobile Agents (MAs), programs that travel inside the network and perform several tasks on behalf of the application that generated them, are often used for such purposes.

Currently, networks are monitored and controlled mainly through SNMP commands that read or set variables in the MIBs of the elements. Current MIB implementations have a significant limitation: The MIB for each device is predefined by its manufacturer, and thus, when the management station needs to compute a value that derives from several variables, it has to fetch those variables and compute the result, rather than defining a new variable in the MIB of the element and shift the computations to this element. In AN environments, this issue can be resolved by installing programs into the

active devices, which will create virtual MIBs that extend the existing ones. In this way, Network Managers will be able to define custom variables, which will be stored and maintained by those programs. SNMP commands will be served transparently either by the physical MIBs, or the virtual ones. The agents will also be able to trigger alarms when customized thresholds are exceeded. Work towards this direction, although in a non-active environment, is presented in [7]; however, these ideas can be also implemented in AN, in an easier and more efficient way. MIBlets, proposed in [8], is another approach to the same problem. MIBlets attempt to address the issue of resource partitioning by providing virtual views of the MIBs of the managed devices; however, customized variables can be maintained in them, as well. This framework also supports legacy devices, since the MIBlet controller does not need to run on the managed device.

However, SNMP has several limitations, too. The network is flooded with messages that, in most cases, report no significant change. Aggregation of data is done centrally, which implies that all devices are polled by the management station, consuming significantly more bandwidth than if aggregation took place inside the network. Besides, congested or unreachable parts of the network cannot be efficiently managed. [9] introduces the NetScript architecture and proposes the use of agents that perform SNMP filtering and aggregation. [10] proposes the use of SNMP proxies. These proxies are installed inside the network and each of them is responsible for some devices, active or legacy. SNMP commands are directed to the proxy and transparently forwarded to the appropriate device. Thus, the proxy may relieve the management station from polling each device, by being programmed to collect and aggregate data. Additionally, the proxy may be configured to trigger customized alerts when certain thresholds, concerning either individual elements or parts of the network, are exceeded. Finally, the proxy could also implement virtual MIBs for legacy devices.

By implementing those proposals, current tools will still be able to carry out management tasks. However, the monitoring will become more efficient and scalable (less data has to be fetched from the MIBs, aggregation takes place, polling can be replaced with alarms, processing is distributed) and more accurate (inconsistency in the data and aggregation drawbacks such as the horizon effect [10] can be eliminated, monitoring can be customized on per-device basis and be adjusted to the network conditions, monitoring is feasible in congested or unreachable areas).

Another limitation of current management techniques is that all resolutions are taken centrally. This approach is deficient for congested, noisy or unreachable areas, since the management commands may arrive late or get lost. However, several decisions do not need to be centralized, since they do not need a global view of the network status; instead, they can be made based on data relating to specific elements or small regions. Active nodes can be programmed to take such decisions, allowing in this way the distribution of the resolution centers. In this way, the decisions are moved closer to the managed entities, and thus, they are less prone to be late or get lost. Besides, parts of the network that are unreachable from the management stations may still remain manageable. Finally, AN also allow the easy redistribution and reorganization of those centers, whenever this is desirable (e.g., when the topology or the status of the network changes significantly). These properties are of great importance for self-manageable or self-healing networks, and may have several applications (e.g., ad-hoc networks).

Finally, the ability of the nodes to perform advanced tasks opens the way for novel applications. For example, with current technologies, it is hard or impossible to make a node execute a "ping" or a "traceroute" command to another node, and return the results to the management station, in order to estimate the network status between two points in the network. With AN, though, this is much simplified. More complex applications can also be anticipated. For instance, agents that reside on critical elements or travel inside the network, guarding and maintaining the nodes can be envisioned. Such uses of agents will be discussed later in this document.

B. Active Networks: Enabling technology for distributed FCAPS

Fault Management: Fault Management deals with detecting, isolating, fixing, and logging network faults, i.e., deviations from the normal operation. Its importance is obvious: faults mean network downtime, poor performance and service degradation. Fault Management tools monitor the network in order to detect or predict abnormal situations, which are either fixed automatically, or reported to the network managers, who resolve the problems manually. A usual situation in Fault Management is that primary faults may raise secondary ones, which produce irrelevant messages that may divert the tools or the administrators from determining the actual source of the problem. Hence, event filtering and alarm correlation are necessary to determine important event and to discard useless information.

Fault Management tools monitor the network and attempt to identify known fault symptoms. When such symptoms are detected, their cause it attempted to be determined. While resolving the problem, temporal backup mechanisms may be

triggered, which will attempt to moderate the problem, even partially. After resolving the problem and ensuring that the network is performing well again, the problem and its solution has to be recorded for future use, if a similar situation arises.

The big number of the managed components and their wide physical distribution is one of the primary burdens for Fault Management [5]. AN may tackle this problem by distributing the management centers inside the network, as discussed previously: In this way, the monitoring becomes accurate, faults can be detected rapidly or be prevented, and the responses are prompt and efficient, even in cases where current (centralized) techniques would fail.

Additionally, the use of smart MAs that move transparently and autonomously increases the robustness of the network. As discussed previously, such agents make the network manageable during fault situations and contribute towards self-managed and self-healing networks. Those properties are important during faults, in order to trigger backup mechanisms and resolve the problem.

Besides, the flexibility of AN allows the replacement of generic, rigid protocols and services with more flexible ones, customized for the specific network. In this way, the reaction to faults may become more rapid, precise and efficient. For instance, multirouting protocols could forward traffic through many secondary, low capacity paths during a primary, high capacity link failure, minimizing in this way the impact of the fault.

Finally, in Fault Management, as well as in other areas of FCAPS such as Configuration and Performance Management, predicting and preventing undesirable situations is important. The existing predictive tools use naïve algorithms to foresee the status of the network, mainly due to two reasons: First, the management stations do not have the computing power to simulate the operation of the whole network in detail; second, the real data, necessary to verify or correct the predictions, may delay significantly. In other words, the management station cannot collect, process, simulate, verify and take corrective actions for fairly large networks, due to processing and time constrains. Thus, the current predictive algorithms are simple and take into consideration only a few parameters. However, AN technologies enable the deployment of efficient predictive management, since the computations can be distributed into the whole network. Research towards predictive active management has been presented in [11]: Each node predicts and transmits to its neighbors its future state. The prediction of each node depends on its current state and on the predictions of its neighbors. When a prediction is not verified by the real data, the prediction mechanism re-initializes from a known, consistent state. In this way, hazards, such as faults or congestion, can be predicted with satisfactory accuracy.

Configuration Management: Configuration Management refers both to the process of gathering configuration information and to the result of configuring the network parameters [5]. Hence, Configuration Management tools have to perform several tasks. Discovering new devices and maintaining accurate topology information is one of them. This task, known as Inventory Management, is crucial for re-arranging the resources for optimal performance. However, in some cases it is even necessary for the operation of the network (e.g., in ad-hoc networks). Software Management is another Configuration Management task. It involves the means of controlling (e.g. installing, updating, reconfiguring etc) the software of the network elements remotely. By automating such tasks (e.g., by a batch update), considerable amount of time is saved and the network is kept consistent. Other tasks involve the control of parameters like resource utilization, delay and jitter (although this overlaps with Performance Management) or setting up VLANS that establish independent user domains.

Configuration Management techniques may be enhanced in an AN environment. For instance, MAs can be used for Inventory Management. Those MAs can be used to discover and report changes to the existing configuration. For example, agents could be programmed to propagate DNS updates to the entire network. In networks that seldom change, this property may be of little significance; however, for several types of networks, such as mobile or ad-hoc networks, the rapid propagation of the updates in the configuration information may be crucial for the reconfiguration (and hence, the operation) of the entire network. In such networks, complex protocols and algorithms currently try to ensure the prompt and accurate Inventory Management; this process can be simplified significantly in AN.

MAs can also be used to enhance Software Management. These agents could travel inside the network and check the installed software on the various network nodes and hosts. These agents could transparently perform the installation, reconfiguration and update of the software of the nodes without the interference of the network manager. The network manager, in this case, would just need to launch the agent with some parameters determining the appropriate software configuration for some nodes. Those tasks can be easily programmed in several of the existing architectures, such as in the Phoenix framework [12] or the ADM architecture [13], [14].

AN also facilitate VLAN deployment. VLANs are independent user domains in the same physical network. Practically, this means that network resources are partitioned

and allocated (dynamically or statically) to each group. In AN, access to the resources of the active nodes can be controlled, hence the partitioning of the resources can be easily implemented. An attempt towards this direction is the VAN architecture [15], [16]. In this architecture, the network managers only need to define the user groups and partition the resources, according to their SLAs. The architecture guarantees the independence and the security of the domains. Moreover, the users can manage their own domain by installing custom protocols and services, without any interference with the network managers. MIBlets [8] is another attempt towards the same goal. In this architecture, the resources of the elements are partitioned. The MIBs of the elements are also partitioned, respectively, into virtual MIBs. Each user group has control over its own virtual MIB, allowing it to control its own portion of the resources in each element. In this way, the user domains can be customized.

Accounting Management: Accounting Management deals with accounting and user administration. It comprises tasks such as name and address administration, granting access to use resources and services, defining costs for the resources and services, logging the network use, and charging the users according to it. Directory servers are commonly used to maintain user and accounting policies.

One of the most important tasks that Accounting Management tools carry out is the monitoring of the network usage. Most AN architectures, for security and safety reasons, authenticate the users before any resources are allocated to them or they are allowed to access any service. In this way, the monitoring of the resources is integrated to the network architecture, rather than being an additional function. The range of the accounted resources also increases. Currently, accounting was mainly based on bandwidth consumption, and probably, some priority schemes. With AN, all resource usage, such as bandwidth, CPU, memory, or scheduling priorities, can be accounted. Moreover, the clients can be charged for the services that they use. In this way, the billing is more accurate. Additionally, in some AN architectures, such as the VLAN framework [15], [16] which was discussed previously, the users may demand specific services and resources on specific nodes. This enables fine-grained SLAs that best meet the users' needs.

Finally, as discussed previously, AN may be manageable even when some areas cannot be reached by the management stations. This is crucial for Accounting Management, because those situations usually lead to unreported network use, therefore, loss of profit. Such situations can be prevented in active environments.

Performance Management: Performance Management attempts to keep the network performance in satisfactory levels. It is strongly related to traffic management and QoS provisioning. Performance management tools measure various parameters, such as network throughput, delays and bandwidth utilization, and attempt to control them. Performance Management also involves gathering performance data, establishing baseline performance levels and thresholds, monitoring them, and ringing alarms when those are exceeded. Such thresholds are usually defined in the SLAs between the provider and its clients.

Traffic Management attempts to control and bound parameters such as delay, jitter and packet loss. In AN, the way the devices handle traffic can be easily customized, for each device individually and in a per-user basis. Hence, scheduling and routing, traffic shaping, admission control and priorities can be easily controlled in order to manipulate traffic. For instance, different routing algorithms could be used for different types of traffic, forwarding time-sensitive information through high-speed links. Prediction is also crucial for Traffic Management. For instance, congestion can be avoided by temporarily readjusting parameters such as routing, traffic shaping or admission control. Proposals for predictive management have been discussed previously in this document. Finally, the deployment of QoS services can be easily achieved in AN, since protocols that perform the necessary reservations and computation can be easily installed on active nodes.

Besides, QoS provisioning can be easily implemented in AN. One of the most appealing features of AN is their flexibility in installing protocols. Complex QoS protocols, such as RSVP or qGSMP, could be deployed easily over AN. Any other custom QoS protocol could be deployed easily, too: The reservation of resources and the scheduling algorithms of active nodes can be manipulated in any desired way. The ability to implement QoS protocols without relying on legacy, rigid protocols (such as IP), makes those protocols light-weighted and efficient.

Security Management: Security Management deals with security and safety in the network. Security involves safeguarding the network from active attacks, i.e., attempts to degrade network performance by overloading, reconfiguring, or causing malfunction to the network elements. Safety attempts to ensure the secure exchange of data through the network, by preventing inappropriate access to resources or data, eavesdropping, spoofing, etc. Security Management also deals with user misbehavior, as well as with protecting the network from unintentional damage or access to unauthorized resources. In order to achieve these goals, Security Management has to identify and classify sensitive resources,

conduct threat analysis, define and enforce security policies, check the users' identities and log the use of the resources and services (especially when inappropriate).

Most AN architectures implement modules that relate to security and safety. These modules authenticate the access to the resources; hence, several of the current Security Management tasks are architecturally integrated into these modules. This relieves NM tools from ensuring the enforcement of the policies and SLAs. In addition, the policies can be defined on per-user and per-device basis. Therefore, policies can become stricter and prevent unnecessary access to resources and services.

Apart from the traditional policing, though, AN also allow the deployment of novel security techniques. For example, intrusion detection can become much easier and effective by agents that reside on the sensitive nodes. Attacks, such as the TCP SYN attack (where the attacker floods a TCP port with SYN messages, causing the targeted machine to spend too much resources in serving such requests and, unavoidably, failing to serve normal connections) can also be efficiently detected and prevented. [17] demonstrates how Self-Checking networks, i.e., networks that attempt to check the content of the packets for correctness, can be implemented in an AN environment, and how these can be used to secure nodes from attacks. Moreover, MAs can easily trace back attackers with faked IP, by following backwards the path of the packets. Several such applications can be envisioned and deployed over most of the proposed architectures. For instance, the Phoenix framework [12] allows the existence of MAs that are programmed to perform specific tasks, one of which may be to safeguard the network. Other architectures [18] allow agents to be programmed to start their execution on any node, making, in this way, the existence of safeguarding agents feasible.

4. DEPLOYMENT ISSUES

The previous part discussed about enhancement to the current NM techniques. Although those ideas are theoretically feasible, there are still several issues to be resolved before actually implementing and deploying them. This section discusses some of these issues.

An important feature of a distributed application is the autonomy of its components. Self-depended components ensure the stability and robustness of the network, since the application can work satisfactorily during fault situations. Independence from the network topology is also important. The topology may change due to abnormal situations (faults such as link failures) or during the normal operation of the network (especially in mobile or ad-hoc networks). In both cases, autonomy is crucial for the rearrangement of the components. An attempt to address autonomy and transparency is presented in [13], [14]. In the proposed architecture, the applications are self-controlled as soon as they leave the management station.

However, even if the components of the tools are coordinated centrally, there still exists the issue of what is the optimal number of components and which is the optimal location to place those components. This problem is NP-hard; however several techniques may be used to take a non-optimal, yet satisfactory, resolution. Such techniques include the use of heuristic algorithms, graph theory, enumeration and mathematical programming. This issue is more extensively discussed in [19].

The deployment of MAs is also an issue that has to be addressed. MAs may be created and distributed centrally, or replicated from other agents. Besides, the MAs can be organized in a flat or multilevel hierarchy. Hence, four basic deployment patterns (combinations) can be distinguished [19]. However, those deployment patterns do not imply that an MA is static; on the contrary, an MA may move inside the network. Several movement patterns exist, such as visiting some nodes in a route, visiting all nodes in a circular path, visiting a node and returning back, etc. Besides, the number of MAs may increase or decrease dynamically. The architectures should consider such issues and facilitate them. The work presented in [13], [14], for instance, provides primitives for the most common navigation schemes.

Another important issue, common in all management architectures, is the trade-off between openness and security and safety. Errors in the code of a management application may cause faults or performance degradation that may be very hard to tackle. The use of MAs makes the situation even more complex, because an agent, generating errors, may move and replicate itself uncontrollably. Isolating such agents may be hard; moreover, the same security mechanisms that protect the agents from malicious attacks may refrain managers from eliminating a badly behaved agent. Besides, the openness of the nodes makes them vulnerable to attacks. In order to tackle such issues, most architectures pose limitations to the resource access, with obvious impact upon the efficiency.

The interoperability between different domains is also an issue. NM applications or services may need to cooperate with peer applications in other domains. The propagation of DNS updates to other domains could be an example. Another one is the QoS provisioning for inter-domain applications. Resource reservation for such an application needs to be

performed over all the domains the application spans. In this case, the underlying active methods that implement those protocols in neighboring networks have to co-operate in order to serve the application.

Finally, since AN are expected to emerge gradually, the co-existence of active devices with legacy ones seems inevitable. Management applications and tools should take this fact into consideration.

5. CONCLUSION

This paper attempted to examine the impact of AN upon the current NM techniques. The FCAPS framework was used in order to sort and organize them. Based on this taxonomy, the limitations and deficiencies of the current applications and tools were outlined, and the way they can be improved and enhanced in an AN environment was presented. The contribution of this paper was to gather, classify, combine and present of the various proposals and ideas, as well as propose some novel ones, in order to facilitate future research in this area.

However, the impact of AN upon NM is much greater that enhancing the current applications and tools. AN change radically the scenery in computer networks, and this affects significantly NM. AN raise several new management issues such as how code is transferred into the active nodes and how long it remains there. One may notice that AN make networks resemble to distributed systems. In this case, it would be reasonable to assume that NM heads towards a Network Operating System (similar to a Distributed Operating System), which will act as a distributed middleware between the network resources and the applications. From this point of view, several questions may arise: Is SNMP a protocol suitable for such networks? Is FCAPS able to describe the new needs? Does it need to be revised, by adding new functions to the existing areas, or by adding new areas? Or, FCAPS needs to be totally replaced by a new framework? Several such questions need to be answered, in order to estimate the full impact of AN upon NM.

REFERENCES

[1]. David L. Tennenhouse, Jonathan M. Smith, W. David Sincoskie, David J. Wetherall, and Gary J. Minden; "A Survey of Active Network Research"; *IEEE Communications Magazine,* Vol. 35, No. 1, pp80-86. January 1997.

[2]. Konstantinos Psounis; "Active Networks: Applications, Security, Safety, and Architectures"; *IEEE Communications Surveys,* Vol. 2, No. 1, First Quarter 1999.

[3]. Smith, J.M., Calvert, K.L., Murphy, S.L., Orman, H.K., and Peterson, L.L.; "Activating networks: a progress report"; *Computer* Vol. 32 4, April 1999, Page(s): 32 – 41

[4]. D. L. Tennenhouse and D. J. Wetherall; "Towards an Active Network Architecture"; *Computer Communication Review,* Vol. 26, No. 2, April 1996.

[5]. Heinz-Gerd Hegering, Sebastian Abeck and Bernhard Neumair; "Integrated Management of Networked Systems"; *Morgan Kaufman,* 1999.

[6]. "FCAPS Overview", http://www.fore.com/products/fv/fv_fcaps_wp.html

[7]. Goldszmidt G. and Yemini Y.; "Computing MIB views via delegated agents"; *Systems Management, 1998. Proceedings of the IEEE Third International Workshop on,* 1998, Page(s): 86 -95

[8]. W. Ng, A. Do-Sung Jun, H. K. Chow, R. Boutaba, and A. Leon-Garcia; "MIBlets: A Practical Approach to Virtual Network Management"; *Proc. of IFIP/IEEE IM'99*, Boston, US, June 1999.

[9]. Y. Yemini and S. da Silva; "Towards Programmable Networks"; IFIP/IEEE *International Workshop on Distributed Systems: Operations and Management,* L'Aquila, Italy, October 1996.

[10]. Sharma R., Keshav S., Wu M., and Wu, L.; "Environments for active networks"; *Network and Operating System Support for Digital Audio and Video, 1997., Proceedings of the IEEE 7th International Workshop on* , 1997 , Page(s): 77 -84

[11]. Bush, S.F.; "Active Virtual Network Management Protocol"; *Parallel and Distributed Simulation, 1999. Proceedings. Thirteenth Workshop on,* 1999, Page(s): 182 -192

[12]. Putzolu D., Bakshi S., Yadav S., and Yavatkar R.; "The Phoenix framework: a practical architecture for programmable networks"; *IEEE Communications Magazine* Vol. 38 3, March 2000, Page(s): 160 -165

[13]. Kawamura R., and Stadler R.; "Active Distributed Management For IP Networks"; *IEEE Communications Magazine* Vol. 38 4, April 2000, Page(s): 114 -120

[14]. Kawamura R., and Stadler R.; "A Middleware Architecture for Active Distributed Management of IP Networks"; *Network Operations and Management Symposium, 2000. NOMS 2000. 2000 IEEE/IFIP, 2000,* Page(s): 291 –304

[15]. Brunner M., and Stadler, R.; "Service Management In Multiparty Active Networks"; *IEEE Communications Magazine* Vol. 38 3, March 2000, Page(s): 144 -151

[16]. Brunner, M.; "A Service Management Toolkit for Active Networks"; *Network Operations and Management Symposium, 2000. NOMS 2000. 2000 IEEE/IFIP, 2000,* Page(s): 265 -278

[17]. Jonathan Smith, David Farber, Carl A. Gunter, Scott Nettles, Mark Segal, Walter D. Sincoskie, David Feldmeier, and Scott Alexander; "SwitchWare: Towards a 21st Century Network Infrastructure"; http://www.cis.upenn.edu/~switchware/papers/sware.ps

[18]. Raz, D., and Shavitt, Y.; "Active Networks For Efficient Distributed Network Management"; *IEEE Communications Magazine* Vol. 38 3, March 2000, Page(s): 138 -143

[19]. Antonio Liotta, Graham Knight and George Pavlou; "On the performance and scalability of Decentralized Monitoring Using Mobile Agents"; *In Proc. of IFIP/IEEE DSOM'99,* Zurich, Switzerland, October 1999.