Projecting Advanced Enterprise Network and Service Management to Active Networks

Raouf Boutaba, University of Waterloo
Andreas Polyrakis, University of Toronto

Abstract

Active networks is a promising technology that allows us to control the behavior of network nodes by programming them to perform advanced operations and computations. Active networks are changing considerably the scenery of computer networks and, consequently, affect the way network management is conducted. Current management techniques can be enhanced and their efficiency can be improved, while novel techniques can be deployed. This article discusses the impact of active networks on current network management practice by examining network management through the functional areas of fault, configuration, accounting, performance and security management. For each one of these functional areas, the limitations of the current applications and tools are presented, as well as how these limitations can be overcome by exploiting active networks. To illustrate the presented framework, several applications are examined. The contribution of this work is to analyze, classify, and assess the various models proposed in this area, and to outline new research directions.

The events in the area of computer networks during the last few years reveal a significant trend toward open architecture nodes, the behavior of which can easily be controlled. This trend has been identified by several developments [1] such as:

- Emerging technologies and applications that demand advanced computations and perform complex operations
- Sophisticated protocols that demand access to network resources
- Research toward open architecture nodes

Active networks, a technology that allows flexible and programmable open nodes, has proven to be a promising candidate to satisfy these needs.

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

Architecturally, AN can be divided into discrete (or programmable), and integrated (or encapsulated) approaches [1, 4]. The main difference between those two approaches is that in the former, programs are sent to active nodes through separate out-of-band channels, while in the latter the code is embedded in data packets. This imposes some differences in the capabilities of the two approaches. The programmable approach seems more appropriate for cases where administrators want to modify the behavior of nodes (e.g., by replacing a protocol or installing a new service). The integrated approach seems more efficient when the network applications require advanced computations or customized manipulation of their packets by the network nodes. In both approaches, though, the architecture usually defines some basic primitives in the active nodes that provide critical or commonly used functions such as packet manipulation, access to the environment of the node and navigation schemes, scheduling, and storage.

An important aspect in the deployment of AN is security and safety, since the open architecture of active nodes makes them vulnerable to malfunctions of the code executed on them and attacks [2–4]. Several techniques have been proposed in order to ensure data and code security and operational safety: authentication of users; safe execution environments and restricted sets of operations; inspection of the integrity of the code; restriction to programs downloaded by trusted servers; and restricted and authenticated access to resources.

0890-8044/02/$17.00 © 2002 IEEE

An elegant modeling concept and programming paradigm that allows implementation of several computational features of AN, including distributed processing, is mobile agents (MAs) [2]. MAs are programs that travel inside the network and perform several tasks on behalf of the application that generated them. AN and MA are two technologies with similar motivation, and they can be considered as complementary to each other: AN provide the background to deploy MAs; while MAs are usually the most efficient way to implement an AN functionality.

AN are introducing radical changes to computer networks, making them resemble a distributed operating system. These changes can be beneficial for a wide range of applications and tools [2, 4]. This article examines the impact of AN on traditional management techniques. Note that AN also introduce several new management issues related to management of the additional functionality, such as access to the open architecture nodes, management of the programmable resources of the nodes (CPU, memory, etc.), safety and security of the programs injected by different users, accounting based on the node resources the injected programs consume, and so on; however, these issues are not discussed here.

The structure of this article is as follows. We introduce network management and the Fault, Configuration, Accounting, Performance, and Security (FCAPS) management functional framework. We discuss how network management in general can be enhanced by AN and present the impact of AN on each of the FCAPS areas individually. We demonstrate how service management is affected. We present some deployment issues, and in the last section we conclude this work.

Network Management: Basic Principles

Network management (NM) deals with the monitoring and controlling of 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 maintaining accounting information for these services.

The monitoring of the network is one of the most crucial NM tasks, since it provides information on network status. The collected data can be used to reveal and prevent abnormal and undesirable situations, and configure the network parameters. Most monitoring tools and applications collect the network data by polling the network devices regularly. 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 Simple Network Management Protocol (SNMP). 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 bases (MIBs) of the managed devices. SNMP also allows managers to set values in the MIBs, thus affecting the behavior of the managed devices.

NM may be divided into several functional areas. The International Organization for Standardization (ISO) has distinguished and standardized five major ones: fault, configuration, accounting, performance, and security management, known as the FCAPS framework [5, 6]:

- **Fault management** deals with detecting, isolating, fixing, and recording network (device) faults 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 system parameters that relate to its normal operation.
- **Accounting management** relates to user management and administration functions, as well as with accounting and billing for the use of the network resources and services.
- **Performance management** aims to maximize the network performance. It is strongly related to quality of service (QoS) provisioning and to parameters like resource utilization, delay, jitter, and packet loss.
- **Security management** deals with ensuring the security and safety of data and operations.

Advancing Network Management with Active Networks

The impact of AN technology can be witnessed in many areas of current network management: it affects the implementation of existing management procedures and architectural solutions, and introduces new ones as well. In this section we will describe some general examples of how the use of AN properties can advance these procedures and architectural solutions. We demonstrate in more detail the effect of AN within the concrete context of FCAPS.

An important property of AN, as far as NM system architecture is concerned, is enabling the distribution of management applications and tools. This might happen at various network levels and groupings. This property has a strong impact on all areas of FCAPS. Due to its great importance, most AN management architectures are addressing it. MAs 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, which are defined by their manufacturers, have several significant limitations. For example, 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 the AN environment, this issue can be resolved by installing programs in the active devices that will create virtual extensions of an existing MIB. In this way managers will be able to define custom variables to be stored and maintained by those programs. SNMP commands will be served transparently by either the physical or virtual MIBs.

Another important use of agents is triggering alarms when customized thresholds of monitored parameters are exceeded. Work in this direction, although in a nonactive environment, is presented in [7]. Such ideas can be implemented with AN in an easier and more efficient way. MIBlets, proposed in [8], is another approach in the same direction that supports resource partitioning and hence network virtualization. A MIBlet provides abstract and selective views of the physical network resources as a means to create and manage virtual networks. The abstract view hides the details of the resource interface that are not relevant to the virtual network management system. The selective view restricts the access of the virtual NM system to selective parts of the network resource. The virtual NM system monitors and controls those parts of the resource through the resource representation in the MIBlet. Like MIBs, MIBlets are accessed through a standard SNMP interface. Unlike MIBs, MIBlets are not rigid in that they can be extended dynamically with new data variables and customized by downloadable programmable control agents.

A well-known limitation of SNMP is related to its inability to handle high volumes of processed network data. Very often networks are flooded with messages that, in many cases, report no significant changes. Aggregation of data is done centrally, which implies that all devices are polled by the man-

agement 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. Reference [9] introduces the NetScript architecture that uses agents to perform SNMP filtering and aggregation. Reference [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 get forwarded to the appropriate devices. In this manner, the proxy collects and aggregates data and relieves the management station from polling each device individually. 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 can also implement virtual MIBs for legacy devices.

By implementing those proposals the current tools 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 a per-device basis and be adjusted to the network conditions, and monitoring is feasible in congested or unreachable areas).

Another limitation of the current management techniques is that all management decisions are usually made centrally. This approach is inefficient when the network is congested, or when a part of it is unreachable, since the management commands may arrive late or get lost. However, several decisions do not need to be centralized, since they are based on the state of a single element or a small region of the network. Active nodes can be programmed to make such decisions, thus allowing the distribution of decision centers across the network. In this manner the decisions are moved closer to the managed entities, and thus are less prone to being late or getting lost. Besides, parts of the network that are unreachable from the management stations may still remain manageable. Finally, AN also allow easy redistribution and reorganization of those centers whenever desirable (e.g., when the topology or status of the network changes significantly). These properties are of great importance for self-manageable and 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 instance, agents that reside in critical elements or travel within the network can guard and maintain the nodes. Such use of agents is discussed in the next few sections.

Active Networks for FCAPS

Fault Management

Fault management deals with detecting, isolating, fixing, and logging network faults, that is, deviations from the normal operation of the network. Its importance is obvious: faults cause 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 trouble administration system, which manages network troubles, and if needed passes trouble tickets to field personnel. A usual situation in fault management is that primary faults may raise secondary ones, which produce redundant messages that may divert the tools or administrators from determining the root cause of the problem. Hence, event filtering and alarm correlation are necessary functions to determine the primary event and discard "noisy" data.

Fault management tools monitor the network and attempt to identify known fault symptoms. If such symptoms are detected, they are used in order to determine the cause of the problem. While resolving the problem temporal backup mechanisms may be triggered, which will attempt to moderate the severity of the problem, even partially. After resolving the problem and ensuring that the network is performing well again, the problem and its solution must be recorded for future use if a similar situation recurs.

The big number of 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 order to achieve accurate monitoring, rapid fault detection and prevention, and prompt efficient responses, even in cases where traditional (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 to 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. Thus, reaction to faults may become more rapid, precise, and efficient. For instance, routing protocols could forward traffic through many secondary low-capacity paths during a primary high-capacity link failure, thus minimizing the impact of the fault.

Finally, in fault management as well as other areas of FCAPS such as configuration and performance management, predicting and preventing undesirable situations is important. Existing prediction tools attempt to predict situations based on algorithms that are quite simplistic, mainly for 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 be delayed 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 constraints. Thus, the current predictive algorithms take into consideration only a few parameters. However, AN technologies enable deployment of efficient predictive management, since the computations can be distributed to the whole network. Research toward 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 the predictions of its neighbors. When a prediction is not verified by real data, the prediction mechanism reinitializes 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 to:

- The process of gathering information about the configuration of the network devices and services
- The process and result of configuring the parameters of such devices and services [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 rearranging the resources for optimal performance. Even more, there are cases where inventory management is necessary for the operation of the network (e.g., ad hoc networks). Software management is another configuration management task. It involves the means of control-

ling (e.g., installing, updating, reconfiguring) the software of the network elements remotely. By automating such tasks (e.g., by a batch update), a 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 virtual private networks (VPNs) that establish independent private user networks over a shared public one.

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, rapid propagation of updates in configuration information may be crucial for the reconfiguration (and hence operation) of the entire network. In such networks, complex protocols and algorithms currently try to ensure 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 in 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 easily be programmed in several existing architectures, such as the Phoenix framework [12] or the ADM architecture [13, 14].

AN also facilitate VPN deployment. VPNs are independent private networks built over a shared public 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, partitioning of resources can easily be implemented. An attempt in this direction is the virtual active network (VAN) architecture [15, 16]. In this architecture the network managers just define the user groups and the way resources are allocated to them. The architecture guarantees the independence and security of the domains. Moreover, users can manage their own domains by installing custom protocols and services, without any interference by network managers. MIBlets [8] have the same goal. In this architecture the resources of the elements are partitioned. The MIBs of the elements are also partitioned 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, 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 resources and services, defining costs for resources and services, auditing network use, and charging users according to it. Directory servers are commonly used to maintain user and accounting policies.

One of the most important tasks accounting management tools carry out is monitoring 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. Thus, 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, clients can be charged for the services they use. In this way, the billing is more accurate. Additionally, in some AN architectures, such as the VAN framework [15, 16] discussed previously, users may demand specific services and resources on specific nodes. This enables fine-grained SLAs that best meet 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, and therefore loss of profit. Such situations can be prevented in active environments.

Performance Management

Performance management aims to keep network performance within predefined 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. With AN, the way devices handle traffic can easily be customized on a per-device and per-user basis. Hence, scheduling and routing, traffic shaping, admission control, and priorities can easily be 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 easily be achieved in AN, since protocols that perform the necessary reservations and computation can easily be installed on active nodes.

Besides, QoS provisioning is facilitated in AN. One of the most appealing features of AN is their flexibility in installing protocols. Complex QoS protocols, such as Resource Reservation Protocol (RSVP) or qGSMP, could easily be deployed 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 and rigid protocols (e.g., IP) makes those protocols lightweight and efficient.

Security Management

Security management deals with security and safety in the network. Security involves safeguarding the network from active attacks, that is, 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, and so on. Security management also deals with user misbehavior, as well as 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 users' identities, and log use of resources and services (especially when inappropriate).

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

Apart from 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. Reference [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 trace back attackers with faked IP, by following backward 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.

Active Networks for Service Management

Routing

Routing is an important task of NM since it affects significantly fault, configuration, and performance management. Traditionally, packets were routed according to their destination only. However, in several cases it would be desired to route packets according to other parameters as well. For instance, time-critical data can be routed from faster and less congested paths than best-effort traffic. AN allow the routers to examine the content of the packets and route them according to it. Even more, the applications themselves may program the network to route their data through particular paths (e.g., for security or accounting reasons). Besides, packets can be forwarded to more than one outgoing link. A number of applications could benefit from this property: accounting and statistical analysis of the traffic can be performed online, information can be cached, and load balancing can easily be performed. Besides, we can envision applications that transmit the same data through more than one path, in order to reduce packet loss and delay.

Active routing has also significant applications for ad-hoc networks. In such networks the topology constantly changes, and those changes should be quickly and efficiently reflected to the routing tables. This demands flexible protocols and advanced computations inside the network, something that can be performed by active devices. Work in this area has been conducted and presented in [6].

End-to-End QoS Deployment

AN can be the enabling technology for end-to-end QoS provisioning. Today, two main QoS frameworks are being defined: integrated services (IntServ) and differentiated services (DiffServ). The former is based on per-flow manipulation of the traffic. It can provide strict QoS guarantees but is not scalable to large networks. The latter classifies the packets in some predefined priority classes and handles them according to their priority; however, this architecture cannot ensure that a flow will receive the QoS it demands. An important issue is how these two architectures can cooperate, and what happens to interdomain traffic. AN allow network managers to program the behavior of the ingress/egress routers and provide standard or customizable mappings between different architectures and domains. Besides AN allow network managers to deploy any of these architectures, customize them (e.g., by modifying the prioritization and scheduling algorithms) and possibly deploy novel QoS protocols that will emerge in the future.

Multicasting

Multicasting is the transmission of the same packets to more than one recipient. Multicast is a one-to-many transmission similar to broadcasting, except that multicasting implies sending to a list of specific users, whereas broadcasting implies sending to everybody. AN significantly affect the deployment of multicasting applications by aggregating multicast streams downstream. Routers can be programmed to process a stream (e.g., video) at any network layer, generate other streams, and transmit them downstream. In this way, from a single input stream several output streams can produced (less frames per second, lower resolution or color depth, audio only, encrypted, etc.) for clients with different requirements and/or capabilities.

Transparent Mirroring

Network managers often cache or mirror popular Web or FTP sites requests from a local server inside the (fast) LAN and decongest their (much slower) links to other networks and the Internet. AN can be programmed to transparently redirect users' traffic to local mirrors (or cache). Besides, even when an FTP or Web site is mirrored locally, the network can be programmed to redirect users to the closest one, where the term *closest* can be evaluated according to several criteria (location, congestion, etc.).

On Deployment of Active Networks

The previous sections discussed about how AN can enhance current network and service management techniques. Although those ideas are feasible in theory, there are still several issues to be resolved before actually implementing and deploying them. This section discusses some of these issues.

First of all, an important issue is how to ensure maximum functional and topological autonomy for the distributed components of the active applications. Self-dependent components ensure the stability and robustness of the network, since the application can work satisfactorily during fault situations. Independence from network topology is also important. The topology may change due to abnormal situations (faults such as link failures) or during normal operation of the network (especially in mobile or ad hoc networks). In both cases, autonomy is crucial for rearrangement of components. On the other hand, the existence of autonomous components inside the network raises several issues, such as how to locate and eliminate them if they start malfunctioning. An attempt to address autonomy and transparency issues 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 the optimal number of "active" components or MAs and the optimal location to place those components or MAs. For instance, supposing that we implement SNMP proxies in some nodes, how many proxies do we need, where these should be placed, and which devices

should they control? Such problems are NP-hard; however, several techniques may be used to make nonoptimal yet satisfactory decisions. 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, and so on. 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 the openness of the nodes (programmability) and security and safety of:

- The code installed on the nodes by the users or administrators
- The data flowing through these nodes

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 on resource access, with obvious impact on efficiency.

The interoperability between different domains is also an issue. NM applications or services may need to cooperate with peer applications in other domains. End-to-end QoS provisioning between hosts in different domains is such an example: resource reservation needs to be performed over all the domains in the path between the hosts, which implies the cooperation of the underlying active methods that implement QoS in each domain.

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

Conclusion

This article examines the impact of AN on current NM techniques. The FCAPS framework was used in order to organize them. Based on this taxonomy, the limitations and deficiencies of the current applications and tools are outlined, and the way they can be improved and enhanced in an AN environment is presented. The contribution of this article is to classify, assess, and present the various proposals and ideas, as well as propose some new directions for future research in this area.

However, the impact of AN on NM is much greater than enhancing the current applications and tools. AN change radically the scenery in computer networks, and this significantly affects 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 operating systems, which provide distributed applications with access to network, processing, and storage resources, as well as a means to control them. 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 adding new areas? Or does FCAPS need to be totally replaced by a new framework? Several such questions need to be answered in order to estimate the full impact of AN on NM.

References

[1] D. L. Tennenhouse *et al.*, "A Survey of Active Network Research," *IEEE Commun. Mag.*, vol. 35, no. 1, Jan. 1997, pp. 80–86.
[2] K. Psounis; "Active Networks: Applications, Security, Safety, and Architectures," *IEEE Commun. Surveys*, vol. 2, no. 1, 1st qtr. 1999.
[3] J. M. Smith, "Activating Networks: A Progress Report," *Comp.*, vol. 32 4, Apr. 1999, pp. 32–41.
[4] D. L. Tennenhouse and D. J. Wetherall, "Towards an Active Network Architecture," *Comp. Commun. Review*, vol. 26, no. 2, Apr. 1996.
[5] H.-G. Hegering, S. Abeck, and B. Neumair, *Integrated Management of Networked Systems*, Morgan Kaufman, 1999.
[6] "FCAPS Overview," http://www.fore.com/products/fv/fv_fcaps_wp.html
[7] G. Goldszmidt and Y. Yemini, "Computing MIB views via Delegated Agents," *Proc. 3rd IEEE Int'l. Wksp. Sys. Mgmt.*, 1998, pp. 86–95.
[8] R. Boutaba, W. Ng, and A. Leon-Garcia; "Web-based Customer Management of Virtual Private Networks," *Int'l. J. Net. Sys. Mgmt.*, Special Issue on Web-Based Management, Kluwer Academic/Plenum Press, vol. 9, no. 1, 2001, pp. 67–87.
[9] Y. Yemini and S. da Silva, "Towards Programmable Networks," *IFIP/IEEE Int'l. Wksp. Dist. Sys.: Ops. and Mgmt.*, L'Aquila, Italy, Oct. 1996.
[10] R. Sharma *et al.*, "Environments for Active Networks," *Proc. 7th Int'l. Wksp. Net. and Op. Sys. Support for Digital Audio and Video*, 1997, pp. 77–84.
[11] S. F. Bush, "Active Virtual Network Management Protocol," *Proc. 13th Wksp. Parallel Dist. Sim.*, 1999, pp. 182–92.
[12] D. Putzolu *et al.*, "The Phoenix Framework: A Practical Architecture for Programmable Networks," *IEEE Commun. Mag.*, vol. 38, no. 3, Mar. 2000, pp. 160–65.
[13] R. Kawamura and R. Stadler, "Active Distributed Management For IP Networks," *IEEE Commun. Mag.*, vol. 38, no. 4, Apr. 2000, pp. 114–20.
[14] R. Kawamura and R. Stadler, "A Middleware Architecture for Active Distributed Management of IP Networks," *Net. Ops. and Mgmt. Symp.*, 2000, pp. 291–304.
[15] M. Brunner and R. Stadler, "Service Management In Multiparty Active Networks," *IEEE Commun. Mag.*, vol. 38, no. 3, Mar. 2000, pp. 144–51.
[16] M. Brunner, "A Service Management Toolkit for Active Networks," *Net. Ops. and Mgmt. Symp.*, 2000, pp. 265–78.
[17] J. Smith et al., "SwitchWare: Towards a 21st Century Network Infrastructure," http://www.cis.upenn.edu/~switchware/papers/sware.ps
[18] D. Raz and Y. Shavitt, "Active Networks For Efficient Distributed Network Management," *IEEE Commun. Mag.*, vol. 38, no. 3, Mar. 2000, pp. 138–43.
[19] A. Liotta, G. Knight, and G. Pavlou, "On the Performance and Scalability of Decentralized Monitoring Using Mobile Agents," *Proc. IFIP/IEEE DSOM '99*, Zurich, Switzerland, Oct. 1999.

Biographies

RAOUF BOUTABA (rboutaba@bbcr.uwaterloo.ca) has been teaching networks and distributed systems in the Department of Computer Science of the University of Waterloo since 1999. He conducts research in integrated network and systems management, wired and wireless multimedia networks, and quality of service control in the Internet. He has been program chair of the Technical Committee on Information Infrastructure of the IEEE Communications Society since 2000 and chair of the IFIP working group on network and distributed systems management since 2001. He was the recipient of Premier's Research Excellence Award in 2000.

ANDREAS POLYRAKIS (apolyr@cs.toronto.edu) works as a network engineer at the National University of Athens, Greece. Recently he received his M.Sc. degree from the University of Toronto,Canada. His main area of interest is policy-based networking, but he is also interested in other areas of networking, such as directory-enabled networking, active networks, and QoS provisioning in IP networks.