The Simple Times™

THE QUARTERLY NEWSLETTER OF SNMP TECHNOLOGY, COMMENT, AND EVENTS
VOLUME 8, NUMBER 1 SEPTEMBER, 2000

The Simple Times is an openly-available publication devoted to the promotion of the Simple Network Management Protocol. In each issue, *The Simple Times* presents technical articles and featured columns, along with a standards summary and a list of Internet resources. In addition, some issues contain summaries of recent publications and upcoming events.

In this Issue:

The Simple Times is available as an online journal in HTML, PDF and PostScript. New issues are announced via an electronic mailing list. For information on subscriptions, see page 18.

Editorial

Aiko Pras, University of Twente
Jürgen Schönwälder, TU Braunschweig

This issue of *The Simple Times* focuses on configuration management, which is still considered an important and insufficiently solved problem in many networked environments. Of course, configuration management is not a new problem. So why is there again so much interest in effective configuration management?

The movement from a simple best-effort Internet towards an Internet which supports multiple service levels is currently the main driving force. The technologies which have been developed to implement QoS in the Internet require much more configuration data to be effective. It is therefore of key importance to solve the configuration management problem in order to deploy an Internet which provides multiple service levels.

The articles in this issue of *The Simple Times* discuss various aspects of the configuration management problem. The authors address questions such as whether SNMP can be used to effectively address configuration management problems or whether policy-based configuration management solutions can address the problem appropriately.

An important aspect is the role standards can play to reduce the amount of time and money spent on configuration management. Some people believe that commonly accepted and implemented configuration management standards are the only way to solve the problems. Others believe that device vendors have little interest to support common configuration management standards in order to differentiate their products and because it takes too long to define configuration management standards for new emerging technologies. You may want to keep these different views in mind while reading the articles in this issue.

In the last issue of *The Simple Times*, we asked our readers to fill out an online survey since we considered to publish the first twenty issues of *The Simple Times* as a reference book. A total of 144 readers completed the form. A majority of 77 percent expressed their interest in such a book and the amount of money they were willing to pay was more or less evenly distributed in the range US $10-$50.

Comparing the survey results with the page hit rates for the last issue of *The Simple Times* (18,500 HTML, 7,750 PDF, 700 PostScript), we concluded that only a minority actually participated in our online survey. Of course, the online HTML version is frequently reloaded by the same group of people. Hence, we believe that the PDF and PostScript page hit counts provide a better base for an estimation of the size of *The Simple Times* readership. We also know that some people continue to distribute paper copies of *The Simple Times*. So a reasonable estimate of the size of *The Simple Times* readership is perhaps 10,000. This, however, implies that our survey only reached about one percent of the total readership and thus we concluded that the interest is not big enough to bundle the first twenty issues of *The Simple Times* in a reference book.

Configuration Management Services for the Large Enterprise Network

John Roese, Enterasys Networks

The need for more robust and effective configuration management tools has always been a pressing issue within the enterprise networking space. Network administrators continue to be under tremendous pressure to make their network infrastructure provide more robust and timely services to more users.

Compounding this pressure is the fact that enterprise IT organizations continue to face growing shortages of qualified staff as well as budgets that are not growing at the same pace as the scale of their systems. Examining these trends, it becomes clear that in order to deliver the services required, with the limited resources available, more intelligent management services are needed.

First Some History...

While network management has been part of the overall network solution for over a decade, beyond basic fault isolation the real value of network management tools has been very limited.

Vendors offer many "configuration management" tools and services, but most have failed to provide real cost versus contribution value. What is clear by looking at which tools are used and which tools are put idle after purchase (whether due to inadequate design, limited device support or excess complexity), is that the vast majority of configuration management tools and services have failed to provide real value. This failure is sometimes due to a disconnect between the management interfaces supported by the tools, e.g. MIB modules, and the management interfaces supported by the devices under their management.

Many configuration tools operate on the flawed principle that the configuration action can be manually configured on a device-by-device basis. Most web-based configuration tools fall into this category. They might be interesting when configuring a single switch in a lab environment, but they do not provide value when used to configure an enterprise consisting of thousands of switches.

The tangible result of the inadequacy of configuration management tools can be seen in the development of the preferred configuration interface, the Command Line Interface (CLI). The fact that most seasoned and technical networking professionals prefer to use the cryptic and non-intuitive CLI versus external software based configuration tools defies logic until you look at the historical record of the effectiveness of configuration management tools. This is a simple but valid observation of the current state of configuration management tools.

We need to look at how we might correct this situation as we move into the next generation of configuration management architectures. A deeper analysis shows that a few specific issues come forward as an explanation of what has gone wrong and how we might correct them as we move into the next generation of configuration management architectures:

- *Too much detail.* Building a huge monolithic configuration management service that offers a direct interface into every attribute of a device will overwhelm the network administrator. Too many tools present raw data without converting it into useful management information.

- *Single Manager Assumption.* Because the CLI is commonly used, and split horizon management concepts are becoming the rule of the day, any assumption that one management interface will have absolute control over a device configuration is incorrect. Configuration Management services that assume exclusive control of the systems under management are destined to failure.

- *Lack of support for standard management interfaces.* Many vendors require customers to learn and to use that vendors proprietary tools, such as the CLI, for configuring their devices. Even many of the core standards developed by the IETF and IEEE considered network management a secondary concern. Lack of standardized management increases the burden of staff training, and increases the likelihood of unintended inconsistencies in the configuration of multiple devices. There are two, not necessarily consequential, effects here. Even a well-trained operator has difficulty ensuring consistency of configuration in a large network of heterogeneous devices. This is particularly problematic in the configuration of network security.

- *Lack of support for remote management in network devices.* Many vendors have traditionally failed to provide the ability to configure a device remotely. Many router vendors have chosen not to instrument their routers with open services that can facilitate remote configuration. This is a significant deficiency in configuration management services because routers are such an important factor in core internetworking.

- *Lack of security.* One excuse to avoid developing and deploying configuration management tools has been the lack of secure network transactions to facilitate the service. The standards bodies have been responding to this need for security with new standards, such as 802.1X, IPSec and SNMPv3.

- *Lack of scalability.* Many tools work well in a demonstration or lab setting but often fail when subjected to the scale of an enterprise network. The design flaws occur either as a result of a failure to develop a distributed model for the software or as a result of heavyweight protocols that cannot operate when thousands of devices must be under system control.

What is Required to Succeed – or – What We Have Learned

There is hope for the next generation of configuration management tools. Many of the needed solutions to the above issues have already been developed and are simply awaiting market adoption and integration into better solutions.

As the CTO for a vendor of both network infrastructure and management tools, I have seen the company and its competitors deliver solutions that provided very effective configuration management services. I have also seen them deliver tools that ended up being under utilized by customers. After a decade of experience in this area, several key factors required for successful configuration management services have become apparent:

- *The best solutions are the ones with the simplest interface and most focused mission.* Tools that hide the underlying complexity and raw data from the network administrator – and use elegant and focused interfaces to present information that allows desired tasks to be accomplished easily – are the tools that will get used.

- *Multiple management interfaces must be supported.* Configuration management tools must be designed to expect configuration modifications by other tools that are using other management interfaces. Even where tools are designed to enforce policy-based configuration throughout the network, the need to support exceptions to the rule exists for troubleshooting and other atypical conditions.

- *Standardized management is critical for coordination of network-wide configuration.* For tools to apply configuration consistently throughout a network, the "language" for expressing the configuration must be consistent throughout the network, even across different vendors devices. The strong future emphasis on policy based configuration will require standardized interfaces.

  The standards organizations recognize the importance of consistent management interfaces. The IETF has made discussion of management a requirement of all protocol proposals; IEEE standards typically include a discussion of how a protocol should be managed. Likewise, many consortia submit proposals to the IETF to provide a standard management interface.

- *Interfaces for remote management are necessary to allow for scalable enterprise configuration.* As networks continue to grow, it is no longer acceptable to expect each device to be configured manually. The use of automation reduces the staff burden and the consistency of configuration throughout a large enterprise network.

  The standards organizations recognize the importance of remote configuration, especially using the Internet Standard Management Protocol, SNMP. Most IETF working groups developing new protocols submit SNMP MIBs for standardized management; IEEE has been submitting SNMP MIB proposals to the IETF to accompany IEEE protocols and many consortia submit proposed SNMP MIBs to the IETF to manage the protocol that is their focus.

- *Security must be approached in a holistic manner.* As configuration management tasks become easier to use, they also become more risky if left exposed. There are many simple techniques to control the access to configuration management interfaces that are rarely utilized today. Some of these will become part of an overall holistic security solution in the future.

Logical side-band management channels via 802.1QVLANs, IPSEC tunnels or other means are an easy way to secure the management interfaces of the devices. More fine-grained user-authentication and data-access control schemes will be possible using SNMPv3. To prevent security holes, it will be important that authentication and data access control be consistent between different management interfaces and between devices of different vendors. All aspects of security must be integrated to create a cohesive whole.

- *Scalable configuration management requires both bulk transfers and minimal sized packets.* An efficient bulk transfer mechanism is needed to distribute network- and device-wide configurations during an initialization phase. Minimal sized packets are needed to distribute incremental updates quickly, using little bandwidth. In both the initialization and the update scenarios, connectionless datagram-based protocols are much better suited for large-scale configuration management tasks.

  Although TCP based transport can offer some efficiency in bulk transfers of data, when system-wide configuration is applied in a large enterprise network, the resulting connection setup and maintenance overhead is unacceptable.

  The connection setup and maintenance overhead is especially unsuitable for incremental configuration management. For example, if a manager wants to change the default QoS services in the 802.1p switches, it is important that the thousands of ports affected be touched quickly, and that only the minimum necessary data be transferred.

- *Policy-based networking and configuration management are not mutually exclusive.* In the confusion over the emergence of policy-based networking as a marketing architecture, the focus of network management has been shifted towards a future "nirvana" of user-based, profile-derived IT services. While that end goal is worth pursuing, the typical swing of focus has failed to recognize that this nirvana will be reachable via a combination of both current and future management technologies.

  Policy services can be viewed as falling into two classifications: edge policy and core policy.

  Edge policy deals with dynamic and individual entities in the IT system. These include mapping policy to authenticated users at their ingress port or possibly delivering policy-based on the dynamic content of a connection (and the users involved) through a firewall. Edge policy is somewhat served by protocols such as RADIUS and more recently via COPS-RSVP and even LDAPv3.

  Core policy, however, deals with static and aggregate entities (subnets, traffic classes, system-wide application, etc.) in the IT system. Core policy is almost exactly the same problem space as traditional configuration management. Leveraging existing configuration management services and technologies is an excellent way to effectively deploy a core policy model. When you evaluate ways to deliver core policy services the existing configuration management protocols of SNMPv1 – and especially SNMPv3 – are more than adequate to deploy system wide or multiple device configuration changes over large systems.

- *Leverage what you have before inventing and imposing new protocols.* Any vendor of network infrastructure solutions should be very cautious about adding any new protocol when an existing protocol could be extended and leveraged to solve the same problem. In development, including standards work, it is often too easy to lose sight of the fact that existing protocols and methods already in devices could be leveraged to solve the problem being studied.

  Adding a new protocol to a device typically has a huge impact in the complexity of the device because the change affects much more than just the service it is tied to. For example, new protocols affect the devices operating system because it must allocate resources to the new service at the expense of other on-board services.

  With the move to upgrade existing SNMPv1 management support to SNMPv3 and tri-lingual SNMP stacks, maximizing the use of that protocol is the most logical direction for the development of management interfaces for configuration and policy-based configuration.

Conclusion

With the increase in the complexity of networking services (DiffServ, 802.1X, L4 Classification, Bandwidth control/rate limiting, policy-based routing, 802.1s Multiple Spanning Trees, ...) the need for effective and usable configuration management services is critical. These evolutions have the opportunity to solve many administration and configuration problems in the next enterprise network.

Attention to these lessons will make the possibility of success for next-generation configuration management services much higher:

- We need to understand the historical failings of the predecessors of our current configuration tools.
- We need to solicit and study customers present and future demands for configuration services.
- We must strive to evolve configuration management to address the failings that plagued early implementations.
- The standards bodies (IEEE and IETF) must continue to pay careful attention to the delivery of management interfaces with their emerging technologies.
- SNMPv3 must be deployed to provide a scalable, secure management architecture.
- Core policy services should be delivered over existing protocols, such as SNMP.

As this article has discussed, we must make configuration management tools that better address customers needs, and obviate the need to rely on proprietary CLI-based tools.

Policy-Based Configuration Management: A Perspective from a Network Management Vendor

Lundy Lewis, Aprisma Management Technologies

One of the principle costs associated with maintaining a network is the time spent on reconfiguration. This is not necessarily the replacement of switches, concentrators, bridges, etc., but also adding, moving and changing of users connected to the network. Simply moving a person from one desk on one floor to another desk on another floor may involve changing router ports, routing tables, IP addresses, making desktop changes and even doing some physical rewiring.

The average cost of adds, moves, and changes on today's router-centric networks has been conservatively estimated at $300-500 per user. With the average company moving each user 1.1 times per year, it is clear where many of the support dollars are going. The administrators overseeing these operations would appreciate a reduction in the time it takes to implement such changes.

As the cost of maintaining networks has risen, the network operators able to oversee such operations are becoming harder to find. Many networks are understaffed to meet the increasing demands placed on them. A management system is needed which allows someone who is not a networking expert to perform the mundane operations such as moving users around, adding users, or changing the access constraints of specific users.

For example, the ability to connect to a network will often depend on the location from which a user is accessing the network and the destination a user is trying to reach. It is a complicated job to control access between what could be thousands of users, and it is made more complicated by the fact that the same user might access the system from different locations and might need different levels of access as a function of the location. The possible combinations of access increase quickly because of these "nomadic" users.

Thus, it is desirable to have a management system for controlling, simplifying, and/or automating various aspects of configuration management tasks so that the cost of maintaining the network, and/or using the network, can be better controlled.

Requirements for Policy-Based Configuration Management

What would a framework for implementing policy in configuration management look like? The framework should include (i) a method for defining network domains, (ii) a method for defining policies, (iii) a method for attaching policies to domains, and (iv) a policy driver to monitor objects, execute policies that apply to the objects, and adjudicate among conflicting policies.

Given this framework, one developing an application in a particular network management area may ask the following questions:

- What are the objects in my application?
- What are the attributes of the objects?
- What (if any) are the ways in which I should group the objects?
- Which attributes do I want to monitor and control?
- What policies apply to attributes, objects, or groups of objects?
- Which events will trigger the policy driver?
- What are the actions I want when policies are violated or close to being violated?

With answers to these questions, one has made a good start towards implementing policy in a particular management domain. For example, in the area of configuration management, we desire a configuration application provided with policies that govern:

- The addition of users and resources on the network;
- The deletion of users and resources from the network;
- Changes in resource operating parameters;
- The access rights of users and end stations to databases, applications, and other users and end stations;
- Authentication of users (for security); and
- Tracking the usage of network resources.

Let us describe such a policy framework in a little more detail .

A Generic Policy Framework

Figure 1 shows a generic policy framework. A domain space and a rule space make up policy space, and together provide input to a policy driver. The output of the policy driver is an action space which generally brings about an enforcement of a policy in the network. The network communicates attribute values to the domain space via SNMP polling or traps.

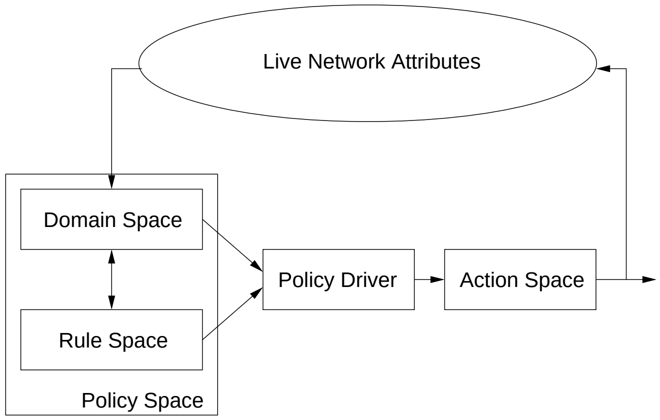

Figure 1: A Generic Policy Framework.

The domain space, at the lowest level of abstraction, consists of objects of interest in the application. Objects are the smallest units in the domain space, and they are defined in terms of their attributes. In access management for example, the objects might be transmissions, where the attributes of transmissions are source Internet Protocol (IP) address, destination IP address, and service type. In fault management, objects might be alarms, where the attributes of alarms are alarm severity, device type, and device location.

At higher levels of abstraction, objects are grouped into domains. A particular grouping principle depends on the objects of interest in the application and the attributes of the objects. Possible domains in access management include all transmissions of a certain service type, or all transmissions whose destination IP address falls into the address block 192.168/16. Possible domains in fault management include all red alarms, or all alarms in Building 2. The domains include both objects and other domains, as one domain may be a member of another domain.

The rule space consists of if-then rules, where the left-hand side of the rule is written in terms of the attributes of objects in the domain space, and the right-hand side is an action. For example, a rule in fault management might be: "If an alarm is red, then forward the alarm parameters to the trouble ticket application." In a security application, an example of a rule is: "If the transmission source is X and the transmission destination is Y, then block the transmission."

The elements of the action space are just the right-hand sides of the rules in the rule space. Actions are dependent on the application. They may include permission or forbiddance of an operation on the network, the modification of attributes in other objects, the display of a console message, or an entry in a log file. For example, there might be just two kinds of actions in fault management: forward an alarm to an external application, or discard the alarm.

It is important to note that a policy in this framework is the attachment of a rule or rule set to an element of the domain space. Thus, a policy is inherently a two-place relation "attaches to." For example, the statement "All kids have to be in bed by 8 p.m." is a rule, but "All kids have to be in bed by 8 p.m. and this applies to you" is a policy. In general, we may say that "rules that do not apply to objects are empty; and objects without rules applying to the them are blind."

The functions of the policy driver are to:

- monitor the attributes of objects in the domain;
- compare the values of attributes with the left-hand sides of rules;
- resolve conflicts when two or more rules are applicable to the same object; and
- execute the right-hand side of a selected rule.

In general, the policy driver is triggered by an event, and takes an element in the domain space as a parameter. In fault management, the policy driver can be triggered by an alarm, where the parameter is just the alarm. In configuration management, the policy driver can be triggered by a device being switched on, and the parameter is the name of the device.

The general operation of the policy driver is as follows:

```
repeat
  for domain element E do:
    1. Collect all domains D of which E is a
       member (either directly or indirectly).
    2. Collect the rules that apply to each domain
       D (if any), plus the rules for E (if any).
    3. Resolve any conflicting rules,
       producing an enforceable rule set.
    4. Execute the action of each rule in the
       enforceable rule set.
```

Note that a single iteration of the policy driver over the policy space may result in actions that change the attributes of elements in the domain. On subsequent iterations of the driver, other policies may be applicable and thus change other attributes.

Conflicts occur when two rules issue two inconsistent actions. Consider Figure 2, which illustrates a simple example of a domain space, a rule space, and a policy space. If the policy driver is triggered for Object 1, and Object 1 inherits policies from parent Domain 1 and grandparent Domain 2, it is possible that Rule 1 and Rule 2 are triggered and that they have inconsistent actions (i.e. y and not-y).

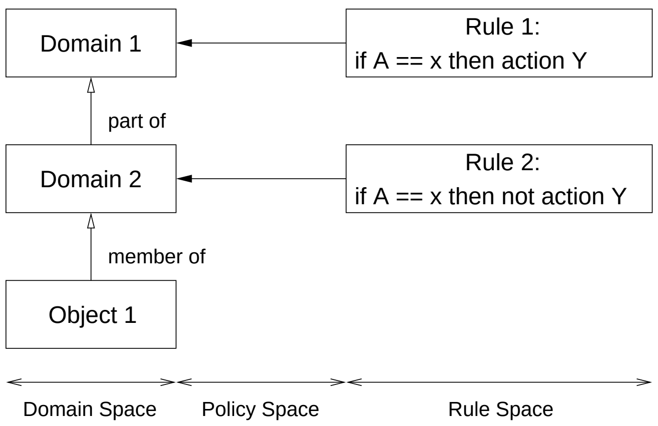

Figure 2: How Conflicts Happen.

Now, the purpose of the conflict resolution strategy is to adjudicate when conflicts occur. Note that conflict resolution strategies are a form of "metapolicy" about policies. There are several ways to specify such strategies in a generic way in order to resolve conflicts. Possible strategies include the following:

Before runtime:

1. Disallow overlapping domains, thereby precluding the possibility of conflicts.
2. Uncover possible conflicts and resolve them via verification/validation algorithms.

During runtime:

1. Select the rule that issues from the most specific domain element.
2. Select the rule that issues from the least specific domain element.
3. Select the rule that satisfies the largest number of conditions.
4. Report conflicting rules to a user and allow the user to adjudicate.
5. Select the rule according to a predefined priority ranking of rules.

For the situation in Figure 2, for example, the strategy "Select the rule that issues from the most specific domain element" would select Rule 1.

Implementing Policy for Device Configuration Management (CM)

In this section we will describe a policy-based configuration manager (PCM) based on the ideas above. The PCM monitors and controls the configuration of network devices with respect to a prescribed policy. The PCM modifies configurations (if needed) under alternative network scenarios, including for example, when a device is added to the network and switched on, when network traffic becomes overstressed, and when an administrator wishes to perform a spot check on the network configuration.

The embodiment of the PCM uses the Spectrum Management System built by Aprisma Management Technologies (formerly Cabletron Spectrum). Spectrum provides the necessary underpinnings for a PCM application, including device modeling, management information base (MIB) compilation, and interfaces for monitoring and controlling devices via SNMP. The system is illustrated in Figure 3.

A live network is monitored and controlled by the network management system, which in turn communicates with a PCM system. The functions of the PCM system are listed in Figure 3. We will describe them in a little more detail below.

Device configuration management in communications networks generally includes the tasks of keeping an

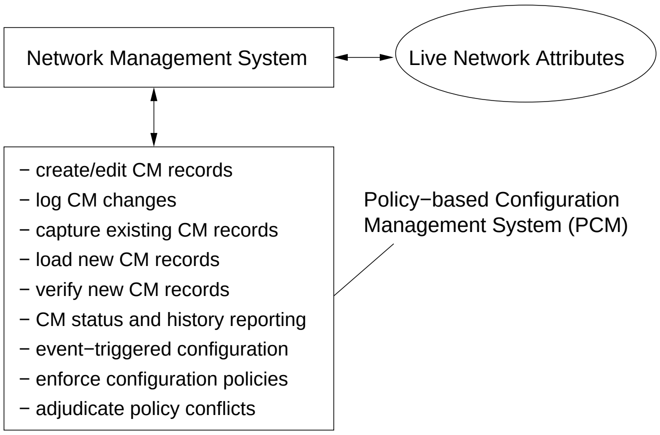

Figure 3: A Policy-Based Configuration Management System.

inventory of network devices, knowing/verifying the configuration of each device, resetting or updating configurations as the need arises, and scheduling configuration changes.

A configuration is a set of particular values of MIB attributes that govern the operational characteristics of a device (e.g., port thresholds, on/off switches, access, security, etc.). Devices that are reconfigured routinely in communications networks are routers, switches, bridges, and hubs.

A configuration record is a copy of a configuration for a particular device, e.g. a Cisco router. The configuration record includes a list of attributes and their corresponding values. A configuration record may be obtained by interrogating a selected device through a template, or by manual construction and editing. The apparatus for doing so exists in the PCM. Note that a configuration record may consist of a list of records that are desired to be in effect for particular devices in a domain. For example, a compound configuration record might consist of a record for SGI workstations and another record for Cisco routers.

A configuration policy expresses a relation between a configuration record and a device; the expression "attaches to" represents this relation. For example, a policy could be that a network administrator wishes a particular configuration record (i.e., rule) to be in force for a particular device (i.e., object), regardless of whether the current configuration of the device is equivalent to the desired configuration record.

As in the generic policy framework in Figure 1, the PCM includes the following components:

- an apparatus for defining a domain space;
- an apparatus for defining configuration records (a rule space);
- an apparatus for attaching configuration records to elements in the domain space to create configuration policies; and
- a policy driver for monitoring and enforcing configuration policies.

The elements in the domain space are network devices such as hubs, bridges, routers, and workstations. Domains are constructed in accordance with an organizational principle by which devices are grouped in the network. In general, network devices may be grouped in any way that serves as an aid in understanding and managing the network. Common grouping principles include grouping with respect to topology, device type, location, managerial domains, and/or the organizational structure of a enterprise network.

The data structure that records domain membership is of the form "X is a member of Y," where X identifies a device or a domain, and Y identifies a domain. For example, Figure 4 shows five individual devices which are grouped in a two-level grouping structure: (i) domains with respect to device type (WS and RTR) and (ii) domains with respect to topology (LAN-1, LAN-2). The arrows in the figure represent "is a member of" relationships.

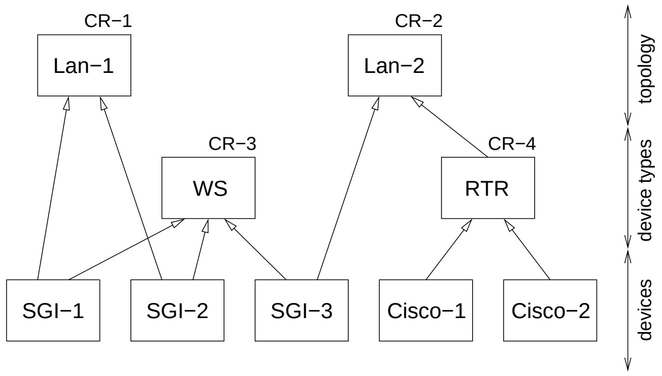

Figure 4: A Sample Structure of a Domain Space.

Configuration policies are attachments of configuration records to elements of the domain space. In Figure 4, a policy is represented by the expression "CR-X " resting on top of an element in the domain space (i.e., CR-1, CR-2, CR-3, CR-4).

The general form of a configuration policy is "X is attached to Y with Ordering Index I if Conditions condition1, condition2, ..." where X is a configuration record, and Y is an element in the domain space. The

Ordering Index and Conditions parameters are optional. The former controls the order in which configurations are loaded into a device, and the latter constrains the enforceability of the attachments. For example:

```
CR-1.1 is attached to Y with Ordering Index 2 if
    segment load (Z) > 40% and
    CR-1.1 is not equal to the current
    configuration of Y.

CR-1.2 is attached to Y with Ordering Index 3 if
    segment load (Z) > 40% and
    CR-1.2 is not equal to the current
    configuration of Y.

CR-1.3 is attached to Y if
    segment load (Z) =< 40%.
```

Here, if the condition `segment load (Z) > 40%` is true and neither CR-1.1 nor CR-1.2 match the existing configuration of Y, then configuration record CR-1.1 is downloaded on Y first, then CR-1.2. Other examples of CM policies include the following:

```
CR-1.4 is attached to LAN-1 if
    "the time is between 8 a.m. and 5 p.m."

CR-1.5 is attached to LAN-1 if
    "the time is between 5 p.m. and 1 a.m."
```

As shown in Figure 5, the function of the policy driver is to monitor objects in the domain space and to enforce configuration policies. The inputs to the driver are a domain space, a trigger issuing from the network management system, and a set of configuration records attached to elements in the domain space.

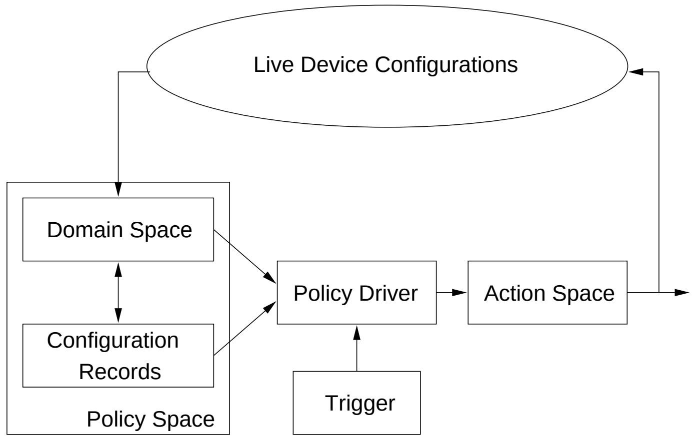

Figure 5: Policy-Based Configuration Management Framework.

The output of the policy driver is an instruction to the network management system, e.g.

- download a configuration;
- issue a notice of conflicting configurations;
- issue a notice of "no action required"; or
- issue a report of the state of overall network configuration.

A note of caution is in order. Looking back at Figure 3, we see that the PCM interacts with network devices via the network management system, requesting certain configuration actions. However, that introduces a big assumption, namely that the network management system can configure devices according to instruction. Unfortunately, not all configuration methods are SNMP-based. Some vendors introduce additional methods such as proprietary command line interfaces that by-pass MIB-based control. Such exceptions, then, may call for additional (ugly) patches to the pure SNMP-based system.

The policy driver may be triggered by one or more of the following events issuing from the network management system:

- a device goes up or down;
- a new device is added to the network;
- the network goes up or down;
- a scheduler triggers the driver; and
- a user manually triggers the driver.

The operation of the driver is a modification of the general operation of the driver described in the Generic Policy Framework section:

```
repeat
  for domain element E do:
    1. Collect all domains D of which E is a
       member.
    2. Collect the CRs that attach to each domain
       D (if any), plus the CRs for E (if any).
    3. From each CR, pick out those that
       attach to the individual devices that
       are members of E.
    4. Resolve conflicting attachments, producing
       one enforceable configuration record (ECR).
    5. Do one of the following (user-selectable):
       (a) Appeal to the administrator with
           conflict explanation/recommendation
           (supervised control).
```

```
(b) Load the ECR into the devices in E and
    report the transaction (unsupervised
    control).
```

Steps 1 to 3 are performed by cycling through a network grouping structure following "is a member of" links and collecting attachments for E recursively with prevention of infinite loops. A method for preventing infinite loops is to keep a record of where you have been and stop if you revisit the same spot. Step 4 is performed by the conflict resolution strategy incorporated into the policy driver. Step 5 is performed by the network management platform.

Configuration conflicts occur when two configuration records issue enforcements for two nonidentical values of a single device attribute. The purpose of the conflict resolution strategy is to adjudicate when this happens. For example, CR-2 issues `ifAdminStatus.5 = up(1)` while CR-4 issues `ifAdminStatus.5 = down(2)`.

There are several strategies one may employ to resolve such conflicts. The PCM provides the following strategies, which are user-selectable:

1. Select the value that issues from the CR which is attached to the most specific network domain.

2. Select the value that issues from the CR which satisfies the greatest number of conditions.

3. If both #1 and #2 issue conflicts, favor #1.

4. Report conflicting values to a user and allow the user to adjudicate among conflicts.

These strategies reflect the common-sense notion that the exception overrides the rule. If this strategy is not acceptable, the burden of conflict resolution rests with the user of the system.

Further Studies

A good sample of early work in policy-based management is the author's paper "Implementing Policy in Enterprise Networks" [1]. One may find additional details of constructing the above policy-based configuration management system in the three patents [2-4], which are available on-line at the United States Patent and Trademark Office.

The IETF Policy Framework Working Group is defining a policy framework and information model to apply to quality-of-service (QoS). The DMTF Information Service Level Agreement Working Group is extending the Common Information Model (CIM) to include policies, rules, and expressions.

Good web sites on policy management in general are at the Imperial College and at the University of Southern California.

Finally, there is a steady body of research in policy-based management reported in the IEEE/IFIP Network Operations and Management Symposium and IEEE/IFIP Integrated Management Symposium. One can go to the IEEE Communications Society web site to find out more, including a brand new "Workshop on Policies for Distributed Systems and Networks". See the "Calendar and Announcements" section at the end of this issue for more details.

References

[1] Lewis, L., *Implementing Policy in Enterprise Networks*, IEEE Communications Magazine, 34(1), January 1996.

[2] Malik, R., Sycamore, S., Tracy, B., *Method and Apparatus for Configuration Management in Communications Networks*, Patent # 5,832,503, United States Patent and Trademark Office, November 1998.

[3] Lewis, L., Malik, R., Sycamore, S., Thebaut, S., Scott, W., Rustici, E., Kaikini, P., *Method and Apparatus for Defining and Enforcing Policies for Configuration Management in Communications Networks*, Patent # 5,872,928, United States Patent and Trademark Office, February 1999.

[4] Thebaut, S., Scott, W., Rustici. E., Kaikini, P., Lewis, L., Malik, R., Sycamore, S., Dev, R., Ibe, O., Aggarwal, A., Wohlers, T., *Policy Management and Conflict Resolution in Computer Networks*, Patent # 5,889,953, United States Patent and Trademark Office, March 1999.

Questions Answered

David T. Perkins, SNMPinfo

What are MODULE-COMPLIANCE and AGENT-CAPABILITIES constructs?

A `MODULE-COMPLIANCE` construct is used to formally write a requirements specification for implementation of objects and/or notifications from one or more MIB modules. An `AGENT-CAPABILITIES` construct is used to formally write an implementation capabilities specification for objects and/or notifications from one or more MIB modules. The definitions of these constructs are in RFC

2580, which is a companion to the other SMIv2 documents - RFC 2578 and RFC 2579.

How is a MODULE-COMPLIANCE construct used?

In IETF standards-track RFC documents, one or more `MODULE-COMPLIANCE` specifications are defined in each MIB module to specify implementation requirements for that MIB module. They provide developers with a precise definition of what support is needed to claim compliance with the RFC. Note that the `MODULE-COMPLIANCE` specifications could be put in another MIB module. However, the current style is to include them inside the MIB modules of standards-track RFCs. MIB modules developed by equipment vendors do not typically include `MODULE-COMPLIANCE` specifications.

Another use of `MODULE-COMPLIANCE` specifications is by equipment users, such as Internet service providers (ISP). ISPs have a set of management applications that are used to manage their networks. These applications use a set of objects and notifications. An ISP can use a `MODULE-COMPLIANCE` construct to specify exactly the objects and notification that must be supported by SNMP agents in the network equipment for the management applications to function. The buyer of equipment for the ISP can use `MODULE-COMPLIANCE` specifications to determine if new equipment can be managed with existing management applications.

How is an AGENT-CAPABILITIES construct used?

A provider of SNMP-managed devices can use one or more `AGENT-CAPABILITIES` specifications to precisely specify the objects and notifications that are supported by an SNMP agent. The `sysORTable` defined in RFC 1907 provides a mechanism for an SNMP agent to list a set of `AGENT-CAPABILITIES` specifications that define what is presently supported. An `AGENT-CAPABILITIES` specification is identified by an OID value, which uniquely defines it for all space and time. (And, thus, once written cannot be modified.) The `sysORTable` allows runtime modification, so that extensible SNMP agents, such as those that use the AgentX protocol, can also be supported.

Are agent capabilities a "good thing"?

What `AGENT-CAPABILITIES` specifications are trying to accomplish is very important and useful. A formal list of supported objects and notifications helps in the understanding between agent developers and all the users of SNMP agents, which include testing groups within the equipment manufacturer (and independent test organizations), product managers writing product description literature, support groups within the manufacturer, evaluators of the equipment, developers of management applications, and integrators that customize generic management applications (such as those that generate trend charts). However, agent capabilities specifications have seen limited usage, and there appears to be a few places for improvement. Even though, they are still quite useful.

The most successful usage of agent capabilities specifications so far is probably in automated agent testing. Given agent capabilities definitions and a few control files, an automated test system can determine if an agent supports everything specified in an agent capabilities specification.

Should I create one or several agent capabilities specifications to describe an SNMP agent?

Typically, when a company starts developing network equipment, there is a single product with a single developer for the SNMP agent. In this case, a single agent capabilities specification seems appropriate. However, over time (if the company is successful), there will be several releases of the agent (with different and probably expanding capabilities), and several products with similar capabilities in the agents. Generally, there will be common chunks of capabilities in each release and across several products.

Back many years ago I worked at a company that had after about five years of development, approximately 15 products with SNMP agents, and over 80 total versions! There was much similarity between versions and between products. There were over 12 engineers developing SNMP agents. Since there was a lot of shared code, the approach that was taken was to have an equivalent of an agent capabilities specification per version of a "library." Note that sometimes the library would be updated, for example to fix a bug or to apply some optimization, without changing SNMP support. So, even without dynamic configuration of SNMP agents, it is still advantageous to have multiple agent capabilities specifications. On the other hand, some developers create a single agent capabilities specification per release of an agent, even if nothing has changed in the support for objects and notifications.

What about agents which can dynamically adapt their capabilities?

Some SNMP agents are able to dynamically adapt their capabilities to the execution environment. This adaptation can either happen at installation time or during runtime. As an example, consider an SNMP agent running on a Linux system. Some capabilities of this agent may depend on the configuration options used by the system administrator when installing the agent. Other capabilities may depend on the presence of certain operating system features which are determined during runtime.

As noted before, agent capabilities specifications are relatively static. Thus, the question is whether an agent capabilities statement indicates exactly what an agent actually supports at any moment or whether an agent capabilities statement defines the maximally possible support. Either interpretation has its problems. The result is that agent capabilities statements are useful in some but not necessarily all environments.

What should MIB compilers do with agent capabilities specifications?

First, lets define the term MIB compiler. The term is simple, but there seem to be many interpretations. A MIB compiler should validate that a collection of MIB modules is syntactically valid, and check that it is semantically valid as far as possible that can be done by a computer program. (Note that much of the semantics associated with a MIB module are specified in text for humans and not for computer programs.) After validation, a MIB compiler generates output in another format that is directly usable or is input to another program. For example, a MIB compiler could create data structure definitions and code stubs to assist in creating an SNMP agent. Note that the data structures and stub code for one agent implementation would probably be of little use for other agent implementations. Alternatively, a MIB compiler could output information that assists a program that graphs values of objects. There are many uses.

So what should a MIB compiler do with an agent capabilities specification? Of course, a MIB compiler should validate agent capabilities specifications like all other SMIv2 constructs. Unfortunately, many MIB compilers "skip over" agent capabilities specifications without any checking to see if they are valid. They do this because implementors could not figure out what to do with the agent capabilities specifications. Other MIB compilers try to use agent capabilities specifications to drive how data structures and stub code is generated for agents. My experience and belief is the opposite. That is, the control mechanisms that are used to specify the implementation characteristics of an agent and used to drive a code generator, should be used to drive the creation of agent capabilities specifications.

What should a MIB browser do with agent capabilities specifications?

A MIB browser is a program that allows a user to examine and change the values of MIB objects. It should also allow the creation of new instances of objects. It is possible for a MIB browser to give hints to its user about which objects can be accessed based on information from an agent capabilities specification. However, the benefit seems to be little. I am not aware of any MIB browser that uses agent capabilities specifications.

I have not seen any agent capabilities specifications from vendors, why not?

There are three primary reasons why equipment vendors do not ship agent capabilities specifications with their devices:

1. It is difficult to get agent implementors to completely document their implementation. It is "extra" work to do after they have finished. And it is often almost impossible to create agent capabilities specifications for old agents without re-engineering the agent implementation.

2. The product marketing managers do not like to publish specifications that appear to look like bug lists, even when an agent conforms to compliance specifications.

3. Since the agent capabilities are identified by OID values, it is difficult for users to find the appropriate MIB modules. One of the most common requests on the SNMP news group is for MIB modules from vendors that define a given OID value.

Book Reviews

The reviews published in this column represent the opinion of the author(s). Please contact the author(s) directly if you want to share your comments. Please contact the editors of *The Simple Times* if you are interested to publish your own book review in this column.

Red Hat Linux Network Management Tools

Reviewed by Aiko Pras, University of Twente

The book starts with the usual explanation of TCP/IP and SNMP. Although the book was published in April 2000, the explanation focuses on SNMPv1 and spends only a few words on SNMPv3. This is a missed opportunity, since the main tools discussed in the book already support SNMPv3 and many readers may have questions related to that version.

After the introduction the book continues with some of the core Linux utilities and tools, such as `arp`, `ifconfig`, `netstat`, `ping`, `tcpdump` and `traceroute`. For each of these tools there are about 10 pages of text; this text may be quite useful for many readers. The following section discusses additional system utilities and tools, such as `arpwatch`, `ethereal`, `fping`, `nmap` and `xtraceroute`. Again, the text is useful, although some of the examples could be shorter without loss of information.

The remaining two third of the book focuses on SNMP. First there is a discussion of 80 pages on MIB-II; unfortunately this discussion is very similar to the text of RFC1213 and similar information is already available from many other sources. After MIB-II, the book continues with a discussion of UCD SNMP and SUN's Solstice Enterprise Agents. Although many readers may find the UCD-SNMP text valuable, it is unclear why a book with the title "Red Hat Linux Network Management Tools" discusses tools for a Solaris platform. Fortunately, the total text on UCD-SNMP is much larger than the text on the SUN agent.

After the discussion on agents, the book continues with some of the most popular Web-enabled tools: `mrtg` and `ntop`. Although there is already many documentation on the Web related to these two packages, the book is still interesting to read. The last chapters of the book concentrate on the Linux `control panel` and on `scotty`'s `tkined`. It is unclear why the discussion on the Linux `control panel` was postponed until the end of the book; it would have been more logical to discuss it earlier. The chapter on `tkined` is quite useful; there are about 50 pages of information which can not easily be found elsewhere. It is not clear, however, why the `Tnm` part of the `scotty` package has not been discussed. It would have been a good replacement for the sections on the SUN agent.

The book is surely a valuable source of information for people relatively new to Linux based network management. What is absolutely missing, however, are references to sources elsewhere. Although all packages that were discussed are contained on the two CDs that accompany the book, the book should at least have included the URLs from where the readers can download the latest versions of these packages. In fact, except for a few examples like UCD-SNMP, the book does not even mention the source and authors of these packages (the publisher even claims that "the software and information on the diskette are the property of The McGraw-Hill Companies"!).

Standards Summary

Please consult the latest version of *Internet Official Protocol Standards*. As of this writing, the latest version is RFC 2700.

SMIv1 Data Definition Language

Full Standards:

- RFC 1155 - Structure of Management Information
- RFC 1212 - Concise MIB Definitions

Informational:

- RFC 1215 - A Convention for Defining Traps

SMIv2 Data Definition Language

Full Standards:

- RFC 2578 - Structure of Management Information
- RFC 2579 - Textual Conventions
- RFC 2580 - Conformance Statements

SNMPv1 Protocol

Full Standards:

- RFC 1157 - Simple Network Management Protocol

Proposed Standards:

- RFC 1418 - SNMP over OSI
- RFC 1419 - SNMP over AppleTalk
- RFC 1420 - SNMP over IPX

SNMPv2 Protocol

Draft Standards:

- RFC 1905 - Protocol Operations for SNMPv2
- RFC 1906 - Transport Mappings for SNMPv2
- RFC 1907 - MIB for SNMPv2

Experimental:

- RFC 1901 - Community-based SNMPv2
- RFC 1909 - Administrative Infrastructure
- RFC 1910 - User-based Security Model

SNMPv3 Protocol

Draft Standards:

- RFC 2571 - Architecture for SNMP Frameworks
- RFC 2572 - Message Processing and Dispatching
- RFC 2573 - SNMP Applications
- RFC 2574 - User-based Security Model
- RFC 2575 - View-based Access Control Model
- RFC 1905 - Protocol Operations for SNMPv2
- RFC 1906 - Transport Mappings for SNMPv2
- RFC 1907 - MIB for SNMPv2

Proposed Standards:

- RFC 2576 - Coexistence between SNMP Versions

Informational:

- RFC 2570 - Introduction to SNMPv3

Experimental:

- RFC 2786 - Diffie-Helman USM Key Management

SNMP Agent Extensibility

Proposed Standards:

- RFC 2741 - AgentX Protocol Version 1
- RFC 2742 - AgentX MIB

SMIv1 MIB Modules

Full Standards:

- RFC 1213 - Management Information Base II
- RFC 1643 - Ethernet-Like Interface Types MIB

Draft Standards:

- RFC 1493 - Bridge MIB
- RFC 1559 - DECnet phase IV MIB

Proposed Standards:

- RFC 1285 - FDDI Interface Type (SMT 6.2) MIB
- RFC 1381 - X.25 LAPB MIB
- RFC 1382 - X.25 Packet Layer MIB
- RFC 1414 - Identification MIB
- RFC 1461 - X.25 Multiprotocol Interconnect MIB
- RFC 1471 - PPP Link Control Protocol MIB
- RFC 1472 - PPP Security Protocols MIB
- RFC 1473 - PPP IP NCP MIB
- RFC 1474 - PPP Bridge NCP MIB
- RFC 1512 - FDDI Interface Type (SMT 7.3) MIB
- RFC 1513 - RMON Token Ring Extensions MIB
- RFC 1515 - IEEE 802.3 MAU MIB
- RFC 1525 - Source Routing Bridge MIB
- RFC 1742 - AppleTalk MIB

SMIv2 MIB Modules

Full Standards:

- RFC 2819 - Remote Network Monitoring MIB

Draft Standards:

- RFC 1657 - BGP version 4 MIB
- RFC 1658 - Character Device MIB
- RFC 1659 - RS-232 Interface Type MIB
- RFC 1660 - Parallel Printer Interface Type MIB
- RFC 1694 - SMDS Interface Type MIB
- RFC 1724 - RIP version 2 MIB
- RFC 1748 - IEEE 802.5 Interface Type MIB
- RFC 1850 - OSPF version 2 MIB
- RFC 1907 - SNMPv2 MIB
- RFC 2115 - Frame Relay DTE Interface Type MIB
- RFC 2571 - SNMP Framework MIB
- RFC 2572 - SNMPv3 MPD MIB
- RFC 2573 - SNMP Applications MIBs
- RFC 2574 - SNMPv3 USM MIB
- RFC 2575 - SNMP VACM MIB
- RFC 2790 - Host Resources MIB
- RFC 2863 - Interfaces Group MIB

Proposed Standards:

- RFC 1611 - DNS Server MIB
- RFC 1612 - DNS Resolver MIB
- RFC 1666 - SNA NAU MIB
- RFC 1696 - Modem MIB
- RFC 1697 - RDBMS MIB
- RFC 1747 - SNA Data Link Control MIB
- RFC 1749 - 802.5 Station Source Routing MIB
- RFC 1759 - Printer MIB
- RFC 2006 - Internet Protocol Mobility MIB
- RFC 2011 - Internet Protocol MIB
- RFC 2012 - Transmission Control Protocol MIB
- RFC 2013 - User Datagram Protocol MIB
- RFC 2020 - IEEE 802.12 Interfaces MIB
- RFC 2021 - RMON Version 2 MIB
- RFC 2024 - Data Link Switching MIB
- RFC 2051 - APPC MIB
- RFC 2096 - IP Forwarding Table MIB
- RFC 2108 - IEEE 802.3 Repeater MIB
- RFC 2127 - ISDN MIB
- RFC 2128 - Dial Control MIB
- RFC 2206 - Resource Reservation Protocol MIB
- RFC 2213 - Integrated Services MIB
- RFC 2214 - Guaranteed Service MIB
- RFC 2232 - Dependent LU Requester MIB
- RFC 2238 - High Performance Routing MIB
- RFC 2266 - IEEE 802.12 Repeater MIB
- RFC 2287 - System-Level Application Mgmt MIB
- RFC 2320 - Classical IP and ARP over ATM MIB
- RFC 2417 - Multicast over UNI 3.0/3.1 / ATM MIB
- RFC 2452 - IPv6 UDP MIB
- RFC 2454 - IPv6 TCP MIB
- RFC 2455 - APPN MIB
- RFC 2456 - APPN Trap MIB
- RFC 2457 - APPN Extended Border Node MIB
- RFC 2465 - IPv6 Textual Conventions and MIB
- RFC 2466 - ICMPv6 MIB
- RFC 2493 - 15 Minute Performance History TCs
- RFC 2494 - DS0, DS0 Bundle Interface Type MIB
- RFC 2495 - DS1, E1, DS2, E2 Interface Type MIB
- RFC 2496 - DS3/E3 Interface Type MIB
- RFC 2512 - Accounting MIB for ATM Networks
- RFC 2513 - Accounting Control MIB
- RFC 2514 - ATM Textual Conventions and OIDs
- RFC 2515 - ATM MIB
- RFC 2558 - SONET/SDH Interface Type MIB
- RFC 2561 - TN3270E MIB
- RFC 2562 - TN3270E Response Time MIB
- RFC 2564 - Application Management MIB
- RFC 2576 - SNMP Community MIB
- RFC 2584 - APPN/HPR in IP Networks
- RFC 2591 - Scheduling MIB
- RFC 2592 - Scripting MIB
- RFC 2594 - WWW Services MIB
- RFC 2605 - Directory Server MIB
- RFC 2613 - RMON for Switched Networks MIB
- RFC 2618 - RADIUS Authentication Client MIB
- RFC 2619 - RADIUS Authentication Server MIB
- RFC 2667 - IP Tunnel MIB
- RFC 2662 - ADSL Line MIB
- RFC 2665 - Ethernet-Like Interface Types MIB
- RFC 2668 - IEEE 802.3 MAU MIB
- RFC 2669 - DOCSIS Cable Device MIB
- RFC 2670 - DOCSIS RF Interface MIB
- RFC 2677 - Next Hop Resolution Protocol MIB

- RFC 2720 - Traffic Flow Measurement Meter MIB
- RFC 2737 - Entity MIB
- RFC 2742 - AgentX MIB
- RFC 2787 - Virtual Router Redundancy Proto. MIB
- RFC 2788 - Network Services Monitoring MIB
- RFC 2789 - Mail Monitoring MIB
- RFC 2873 - Fibre Channel Fabric Element MIB
- RFC 2851 - Internet Network Address TCs
- RFC 2856 - High Capacity Data Type TCs
- RFC 2864 - Interfaces Group Inverted Stack MIB
- RFC 2895 - RMON Protocol Identifier Reference
- RFC 2925 - Ping, Traceroute, Lookup MIBs
- RFC 2932 - IPv4 Multicast Routing MIB
- RFC 2933 - IGMP MIB
- RFC 2940 - COPS Client MIB
- RFC 2954 - Frame Relay Service MIB
- RFC 2955 - Frame Relay / ATM PVC MIB
- RFC 2959 - Real-Time Transport Protocol MIB

Informational:

- RFC 1628 - Uninterruptible Power Supply MIB
- RFC 2620 - RADIUS Accounting Client MIB
- RFC 2621 - RADIUS Accounting Server MIB
- RFC 2666 - Ethernet Chip Set Identifiers
- RFC 2707 - Print Job Monitoring MIB
- RFC 2896 - RMON Protocol Identifier Macros
- RFC 2922 - Physical Topology MIB

Experimental:

- RFC 2758 - SLA Performance Monitoring MIB
- RFC 2786 - Diffie-Helman USM Key MIB
- RFC 2934 - IPv4 PIM MIB

IANA Maintained MIB Modules

- Interface Type Textual Convention
  ftp://ftp.iana.org/mib/ianaiftype.mib
- Address Family Numbers Textual Convention
  ftp://ftp.iana.org/mib/ianaaddressfamilynumbers.mib
- TN3270E Textual Conventions
  ftp://ftp.iana.org/mib/ianatn3270etc.mib
- Language Identifiers
  ftp://ftp.iana.org/mib/ianalanguage.mib
- IP Routing Protocol Textual Conventions
  ftp://ftp.iana.org/mib/ianaiprouteprotocol.mib

Related Documents

Informational:

- RFC 1270 - SNMP Communication Services
- RFC 1321 - MD5 Message-Digest Algorithm
- RFC 1470 - Network Management Tool Catalog
- RFC 2039 - Applicability of Standard MIBs to WWW Server Management
- RFC 2962 - SNMP Application Level Gateway for Payload Address Translation

Experimental:

- RFC 1187 - Bulk Table Retrieval with the SNMP
- RFC 1224 - Techniques for Managing Asynchronously Generated Alerts
- RFC 1238 - CLNS MIB
- RFC 1592 - SNMP Distributed Program Interface
- RFC 1792 - TCP/IPX Connection MIB Specification
- RFC 2593 - Script MIB Extensibility Protocol

Recent Publications

Directory Enabled Networks

- Authors: John Strassner <jstrassn@cisco.com>
- Publisher: Macmillan Technical Publishing
  http://www.macmillantech.com/
- ISBN: 1-57870-140-6
- Available: September, 1999

This 700+ pages book serves as an authoritative guide for the Directory Enabled Networks (DEN) technology. Directory enabled networks store network and service management information in a common repository in an agreed-upon data format. The book first introduces the object-oriented concepts that provide the foundation for the Common Information Model (CIM) defined by the Distributed Management Task Force (DMTF). The second part of the book explains in detail how the CIM model has been extended to realize the DEN idea. The last part of the book describes DEN extensions to support policy-based networking and it presents some case studies how DEN technology is used by various products.

Inter-Domain Management: Specification Translation & Interaction Translation

- Author: The Open Group
  http://www.opengroup.org/
- Publisher: The Open Group
  http://www.opengroup.org/
- ISBN: 1-85912-256-6
- Available: January, 2000

This document contains the specification of technologies that enable interworking between OSI, SNMP and OMG CORBA-based management systems, also known as the Joint Inter-Domain Management (JIDM) standards. The first part of the book introduces the Specification Translation rules which describe how GDMO and SMIv2 data definitions are mapped to CORBA IDL definitions. The second part of the book covers the Interaction Translation rules which define how CMIP and SNMP protocol operations can be mapped to CORBA operations. The Open Group adopted both translation rules as Technical Standard C802.

Red Hat Linux Network Management Tools

- Author: Steve Maxwell
- Publisher: McGraw-Hill Professional Publishing
  http://www.mcgraw-hill.com/
- ISBN: 0-07-212262-5
- Available: February, 2000

This book discusses some of the main network management packages that are available for the Red Hat as well as other Linux platforms. It explains how to use utilities like `UCD-SNMP`, `scotty`, `mrtg`, `ntop` and the Linux `control panel`. These utilities, as well as many others, are included on two CD-ROMs. (For more information, see the book review elsewhere in this issue.)

Calendar and Announcements

IETF Meetings:

- 49th Meeting of the IETF
  December 11-15, 2000, San Diego, CA, USA
- 50th Meeting of the IETF
  March 19-23, 2001, Minneapolis, MN, USA
- 51st Meeting of the IETF
  August 5-10, 2001, London, England

Conferences and Workshops:

- Workshop on IP-oriented Operations and Management (IPOM 2000)
  September 4-6, 2000, Cracow, Poland
- Workshop on Distributed Systems Operations and Management 2000 (DSOM 2000)
  December 4-6, 2000, Austin, Texas, USA
- Workshop on Policies for Distributed Systems and Networks (Policy 2001)
  January 29-31, 2001, Bristol, England
- Integrated Network Management (IM 2001)
  May 14-18, 2001, Seattle, WA, USA

Exhibitions and Trade Shows:

- NetWorld + Interop Atlanta
  September 25-29, 2000, Atlanta, USA
- NetWorld + Interop Paris
  November 6-9, 2000, Paris, France
- NetWorld + Interop Sydney
  March 7-9, 2001, Sydney, Australia
- NetWorld + Interop Las Vegas
  May 6-11, 2001, Las Vegas, USA
- NetWorld + Interop Tokyo
  June 4-8, 2001, Tokyo, Japan

Publication Information

Editors	
Aiko Pras	University Twente
Jürgen Schönwälder	TU Braunschweig
Editorial Board	
David Harrington	Cabletron Systems Inc.
Keith McCloghrie	Cisco Systems Inc.
Bob Natale	ACE*COMM
David Perkins	SNMPinfo
Randy Presuhn	BMC Software Inc.
Steve Waldbusser	
Bert Wijnen	Lucent Technologies
Contact Information	
E-mail	`st-editorial@simple-times.org`
ISSN	1060–6068

Submissions

The Simple Times solicits high-quality articles of technology and comment. Technical articles are refereed to ensure that the content is marketing-free. By definition, commentaries reflect opinion and, as such, are reviewed only to the extent required to ensure commonly-accepted publication norms.

The Simple Times also solicits terse announcements of products and services, publications, and events. These contributions are reviewed only to the extent required to ensure commonly-accepted publication norms.

Submissions are accepted only via electronic mail, and must be formatted in HTML version 1.0. Each submission must include the author's full name, title, affiliation, postal and electronic mail addresses, telephone, and fax numbers. Note that by initiating this process, the submitting party agrees to place the contribution into the public domain.

Subscriptions

The Simple Times is available in HTML, PDF and PostScript. New issues are announced via an electronic mailing list. Send electronic mail to

```
st-request@simple-times.org
```

with

```
subscribe simple-times
```

in the body if you want to subscribe to this list. Back issues are available via *The Simple Times* Web server:

```
http://www.simple-times.org/
```