State Decision Of Managed Objects Using Object Hierarchy And Priorities Among State Values

Seong Ik Hong, Korea Telecom

Abstract

States of the resources are very essential because they represent the basic operability, usage and administration of network elements. But the complex relationships among resources make it very difficult to know whether the resources are really available or not. We propose a simple state determination scheme using object hierarchy and the priorities among state values according to their natural born properties. We convert the priorities into numeric values and get the result state with arithmetic comparison.

Index terms

State Management, Properties in State Values, Ojbect Hierarcy, TMN

A. Introduction

Generic states which represent the basic availability of network resources become critical as networks evolve, become larger and more complex. States can be used as the standardized metrics of physical availability, usage and administration for heterogeneous resources such as ATM(Asynchronous Transfer Mode) switches, IP routers and DSLAM(Digital Subscriber Line Access Multiplexer)s. Layered network applications necessitate exact state of resources, but the amount and complexity of state informations of network make it very difficult that network management systems deal with such enormous information data efficiently.

Now, three generic states are defined in ITU-T Recommendation X.731 [1]. The operational state represents whether or not the resource is physically installed and working. The usage state represents if the resource is actively in use at a specific instant, and if so, whether or not it has spare capacity for additional users at that instant. The administrative state is used to permit or prohibit against using the resource, imposed through the management services.

Until now, studies on these states have been concentrated on the operational state by the name of fault identification, event correlation and alarm surveillance. Fault localization nalysis results were shown based on graph based network model by a heuristic algorithm[2]. Another study to correlate burst events is coding approach with a causality graph, correlation graph and correlation matrix[3]. And many other studies[4],[5] have been done but they have focused on the correlation of symptoms and problems of faults only.

Since faults are inevitable, quick detection, identification and recovery are crucial to make the systems more robust and their operation more reliable. But the usage monitoring and redirection of the resource which has faults, and the administrations not to use those resources are the final goals of efforts of alarm correlation.

The purpose of this work is to propose a management scheme of all the generic states for MO(Managed Object) in general networks which are controled by TMN based management system. We gather all the state informations and apply them to the corresponding objects. And then we decide the result state of the object using object hierarchy and state definitions. And so we can determine the state of an object not knowing which are root problem events and which are symptoms.

The work is organized as follows: In section B, we describe a network resource hierarchy and layered management. In section C, we analyze state definitions. In section D, we propose a different scheme to determine the state of a network resource and show an example of state transitions. Section E concludes this work with a summary of the results. .

B. Network and Tmn

1) *Managed Object Hierarchy*

A telecommunications network is a complex one which can be described in a number of different ways. We defined MOs to represent every resource of the network and applying the partitioning concept of ITU-T recommendation G.805 [6], we can abstract several low level MOs into one high level MO to form a hierarchical architecture. In this hierarchy, state changes of MOs must be affected to its lower level MOs and associated MOs. MOs have relationships with one another such as container-contained or association. This paper demonstrates a method for state management by hierarchical naming structure and arithmetic comparison of priorities of states. A container-contained relationship is represented by the MO's naming scheme as shown in Figure 1. We can determine priorities among states due to their definitions, so that makes it very easy to say an MO's state by a simple arithmetic comparison.

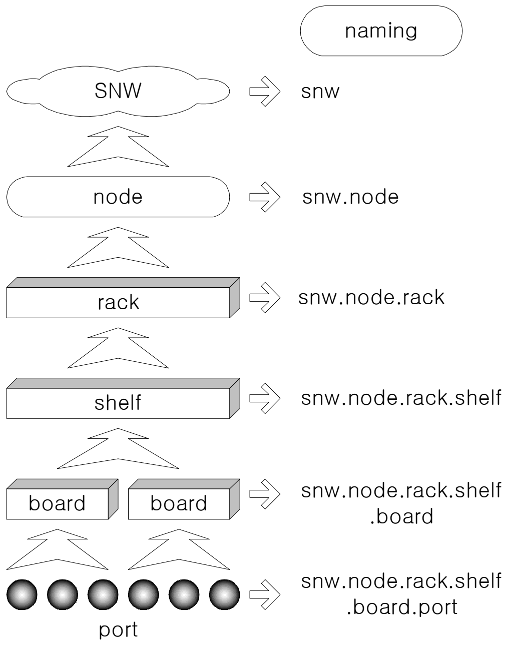

Figure 1. Object Hierarchy

2) TMN Logical Layer

ITU-T recommendation M.3010[7] describes TMN Logical Layered Architecture as shown in Figure 2. To deal with the complexity of telecommunication management, the management functionality may be considered as forming logical layers. Every layer management system has its own specific information and exchange data to establish basic functionalities. EMS(Element Management System) controls NE (Network Element)s directly. A boundary of EMS is determined by its regional area or physical class of resources. Nation-wide NMS(Network Management System) such as ATM NMS, IP NMS gathers all information about physical resources from EMS and make network view information. It controls all the network elements through EMSs. SMS(Service Management System) which has human interface controls service informations based on the network data through all NMSs. Its role is divided into a few parts like service configurations, fault managements and so on. In each part, there are operators to handle user-request and network problems. In this layered architecture, state changes come from just one layer according to their definitions and must be propagated to adjacent layer management systems to make an actual effect as shown in Figure 2.

Each system construct its own database for the resources. Provisionings such as port activation and connection set-ups are done by top-down approach. SMS operators should have service contract of customers and know available resources from NMS. When NMS and EMS get operations from SMS, they do their work such as routing, access profile check and keep the informations in their storage.

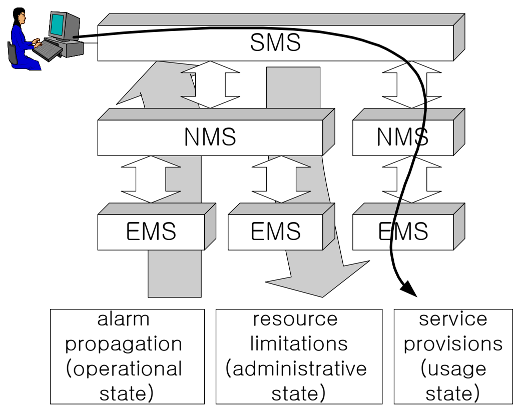
SMS
NMS
NMS
EMS
EMS
EMS
alarm propagation (operational state)
resource limitations (administrative state)
service provisions (usage state)

Figure 2. Logically Layered System

C. STATE ANALYSIS

1) Operational State

The operational state is a problem of operability by physical fault. It has two possible values: disabled and enabled. When a resource ceases to exist, but there is still a managed object maintaining state attributes about that resource, then the operational state will be disabled. Enable event from the disabled state to the enabled state consists of an action being taken to render the resource partially or fully operable and makes the resource enabled state as shown in Figure 3. The management system can only gather information about the operational state of a managed object; i.e. the operational state is read-only in nature. This state depends on the state of NE, so it must be propagated from EMS to NMS and SMS as shown in Figure 2.

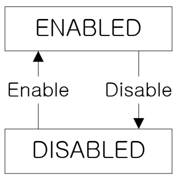

Figure 3. Operational State Transition Map

2) Usage State

The usage state is a problem of usage of MO, which has three possible values: idle, active and busy. If a resource has no users, the usage state becomes idle. When a new user comes to the resource, after the event, if the resource still has sufficient operating capacity to provide for additional users, the usage state becomes active; however if no capacity exists, it becomes busy as shown in Figure 4. The usage state is read-only, too.

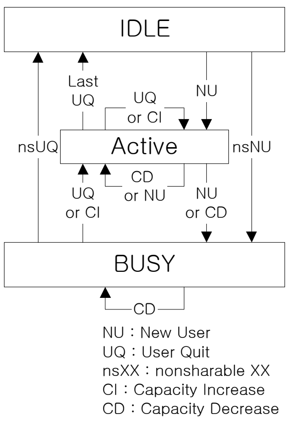

Figure 4. Usage State Transition Map

3) *Administrative State*

The administration of managed objects operates independently of the operability and usage of managed objects and is described by the administrative state attribute, which has three values. These are called locked, unlocked and shutting down as shown in Figure 5. These states limit the access to the resources. Operators can replace the physical resources or protect the existing services using these states. The unlocked state puts no limitation to the resource. In shutting down state, no new service can be provided to the resource but existing services can be released. It can occur only if the managed object's administrative state is unlocked. But at that moment, if the resource has no users, the administrative state becomes locked. In the locked state, we can make no changes to the resource.

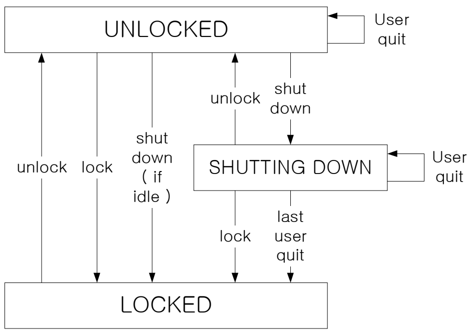

Figure 5. Administrative State Transition Map

D. PROPOSED STATE DECISION SCHEME

1) *A Common Rule*

As a consequence of the previsously stated definitions, we can catch some common properties of the state using their hierarchy. Mentioning the operational state first, MOs cannot survive if the states of their higher objects have gone to the disabled state. So if a managed object has at least one parent object which is disabled, its operational state becomes disabled. But it doesn't matter whether the states of its children are disabled or not. To decide a usage state, we don't have to know another object's state but the states of its lowest children (bottom objects), ports because all the services are directly provided by the bottom objects. Higher objects are abstractions or aggregations of bottom objects in fact. Administrative states are top-down read-write states that operators may set. If an operator intends to set the state and limit the access of specific resources, it means he or she likes to limit all the services related to that object. So it depends on the states of higher objects and is shown in Table 1.

Table 1. Dependency on Object Hierarchy

State\Dependency	Own state	Parent state	Child state
Operational state	O	O	X
Usage state	O (if bottom)	X	O (bottoms)
Administrative state	O	O	X

And we can catch second important property, the priorities among state values. There are two values in operational state. We can consider disabled state as stronger. For example, assume I were a port. I couldn't survive my dead board(one of the parents in the object hierarchy). If I were dead, the enabled board couldn't revive me. In the administrative state, the locked state is the strongest, shutting down is next and the unlocked state is the weakest according to the degree of limitations. As usage state depends on the states of the bottom objects, we have to join all children's states. If all my bottom states are idle, my state is idle. If all of them are busy, I'm busy. In all the other cases, I'm active because I have users but I can accept another user of resources that are not busy. So, priorities can be expressed as follows.

Op : Disabled > Enabled

Ad : Locked > Shutting Down > Unlocked

Us : If(Idle && Idle && ...) → Idle
Else if (Busy && Busy && ...) → Busy
Else,Other Mixture(Idle, Active, Busy)
→ Active

Here, we can make a common rule to decide the state of an object using the dependency on object hierarchy and the priorities among state values. In operational state and administrative state, we are indifferent to the states of the children. We can put arithmetic values to the state due to their priorities. These two properties make a state of one object as shown below.

resultOpState = max(MyOpState,
father'sOpState,
grandFa'sOpState,
...)
with Disabled(1), Enabled(0)

resultAdState = max (MyAdState,
father'sAdState,
grandFa'sAdState,
...)
with Locked(2), ShuttingDown(1), Unlocked(0)

In case of the usage state, it is different from above two states. We cannot put arithmetic priorities to their state values. Instead, we can get result state with logical AND operations of all the states of bottom objects.

ResultUsState = Idle, if(all Idle)
= Busy, else if(all busy)
= Active, else

Changes of operational state are propagated from EMS to upper management systems and those of administrative state are propagated in opposite direction as shown in Figure 2. But automatic transition of shutting down to the locked state is of the same direction as operational state. By the state decision rules mentioned above, there needs no redundant state information such as states of children or related objects for each system to decide a state of an object. If a state of one of the resources has changed, SMS or EMS has nothing to do but notify the other systems of just that event. Each system can determine the states of its related objects using fast search of the states of the related objects through naming tree as shown in Figure 1.

2) *Example*

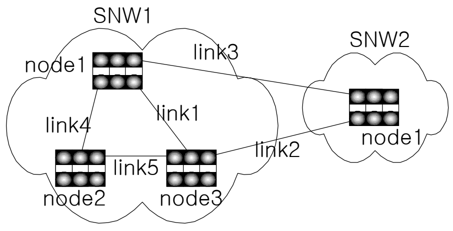

Figure 6. Simple Network

Now, we will give you an example. Let's assume a simple network shown in Figure 6 with nodes of one shelf which consists of three boards. Each board has two ports. The numbering of boards is from left to right and ports up to down. To make this a simple example, let's assume all objects above boards are enabled and unlocked. Then we can only care the states of boards and ports. Node 1.1 (In our naming scheme, node is represented by 'snw.node') is connected with node 2.1 through link 3. One end point is port 1.1.1.1.3.1 and the other is 2.1.1.1.1.1. If a port at the left end of link 3 has gone by a bad board, the restoration procedures are as shown in Figure 7.

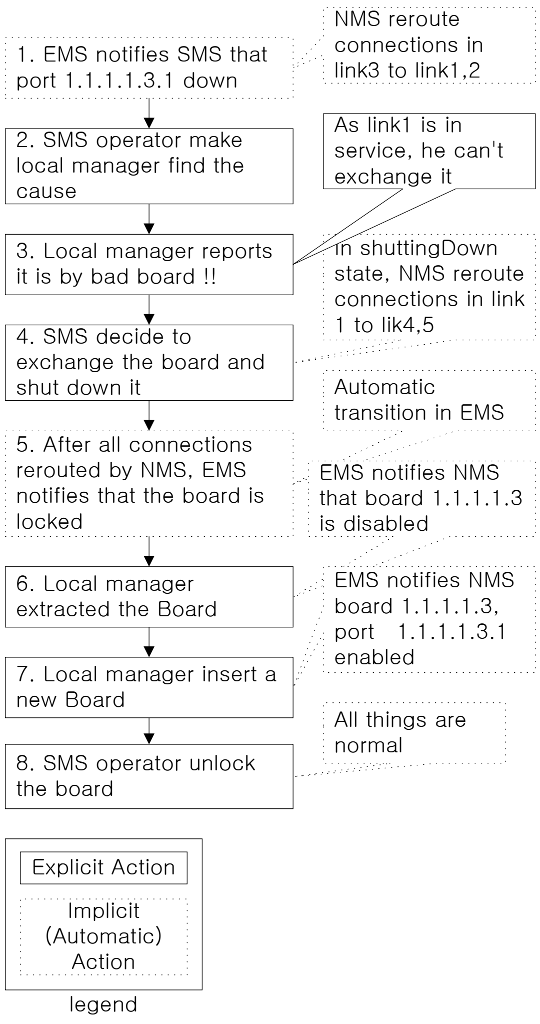
1. EMS notifies SMS that port 1.1.1.1.3.1 down
NMS reroute connections in link3 to link1,2
2. SMS operator make local manager find the cause
As link1 is in service, he can't exchange it
3. Local manager reports it is by bad board !!
In shuttingDown state, NMS reroute connections in link 1 to lik4,5
4. SMS decide to exchange the board and shut down it
Automatic transition in EMS
5. After all connections rerouted by NMS, EMS notifies that the board is locked
EMS notifies NMS that board 1.1.1.1.3 is disabled
6. Local manager extracted the Board
EMS notifies NMS board 1.1.1.1.3, port 1.1.1.1.3.1 enabled
7. Local manager insert a new Board
All things are normal
8. SMS operator unlock the board
Explicit Action
Implicit (Automatic) Action
legend

Figure 7. State Control Procedure

In each step of Figure 7, the state is as in Table 2. For the usage state, let’s assume the bandwidth of all the links are same of 1 unit and there is a connection of 0.5 unit in link1 and link3. Other links are empty. A detailed procedures for port 1.1.1.1.3.1 are as follows:

OpState : max(boardOpState,portOpState)
AdState : max(boardAdState,portAdState)

Step 1 : opState : max(0,1) = 1
adState : max(0,0) = 0
Step 2 : opState : max(0,1) = 1
adState : max(0,0) = 0
Step 3 : opState : max(0,1) = 1
adState : max(0,0) = 0
Step 4 : opState : max(0,1) = 1
adState : max(1,0) = 1
Step 5 : opState : max(0,1) = 1
adState : max(2,0) = 2
Step 6 : opState : max(1,1) = 1
adState : max(2,0) = 2
Step 7 : opState : max(0,0) = 0
adState : max(2,0) = 2
Step 8 : opState : max(0,0) = 0
adState : max(0,0) = 0

And the usage state of board 1.1.1.1.3 is as follows:

Step 1 : (a, a) → Active
Step 2 : (i, b) → Active
Step 3 : (i, b) → Active
Step 4 : (i, b) → Active
Step 5 : (i. i) → Idle
Step 6 : (i. i) → Idle
Step 7 : (i. i) → Idle
Step 8 : (i. i) → Idle with i(dle), a(ctive), b(usy)

Table 2. Result State

	Board 1.1.1.1.3			Port 1.1.1.1.3.1			Port 1.1.1.1.3.2		
State	op	us	ad	op	us	ad	op	us	ad
step1	0	a	0	1/1	a	0/0	0/0	a	0/0
step2	.	.	.	.	i	.	.	b	.
step3	.	.	.	.	.	.	.	.	.
step4	.	.	1	.	.	0/1	.	.	0/1
step5	.	i	2	.	.	0/2	.	i	0/2
step6	1	.	.	1/1	.	.	0/1	.	.
step7	0	.	.	0/0	.	0/2	0/0	.	.
step8	.	.	0	.	.	0/0	.	.	0/0

op(erational), us(age), ad(ministrative) state
Own state / result state
Period (.) : same as before

We can summarize the result states of board 1.1.1.1.3, port 1.1.1.1.3.1 and port 1.1.1.1.3.2 as shown in Table 2.

E. Conclusion

In this paper, a state decision scheme using object hierarchy and priorities among state values is presented. This is due to the natural born properties of state definitions. With our scheme, operators and management systems are able to know whether the resources are really available in complex relationships among resources with minimum inter-system state propagations.

References

[1] ITU-T, Information Technology - Open Systems Interconnection - Systems Management: State Management Function, X.731, Jan. 1992.
[2] Irene Katzela, Mischa Schwartz, “Schemes for Fault Identification in Communication Networks”, IEEE/ACM Trans. Networking, Vol.3, No.6, Dec. 1995.
[3] S.A.Yemini et al., “High Speed and Robust Event Correlation”, IEEE Commun. Magazine, 82-90, 1996.
[4] Chi-Chun Lo, Shing-Hong Chen, “Robust Event Correlation Scheme for Fault Identification in Communications Network”, Proc. IEEE Globecom’98, pp.3745-3750, 1998.
[5] Jaesung Choi, Myungwhan Choi, Sang-Hyuk Lee, “An alarm correlation and fault identification scheme based on OSI managed object classes”, Proc. ICC '99, pp. 1547 –1551, 1999
[6] ITU-T, Generic Functional Architecture of Transport Networks, G.805, Nov. 1995.
[7] ITU-T, Principles for a Telecommunications management network, M.3010, May 1996.