INTERNATIONAL TELECOMMUNICATION UNION

TELECOMMUNICATION STANDARDIZATION SECTOR

STUDY PERIOD 2017-2020

SG2-C164

STUDY GROUP 2

Original: English

Question(s): Q7/2 Geneva, 19-28 February 2019

CONTRIBUTION

Source:	Beijing University of Posts and Telecommunications	
Title:	Draft Q.rest: REST-based management services	
Purpose:	Proposal	

Contact:	WANG Zhili BUPT China	Tel: +86 10 61198090 ext. 8726 Fax: +86 10 62283412 Email: zlwang@bupt.edu.cn
Contact:	SHAO Sujie BUPT China	Tel: +86 10 61198090 ext. 8726 Fax: +86 10 62283412 Email: sjshao@bupt.edu.cn
Contact:	GUO Shaoyong BUPT China	Tel: +86 10 61198090 ext. 8726 Fax: +86 10 62283412 Email: syguo@bupt.edu.cn
Contact:	LIN Wei Inspur China	Tel: +86 10 56701901 Fax: +86 10 56701818 Email: lin.wei@inspur.com
Contact:	DONG Gangsong State Grid Henan Electric Power Company - Information and Communication Company China	Tel: +86 371 67905372 Fax: Email:donggangsong@ha.sgcc.com.cn

Keywords: Distributed processing; REST, managed objects, network management interfaces, JSON, JSON Schema.

Abstract: This document defines a set of services required to support REST-based interfaces and along with draft Recommendation X.rest composes a framework for REST-based network management interfaces. It specifies protocol requirements, and defines some network management-specific support services. The REST/JSON interface definitions for the network management-specific support services are also provided.

1 Scope

The network management architecture defined in [ITU-T M.3010] introduces the use of multiple management protocols. So far, the GDMO/CMIP, CORBA GIOP/IIOP, SMI/SNMP, Web Service/SOAP are possible choices at the application layer. Based on the management interface specification methodology defined in [ITU-T M.3020], more technology-based paradigms can be introduced into network management interfaces, and REST/SOAP is now an additional paradigm for network management.

This draft document, together with draft [ITU-T X.rest] sets out to define a framework for defining how interfaces supported by management systems and network elements should be modelled using REST/JSON schema. It is within the scope of this Recommendation to provide the following guidelines or instructions:

– protocol requirements for using REST in network management;

– how a REST-based notification service is used in network management interfaces;

– how to monitor the availability of a REST notification forwarding mechanism;

– how to access multiple managed objects in one operation;

– how to access the containment information through a REST containment service;

– compliance and conformance requirements.

2 References

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published. The reference to a document within this Recommendation does not give it, as a stand-alone document, the status of a Recommendation.

[ITU-T M.3010] Recommendation ITU-T M.3010 (2000), *Principles for a telecommunications management network.*

[ITU-T M.3020] Recommendation ITU-T M.3020 (2011), *Management interface specification methodology*.

[RFC 4230] IETF RFC 7230: *Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing*.

[RFC 7231] IETF RFC 7231: *Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content*.

[RFC 3986] IETF RFC 3986: *Uniform Resource Identifier (URI): Generic Syntax*.

[RFC 7232] IETF RFC 7232, *Hypertext Transfer Protocol (HTTP/1.1): Conditional Requests*.

[RFC 5789] IETF RFC 5789, *PATCH Method for HTTP*.

[RFC 6902] IETF RFC 6902, *JavaScript Object Notation (JSON) Patch.*

[RFC 6901] IETF RFC 6901, *JavaScript Object Notation (JSON) Pointer*.

Editors’s Note: to be extended.

3 Definitions

3.1 Terms defined elsewhere

This Recommendation uses the following terms defined elsewhere:

Editors's Note: to be extended.

3.2 Terms defined in this Recommendation

This Recommendation does not define any new terms.

Editors's Note: to be extended.

4 Abbreviations and acronyms

This Recommendation uses the following abbreviations and acronyms:

BNF	Backus-Naur Form
CMIP	Common Management Information Protocol
CORBA	Common Object Request Broker Architecture
GDMO	Guidelines for the Definition of Managed Objects
GIOP	General Inter-ORB Protocol
HTTP	Hyper Text Transfer Protocol
IIOP	Internet Inter-ORB Protocol
JSON	JavaScript Object Notation
REST	REpresentational State Transfer
SMI	Structure of Management Information
SNMP	Simple Network Management Protocol
SOAP	Simple Object Access Protocol
TMN	Telecommunications Management Network
URI	Unified Resource Identifier
XML	extensible Markup Language
XSD	XML Schema Definition

Editor's Note: to be extended.

5 Conventions

A few conventions are followed in this Recommendation to make the reader aware of the purpose of the text. While most of the Recommendation is normative, paragraphs succinctly stating mandatory requirements to be met by a management system (managing and/or managed) are preceded by a boldface "R" enclosed in parentheses, followed by a short name indicating the subject of the requirement, and a number. For example:

(R) EXAMPLE-1 An example mandatory requirement.

Requirements that may be optionally implemented by a management system are preceded by an "O" instead of an "R". For example:

(O) EXAMPLE-2 An example optional requirement.

The requirement statements are used to create compliance and conformance profiles.

Examples of JSON are included in this Recommendation and normative JSON schema specifying the data types, base classes and other modelling constructs of the framework are included in Annex A. The JSON are written in a 10 point courier typeface:

```
{
    "title": "root",
    "items": {
        "title": "array item"
    }
}
```

6 REST-based network management services goal and requirements

This clause describes the key goals of the services framework and the requirements that help the REST-based network management services support these goals. Clause 6.1 introduces the goals of the REST-based services framework. Clause 6.2 then provides terminology and requirements. The requirements in this clause are requirements that the framework must satisfy and which are based on telecommunications management needs. Clauses 7, 8, 9 then describe a framework that meets these needs and defines how to achieve the requirements of this clause by using REST-based services in a certain way.

6.1 Goals

This draft Recommendation sets out to define a framework for defining how interfaces supported by management systems and network elements should be modelled. Some key goals of the framework are identified here:

- application interoperability
- common REST-based support services.

This clause elaborates on these two goals.

6.1.1 Application interoperability

A key goal of the network management architecture and in particular the information architecture, is to promote a standard framework for providing interoperability and information exchange between systems from a diverse set of network management system suppliers. Interoperability between systems involves many aspects of development. At its lowest layer, a common communication mechanism must be in place to support a common syntax, the establishment of connectivity and the exchange of operation requests/replies between systems. This aspect of interoperability is inherently supported by the REST/HTTP specifications.

For network management, there is the need to provide application interoperability. That is, management systems from diverse suppliers will be utilized within a single administration's TMN to support different functions necessary to support the management of its networks. To simplify integration of these various suppliers' systems, they must agree on the semantics of the information being exchanged. This is accomplished with the specification of an information model. Guidelines

for the definition of REST-based information models are specified in [ITU-T X.rest], but the services defined here should support those guidelines.

6.1.2 Common REST-based support services

A second aspect of this framework is the definition of common usage and profiling of the distributed processing environment of choice. This aspect of the framework should indicate the reasonable expectations network management system suppliers may have for one another. Rather than redefining the interface capabilities needed to support common network management functions such as notification service with each information model, the modelling guidelines in [ITU-T X.rest] rely upon a set of support services. These support services enable the information models to be simpler, and also enhance interoperability.

6.2 Information modelling dependencies

As described in the previous clause, the explicit modelling of resources that are manageable across an interface is central to application interoperability. The guidelines for defining REST-based managed objects detailed in [ITU-T X.rest] describe the rules for modelling manageable resources. They also embody several decisions that must be supported by the network management REST-based services framework. This clause summarizes those points.

6.2.1 Access granularity

REST-based service interface granularity refers to the relationship between the resources that are modelled on an interface and the means by which they are accessed using REST-based services. [ITU-T X.rest] uses a service-grain modelling approach which means each modelled resource is only accessible through a REST-based service. The objects that represent manageable resources are called managed objects.

6.2.2 Representation of containment and naming

Containment is a logical representation of how modelled resources contain other modelled resources. Containment has traditionally been a very important relationship in network management applications because it is a convenient means of identifying the large number of resources that typically must be managed. [ITU-T X.rest] guidelines require that a unique name be assigned to each managed object, based in part on the name of the object that contains it. The REST-based management services must provide the means to store these names (and hence the containment relationships they represent).

6.2.3 Object creation and deletion

The REST-based platforms do not provide clients with the means to create or delete objects on remote systems. Instead, these functionalities are provided by remote systems for clients to create or delete objects on the remote system. [ITU-T X.rest] provides a common object accessing service, so object creation and deletion will be model-independent. When deleting an object that contains other objects, the containment relationship shall be kept, so that there will not be any orphan objects existing in the remote systems. The containment information should be stored in the remote system and be maintained when creation and deletion happen.

6.3 Scoping

The ability to perform complex queries (i.e., GET operations), updates (i.e., SET operations) and delete operations on a group of entities with a single operation request is a valuable component of TMN. Management systems may have to manage up to 10^7 instances of managed objects. Due to the size of the management information base, a managing system cannot efficiently perform ad hoc queries on individual instances of managed objects (i.e., entities). Rather, the managing system expects a level of intelligence to be supported by the managed system.

The intelligence in the managed system allows the managing system to select a group of managed entities on which an operation will be performed. Managed entity selection involves two phases: scoping. This managed entity selection process is supported by a service defined later in this Recommendation. This service allows a managing system to select a scope of objects to act on (scope is defined through containment relationships; see clause 6.2.2). Once the scope of entities is determined, the operation (specified by the scope request) is performed only on those entities.

The use of scoping in this framework supports:

– Scoped get: Returns the values (for a list of attributes) from each of the entities that meet the scope.

– Scoped update: Replaces an attribute value or adds/removes values to/from set-valued attributes, in the group of entities meeting the scope, to the values specified in the scoped request. May be used to update one or multiple attributes in a single object or multiple objects.

– Scoped deletion: Deletes all entities that meet the scope.

Scoping entails the identification of the entities to which a filter is to be applied. (Editor's Note: whether filter is needed is to be considered.) Scoping is applied based on the containment hierarchy as defined in clause 6.2.2. The scope is applied from a base managed entity down to a particular depth in the containment tree.

The base entity for the scope is defined as the root of the containment tree from which the search is to commence. A scoped request must specify the base managed entity of the scope. The depth of the scoping level can then be specified in one of four manners within the scoped request:

1) the base entity

2) the n-th level subordinates of the base entity

3) the base entity and all of its subordinates down to and including the n-th level

4) the base entity and all of its subordinates (i.e., the whole subtree).

6.4 Notifications

The framework needs to support the ability to:

- deliver notifications
- subscribe for notification types
- forward notifications to multiple destinations
- filter notifications (Editor's Note: whether filter is needed is to be considered.)
- uniquely identify the resource that emits the notification.

7 Framework overview and protocol requirements

7.1 Framework overview

This framework for REST-based management interfaces is a collection of capabilities. The framework defines the support services to be standards on network management interfaces conforming to this framework. The REST interfaces for these services will be defined in Annex A.

To support the software objects representing manageable resources, the framework requires that they implement some common basic capabilities. Therefore, one base class (JSON-based) is defined in draft [ITU-T X.rest] for use in modelling network management resources. Managed object classes in information models must inherit and implement a basic set of capabilities from the base class in order to operate within this framework. Finally, some rules and conventions are defined for

information modellers developing models for use with this framework. These consist of modelling guidelines and JSON style idioms. All of these are depicted graphically in Figure 1.

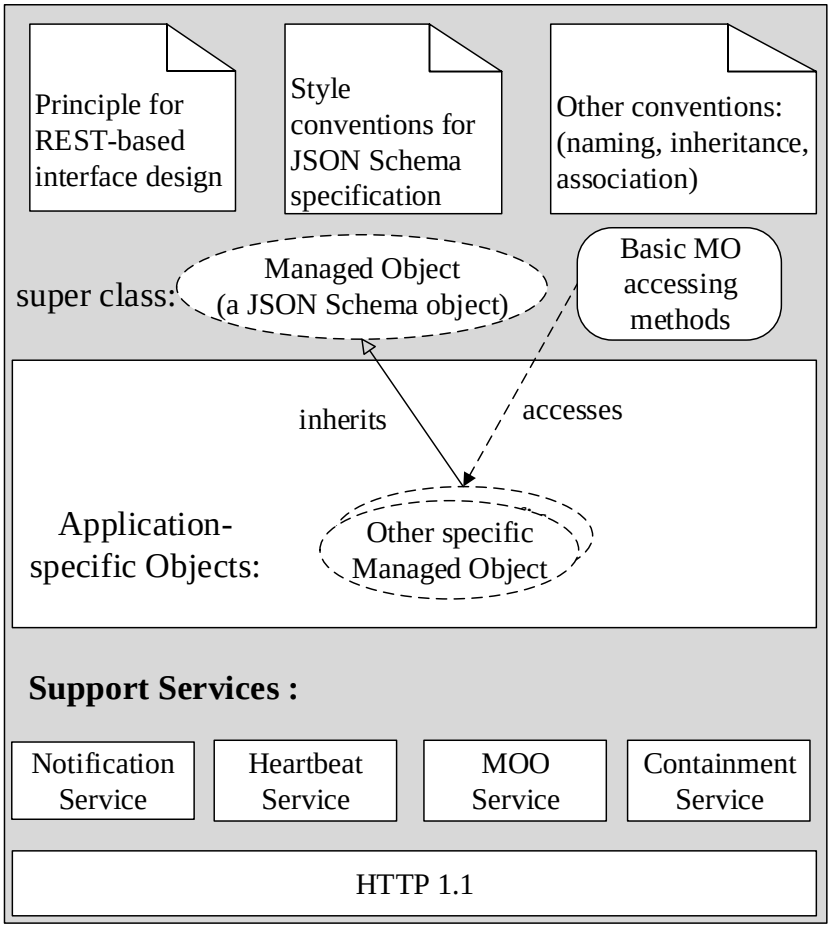

Figure 1 Framework overview.

Figure 1 shows the framework in grey. In the middle are the application-specific objects that are supported by the framework. Along the bottom is a box representing the communication protocol: HTTP. Above that are a number of boxes with names in them representing the services that compose the framework. Along the top of the figure are icons representing the super class managed object, and the basic MO accessing methods. Each managed object supported by this framework must ultimately inherit from the super class. Also shown on Figure 1 are icons of pages with up-turned corners representing standard object modelling conventions.

The framework services, represented as boxes with square corners, are defined in this draft Recommendation. The super class, object modelling conventions and other conversions are defined in draft [ITU-T X.rest].

7.2 Protocol, languages and services requirements

This clause defines the versions of the languages and protocols that are required to support this framework. REST-based technology and protocol specifications are defined by IETF. Table 1 shows which version (including subsequent versions released up until the date of this Recommendation) of the applicable specifications must be supported to comply with this framework.

Table 1 – REST-based language, protocol and versions

Service	Version
JSON Schema Core (IETF)	Draft-07 (March, 2018)
HTTP protocol (IETF)	1.0 (1996) 1.1 (1999) 2.0 (August, 2013)

8 REST-based notification service and notification format

8.1 REST-based notification service overview

REST-base notification service provides a standard approach for notification consumers and providers to transfer notification information using a topic-based publish/subscribe pattern. The contents of notification service include standard message exchanges to be implemented by notification providers, operational requirements expected of service providers and requesters that participate in notifications, and an JSON model that describes topics.

The REST-based notification service supports the asynchronous exchange of event messages between clients using a subscribe-and-publish paradigm. It may also support the filtering of notifications which are also needed in network management interfaces.

8.2 Mapping of operations from Recommendation ITU-T M.3702

[ITU-T M.3702] provides protocol neutral requirements and analysis for the generic notification management service, which defines the generic functions that are to be implemented by a protocol-specific management paradigm. In this framework, the generic notification management model is mapped to HTTP/JSON definition.

The following table indicates the mapping from [ITU-T M.3702] to the REST-based notification service.

No.	**[ITU-T M.3702]**		**ITU-T Q.rest**	
	Operation name		**Operation name**	
	Request/ response	**Parameters**	**Request/ response**	**Parameters**
1	subscribeNotification		HTTP POST	
	Input parameter	– managerId – notificationTypes – filteringCriteria – destination	Request	"POST" {notificationServiceURI}/"subscriptions" {HTTPVersion} request body schema: $ref: "notificationApiDataSchema.json#/definitions/SubscribeNotificationRequest"
	Output parameter	– subscriptionId – status	Response	{HTTPVersion} {statusCode} {reasonPhrase} Location: { notificationServiceURI }/"subscriptions"/{subscriptionId}{response body}
2	unsubscribeNotification		HTTP DELETE	
	Input parameter	– managerId – subscriptionId	Request	"DELETE" {notificationServiceURI}/"subscriptions"/{subscriptionId} {HTTPVersion}

No.	[ITU-T M.3702]		ITU-T Q.rest	
	Operation name		Operation name	
	Request/ response	Parameters	Request/ response	Parameters
	Output parameter	– status	Response	{HTTPVersion} {statusCode} {reasonPhrase} {response body}
3	suspendSubscription		HTTP POST	
	Input parameter	– managerId – subscriptionId	Request	"POST" { notificationServiceURI }/"subscriptions"/{subscriptionId}/"suspendSubscription" {HTTPVersion}
	Output parameter	– status	Response	{HTTPVersion} {statusCode} {reasonPhrase} {response body} Response body schema: $ref:"notificationApiDataSchema.json#/definitions/NotificationInfo"
4	resumeSubscription		HTTP POST	
	Input parameter	– managerId – subscriptionId	Request	"POST" { notificationServiceURI }/"subscriptions"/{subscriptionId}/"resumeSubscriptions" {HTTPVersion}
	Output parameter	– status	Response	{HTTPVersion} {statusCode} {reasonPhrase} {response body}
5	getNotificationTypes		HTTP GET	
	Input parameter	– notificationIRPId	Request	"GET" {notificationServiceURI}/"subscriptions"/{subscriptionId}/"getTypes" {HTTPVersion}
	Output parameter	– status – notificationType List	Response	{HTTPVersion} {statusCode} {reasonPhrase} {response body} Response body schema: $ref:"notificationApiDataSchema.json#/definitions/GetNotificationTypesResponse"
6	querySubscription		HTTP GET	
	Input parameter	– subscriptionId	Request	"GET" {notificationServiceURI}/"subscriptions"/{subscriptionId}{HTTPVersion}
	Output parameter	– subscribed NotificationType – subscriptionStatus – destination – filteringCriteria – status	Response	{HTTPVersion} {statusCode} {reasonPhrase} {response body} Response body schema: $ref: "notificationApiDataSchema.json#/definitions/QuerySubscriptionResponse"
7	modifySubscription		HTTP PATCH	
	Input parameter	– subscriptionId – filteringCriteria – destination – notificationTypes	Request	"PATCH" {notificationServiceURI}/"subscribtions"/{subscriptionId} {HTTPVersion}{request body} Request body schema: $ref:"notificationApiDataSchema.json#/definitions/ModifySubscriptionRequest"
	Output parameter	– status	Response	{HTTPVersion} {statusCode} {reasonPhrase} {response body} Response body schema: $ref: "notificationApiDataSchema.json#/definitions/NotificationInfo"

No.	[ITU-T M.3702]		ITU-T Q.rest	
	Operation name		Operation name	
	Request/ response	Parameters	Request/ response	Parameters
8	listAllSubscriptionIds		HTTP PATCH	
	Input parameter	– managerId	Request	"POST" {notificationServiceURI}/"subscribtions"/listAll SubscriptionIds {HTTPVersion} {request body} Request body schema: $ref: "notificationApiDataSchema.json# /definitions/ListAllSubscriptionsIdsRequest"
	Output parameter	– subscriptionIdSet – status	Response	{HTTPVersion} {statusCode} {reasonPhrase} {response body} Response body schema: $ref:"notificationApiDataSchema.json#/definitio ns/ListAllSubscriptionsIdsResponse"
9	report notification:		HTTP POST (from provider to consumer)	
			Request	"POST" {destinationURI} {HTTPVersion} {request body} Request body schema: $ref: "notificationApiDataSchema.json# /definitions/NotifyRequest"
			Response	{HTTPVersion} {statusCode} {reasonPhrase} {response body}

A detailed interface definition can be found in clause A.1.

8.3 Notification format definitions

8.3.1 Notification format to be used in this Recommendation

This clause will provide a description for noification format tobe used in this Recommendation.

Editor's Note: to be extended.

8.1.3.2 Common notification header definition

This clause provides the detailed common notification header described in [ITU-T M.3702].

The following table provides the definition of the parameters of the common notification header.

Table 6 – Common notification header definition

Parameter name	Qualifiers	JSON schema type	Descriptions
objectClass	M	`string`	It is the class name of the MO which sends out this notification. See clause 7.3.5 of [ITU-T M.3702]
objectInstance	M	`uri`	It is the DN of the MO which sends out this notification. See clause 7.3.5 of [ITU-T M.3702]
notificationId	M	`string`	It is the Identifier of the notification for the purpose of notification correlation. See clause 7.3.5 of [ITU-T M.3702]
eventTime	M	`string`	It is the time when this event happened. See clause 7.3.5 of [ITU-T M.3702]
systemDN	M	`uri`	It is the DN of the system which sends out this notification. See clause 7.3.5 of [ITU-T M.3702]
notificationType	M	`string (enum)`	It indicates the type of notification. See clause 7.3.5 of [ITU-T M.3702]. The possible common notification types defined in this Recommendation can be the following: – attributeValueChange – objectCreation – objectDeletion – stateChange – communicationsAlarm – environmentalAlarm – equipmentAlarm – processingErrorAlarm – qualityOfServiceAlarm – integrityViolation – operationalViolation – physicalViolation – securityViolation – timeDomainViolation – relationshipChange – heartbeat This list can be extended to include new notification types as and when they are defined.

8.1.3.3 Common notification contents definition

1) Notification contents for the objectCreation and objectDeletion notifications.

Table 7 – Notification contents for objectCreation and objectDeletion

Notification parameter	Qualifiers	Data type	Descriptions
correlatedNotifications	O	`string`	List of correlated notifications.
additionalText	O	`string`	Text message.
additionalInfo	O	TBD	Additional information not in text format.
sourceIndicator	O	TBD	Cause of event.
attributeList	O	TBD	Attribute values.

2) Notification contents for the attributeValueChange notification.

Table 8 – Notification contents for attributeValueChange

Notification parameter	Qualifiers	Data type	Descriptions
correlatedNotifications	O	TBD	List of correlated notifications.
additionalText	O	TBD	Text message.
additionalInfo	O	TBD	Additional information not in text format.
sourceIndicator	O	TBD	Cause of event.
attributeChanges	M	TBD	Changed attributes.

3) Notification contents for the stateChange notification.

Table 9 – Notification contents for stateChange

Notification parameter	Qualifiers	Data type	Descriptions
correlatedNotifications	O	TBD	List of correlated notifications.
additionalText	O	TBD	Text message.
additionalInfo	O	TBD	Additional information not in text format.
sourceIndicator	O	TBD	Cause of event.
stateChanges	M	TBD	Changed states.

4) Notification contents for the communicationAlarm, environmentalAlarm, equipmentAlarm, processingErrorAlarm, qualityOfServiceAlarm notifications.

Table 10 – Notification contents for alarms

Notification parameter	Qualifiers	Data type	Descriptions
correlatedNotifications	O	TBD	List of correlated notifications.
additionalText	O	TBD	Text message.
additionalInfo	O	TBD	Additional information not in text format.
probableCause	M	TBD	The probable cause of the alarm.
specificProblems	O	TBD	Non standarized problems.
perceivedSeverity	M	TBD	It indicates the perceived severity of the alarm. See type definition for details.
backedUpStatus	O	TBD	"True" if backed up.
backUpObject	O	TBD	It indicates the DN of the backup object if the backUpStatus is "false".
trendIndication	O	TBD	See type for details.
thresholdInfo	O	TBD	See type for details.
stateChangeDefinition	O	TBD	It indicates the state changes in this alarm.
monitoredAttributes	O	TBD	See type for details.
proposedRepairActions	O	TBD	It indicates the proposed actions to repair this fault.
alarmEffectOnService	O	TBD	True if alarm is service effecting.
alarmingResumed	O	TBD	True if alarming was just resumed, possibly resulting in delayed reporting.
suspectObjectList	O	TBD	Objects possibly involved in failure.

5) Notification contents for the integrityViolation, operationalViolation, physicalViolation, securityViolation, timeDomainViolation notifications.

Table 11 – Notification contents for violations

Notification parameter	Qualifiers	Data type	Descriptions
correlatedNotifications	O	TBD	List of correlated notifications.
additionalText	O	TBD	Text message.
additionalInfo	O	TBD	Additional information not in text format.
securityAlarmCause	M	TBD	The cause of the security alarm.
securityAlarmSeverity	M	TBD	Clears allowed? ITU-T X.721 appears to restrict the cleared value on this alarm, but clears should be allowed.
securityAlarmDetector	M	TBD	See type for details.
serviceUser	M	TBD	The user of the service which is violated.
serviceProvider	M	TBD	The service provider of the service which is violated.

6) Notification contents for the relationshipChange notification.

Table 12 – Notification contents for relationshipChange

Notification parameter	Qualifiers	Data type	Descriptions
correlatedNotifications	O	TBD	List of correlated notifications.
additionalText	O	TBD	Text message.
additionalInfo	O	TBD	Additional information not in text format.
sourceIndicator	O	TBD	Cause of event.
relationshipChanges	M	TBD	Changed relationship attributes.

The detailed data type definitions for notification contents can be found in clause A.1.

9 REST-based heartbeat service

The heartbeat service is used to verify the operation of the notification forwarding mechanism in a managed system, as well as the communications network between the managed system and managing system.

It periodically sends a small notification to a managing system interested in receiving it and that notification identifies the system that emitted the heartbeat. After configuring this service a

managing system can ensure the notification service is functioning. Since these notifications flow through the same software and networks as notifications from other resources, they periodically verify the operation of these resources.

Editor's Note: to be extended.

10 REST-based multiple objects operation service

The multiple object operation (MOO) service provides a set of generic capabilities that may be invoked on any kinds of sets of managed objects (of any kind). The operations supported are listed below.

- Scoped get: Returns the values from each of the objects for a list of attributes.
- Scoped update: Used to replace an attribute value or to add or remove values to/from set-valued attributes. May be used to update one or multiple attributes in a single object or multiple objects.
- Scoped delete: Deletes multiple objects.

A basic service only needs to implement the scoped get operation. The other two operations are optional.

Editor's Note: to be extended.

11 REST-based containment service

In the network management field a function is needed to be able to report which objects are contained by a superior object, to verify that a superior object exists before a subordinate is created, to make sure two objects with the same name are not created, etc. The framework will be extended to support this function by adding a new service, the containment service.

The main function to be supported by the containment service is to enable a managing system to query a managed system with the name of an object, and receive back the names of the objects contained by that object. In addition, a means of getting names added to and removed from the service will be defined. These are not for use by managing systems but internally by managed objects, factories, and other parts of a managed system. They are provided to promote the development of reusable components, possibly by third parties, and are defined on an interface separate from that used by managing systems.

Editor's Note: to be extended.

12 Compliance and conformance

12.1 System conformance points

12.2 Basic conformance profile

Annex A
REST/JSON Scheme definition of framework support services

(This annex forms an integral part of this Recommendation.)

A.1 REST-based notification service interface and JSON schema definition

A.1.1 REST-based notification service interface

The following provides the HTTP requests and responses for REST-based notificaiton service interface.

1) subscribeNotification

REQUEST:

"POST" {notificationServiceURI}/"subscriptions" {HTTPVersion}

response body schema:

$ref: "notificationApiDataSchema.json#/definitions/SubscribeNotificationRequest"

RESPONSE:

{HTTPVersion} {statusCode} {reasonPhrase}

Location: { notificationServiceURI }/"subscriptions"/{subscriptionId}

{response body}

statusCode	reasonPhrase	response body schema
201	Created	null or $ref:"notificationApiDataSchema.json#/definitions/NotificationInfo"
400	Bad Request	$ref:”commonApiDataSchema.json#/definitions/ErrorInformation”
404	Not Found	null
409	Conflict	null
500	Internal Server Error	null

2) unsubscribeNotification

REQUEST:

"DELETE" {notificationServiceURI}/"subscriptions"/{subscriptionId} {HTTPVersion}

RESPONSE:

{HTTPVersion} {statusCode} {reasonPhrase}

{response body}

statusCode	reasonPhrase	response body schema
200	OK	null
400	Bad Request	$ref:”commonApiDataSchema.json#/definitions/ErrorInformation”

404	Not Found	null
405	Method Not Allowed	null
500	Internal Server Error	null

3) suspendNotification

REQUEST：

"POST" { notificationServiceURI }/"subscriptions"/{subscriptionId}/ "suspendSubscription" {HTTPVersion}

RESPONSE:

{HTTPVersion} {statusCode} {reasonPhrase}

{response body}

statusCode	reasonPhrase	response body schema
200	OK	null
400	Bad Request	$ref:”commonApiDataSchema.json#/definitions/ErrorInformation”
404	Not Found	null
405	Method Not Allowed	null
409	Conflict	null
500	Internal Server Error	null

4) resumeSubscriptions

REQUEST:

"POST" { notificationServiceURI }/"subscriptions"/{subscriptionId}/ "resumeSubscriptions" {HTTPVersion}

RESPONSE:

{HTTPVersion} {statusCode} {reasonPhrase}

{response body}

statusCode	reasonPhrase	response body schema
200	OK	null
400	Bad Request	$ref:“commonApiDataSchema.json#/definitions/ErrorInformation”
404	Not Found	null
405	Method Not Allowed	null
409	Conflict	null
500	Internal Server	null

	Error	

5) getNotificationTypes

REQUEST:

"GET" {notificationServiceURI}/"subscriptions"/{subscriptionId}/"getTypes" {HTTPVersion}

RESPONSE:

{HTTPVersion} {statusCode} {reasonPhrase}

{response body}

statusCode	reasonPhrase	response body schema
200	OK	$ref: "notificationApiDataSchema.json#/definitions/GetNotificationTypesResponse"
400	Bad Request	$ref:”commonApiDataSchema.json#/definitions/ErrorInformation”
404	Not Found	null
405	Method Not Allowed	null
500	Internal Server Error	null

6) querySubscription

REQUEST：

"GET" {notificationServiceURI}/"subscriptions"/{subscriptionId} {HTTPVersion}

RESPONSE:

{HTTPVersion} {statusCode} {reasonPhrase}

{response body}

statusCode	reasonPhrase	response body schema
200	OK	$ref: "notificationApiDataSchema.json#/definitions/QuerySubscriptionResponse"
400	Bad Request	$ref:”commonApiDataSchema.json#/definitions/ErrorInformation”
404	Not Found	null
405	Method Not Allowed	null
500	Internal Server Error	null

7) modifySubscription

REQUEST:

"PATCH" {notificationServiceURI}/"subscribtions"/{subscriptionId} {HTTPVersion}

{request body}

Request body schema:

$ref:"notificationApiDataSchema.json#/definitions/ModifySubscriptionRequest"

RESPONSE:

{HTTPVersion} {statusCode} {reasonPhrase}

{response body}

statusCode	reasonPhrase	response body schema
200	OK	null or $ref: "notificationApiDataSchema.json# /definitions/NotificationInfo"
400	Bad Request	$ref: "commonApiDataSchema.json# /definitions/ErrorInformation"
404	Not Found	null
405	Method Not Allowed	null
500	Internal Server Error	null

8) listAllSubscriptionIds

REQUEST：

"POST" {notificationServiceURI}/"subscribtions"/listAllSubscriptionIds {HTTPVersion} {request body}

Request body schema:

$ref: "notificationApiDataSchema.json#/definitions/ListAllSubscriptionsIdsRequest"

RESPONSE:

{HTTPVersion} {statusCode} {reasonPhrase}

{response body}

statusCode	reasonPhrase	response body schema
200	OK	$ref:"notificationApiDataSchema.json# /definitions/ListAllSubscriptionsIdsResponse"
400	Bad Request	$ref:”commonApiDataSchema.json#/definitions/ErrorInformation”
404	Not Found	null
405	Method Not Allowed	null

500	Internal Server Error	null

9) notify

REQUEST:

"POST" {destinationURI} {HTTPVersion}

{request body}

Request body schema:

$ref: "notificationApiDataSchema.json#/definitions/NotifyRequest"

RESPONSE:

{HTTPVersion} {statusCode} {reasonPhrase}

{response body}

response body schema:

statusCode	reasonPhrase	response body schema
200	OK	null
400	Bad Request	$ref:”commonApiDataSchema.json# /definition/ErrorInformation”
404	Not Found	null
405	Method Not Allowed	null
500	Internal Server Error	null

A.1.2

A.1.2 JSON-schema definitions for REST-based notification service

The following provides the JSON-schema definitions for REST-based notificaiton service interface.

FileName: notificationApiDataSchema.json

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "definitions": {
        "SubscribeNotificationRequest": {
            "type": "object",
            "properties": {
                "managerId": {
                    "type": "string"
                },
                "notificationTypes": {
```

```
            "type": "array",
            "items": {
                "type": "string"
            }
        },
        "filteringCriteria": {
            "type": "string"
        },
        "subscriberUri": {
            "type": "string",
            "format": "uri"
        }
    }
},
"SubscriptionInfo ": {
    "type": "object",
    "properties": {
        "subscriptionId": {
            "type": "string",
            "format": "uri"
        },
        "managerId": {
            "type": "string"
        },
        "notificationTypes": {
            "type": "array",
            "items": {
                "type": "string"
            }
        },
        "filteringCriteria": {
            "type": "string"
        },
        "destination": {
            "type": "string",
            "format": "uri"
```

```
        },
        "subscriptionStatus": {
            "type": "string",
            "enum": [
                "locked",
                "unlocked"
            ]
        }
    }
},
"notificationTypeList": {
    "type": "array",
    "items": {
        "type": "string"
    }
},
"ModifySubscriptionRequest": {
    "type": "object",
    "properties": {
        "filteringCriteria": {
            "type": "string"
        },
        "destination": {
            "type": "string",
            "format": "uri"
        },
        "notificationTypes": {
            "type": "string"
        }
    }
},
"ListAllSubscriptionIdsRequest": {
    "type": "object",
    "properties": {
        "managerId": {
            "type": "string"
```

```
        }
      }
    },
    "ListAllSubscriptionIdsResponse": {
      "type": "array",
      "items": {
        "type": "string",
        "format": "uri",
        "description": "..."
      }
    },
    "NotifyRequest": {
      "$ref": "#/definitions/NotificationBaseType"
    },
    "NotificationBaseType": {
      "type": "object",
      "required": [
        "objectClass",
        "objectInstance",
        "notificationId",
        "eventTime",
        "notificationTypes",
        "systemDN"
      ],
      "properties": {
        "objectClass": {
          "type": "string"
        },
        "objectInstance": {
          "type": "string"
        },
        "notificationId": {
          "type": "string"
        },
        "eventTime": {
          "type": "string"
```

```
        },
        "notificationTypes": {
            "type": "string"
        },
        "systemDN": {
            "type": "string",
            "format": "uri"
        },
        "correlatedNotifications": {
            "type": "array",
            "items": {
                "$ref": "#/definitions/CorrelatedNotificationsType"
            }
        },
        "additionalText": {
            "type": "string"
        },
        "additionalInfo": {
            "type": "array",
            "items": {
                "$ref": "#/definitions/ManagementExtensionType"
            }
        }
    }
},
"CorrelatedNotificationsType": {
    "type": "array",
    "items": {
        "type": "object",
        "properties": {
            "source": {
                "type": "string",
                "format": "uri"
            },
            "notifIDs": {
                "type": "array",
```

```
                "items": {
                    "type": "string"
                }
            }
        }
    },
    "ManagementExtensionType": {
        "type": "object",
        "properties": {
            "id": {
                "ref": "#/definitions/UIDType"
            },
            "info": {
                "type": "object"
            }
        }
    },
    "UIDType": {
        "type": "object",
        "properties": {
            "moduleName": {
                "type": "string"
            },
            "value": {
                "type": "integer"
            }
        }
    },
    "UIDSetType": {
        "type": "array",
        "items": {
            "ref": "#/definitions/UIDType"
        }
    },
    "SourceIndicatorType": {
```

```
        "type": "string",
        "enum": [
            "resourceOperation",
            "managementOperation",
            "unknown"
        ]
    },
    "AttributeChangeSetType": {
        "type": "array",
        "items": {
            "type": "object",
            "properties": {
                "attributeName": {
                    "type": "string"
                },
                "oldValue": {},
                "newValue": {}
            },
            "required": [
                "attributeName",
                "oldValue",
                "newValue"
            ]
        }
    },
    "AttributeSetType": {
        "type": "array",
        "items": {
            "type": "object",
            "properties": {
                "attributeName": {
                    "type": "string"
                },
                "value": {}
            },
            "required": [
```

```
            "attributeName",
            "value"
        ]
    }
},
"AttributeValueChangeMessageType": {
    "allOf": [
        {
            "ref": "#/definitions/NotificationMessageType"
        },
        {
            "properties": {
                "sourceIndicator": {
                    "ref": "#/definitions/SourceIndicatorType"
                },
                "attributeChanges": {
                    "ref": "#/definitions/AttributeChangeSetType"
                }
            }
        }
    ]
},
"ObjectCreationMessageType": {
    "allOf": [
        {
            "ref": "#/definitions/NotificationMessageType"
        },
        {
            "properties": {
                "sourceIndicator": {
                    "ref": "#/definitions/SourceIndicatorType"
                },
                "attributeList": {
                    "ref": "#/definitions/AttributeSetType"
                }
            }
```

```
        }
      ]
    },
    "ObjectDeletionMessageType": {
      "ref": "#/definitions/ObjectCreationMessageType"
    },
    "StateChangeMessageType": {
      "allOf": [
        {
          "ref": "#/definitions/NotificationMessageType"
        },
        {
          "properties": {
            "sourceIndicator": {
              "ref": "#/definitions/SourceIndicatorType"
            },
            "stateChanges": {
              "ref": "#/definitions/AttributeChangeSetType"
            }
          }
        }
      ]
    },
    "AlarmMessageType": {
      "allOf": [
        {
          "ref": "#/definitions/NotificationMessageType"
        },
        {
          "properties": {
            "probableCause": {
              "ref": "#/definitions/UIDType"
            },
            "specificProblems": {
              "ref": "#/definitions/UIDSetType"
            },
```

```
"perceivedSeverity": {
    "ref": "#/definitions/PerceivedSeverityType"
},
"backedUpStatus": {
    "type": "boolean",
    "description": "optional item"
},
"backUpObject": {
    "type": "string",
    "format": "uri"
},
"trendIndication": {
    "ref": "#/definitions/TrendIndicationType",
    "description": "optional item"
},
"thresholdInfo": {
    "ref": "#/definitions/ThresholdInfoType"
},
"stateChangeDefinition": {
    "ref": "#/definitions/AttributeChangeSetType"
},
"monitoredAttributes": {
    "ref": "#/definitions/AttributeSetType"
},
"proposedRepairActions": {
    "ref": "#/definitions/UIDSetType"
},
"alarmEffectOnService": {
    "type": "boolean",
    "description": "optional item"
},
"alarmingResumed": {
    "type": "boolean",
    "description": "optional item"
},
"suspectObjectList": {
```

```
                "ref": "#/definitions/SuspectObjectSetType"
            }
        },
        "required": [
            "probableCause",
            "specificProblems",
            "perceivedSeverity",
            "backUpObject",
            "thresholdInfo",
            "stateChangeDefinition",
            "monitoredAttributes",
            "proposedRepairActions",
            "suspectObjectList"
        ]
    }
  ]
},
"PerceivedSeverityType": {
  "type": "string",
  "enum": [
    "indeterminate",
    "critical",
    "major",
    "minor",
    "warning",
    "cleared"
  ]
},
"TrendIndicationType": {
  "type": "string",
  "enum": [
    "lessSevere",
    "noChange",
    "moreSevere"
  ]
},
```

```
"ThresholdInfoType": {
    "type": "object",
    "properties": {
        "attributeID": {
            "type": "string"
        },
        "observedValue": {
            "type": "integer"
        },
        "thresholdLevel": {
            "ref": "#/definitions/ThresholdLevelIndType",
            "description": "optional item"
        }
    },
    "required": [
        "attributeID",
        "observedValue"
    ]
},
"ThresholdLevelIndType": {
    "type": "object",
    "properties": {
        "indication": {
            "type": "string",
            "enum": [
                "up",
                "down"
            ]
        },
        "low": {
            "type": "integer",
            "description": "optional item"
        },
        "high": {
            "type": "integer"
        }
```

```
        },
        "required": [
            "indication",
            "high"
        ]
    },
    "SuspectObjectSetType": {
        "type": "array",
        "items": {
            "type": "object",
            "properties": {
                "objectClass": {
                    "type": "string"
                },
                "suspectObjectInstance": {},
                "failureProbability": {
                    "type": "integer",
                    "description": "optional item"
                }
            },
            "required": [
                "objectClass",
                "suspectObjectInstance"
            ]
        }
    }
  }
}
```

A.2 REST-based heartbeat service interface and JSON schema definition

A.3 REST-based MOO service interface and JSON schema definition

A.4 REST-based containment service interface and JSON schema definition

_____