Frameworx Specification

SLA Management API REST Specification

TMF623
Release 14.5.0
September 2014

Latest Update: Frameworx Release 14.5	**Member Evaluation**
Version 1.0.0	**IPR Mode: RAND**

NOTICE

Direct inquiries to the TM Forum office:

240 Headquarters Plaza,
East Tower – 10th Floor,
Morristown, NJ 07960 USA
Tel No. +1 973 944 5100
Fax No. +1 973 944 5110
TM Forum Web Page: www.tmforum.org

TABLE OF CONTENTS

LIST OF TABLES

N/A

INTRODUCTION

The following document is the specification of the REST API for the SLA and SLA Violation resources. It includes the model definition as well as all available operations. Possible actions are creating and retrieving a SLA or SLA Violation, updating the whole SLA or only do a patch update. Furthermore the HTTP GET allows filtering.

The SLA API provides a standardized interface for SLA life cycle Management (SLA Negotiation, SLA configuration, SLA Activation/enforcement, SLA Operations, SLA violation / consequence handling, SLA reporting) between a Customer and a Service Provider which provides offers (product with attached SLA in its catalogue) the customer can discover, browse, trigger and order.

It will be also useful in a multi-partner environment where exchanging SLA is needed in order to allow rapid and efficient SLA life cycle management across a partner's environment. From SLA perspective, duties and rights are assigned to each actor & associated roles mainly in the case where a service is composed of various components brought by different partners within federation or / and syndication models.

SLA Management API manages the following resources:

- Service Level Agreement (SLA)
    - Part of a business agreement between a Service Provider and a Customer, quantitatively specifying the service performance level the Service Provider commits to deliver. Other actors & roles can be involved such as SLA Auditor or SLA Integrator. SLA includes rules or Service Level Specifications (SLA Parameters, Metrics and Thresholds), as well as a description of measuring, reporting and violation handling processes. For the purpose of the specification, it can be expressed in terms of validity of period, related parties, and rules (metrics, reference value, tolerance, consequence …).
    - From the Customer perspective, this means that the end Customer provides Quality of Service requirements associated to its business applications to a Service Provider. The two parties negotiate the specific set of SLA parameters and parameter values that best serves them.
    - From the Service Provider perspective, each offered product or service can be provided with a standard Product SLA.
- SLA Violation
    - It represents any SLA failures observed through a metric threshold crossing restricted to what has been agreed in the SLAs for the given service (consequences, penalties, remedies...). SLA Violation is composed of Metrics, reported date, period,. The Related Parties are represented in the same way as for "SLA" resource. This practical and operational view allows the related parties to react and perform an immediate and direct analysis of potential impacts of the violation.
    - There is also an "Attachment" which represents statistics, a dashboard or reporting data to be presented to the target parties, for deeper analysis purpose.

SLA API performs the following operations on Service Level Agreement:

- Retrieval
    - all SLAs (with “SLA Provider”, “SLA Customer” or “End User” role)
    - SLAs based on template
    - SLAs with specified ID – only one SLA is returned
- Creation of a SLA (planned)
- Full update of a SLA (planned)
- Partial update of a SLA (planned)
- Creation of a SLA violation (planned)
- Retrieval of a SLA violation
- Notification of SLA Violation Creation

SAMPLE USE CASES

Reader will find examples of use cases using SLA management API in “Open Digital Business Scenarios and Use Cases” document.

RESOURCE MODEL

This specification involves two resources: Service Level Agreement “SLA” and “SLA Violation”

Service Level Agreement “SLA” Resource

a) Description

This Service Level Agreement “SLA” Resource Model aims at illustrating the way to translating “SLA Resource Model” in JSON representation, reflecting the resource aspect described in the “SLA API Profile” (Management Requirements) document V0.2. Service Level Agreement (SLA) Resource Model and associate attributes are aligned with those described in SID Service Level Agreement modeling. Meaning no Resources are imported from other sources. Besides, the entities are represented in terms of roles, each with its rights & duties covering Multi-partner model illustrating B2BX model (SLA Provider, SLA Consumer, SLA Auditor, EndUser roles). In this model, “SLA Provider” is referring to a CSP while “SLA Consumer” is referring to a DSP. The “EndUser” is referring to the customer of the DSP, in some cases he could be a customer of both (CSP and DSP). “SLA Auditor” role is to monitor SLA as described in TR178V2. It could be played either by the CSP himself or delegated to a 3rd party. The “Agreed” or “Approved SLA” is described in terms of SLA rules which contains the Metrics, their related values or range, thresholds, valid period or date, consequences in case of violation of any clause of the SLA. It is also assumed all Metrics are the existing ones which are stored in the Service Provider “Metrics Library” with their attached references. Besides, each metric is attached to a given Product in the Catalogue with a dedicated reference e.g. ”URL”. The Service Level Agreement “SLA” resource model is also characterized by its “validity period”. This use case covers the situation where the validity period is pre-determined (planned) which excludes the case of Time-variant SLA that could be attached to a “SLA on-Demand” use case. The latter could be considered in another release for specific use cases (Cloud, virtualization) for instance.

In order to optimize the SLA resource, there is a need for defining a common pattern or Template “rule”. This pattern is structured as following: the Id of the metric, its name, the measurement unit attached to the considered metric, its reference value, the tolerance value when the threshold is crossed and the consequence in case of violation. Regarding the financial-related aspect and penalties associated to a consequence, a pointer is simply mentioned to the Service Level Agreement “SLA” contract.

b) Example of the JSON representation of an Service Level Agreement “SLA” resource

```
{
   "id": "123444",
   "href": "http/www.acme.com/SlaManagement/sla/123444",
```



```
"name": "HighSpeedDataSLA",
"description": "SLA for high speed data.",
"version": "0.1",
"validFor":
{
      "startDateTime": "2013-04-19T16:42:23.0Z",
      "endDateTime": "2013-04-21T09:43:54.0Z"
},

"relatedParty":
[
      {
            "href": "http://",
            "role": "SLAProvider"
      },

      {
            "href": "http://",
            "role": "SLAConsumer"
      },

      {
            "href": "http://",
            "role": "SLAAuditor"
      },

      {
            "href": "http://",
            "role": "SLABusinessBroker"
      },

      {
            "href": "http://",
            "role": "SLATechnicalBroker"
      },

      {
            "href": "http://",
            "role": "ThirdPartySLAManager"
      },

      {
            "href": "http://",
            "role": "EndUser"
      }
],

"rule":
[
      {
            "id": "availability",
            "metric": "http://IEEE99.5/Availability",
```



```
                "unit": "string",
                "referenceValue": "available",
                "operator": ".eq",
                "tolerance": "0.05",
                "consequence": "http://ww.acme.com/contract/clause/42"
        },

        {
                "id": "downstream_GBR",
                "metric": "http://IEEE99.5/Data/bitrates/GBR/down",
                "unit": "kbps",
                "referenceValue": "1024",
                "operator": ".ge",
                "tolerance": "0.20",
                "consequence": "http://ww.acme.com/contract/clause/45"
        }
    ],

    "template":
    {
            "href": "http/www.acme.com/slaManagement/slaTemplate/42",
            "name": "DataSLATemplate",
            "description": "basic template for Data SLA"
    }
}
```


Field description

Field	Description
id	Unique identifier of the Service level Agreement (SLA)
name	Name of the Service level Agreement (SLA)
description	Description of the Service level Agreement (SLA)
version	Version of the Service level Agreement (SLA)
validityPeriod	Period where the clauses of the SLA are applicable
startTime	Date/Time of the beginning of the validityPeriod
endTime	Date/Time of the end of the validityPeriod
template	SLA Template characteristics
href	Reference of the Template
name	Name of the template
description	Description of the template
relatedParty	Parties engaged in the SLA (SLA provider, SLA consumer, …)
Role	Role attached to each party (SLA provider, SLA consumer, …)
href	Reference of the party
state	Service Level Agreement state
approved	Indicates if the service level agreement is approved or not (True or false)
rule	Common pattern or Template for the SLA parameters, metrics, and thresholds
id	Unique identifier of the metric
metric	Reference of metric stored in the Service Provider “Metrics Library”
unit	Unit of measure of metric
referenceValue	Reference value of metric
operator	Operator used when calculating the rule

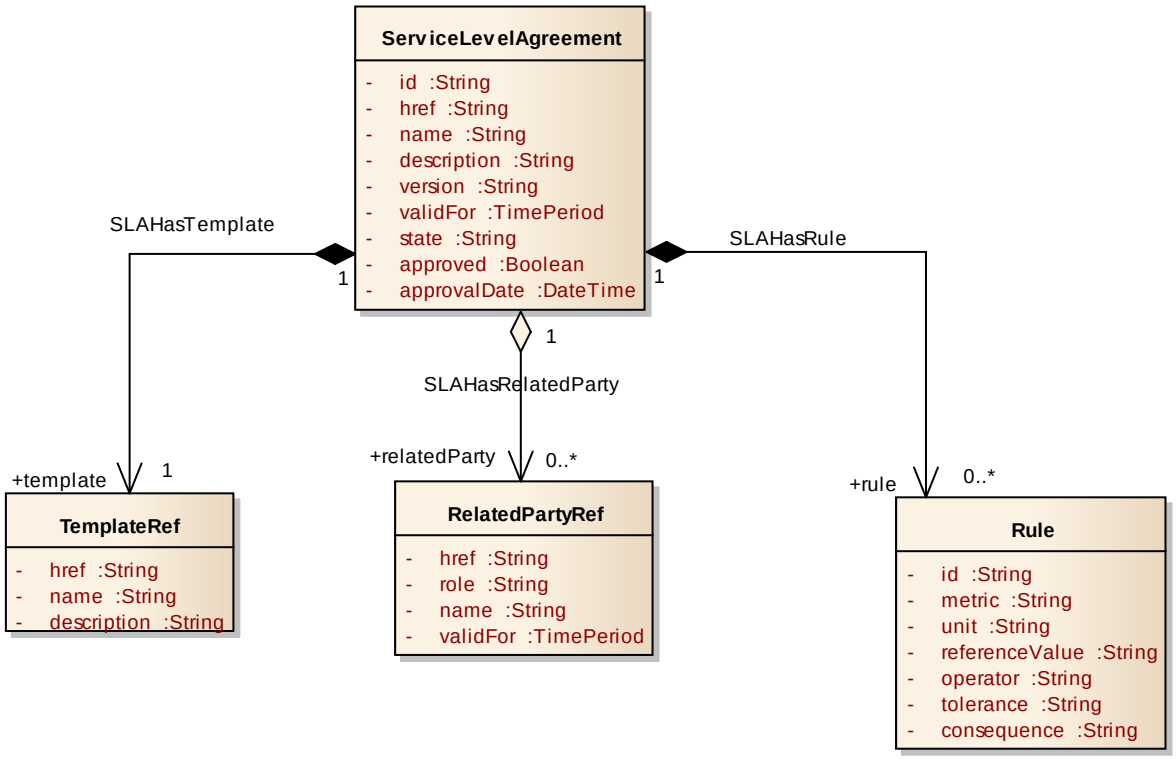
tolerance	Allowable variation of metric
consequence	defines the action to be applied as a result of a threshold crossing

Service Level Agreement “SLA” States

Regarding “SLA States”, the source considered is “WS-Agreement” diagram (state graphic) from which the allowed status changes as recommended in the “SLA Profile” document (section 4) when applicable.

For instance, it is assumed that the processes that are linked to the “Negotiation Phase” of the SLA Management lifecycle such as SLA “Approval”, “Terms or Conditions” and “Consequence” are in the status of “completed”. That means, only one state “Observed” is considered which supposes the SLA is approved and accepted by the related Parties.

SLA resource model



SLAViolation Resource

a) SLA Violation resource description

This “SLAViolation” is the second resource considered in this translation into JSON representation. The same representation used for “SLA” resource is applied and is aligned with the SID modeling as well.

It is assumed all Metrics are the ones already stored in the Service Provider “Metrics Library” with their attached references and associated with products on-boarded in the Catalogue.

The “violation” is defined in terms of ID, description (Metrics, reported date, period). Besides, the RelatedParty such as SLA Provider (CSP), SLA Consumer (DSP) and SLA Auditor (for SLA monitoring) is represented in the same way as for Service Level Agreement “SLA” resource. The same goes for “EndUser” role. Two levels of event representation could be useful to introduce:

- an immediate view of the event result impact. This practical and operational view allows the Related Party to react and perform an immediate and direct analysis of potential impacts of the violation.

* The “attachment” is associated with the event and represents statistics, a dashboard or reporting data to be presented to the target Related Party, DSP (SLA Consumer) or / and to CSP (SLA Provider) for deeper analysis purpose.

b) Example of the JSON representation of an SLAViolation


```
{
    "id": "7467",
    "href": "http/www.acme.com/slaManagement/slaViolation/123444",
    "relatedParty":
    [
          {
                "href": "http://",
                "role": "SLAProvider"
          },

          {
                "href": "http://",
                "role": "SLAConsumer"
          },

          {
                "href": "http://",
                "role": "SLAAuditor"
          },

          {
                "href": "http://",
                "role": "SLABusinessBroker"
          },

          {
                "href": "http://",
                "role": "SLATechnicalBroker"
          },

          {
                "href": "http://",
                "role": "ThirdPartySLAManager"
          },

          {
                "href": "http://",
```



```
                "role": "EndUser"
            }
        ],

        "sla":
        {
            "href": "http/www.acme.com/slaManagement/sla/123444",
            "description": "sla of premium video"
        },

        "violation":
        {
            "unit": "string",
            "referenceValue": "available",
            "operator": ".eq",
            "actualValue": "available",
            "tolerance": "0.05",
            "violationAverage": "0.1",
            "comment": "Availability below agreed level.",
            "consequence": "http://ww.acme.com/contract/clause/42",
            "attachment":
            {
                "href": "https://foo.bar/screenshot.123",
                "description": "availability statistics for August 2013"
            },

            "rule":
            {
                "href": "http/www.zak.com/slaManagement/sla/123444/rules/availability",
                "description": "availability"
            }
        }
    }
```

Field description

Field		Description
id		The id of the resource model (SLAViolation)
SLA		
	description	Description of the Service level Agreement (SLA)
	href	Reference of the Service Level Agreement
RelatedParty		Party engaged in the SLA (SLA provider, SLA consumer, …)
	role	Role attached to each party (SLA provider, SLA consumer, …)
	href	Reference of the party
Violation		A discrepancy identified by applying rules to Service Level Agreement related entities.
	rule	The rule represents the value of the threshold of the metric. Once crossed it triggers a violation.
		description	Description of the rule
		href	Reference of the rule
	unit	Unit of measure of metric
	referenceValue	Reference value of metric

	operator	Operator used when calculating the rule
	actualValue	Actual value of the metric
	tolerance	Allowable variation of metric
	violationAverage	TBD
	Comment	Comment about violation
	consequence	defines the action to be applied as a result of a threshold crossing
	attachment	represents statistics, a dashboard or reporting data to be presented to the target parties
	description	Description of the attachment
	href	Reference of the attachment

SLAVIOLATION RESOURCE MODEL UML

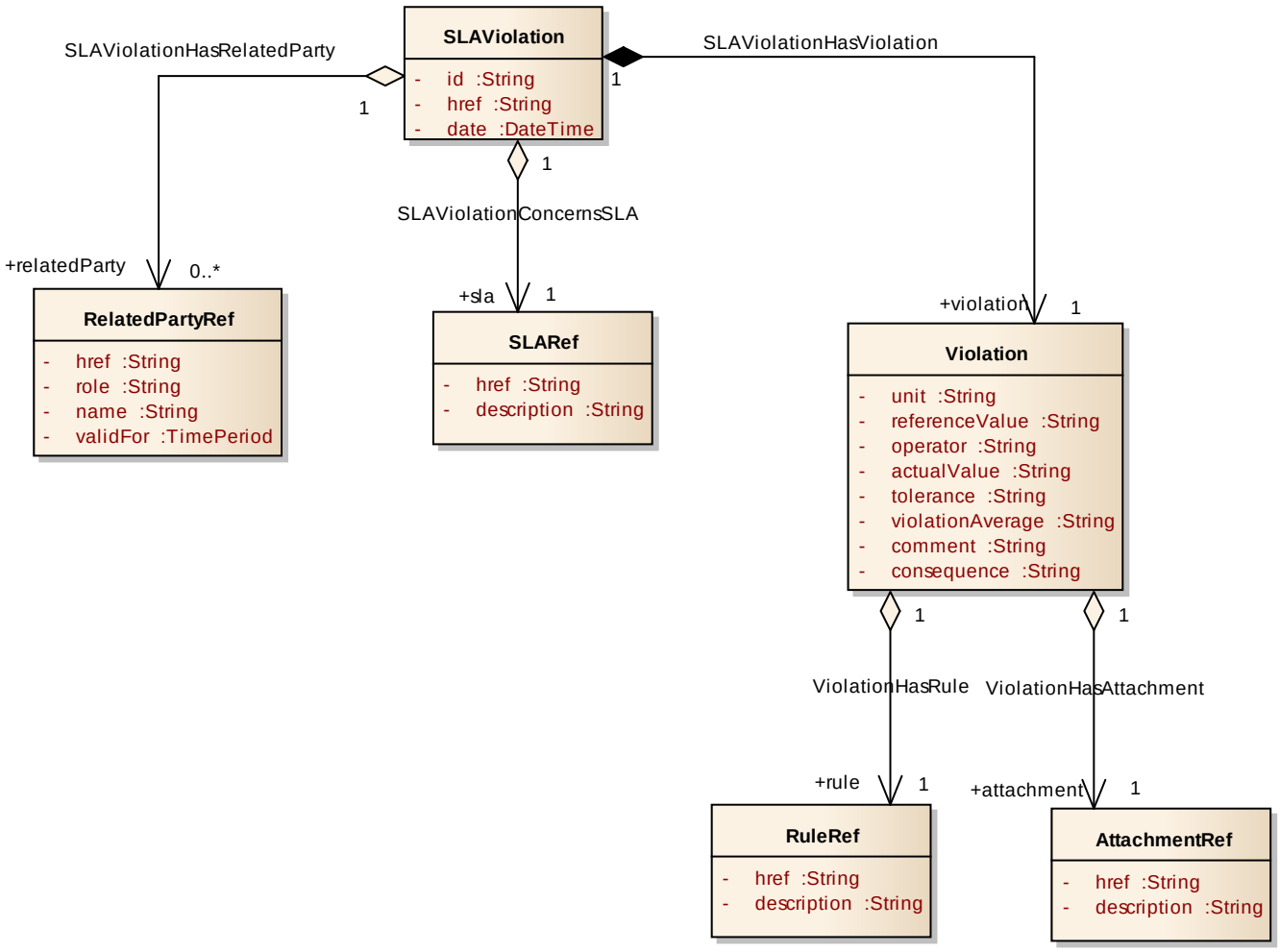


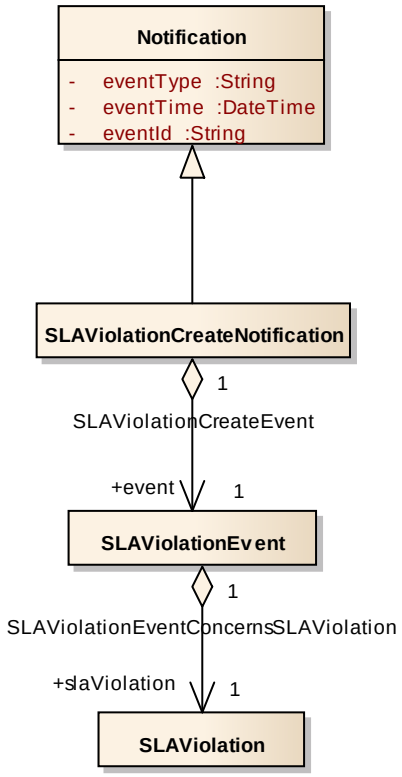
Figure 2 – SLAViolation resource model

EVENT MODEL

SLA VIOLATION CREATE NOTIFICATION

This event is generated whenever a SLAViolation resource is created. In addition to the occurrence date, it contains the created SLAViolation.

UML Model:



a) Example of the JSON representation of an SLAViolationCreation event

```
{
      "eventType": "SLAViolationCreateNotification",
      "eventTime": "2014-09-27T05:46:25.0Z",
      "eventId": "430958",
      "event":
      {
```



```
            "slaViolation":
            {
                  "id": "89045",
            Following a whole representation of the SLA Violation with all its attributes
            See SLA Violation Resource.
            }
      }
}
```

Field	Description
eventTime	Date/time when the event was generated (may be posterior to SLAViolation creation date)
SLAViolation	The created SLAViolation resource
eventType	Type of the event. Must be "SLAViolationCreateNotification"

SLA OPERATIONS

For every single of operation on the entities use the following templates and provide sample REST requests and responses.

Remember that the following Uniform Contract rules must be used :

Operation on Entities	Uniform API Operation	Description
Query Entities	GET Resource	GET must be used to retrieve a representation of a resource.
Create Entity	POST Resource	POST must be used to create a new resource
Partial Update of an Entity	PATCH Resource	PATCH must be used to partially update a resource
Complete Update of an Entity	PUT Resource	PUT must be used to completely update a resource identified by its resource URI
Remove an Entity	DELETE Resource	DELETE must be used to remove a resource
Execute an Action on an Entity	POST on TASK Resource	POST must be used to execute Task Resources
Other Request Methods	POST on TASK Resource	GET and POST must not be used to tunnel other request methods.

Filtering and attribute selection rules are described in the TMF REST Design Guidelines.

Notifications are also described in a subsequent section.

GET API/SLA/{ID}

This Uniform Contract operation is used to retrieve the representation of a managed entity or a task.

Note that collections can be retrieved via GET API/SLA with no {ID}

Description :

- Provide an overall description of the Operation
- Describe the returned representation of the <resource> instance(s).
- Describe if filtering is enabled and what can be done using query parameters.
- Describe if attribute selection is enabled.
- Describe if the resource represents a managed entity, a collection or a task.
- Describe the structure of the identifier.

Behavior :

- What status and exception codes are returned.
- Returns HTTP/1.1 status code 200 if the request was successful.
- Any other special return and/or exception codes.
- Specify what level of attribute filtering can be used. In fact we mandate L0 (equality) filtering in every specification as per REST Guidelines. Add this to Template.

REQUEST
GET /api/sla Accept: application/json
RESPONSE
200 Content-Type: application/json {"sla": [JSON Resource Representation with every attributes }] }

Retrieving all SLAs – returns an array/ a list of SLAs:

- GET /api/sla

Retrieving all SLAs where ‘http://..//acme.com’ is involved with “SLAProvider” role:

- GET /api/sla? ….

Retrieving all SLAs where ‘http://…//mycompany.com’ is involved with “SLAConsumer” role:

- GET /api/sla? ….

Retrieving all SLAs where ‘http://…//John.Doe’ is involved with “EndUser” role:

- GET /api/sla? ….

Retrieving all SLAs based on template ‘"http/www.acme.com/slaManagement/slaTemplate/42":

- GET /api/sla? ….

Retrieve sla with specified ID – only one SLA is returned:

- GET /api/sla/1

PUT API/SLA/{ID}

This Uniform Contract operation is used to completely update the representation of a managed entity or a task.

Description :

- Provide an overall description of the Operation
- Describe the input representation of the <resource> instance.
- Describe if the resource represents a managed entity or a task.
- Describe the structure of the identifier.

Behavior :

- What status and exception codes are returned.
- Returns HTTP/1.1 status code 201 if the request was successful.
- Any other special return and/or exception codes.

Description:

- This Uniform Contract operation is used to completely update the representation of a SLA.
- Resource represents a managed entity.

Behavior:

- Returns HTTP/1.1 status code 201 if the request was successful.
- Returns HTTP/1.1 status code 400 (Bad request) if content is invalid (missing required attributes, …).

Updating the whole SLA – if you try to change the SLA ID itself an exception is returned. All fields with different values will be changed. If the request contains the same values like the current SLA representation, nothing is changed. If an element is empty in the request, the value of the element will be deleted. If it is a required element, an exception is returned.

REQUEST
PUT API/SLA/1 Content-type: application/json {"sla": { JSON Resource Representation with every attributes }}
RESPONSE
201 Content-Type: application/json {"sla": { JSON Resource Representation with every attributes }

PATCH API/SLA/{ID}

Description:

- This Uniform Contract operation is used to partially update the representation of a SLA.
- Resource represents a managed entity.

Table of patchable attributes

Attribute name	Patchable	Rule
id	N	

name	Y	
description	Y	
version	Y	
validityPeriod	Y	
template	Y	
href	Y	
name	Y	
description	Y	
relatedParty	Y	
state	Y	
approved	Y	
rules	Y	
id	Y	
metric	Y	
unit	Y	
referenceValue	Y	
operator	Y	
tolerance	Y	

Consequence	Y	

Further document any rules that must be implemented when patching attributes. It is use-case driven

Rule name	Rule/Pre Condition/Side Effects/Post Conditons

REQUEST
PATCH API/sla/{ID} Content-type: application/json {"sla": { JSON Resource Representation with every attributes }}
RESPONSE
201 Content-Type: application/json {"sla": { JSON Resource Representation with every attributes }}

POST API/SLA/{ID}

Description:

- This Uniform Contract operation is used to create a SLA.
- Resource represents a managed entity.
- Mandatory attributes that must be provided when you create the SLA :

  Description, severity, type

Behavior:

- Returns HTTP/1.1 status code 201 if the request was successful.
- Returns HTTP/1.1 status code 400 (Bad request) if content is invalid (missing required attributes, …).

The requester cannot generate the id – the id to identify the REST resource is generated automatically in the back-end. The correlationId can be set from external but is not mandatory.

Specify the attributes required when an entity is created (and their default values if not):

Attribute name	Mandatory	Default	Rule
id	N		
name	Y		
description	N		
version	N		
validityPeriod	N		
template	N		
href	N		
name	N		
description	N		
relatedParty	N		
state	N		
approved	N		
rules	N		

id	N		
metric	N		
unit	N		
referenceValue	N		
operator	N		
tolerance	N		
Consequence	N		

- Further specify any rules on the creation of the entity

Rule name	Rule

REQUEST
POST API/sla Content-type: application/json {"sla": { JSON Resource Representation with every attributes ", }}
RESPONSE
201 Content-Type: application/json {"sla": {

JSON Resource Representation with every attributes }}

DELETE API/SLA/{ID}

This Uniform Contract operation is used to delete a managed entity or a task.

Description :

- Provide an overall description of the Operation
- Describe if the resource represents a managed entity or a task.
- Describe the structure of the identifier.

Behavior :

- What status and exception codes are returned.
- Returns HTTP/1.1 status code 200 if the request was successful.
- Any other special return and/or exception codes.

REQUEST
DELETE API/SLA/{ID}
RESPONSE
200

Example see TMF REST Design Guidelines.

SLAVIOLATION OPERATIONS

For every single of operation on the entities use the following templates and provide sample REST requests and responses.

Remember that the following Uniform Contract rules must be used :

Operation on Entities	Uniform API Operation	Description
Query Entities	GET Resource	GET must be used to retrieve a representation of a resource.
Create Entity	POST Resource	POST must be used to create a new resource
Partial Update of an Entity	PATCH Resource	PATCH must be used to partially update a resource
Complete Update of an Entity	PUT Resource	PUT must be used to completely update a resource identified by its resource URI
Remove an Entity	DELETE Resource	DELETE must be used to remove a resource
Execute an Action on an Entity	POST on TASK Resource	POST must be used to execute Task Resources
Other Request Methods	POST on TASK Resource	GET and POST must not be used to tunnel other request methods.

Filtering and attribute selection rules are described in the TMF REST Design Guidelines.

Notifications are also described in a subsequent section.

GET API / SLAVIOLATION / {ID}

This Uniform Contract operation is used to retrieve the representation of a managed entity or a task.

Note that collections can be retrieved via GET /API/SLAVIOLATION with no {ID}

REQUEST
GET /api/SLAVIOLATION/{ID}/{attributeSelector}?{filtering expression} Accept: application/json
RESPONSE

200
Content-Type: application/json
 	
{
JSON Resource Representation
}

Example see TMF REST Design Guidelines.

PUT API/ SLAVIOLATION /{ID}

This Uniform Contract operation is used to completely update the representation of a managed entity or a task.

REQUEST

PUT API/SLAVIOLATION /{ID}
Content-type: application/json
 	
{
JSON Resource Representation with every attributes
}

RESPONSE
201 Content-Type: application/json


```
{  JSON Resource Representation with every attributes
}
```

Example see TMF REST Design Guidelines.

PATCH API/ SLAVIOLATION /{ID}

This Uniform Contract operation is used to partially update the representation of a managed entity or a task.

Specify which attributes are patchable using the following table (to capture RO attributes)

Attribute name	Patchable	Rule
id	N	
SLA	Y	
description	Y	
href	Y	
RelatedParty	Y	
Violation	Y	
rule	Y	
unit	Y	
referenceValue	Y	
operator	Y	
actualValue	Y	
tolerance	Y	

violationAverage	Y	
Comment	Y	
consequence	Y	
attachments	Y	

Further document any rules that must be implemented when patching attributes.

Rule name	Rule/Pre Condition/Side Effects/Post Conditons

REQUEST

PATCH API/ SLAVIOLATION /{ID}
Content-type: application/json
 	
{
JSON Resource Representation with every attributes
}

RESPONSE
201 Content-Type: application/json { JSON Resource Representation with every attributes }

Example see TMF REST Design Guidelines.

POST API/ SLAVIOLATION /{ID}

This Uniform Contract operation is used to create a managed entity or a task.

ID Management :

Specify the ID Management Rule POST without specifying the ID must result in the system generating the ID for the <Entity>. In a specific case, the ID can also be included in the POST (optional)

Attributes required when an entity is created (and their default values if not):

Attribute name	Mandatory	Default	Rule
id	N		
SLA	N		
description	N		
href	N		
RelatedParty	N		
Violation	N		
rule	N		
unit	N		
referenceValue	N		
operator	N		
actualValue	N		
tolerance	N		
violationAverage	N		
Comment	N		

consequence	N		
attachments	N		

Further specify any rules on the creation of the entity

Rule name	Rule

REQUEST

POST API/SLAVIOLATION
Content-type: application/json
 	
{
JSON Resource Representation with every mandatory attributes
}

RESPONSE
201 Content-Type: application/json { JSON Resource Representation with every provided and default attributes }

Example see TMF REST Design Guidelines.

DELETE API/ SLAVIOLATION /{ID}

This Uniform Contract operation is used to delete a managed entity or a task.

REQUEST

DELETE API/ SLAVIOLATION /{ID}
RESPONSE
200

Example see TMF REST Design Guidelines.

SLA NOTIFICATIONS

For every single of operation on the entities use the following templates and provide sample REST notification POST calls.

It is assumed that the Pub/Sub uses the Register and UnRegister mechanisms described in the REST Guidelines reproduced below.

REGISTER LISTENER POST /HUB

Description :

Sets the communication endpoint address the service instance must use to deliver information about its health state, execution state, failures and metrics. Subsequent POST calls will be rejected by the service if it does not support multiple listeners. In this case DELETE /api/hub/{id} must be called before an endpoint can be created again.

Behavior :

Returns HTTP/1.1 status code 204 if the request was successful.

Returns HTTP/1.1 status code 409 if request is not successful.

REQUEST
POST /api/hub Accept: application/json {"callback": "http://in.listener.com"}
RESPONSE
201 Content-Type: application/json Location: /api/hub/42 {"id":"42","callback":"http://in.listener.com","query":null}

UNREGISTER LISTENER DELETE HUB/{ID}

Description :

Clears the communication endpoint address that was set by creating the Hub.

Behavior :

Returns HTTP/1.1 status code 204 if the request was successful.

Returns HTTP/1.1 status code 404 if the resource is not found.

REQUEST
DELETE /api/hub/{id} Accept: application/json
RESPONSE
204

PUBLISH {EVENTTYPE} POST /LISTENER

Description :

Provide the Event description

Behavior :

Returns HTTP/1.1 status code 201 if the service is able to set the configuration.

REQUEST

POST /client/listener
Accept: application/json

```
{
      "eventType": "EventType",
      "eventTime": "2014-09-27T05:46:25.0Z",
      "eventId": "1562231",
      "event":
      {
  EVENT BODY
      }
```


}
RESPONSE
201 Content-Type: application/json

Example see TMF REST Design Guidelines.

SLAVIOLATION NOTIFICATIONS

For every single of operation on the entities use the following templates and provide sample REST notification POST calls.

It is assumed that the Pub/Sub uses the Register and UnRegister mechanisms described in the REST Guidelines reproduced below.

REGISTER LISTENER POST /HUB

Description :

Sets the communication endpoint address the service instance must use to deliver information about its health state, execution state, failures and metrics. Subsequent POST calls will be rejected by the service if it does not support multiple listeners. In this case DELETE /api/hub/{id} must be called before an endpoint can be created again.

Behavior :

Returns HTTP/1.1 status code 204 if the request was successful.

Returns HTTP/1.1 status code 409 if request is not successful.

REQUEST
POST /api/hub Accept: application/json {"callback": "http://in.listener.com"}
RESPONSE
201 Content-Type: application/json Location: /api/hub/42 {"id":"42","callback":"http://in.listener.com","query":null}

UNREGISTER LISTENER DELETE HUB/{ID}

Description :

Clears the communication endpoint address that was set by creating the Hub.

Behavior :

Returns HTTP/1.1 status code 204 if the request was successful.

Returns HTTP/1.1 status code 404 if the resource is not found.

REQUEST
DELETE /api/hub/{id} Accept: application/json
RESPONSE
204

PUBLISH {EVENTTYPE} POST /LISTENER

Description :

Provide the Event description

Behavior :

Returns HTTP/1.1 status code 201 if the service is able to set the configuration.

REQUEST

POST /client/listener
Accept: application/json
 	
{
"eventType": "EventType",
"eventTime": "2014-09-27T05:46:25.0Z",
"eventId": "1562231",
"event":
{
EVENT BODY
}
}

RESPONSE
201 Content-Type: application/json

Example see TMF REST Design Guidelines.

RELEASE HISTORY

Release Number	Date	Release led by:	Description
Release 0.7	11/09/2013	zavisa.bjelogrlic@accenture.com tayeb.benmeriem@orange.com jerome.hannebelle@orange.com pgauthier@tmforum.org	First Spec Jam in Paris.
Release 1.0	15/05/2014	tayeb.benmeriem@orange.com jerome.hannebelle@orange.com	Reformatted to Template 1.1

CONTRIBUTORS TO DOCUMENT

Veronique Mauneau	Orange
Jean-Luc Tymen	Orange
Pierre Gauthier	TM Forum
John Morey	Ciena
Cliff C Faurer	AMKB Cloud