Frameworx Specification

Service Problem Management API REST Specification

TMF656
Release 16.5.0
December
2016

Latest Update: Frameworx Release 16.5	**Member Evaluation**
Version 1.2.0	**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 FIGURES

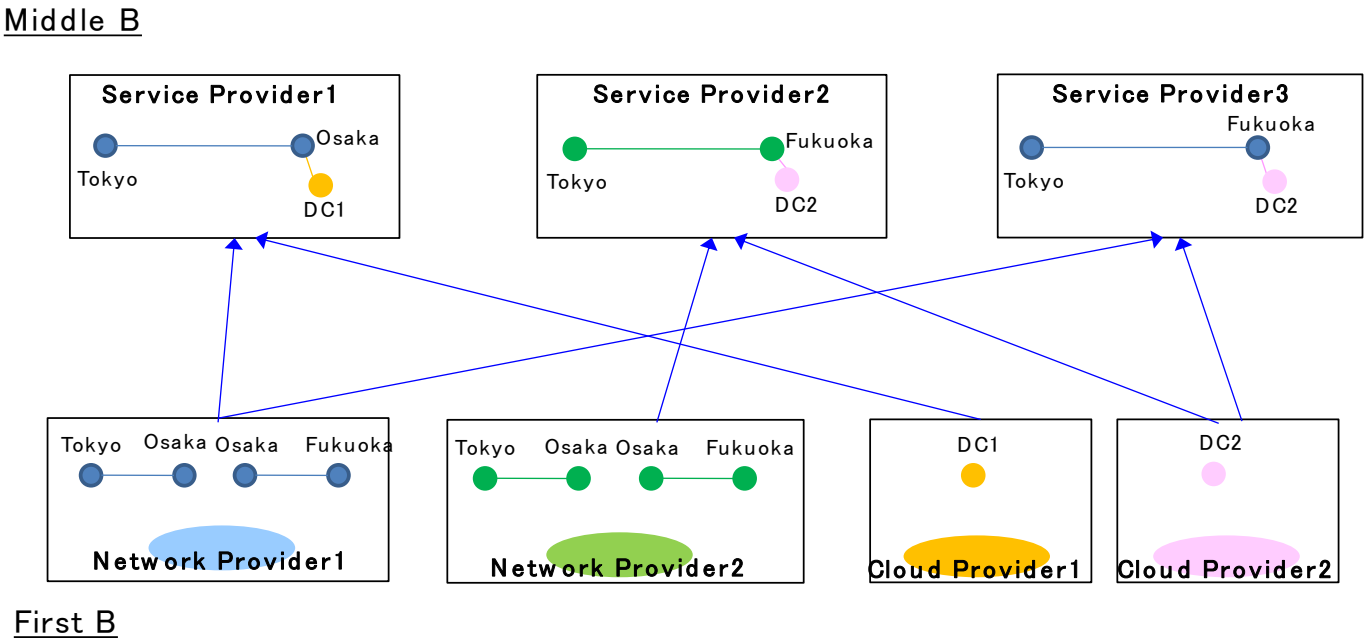
INTRODUCTION

This SPM API is used for the service providers (Defined as the Middle B) to manage the service problems in their service area. Service problem is generated based on the information declared by Middle B or the event information notified from infrastructure providers (Defined as the First B) who provide the infrastructure of cloud or network. The event information includes alarm information, performance anomaly information, trouble ticket information, SLA violation, maintenance information and prediction information. Middle Bs can refer the service problems and the event information from First Bs and when the service problems occur or its status have been changed, Middle Bs can receive notifications. According to these functions, Middle Bs are able to grasp the service problems quickly and accurately.

SAMPLE USE CASES

Sample Use Case

We assume following situation:
There are Network Provider 1 (NP1) and 2 (NP2), which provide network infrastructure, and Cloud Provider 1, 2, which provides cloud infrastructure, as First Bs. Using these infrastructure, Service Provider 1 (SP1), 2 (SP2) and 3 (SP3) are providing their services to their end-users as Middle Bs.

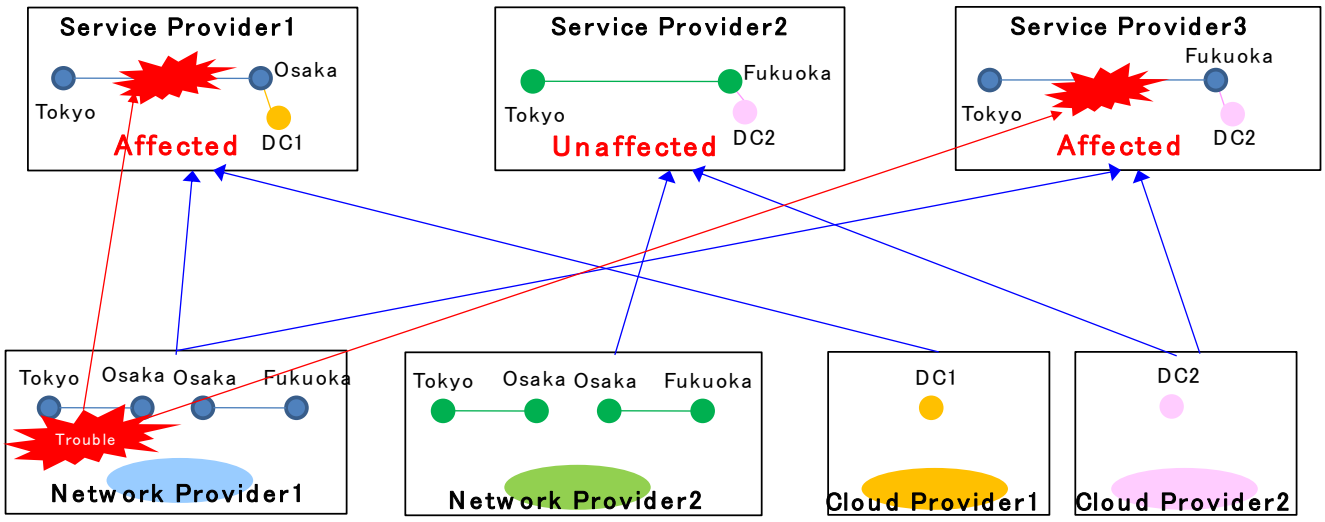


■ Use case 1

When trouble happened in any resources of NW/Cloud Providers, Service Providers can know their services are affected or not. The specific use case is following:

1. SPM collects configuration information of services provided by service providers in advance using Product Inventory Management API etc.
2. Each of Middle Bs(Service Provider1(SP1), Service Provider2(SP2), Service Provider3(SP3)) registers the notification destination to SPM SPI.
3. When a fault occurs, SPM receives a trouble ticket from the Network Provider1(NP1).
4. SPM creates a Service Problem based on the Trouble Ticket.
5. SPM notifies the Service Problem creation notification to Middle B (SP1, SP3) to notify expected service impact, based on the configuration information collected in advance.
6. When SPM receives a notification that the trouble ticket has changed to "In Progress" state, update the status of the relevant Service Problem. Notify the Service Problem state change notification to Middle B(SP1, SP3).

Middle B



First B

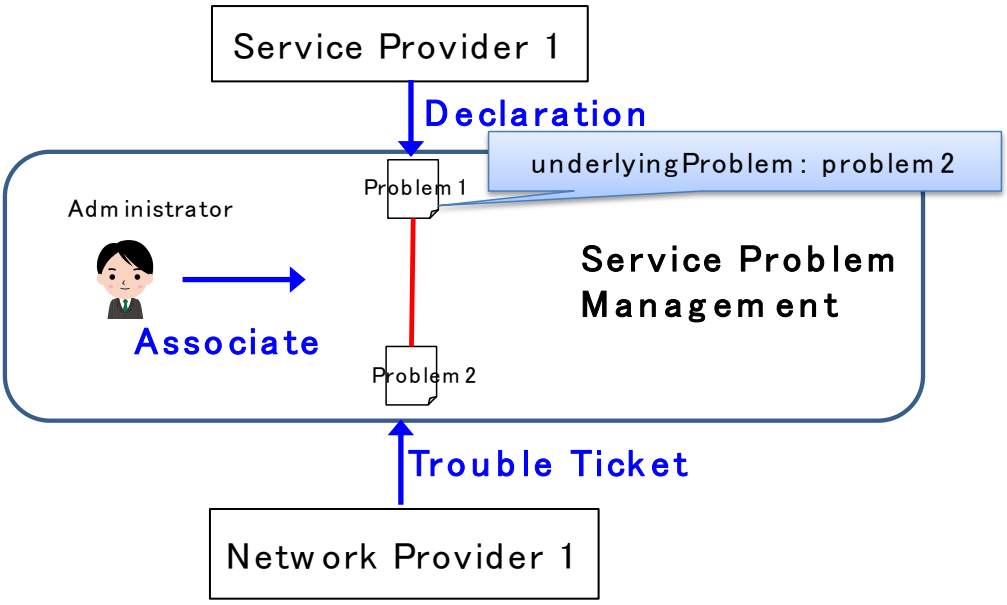
- Use case 2

To analyze the past problems, Middle B collects the problem information in the past one year.

- Use case 3

Service providers can declare a new service problem based on trouble declarations from their end-users. In addition, the SPM administrator can associate the service problem, based on the Middle B declaration, with another problem based on a First B event such as a Trouble Ticket. The specific use case is following:

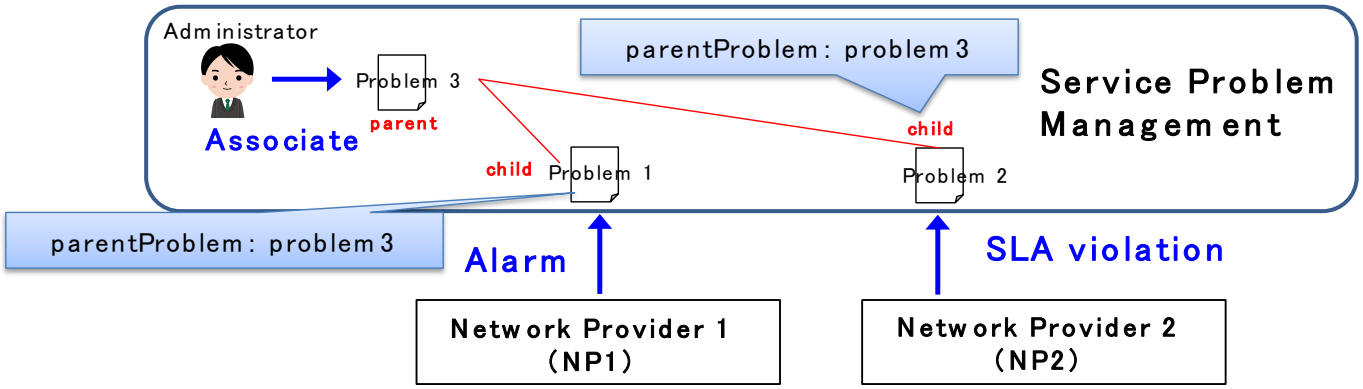
1. Based on the report from the user that there is a problem in the Internet access, Middle B (SP1) gets the current service problem.
2. After SPM collects the current Service Problems, returns that there are no problems related to the service of the Middle B (SP1).
3. In order to request the analysis of this event, Middle B declares a new service problem.
4. Since the SPM administrator found that necessary detailed information was insufficient, SPM administrator requests additional information about the behavior of the Middle B side.
5. SP1 collects the specified additional information and registers it.
6. SPM administrator checks the additional information, and accepts the Service Problem(Problem 1).
7. First B(NP1) registers a detected problem event to the trouble ticket, and notice a new trouble ticket generation to SPM. The SPM creates a Service Problem (Problem 2) based on the trouble ticket.
8. Since the two problems affects the same location, SPM administrator determines that the declared problem (Problem 1) and the new problem based on the new trouble ticket (Problem 2) is associated. SPM administrator associates Problem 1 with 2. In this case, Problem 1 will have Problem2 as the association “underlyingProblem”. Note that Problem 1 can have an “parentProblem” as another association if he would like to group those problems.
9. Since the Problem 1 was changed to add a “underlyingProblem”, Service Problem Change Notification is sent to SP1.



■ Use case 4

The SPM administrator can associate and group multiple service problems so that service providers can easily recognize what the real problem is. The specific use case is following:

1. SPM receives an alarm from NP1, and creates a Service Problem based on it (Problem 1).
2. SPM recieves an SLA violation from NP2 and creates a Service Problem based on it (Problem 2).
3. By analyzing problems, SPM administrator determines that Problem 1 and 2 are the same problem. SPM administrator creates a new Service Problem (Problem 3) in order to group and associate Problem 1, 2 and 3. In this case, Problem 3 is a parent and Problem 1 and 2 are children.



RESOURCE MODEL

Managed Entity and Task Resource Models

SERVICE PROBLEM

Example of the JSON representation of Service Problem Resource

```
{
  "id" : "problemxxxx0000",
  “correlationId”: “xxxxxx”,
  “originatingSystem”: “System_001”,
  “category”: “supplier.originated”,
  "href" : "http://api/serviceProblem/problemxxxx0000",
  "impactImportanceFactor" : "0",
  "priority" : "1",
  "description" : "connection failure between Tokyo and Osaka",
  “problemEscalation”: “0”,
  "timeRaised" : "2016-07-20 xx:xx:xx.xxx",
  "timeChanged" : "2016-07-20 xx:xx:xx.xxx",
  "statusChangeDate" : "2016-07-20 xx:xx:xx.xxx",
  “statusChangeReason”: “problem analysis has been completed in NP1”,
  "resolutionDate" : "2016-07-21 xx:xx:xx.xxx ",
  "status" : "resolved",
  “reason”: “Failure of resource ‘NP1_Resource_1’ in NP1”,
  “affectedServiceNumber”: “2”,
  “firstAlert”: {
      “type”: “Trouble Ticket”,
      "id" : "NP1_TT_0000000",
      "href" : "http://api/troubletiket/NP1_TT_000000"
   },
```



```
"responsibleParty" : {
   "role" : "Network Provider",
   "id" : "NP1",
   “href”: “http://api/party/NP1”
},
"originatorParty" : {
   "role" : "Network Provider",
   "id" : "NP1",
   “href”: “http://api/party/NP1”
 },
"relatedParty" : [
  {
     "role" : "Network Provider",
     "id" : "NP1",
     “href”: “http://api/party/NP1”
  },
  {
     "role" : "Service Provider",
     "id" : "SP1",
     “href”: “http://api/party/SP1”
  },
  {
     "role" : "Service Provider",
     "id" : "SP3",
     “href”: “http://api/party/SP3”
  }
 ],
"affectedService" : [
  {
     "id" : "NP1_Tokyo_Osaka",
     "href" : " http://api/NP1/service/NW1_Tokyo_Osaka",
```



```
  },
  {
    "id" : "NP1_Tokyo_xxxx",
    "href" : " http://api/NP1/service/NW1_Tokyo_xxxx",
  }
],
"affectedResource" : [
  {
    "id" : "NP1_RES_0001",
    "href" : "http://api/NP1/resource/NW1_RES_0001"
  },
],
"affectedLocation" : [
  {
   "id" : "Loc000000",
   "href" : "http://api/location/Loc000000/"
  },
  {
   "id" : "Loc000001",
   "href" : "http://api/location/Loc000001/"
  }
],
"associatedTroubleTicket" : [
  {
    "id" : "NP1_TT_0000000",
    "href" : "http://api/troubletiket/NP1_TT_000000"
  }
],
"underlyingAlarm" : [
   {
     "id" : "NP1_A_0000000",
```



```
      "href" : “http://api/alarm/NP1_A_000000”
    }
],
"associatedSLAViolation" : [
    {
      "id" : "NP1_SLA_0000000",
      "href" : “http://api/slaviolation/NP1_SLA_000000”
    }
],
“relatedEvent”: [
    {
         “eventType”: “prediction”,
         “eventId”: “prediction_0001”,
         “eventTime”: “2014-12-20T17:00:00Z”,
         “event”: { … }
    }
],
"relatedObject" : [
   {
     "id" : "product0001",
     “involvement”: “affected product”,
     "href" : "http://api/productinventory/product0001"
   }
 ],
"rootCauseService" : [
  {
     "id" : "NP1_Tokyo_Osaka",
     "href" : " http://api/NP1/service/NW1_Tokyo_Osaka",
  }
],
"rootCauseResource" : [
```



```
   {
     "id" : "NP1_Resource_1",
     "href" : " http://api/NP1/resource/NP1_Resource_1",
   },
 ],
"parentProblem" : [
   {
     "id" : "problemxxxx0001",
     “correlationId”: "xxxxxxxx",
     "href" : "http://api/serviceproblem/problemxxxx0001"
    }
 ],
 "underlyingProblem" : [
   {
     "id" : "problemxxxx0001",
     “correlationId”: "xxxxxxxx",
     "href" : "http://api/serviceproblem/problemxxxx0001"
    }
 ],

 "trackingRecord" : [
   {
     "description" : "yyy cleared the problem",
     “systemId”: “xxxx”,
     “time”: “2016-xx-xx xx:xx:xx”,
     “user”: {
        "id" : "NP1",
        “href”: “http://api/party/NP1”
    }
 ],
"comment" : [
```



```
    {
      “user”: {
         “id”: “SPM_handler_01”,
         “href”: “http://api/party/SPM_handler_01”,
      },
      “time”: "2016-xx-xx xx:xx:xx",
      “systemId”: “System_002”,
      “comment”: "receive trouble ticket from NP1, and create this Service Problem"
    },
    {
      “user”: {
        “id”: “NP1”,
        “href”: “http://api/party/NP1”
      },
      “time”: "2016-xx-xx xx:xx:xx",
      “systemId”: “System_002”,
      “comment”: "status changed to Progress-Held"
    }
  ],
  “impactPatterns”: “”,
  “extensionInfo”: [
     {
        “name”: “”,
        “value”: “”
     }
  ]
}
```


Fields Description

serviceProblem : The problem information for Middle B which is abstracted in the service layer from the issued event information by First B

Field	M/O	Description	SID
id	M	Identifier of the service problem	Y
correlationId	O	Additional identifier coming from an external system	N
originatingSystem	O	Indicates where the problem was generated	Y
category	M	Classifier for the problem. Settable. For example, this is used for distinguish the category of problem originator in [role].[category] format. ex) “serviceProvider.declared”, “supplier.originated”, “system.originated”…	Y
href	M	Reference to the Service Problem	N
impactImportanceFactor	O	Impact Importance is characterized by an Impact Importance Factor: overall importance of the impact of all the affected services, e.g. 0 (zero impact) to 100 (worst impact). The Impact Importance is a calculated field which is set by the OSS determining the impact.	Y
priority	O	An indication varying from 1(highest) to 10(lowest) of how important it is for the service provider to correct the Service Problem.	Y
description	O	Free form text describing the Service Problem	Y
problemEscalation	O	Indicates if this service problem has been escalated or not. Possible values are 0 to 10. A value of zero means no escalation. The meanings of values 1-10 are to be determined by the user of the interface, but they show increasing levels of escalation.	Y
timeRaised	M	Time the problem was raised	Y
timeChanged	M	Time the problem was last changed	Y
statusChangeDate	M	Time the problem was last status changed	Y
statusChangeReason	O	The reason of state change	N
resolutionDate	M	Time the problem was resolved	N
status	M	The current status of the service problem. Possible values are Submitted, Rejected, Acknowledged, In Progress [Held, Pending], Resolved, Closed, and Cancelled.	N

reason	M	Free text or optionally structured text. It can be Unknown.	Y
affectedServiceNumber	M	Number of affected services	Y
firstAlert	O	Indicates what first alerted the system to the problem. It is not the root cause of the Service Problem. Examples: Threshold crossing alert	Y
responsibleParty	M	Person or organization responsible for handling this problem.	Y
originatorParty	M	Person or organization that created the problem.	N
relatedParty	O	Party playing a role within the service problem	N
relatedObject	O	Objects linked with service problem	N
relatedEvent	O	Events linked with service problem	N
affectedService	O	List of affected services. Either "affectedResource", “affectedService” or "affectedLocations" should at least be present.	Y
affectedResource	O	List of affected resources. Either "affectedResource", “affectedService” or "affectedLocations" should at least be present.	Y
affectedLocation	O	List of affected locations. Either "affectedResource", “affectedService” or "affectedLocations" should at least be present.	Y
associatedTroubleTicket	O	List of trouble tickets associated with this problem.	Y
underlyingAlarm	O	A set of alarm ids identifying the alarms that are underlying this problem.	Y
associatedSLAViolation	O	List of SLA violation associated with this problem.	N
relatedEvent	O	List of events associated to this problem	N
relatedObject	O	List of objects associated to this problem	N
rootCauseService	O	Service(s) that are associated to the underlying service problems that are the Root Cause of this one if any (used only if applicable).	Y
rootCauseResource	O	Resource(s) that are associated to the underlying service problems that are the Root Cause of this one if any (used only if applicable).	Y
parentProblem	O	The parent problem to which this problem is attached.	Y
underlyingProblem	O	References to the underlying service problems. Only if this	Y

		problem is derived from other problems	
trackingRecord	O	Tracking records allow the tracking of modifications on the problem.The tracking records should not be embedded in the problem to allow retrieving the problem without the tracking records.	Y
comment	O	Comments on problem, as a list of comments	Y
impactPatterns	O	Define the patterns of impact (optional)- e.g. other service characteristics- Used when defining impact through another pattern than the pre-defined attributes.	Y
extensionInfo	O	A generic list of any type of elements. Used for vendor extensions or loose element encapsulation from other namespaces.	Y

firstAlert: Indicates what first alerted the system to the problem. It is not the root cause of the Service Problem. Examples: Threshold crossing alert.

Field	Description
type	Type of the object
id	Unique identifier of the object
href	Reference of the object

responsibleParty: Person or organization responsible for handling this problem

Field	Description
role	Role of the party
id	Unique identifier of the party
href	Reference of the party

originatorParty: Person or organization that created the problem

Field	Description
role	Role of the party
id	Unique identifier of the party
href	Reference of the party

relatedParty : A party playing a role within service problem

Field	Description
role	Role of the related party
id	Unique identifier of party

href	Reference of the party

affectedService: List of affected services. Either "affectedResource", “affectedService” or "affectedLocations" should at least be present.

Field	Description
id	Unique identifier of the service
href	Reference of the service

affectedResource: List of affected resources. Either "affectedResource", “affectedService” or "affectedLocations" should at least be present.

Field	Description
id	Unique identifier of the resource
href	Reference of the resource

affectedLocation : List of affected locations. Either "affectedResource", “affectedService” or "affectedLocations" should at least be present.

Field	Description
id	Unique identifier of the location
href	Reference of the location

associatedTroubleTicket : List of trouble tickets associated with this problem.

Field	Description
id	Unique identifier of the Trouble Ticket
href	Reference of the Trouble Ticket

underlyingAlarm : A set of alarm ids identifying the alarms that are underlying this problem.

Field	Description
id	Unique identifier of the Alarm
href	Reference of the Alarm

associatedSLAViolation : List of SLA violation associated with this problem.

Field	Description
id	Unique identifier of the SLA violation
href	Reference of the SLA violation

relatedEvent: Events linked with service problem.

Field	Description

eventType	Type of the event
eventId	ID of the event
eventTime	Time the event occurred
event	event itself

relatedObject: Objects linked with service problem.

Field	Description
type	Type of the object
id	Unique identifier of the object
href	Reference of the object

rootCauseService : Service(s) that are associated to the underlying service problems that are the Root Cause of this one if any (used only if applicable).

Field	Description
id	Unique identifier of the Service
href	Reference of the Service

rootCauseResource : Resource(s) that are associated to the underlying service problems that are the Root Cause of this one if any (used only if applicable).

Field	Description
id	Unique identifier of the Resource
href	Reference of the Resource

parentProblem : The parent problem(s) to which this problem is attached.

Field	Description
id	Unique identifier of the Problem
href	Reference of the Problem

underlyingProblem : References to the underlying service problems. Only if this problem is derived from other problems.

Field	Description
id	Unique identifier of the Problem
href	Reference of the Problem

trackingRecord : Tracking records allow the tracking of modifications on the problem. The tracking records should not be embedded in the problem to allow retrieving the problem without the tracking records.

Field	Description
description	Describes the action being done (ack,clear...)

systemId	Describes the system Id from which the action was done.
time	Describes the time at which the action was done
user	Describes the user of the user doing the action.
id	User id of the user doing the action.
href	Reference to the user doing the action.
identifier	The entity instance identifier EID
extensionInfo	A generic list of any type of elements. Used for vendor Extensions or loose element encapsulation from other namespaces

comment : Comments on problem, as a list of comments

Field	Description
user	Describes the user of the user doing the action
id	Describes the user id of the user doing the action
href	Reference to the user doing the action
time	Describes the time at which the action was done
systemId	Describes the system id from which the action was done
comment	Text of the comment

impactPatterns : Define the patterns of impact (optional)- e.g. other service characteristics- Used when defining impact through another pattern than the pre-defined attributes.

Field	Description
description	Basic description of the impact pattern
extensionInfo	a generic list of any type of elements. Used for vendor extensions or loose element encapsulation from other namespaces.

extensionInfo : A generic list of any type of elements. Used for vendor extensions or loose element encapsulation from other namespaces.

Field	Description
name	Name of the attribute
value	Value of the attribute

UML model:

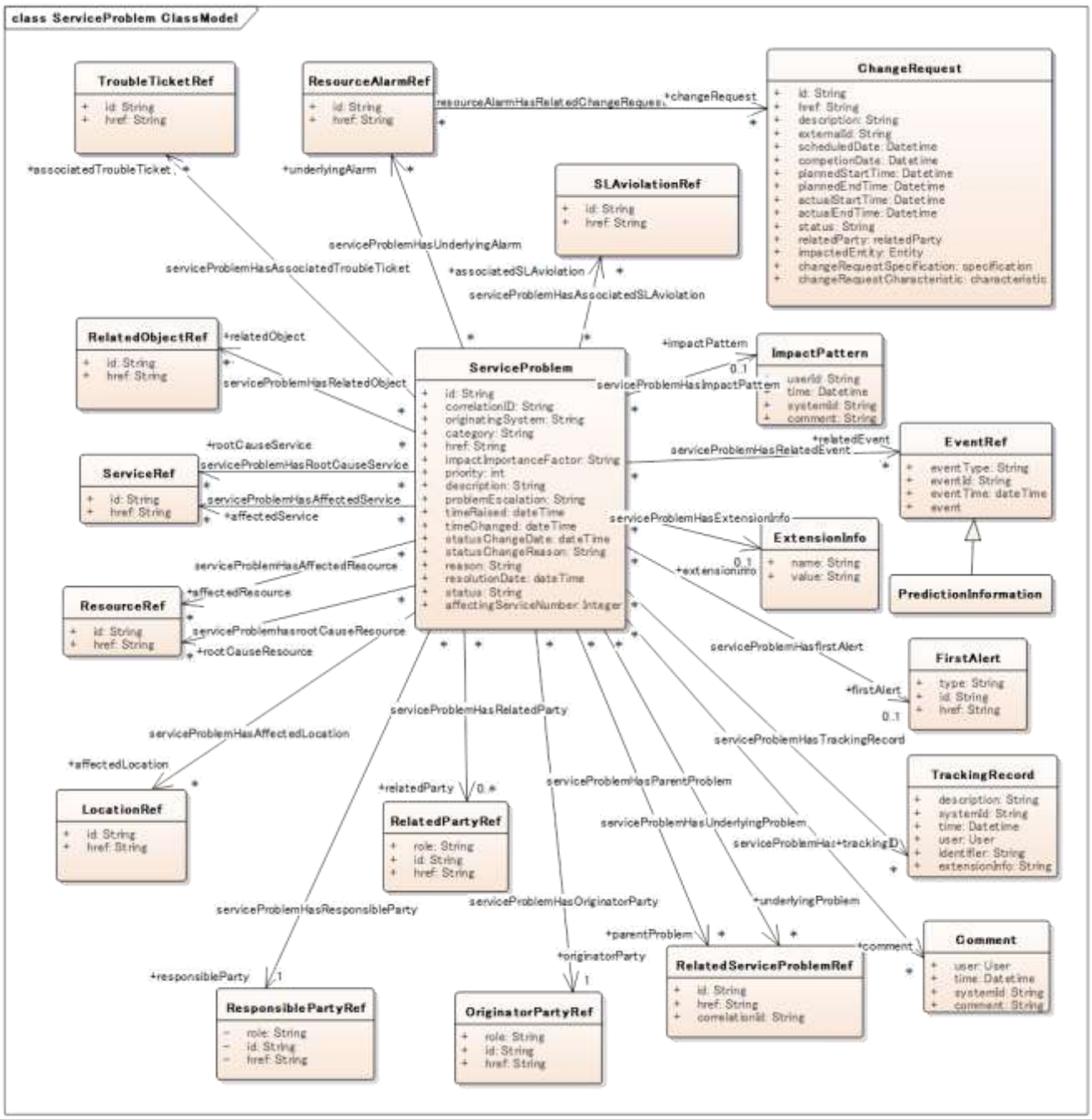


Figure 1 – Service Problem resource model

SERVICEPROBLEM LIFECYCLE

ServiceProblem states:

Following the available status values for a service problem are listed. The status value is in accordance with Trouble Ticket API. The state graphic gives an overview of the allowed status changes

- Submitted

- Rejected
- Acknowledged
- In Progress
  - Held
  - Pending
- Resolved
- Closed
- Cancelled

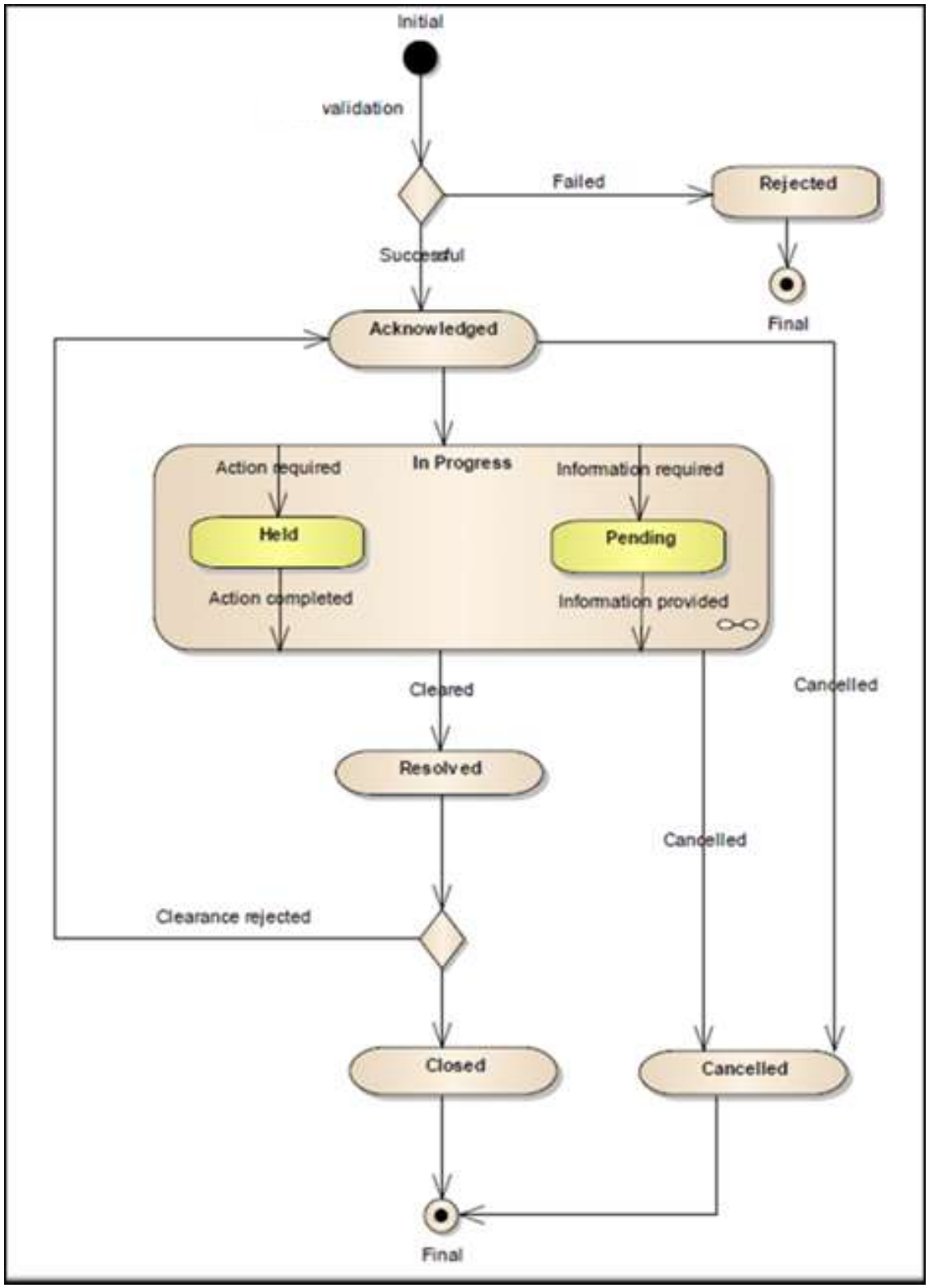


Figure 2 - LifeCycle

State	Description
Submitted	The initial state of a service problem when created by a service problem originator
Acknowledged	The Service Problem was accepted and allocated a unique service problem id by Service problem handler.
In Progress	The service problem was validated by the service problem handler and is being processed.
Resolved	The fault indicated in the service problem was corrected by the service problem handler and acknowledgement is awaited from its originator.
Closed	The service Problem's originator has acknowledged the 'Resolved' state of the service problem, or the timeframe for acknowledgement has passed without response from service problem originator.
Rejected	The service problem was rejected because it : • is not submitted • provides invalid information • fails to meet the Business rules in respect of the product which originator is raisinig a service problem against • is otherwize defective
In Progress – Pending	Service problem handler is awaiting further confirmation on details of a Fault from originator before it can progress the Fault. An example is where appointment information is required.
In Progress - Held	Service problem handler is confirming further details internally before completing a service problem. An example is where service problem handler for network infrastructure spare parts to progress with the fault rectification.
Cancelled	The service problem was cancelled because it : • Was cancelled by service problem originator

SERVICEPROBLEMEVENTRECORD

Example of the JSON representation of ServiceProblemEventRecord Resource

```
{
     "id" : "42",
     "href": "http://api/serviceProblem/serviceProblemEventRecord/42",
     "recordTime": "2016-08-08T10:45:30.0Z",
     "eventType": "ServiceProblemCreationNotification",
     "eventTime": "2016-08-08T10:45:25.0Z",
     "serviceProblemId": "SP001",
     "notification": {
       "eventType": "ServiceProblemCreationNotification",
       "eventTime": "2016-08-08T10:45:25.0Z",
       "eventId": "92775",
       "event":
       {
        "serviceProblem":
        {
            "id": "SP001",
            "href": "http://api/serviceProblem/SP001",
        Following a whole representation of the Service Problem with its all attributes
        See ServiceProblem resource.
      }
   }
```

Fields Description

serviceProblemEventRecord : Record of the ServiceProblemEvent

Field	M/O	Description	SID
id	M	Identifier of the service problem event record.	Y
href	M	Reference to the ServiceProblemEventRecord,	N
recordTime	M	Time at which the record was	N

		created	
eventType	M	Equals to event type of the recorded event	N
eventTime	M	Time of the record	N
serviceProblemId	M	ID of the service problem related to the event	N
notification	M	Must be one of the ServiceProblemNotification i.e. Service Problem Creation Notification, Service Problem Status Change Notification, Service Problem Change Notification, Service Problem Information Required Notification	Y (event payload)

UML model:

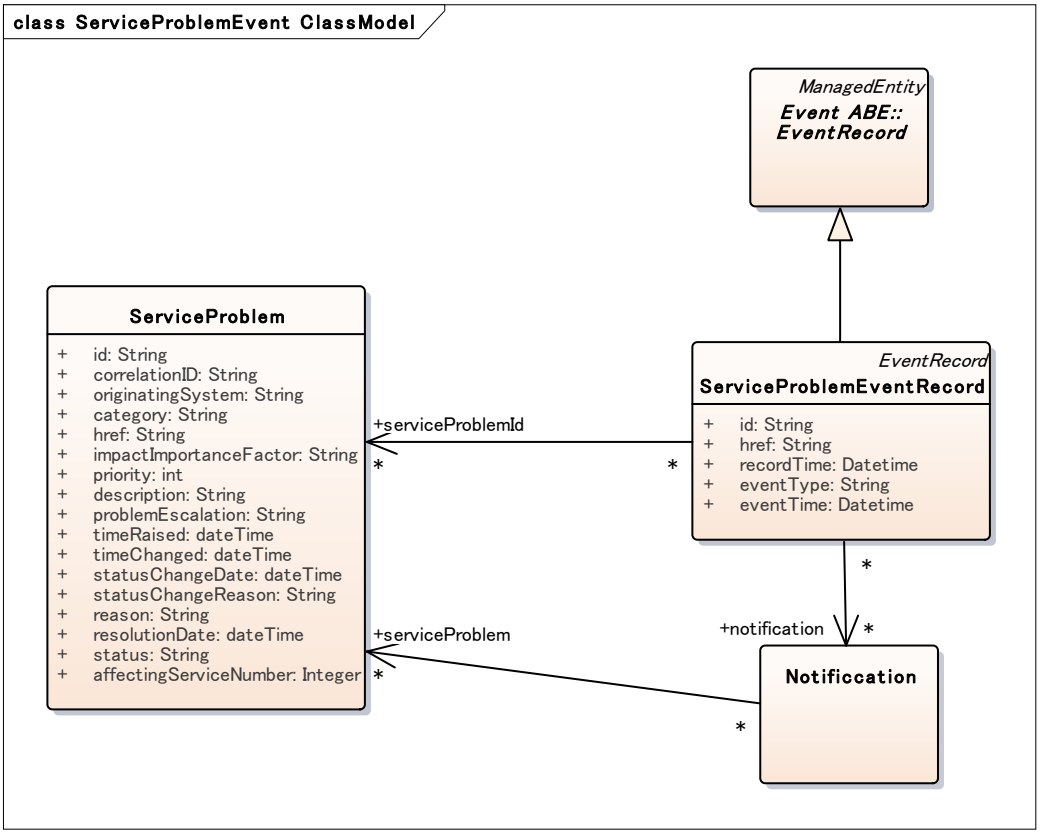


Figure 3 - Service Problem Event Record resource model

Event Models

The notification models for Service Problem Management are as follows:

UML model:

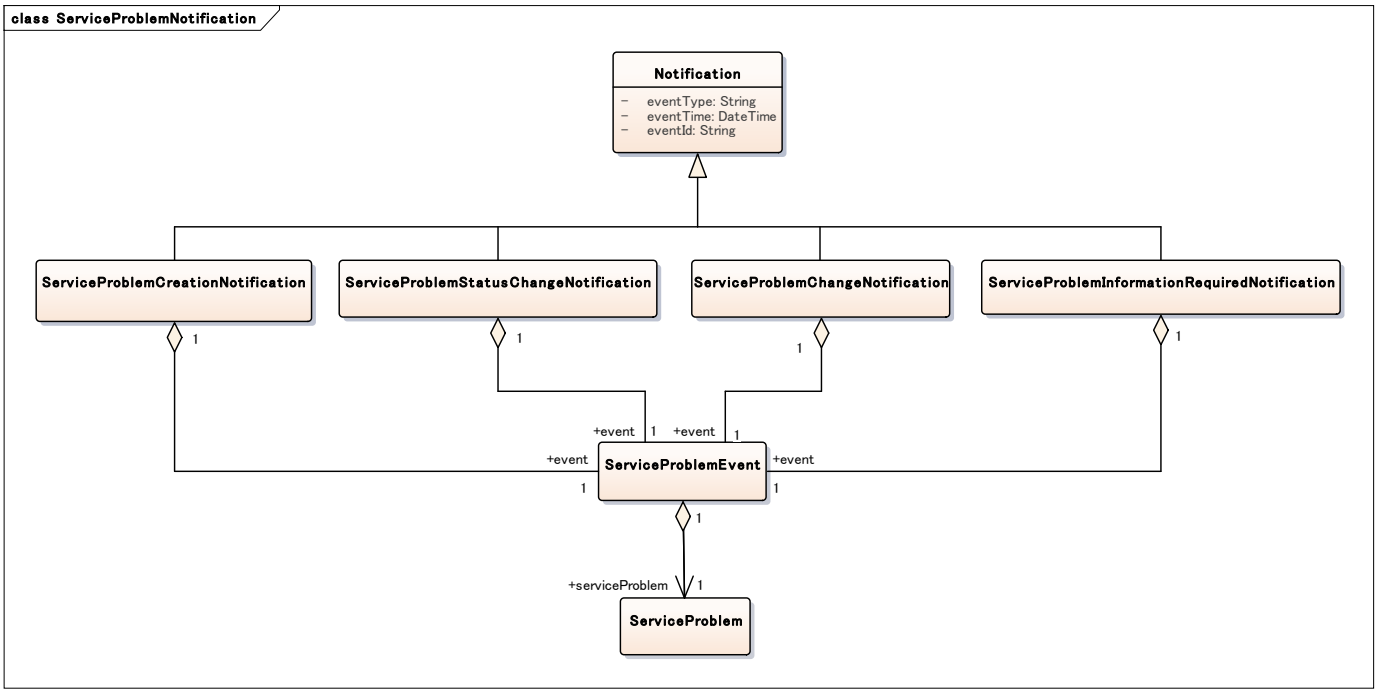


Figure 4 – Service Problem Notification Resource model

SERVICEPROBLEMCREATIONNOTIFICATION

Example of the JSON representation of ServiceProblemCreationNotification:

```
{
    "eventType": "ServiceProblemCreationNotification",
    "eventTime": "2016-08-08T10:45:25.0Z",
    "eventId": "92775",
    "event":
    {
        "serviceProblem":
        {
```



```
            "id": "problemxxxx0000",
            “href”: “http://api/serviceProblem/problemxxxx0000”,
            Optionally a whole representation of the Service Problem with its all attributes
        See ServiceProblem resource.
      }
   }
}
```

SERVICEPROBLEMSTATUSCHANGENOTIFICATION

Example of the JSON representation of ServiceProblemStatusChangeNotification:

```
{
    "eventType": "ServiceProblemStatusChangeNotification",
    "eventTime": "2016-08-08T10:45:25.0Z",
    "eventId": "92775",
    "event":
    {
        "serviceProblem":
        {
            "id": "problemxxxx0000",
            “href”: “http://api/serviceProblem/ problemxxxx0000”,
            "statusChangeDate" : "2016-08-08 T10:45:20.0Z",
            "status" : "Resolved",
            “statusChangeReason”: “”
      }
   }
}
```


SERVICEPROBLEMCHANGENOTIFICATION

Example of the JSON representation of ServiceProblemChangeNotification:

```
{
    "eventType": "ServiceProblemChangeNotification",
    "eventTime": "2016-08-08T10:45:25.0Z",
    "eventId": "92775",
    "event":
    {
        "serviceProblem":
        {
            "id": "problemxxxx0000",
            “href”: “http://api/serviceProblem/ problemxxxx0000”,
            "affectedService" : [
                {
                  "id" : "NP1_Tokyo_Osaka",
                  "href" : " http://api/NP1/service/NP1_Tokyo_Osaka"
                },
                {
                  "id" : "NP1_Tokyo_Yamanashi",
                  "href" : " http://api/NP1/service/NP1_Tokyo_Yamanashi"
                },
              ],
        Only changed attributes are provided
      }
   }
}
```

SERVICEPROBLEMINFORMATIONREQUIREDNOTIFICATION

Used to notify that some data in the order needs to be filled / is missing.

- “resourcePath” allows to precise if it is a data at the service problem level or at array level (and which one of them) that is missing
- “fieldPath” details which field is missing, its structure is quite similar to GET filter criteria
  - “missing=” points at the missing field
  - “&<criteria>” can be used to identify a specific element in lists

Simple example: originatorParty is missing

```
{
   "eventId":"00005",
   "eventTime":"2013-04-19T16:42:25-30:00",
   "eventType":"serviceProblemInformationRequiredNotification",
   "resourcePath":"/serviceProblem/42 ",
   "fieldPath":"missing=originatorParty",
   "event": {
     "serviceProblem":{
        "id":" 42",
        "href":"http://api/serviceProblem/42",
        "correlationID":  "CID_00001"
     }
   }
}
```

Complex example: in the extensionInfo, the name characteristic value is missing for the service problem 42.

```
{
   "eventId":"00006",
   "eventTime":"2013-04-19T16:42:25-30:00",
   "eventType":"serviceProblemInformationRequiredNotification",
   "resourcePath":"/serviceProblem/42/extensionInfo”,
   "fieldPath":"missing=serviceProblem.extensionInfo.name=”productName” ",
   "event": {
     "serviceProblem":{
```



```
        "id":" 42",
        "href":"http://api/serviceProblem/serviceProblem /42",
        "correlationId":"CID_0001"
     }
   }
}
```


API OPERATION TEMPLATES

For every single of operation on the entities use the following templates.

The following Uniform Contract rules are 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/serviceProblem/?{filter}&{attributeSelector}

This Uniform Contract operation is used to retrieve the representation of a service problem.

The operation is used to retrieve the service problems using some filtering information.

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

Description:

- As a search condition by specifying the “period”, makes it possible to acquire the service problems encountered specified past “period”.
- Identify the service problem from the services and resources that is affected by the trouble ticket or the alarm to associate with the service problem

Behavior:

- Return status codes
  - 200 OK - the request was successful
  - 400 Bad Request - error

REQUEST

GET /api/serviceProblem/
Accept: application/json

RESPONSE

200
Content-Type: application/json

```
[
  {
  "id" : "problemxxxx0000",
  “correlationId”: “xxxxxx”,
  “originatingSystem”: “System_001”,
  “category”: “supplier.originated”,
  "href" : "http://api/problem/problemxxxx0000",
  “problemEscalation”: “0”,
   “firstAlert”: {
      “type”: “Trouble Ticket”,
      "id" : "NP1_TT_0000000",
      "href" : "http://api/troubletiket/NP1_TT_000000"
```



```
 },
"responsibleParty" : {
   "role" : "Network Provider",
   "id" : "NP1",
   “href”: “http://api/party/NP1”
},
"originatorParty" : {
   "role" : "Network Provider",
   "id" : "NP1",
   “href”: “http://api/party/NP1”
 },
"relatedParty" : [
  {
     "role" : "Network Provider",
     "id" : "NP1",
     “href”: “http://api/party/NP1”
   },
  {
     "role" : "Service Provider",
     "id" : "SP1",
     “href”: “http://api/party/SP1”
   },
  {
     "role" : "Service Provider",
     "id" : "SP3",
     “href”: “http://api/party/SP3”
   }
 ],
"impactImportanceFactor" : "0",
"priority" : "1",
"description" : "connection failure between Tokyo and Osaka",
```



```
  "timeRaised" : "2016-07-20 xx:xx:xx.xxx",
  "timeChanged" : "2016-07-20 xx:xx:xx.xxx",
  "statusChangeDate" : "2016-07-20 xx:xx:xx.xxx",
  “statusChangeReason”: “SPM received the trouble ticket and created the service
problem”,
  "resolutionDate" : "2016-07-21 xx:xx:xx.xxx ",
  "status" : "Acknowledged",
  “reason”: “unknown”,
  “affectedServiceNumber”: “2”,
  "affectedService" : [
    {
      "id" : "NP1_Tokyo_Osaka",
      "href" : " http://api/NP1/service/NW1_Tokyo_Osaka",
    },
    {
      "id" : "NP1_Tokyo_xxxx ",
      "href" : " http://api/NP1/service/NW1_Tokyo_xxxx",
    }
  ],
  "affectedLocation" : [
   {
     "id" : "Loc000000",
     "href" : "http://api/location/Loc000000/"
   },
   {
     "id" : "Loc000001",
     "href" : "http://api/location/Loc000001/"
   },
  ],
  "associatedTroubleTicket" : [
    {
```



```
    "id" : "NP1_TT_0000000",
    "href" : "http://api/troubletiket/NP1_TT_000000"
  },
],
"relatedObject" : [
  {
    "id" : "product0001",
    “involvement”: “affected product”,
    "href" : "http://api/productinventory/product0001"
   }
 ],
},
{
"id" : "problemxxxx0001",
“category”: “serviceProvider.declared”,
"href" : "http://api/problem/problemxxxx0001",
“problemEscalation”: “0”,
"responsibleParty" : {
  "role" : "Service Provider",
  "id" : "SP1",
  “href”: “http://api/party/SP1”
},
"originatorParty" : {
  "role" : "Service Provider",
  "id" : "SP1",
  “href”: “http://api/party/SP1”
 },
"impactImportanceFactor" : "0",
"priority" : "1",
"description" : "trouble in the internet connection",
"timeRaised" : "2016-07-20 xx:xx:xx.xxx",
```



```
"timeChanged" : "2016-07-20 xx:xx:xx.xxx",
"statusChangeDate" : "2016-07-20 xx:xx:xx.xxx",
"resolutionDate" : "",
"status" : "Submitted",
“reason”: “unknown”,
“affectedServiceNumber”: “1”,
"affectedService" : [
  {
    "id" : "SP1_Service_001",
    "href" : " http://api/SP1/service/SP1_Service_001 ",
  }
]
```

POST API/SERVICEPROBLEM/

This Uniform Contract operation is used to completely update the representation of a Service Problem.

Middle B can declare a service problem that was not detected by the First B operators from the Middle B, It is possible to create and manage service problem

Description:

- This operation is used to create new service problem.

Behavior:

- Return status codes
  - 201 Created - the request was successful
  - 400 Bad Request - error

Specify which attributes are mandatory using the following table.

Field	Mandatory	Rule
id	N	id is generated by SPM
correlationId	N	correlationId is provided by external system
originatingSystem	N	
category	Y	
href	N	

impactImportanceFactor	N	
priority	Y	
description	Y	
problemEscalation	N	
timeRaised	N	
timeChanged	N	
statusChangeDate	N	
statusChangeReason	N	
resolutionDate	N	
status	N	
reason	Y	
affectedServiceNumber	N	Default value is 0.
firstAlert	N	
responsibleParty	N	
originatorParty	Y	
relatedParty	N	
relatedObject	N	
relatedEvent	N	
affectedResource	N	
affectedService	N	
affectedLocation	N	
associatedTroubleTicket	N	
underlyingAlarm	N	
associatedSLAviolation	N	
rootCauseService	N	
rootCauseResource	N	
parentProblem	N	
underlyingProblem	N	
trackingRecord	N	
comment	N	
impactPatterns	N	
extensionInfo	N	

REQUEST

```
POST API/serviceProblem/
Content-type: application/json

{
  "category": “serviceProvider.declared”,
  “priority”: “1”,
  "description" : "Internet connection error",
  “reason”: “unknown”,
  “originatorParty”: {
      “role”: “Service  Provider”,
      “id”:   “SP_00001”,
      “href”: “http://api/partymanagement/SP_00001”
  },
  "affectedService" : [
```



```
  {
    "id" : "SP00001_Service_001",
    "href" : "http://api/NP1/service/SP00001_Service_001“,
  },
 ],
 }
```


RESPONSE

201
Content-Type: application/json


```
{
 ‘‘id”: “sp_001”,
 ‘‘href”: ‘‘http://api/serviceproblem/sp_001”,
 "category”: “serviceProvider.declared”,
 “priority”: “1”,
 "description" : " Internet connection error",
 “reason”: “unknown”,
 “originatorParty”: {
     “role”: “Service Provider”,
     “id”:  “SP_00001”,
     “href”: “http://api/partymanagement/SP_00001”
  },
 "affectedService" : [
  {
    "id" : "SP00001_Service_001",
    "href" : " http://api/NP1/service/SP00001_Service_001 “
  },
}
```


PUT API/SERVICEPROBLEM/{ID}

Description :

- This Uniform Contract operation is used to completely update the representation of a service problem.
- 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 service problem – if you try to change the service problem ID itself an exception is returned. All fields with different values will be changed. If the request contains the same values like the current service problem 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/serviceProblem/1
Content-type: application/json

{
   “id”:  “1”,
   "category”: “supplier.originated”,
   “priority”: “1”,
   "description" : "connection failure between Tokyo and Osaka at 17:00",
   “reason”: “resource trouble in network provider 1”,
   “originatorParty”: {
      “role”: “Network Provider”,
      “id”:   “NP_00001”,
      “href”: “http://api/partymanagement/NP_00001”
  },
 "affectedService" : [
   {
     "id" : "NP1_Tokyo_Osaka",
     "href" : " http://api/NP1/service/NP1_Tokyo_Osaka"
   },
 ]
}
```

RESPONSE

```
201
Content-Type: application/json

{
   “id”:  “1”,
   "category”: “supplier.originated”,
   “priority”: “1”,
   "description" : "connection failure between Tokyo and Osaka at 17:00",
   “reason”: “resource trouble in network provider 1”,
   “originatorParty”: {
```



```
      “role”: “Network Provider”,
      “id”:   “NP_00001”,
      “href”: “http://api/partymanagement/NP_00001”
   },
  "affectedService" : [
    {
      "id" : "NP1_Tokyo_Osaka",
     "href" : "http://api/NP1/service/NP1_Tokyo_Osaka"
    },
  ]
}
```


PATCH API/SERVICEPROBLEM/{ID}

This Uniform Contract operation is used to partially update the representation of a Service Problem.

Description:

- This operation used to modify declared service problem .

Behavior:

- Return status codes
  - 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).

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

Field	Patchable	Rule
id	N	id is generated by SPM
correlationId	N	
originatingSystem	N	
category	Y	
href	N	
impactImportanceFactor	Y	
priority	Y	
description	Y	
problemEscalation	Y	
timeRaised	N	
timeChanged	Y	
statusChangeDate	Y	
statusChangeReason	Y	

resolutionDate	Y
status	Y
reason	Y
affectedServiceNumber	Y
firstAlert	N
responsibleParty	Y
originatorParty	Y
relatedParty	Y
relatedObject	Y
relatedEvent	Y
affectedService	Y
affectedResource	Y
affectedLocation	Y
associatedTroubleTicket	Y
underlyingAlarm	Y
associatedSLAviolation	Y
rootCauseService	Y
rootCauseResource	Y
parentProblem	Y
underlyingProblem	Y
trackingRecord	N
comment	Y
impactPatterns	Y
extensionInfo	Y

REQUEST

```
PATCH API/serviceProblem/sp_001
Content-type: application/merge-patch+json

{
  "description" : "connection failure between Tokyo and Osaka at 5:00"
}
```

RESPONSE

```
201
Content-Type: application/json

{
    ”id”:  “sp_001”,
    ”href”: “http://api/serviceProblem/sp_001”,
    "category”: “supplier.originated”,
    “priority”: “1”,
    "description" : "connection failure between Tokyo and Osaka at 5:00",
    “reason”: “unknown”,
    “originatorParty”: {
```



```
      “role”: “Network Provider”,
      “id”:   “NP_00001”,
      “href”: “http://api/partymanagement/NP_00001”
    },
 "affectedService" : [
   {
     "id" : "NP1_Tokyo_Osaka",
     "href" : " http://api/NP1/service/NP1_Tokyo_Osaka"
   }
  ]
}
```


REQUEST

PATCH /api/serviceProblem/sp_001
Content-type: application/json-patch+json


```
{
        "op": "add",
        "path": "/affectedService",
        "value": {
                "id": "44",
                "href": "http://api/serviceInventory/service/44",
        }
}
```


RESPONSE

201
Content-Type: application/json


```
{
    ”id”:  “sp_001”,
    ”href”: “http://api/serviceProblem/sp_001”,
    "category”: “supplier.originated”,
    “priority”: “1”,
    "description" : "connection failure between Tokyo and Osaka",
    “reason”: “unknown”,
    “originatorParty”: {
      “role”: “Network Provider”,
      “id”:   “NP_00001”,
```



```
      “href”: “http://api/partymanagement/NP_00001”
    },
 "affectedService" : [
  {
    "id" : "NP1_Tokyo_Osaka",
    "href" : " http://api/NP1/service/NP1_Tokyo_Osaka"
  },
  {
    "id" : "44",
    "href" : "http://api/serviceInventory/service/44"
  }
 ]
}
```


DELETE API/SERVICEPROBLEM/{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/serviceProblem/1
RESPONSE
200

GET /api/serviceProblem/serviceProblemEventRecord?{filter}&{attributeSelector}

Retrieve history of Service Problem resources.

Description:

- As a search condition by specifying the “period”, makes it possible to acquire the service problem event records encountered specified past “period”.

Behavior:

- Return status codes
  - 200 OK - the request was successful
  - 400 Bad Request - error

Note: This request should support range-based iterator pattern as described in TMF630.

REQUEST

```
GET /api/serviceProblem/serviceProblemEventRecord?eventTime>=2016-08-05T10:00:00.0Z&eventTime<=2016-08-10T10:00:00.0Z
Accept: application/json
```

RESPONSE

```
200
Content-Type: application/json
Content-Range: items 1-2/2

[
   {
      "id" : "42",
      "href": "http://api/serviceProblem/serviceProblemEventRecord/42",
      "recordTime": "2016-08-05T10:00:05.0Z",
      "eventType": "ServiceProblemCreationNotification",
      "eventTime": "2016-08-05T11:00:00.0Z",
      "serviceProblemId": "SP001",
      "notification": {
         "eventType": "ServiceProblemCreationNotification",
         "eventTime": "2016-08-05T11:00:00.0Z",
         "eventId": "92775",
```



```
        "event": {
           "serviceProblem": {
              "id": "problemxxxx0000",
      Following a whole representation of the Service Problem with its all attributes
      See ServiceProblem resource.
           }
        }
     }
  },
  {
     "id" : "43",
     "href": "http://api/serviceProblem/serviceProblemEventRecord/43",
     "recordTime": "2016-08-07T15:00:05.0Z",
     "eventType": "ServiceProblemChangeNotification",
     "eventTime": "2016-08-07T15:00:00.0Z",
     "serviceProblemId": "SP001",
     "notification": {
       "eventType": "ServiceProblemChangeNotification",
       "eventTime": "2016-08-07T15:00:00.0Z",
       "eventId": "92776",
       "event": {
         "serviceProblem": {
           "id": "problemxxxx0000",
      Following a whole representation of the Service Problem with its all attributes
      See ServiceProblem resource.
         }
       }
     }
  }
]
```


POST API/SERVICEPROBLEM/ACK

Description:

- This operation used to acknowledge service problems.

Behavior:

- Return status codes
  - 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).

Parameter:

Field	Input/Output	M/O	Multiplicity	Description
problems	I	M	1..*	
trackingRecord	I	O	0..1	
ackProblems	O	M	0..*	acknowledged Problems, i.e. the ones really acknowledged

REQUEST

```
POST API/serviceProblem/ack
Content-type: application/json

{
  “problems”: [
    {
        “id”: “41”,
        “href”: “http://api/serviceProblem/41”
    },
    {
        “id”: “42”,
        “href”: “http://api/serviceProblem/42”
    },
    {
        “id”: “43”,
```



```
        “href”: “http://api/serviceProblem/43”
    }
  ],
  “trackingRecord”: {
      "description" : "yyy ack the problem",
      “systemId”: “xxxx”,
      “time”: “2016-xx-xx xx:xx:xx”,
      “user”: {
          "id" : "NP1",
          “href”: “http://api/party/NP1”
      }
  }
}
```


RESPONSE


```
201
Content-Type: application/json

{
    “ackProblems”: [
      {
          “id”: “41”,
          “href”: “http://api/serviceProblem/41”
      },
      {
          “id”: “42”,
          “href”: “http://api/serviceProblem/42”
      },
      {
          “id”: “43”,
          “href”: “http://api/serviceProblem/43”
      }
  ]
}
```


POST API/SERVICEPROBLEM/UNACK

Description:

- This operation used to unacknowledge service problems.

Behavior:

- Return status codes
  - 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).

Parameter:

Field	Input/Output	M/O	Multiplicity	Description
problems	I	M	1..*	input list of ids to unack
trackingRecord	I	O	0..1	
unackProblems	O	M	0..*	problems really unacked

REQUEST

```
POST API/serviceProblem/unack
Content-type: application/json

{
  “problems”: [
    {
        “id”: “41”,
        “href”: “http://api/serviceProblem/41”
    },
    {
        “id”: “42”,
        “href”: “http://api/serviceProblem/42”
    },
    {
        “id”: “43”,
        “href”: “http://api/serviceProblem/43”
    }
  ],
   “trackingRecord”: {
```



```
        "description" : "yyy unack the problem",
        “systemId”: “xxxx”,
        “time”: “2016-xx-xx xx:xx:xx”,
        “user”: {
            "id" : "NP1",
            “href”: “http://api/party/NP1”
        }
    }
}
```


RESPONSE

201
Content-Type: application/json


```
{
    “unackProblems”: [
        {
            “id”: “41”,
            “href”: “http://api/serviceProblem/41”
        },
        {
            “id”: “42”,
            “href”: “http://api/serviceProblem/42”
        },
        {
            “id”: “43”,
            “href”: “http://api/serviceProblem/43”
        }
    ]
}
```


POST API/SERVICEPROBLEM/GROUP

Description:

- This operation used to group one or more child problems to a parent problem.

Behavior:

- Return status codes
    - 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).

Parameter:

Field	Input/Output	M/O	Multiplicity	Description
parentProblem	I	M	1	parent problem
childProblems	I	M	1..*	child problem to be associated with the parent

REQUEST

```
POST API/serviceProblem/group
Content-type: application/json

{
    “parentproblem”: {
          “id”: “41”,
         “href”: “http://api/serviceProblem/41”
    },
   “childproblems”: [
       {
          “id”: “42”,
          “href”: “http://api/serviceProblem/42”
       },
       {
          “id”: “43”,
          “href”: “http://api/serviceProblem/43”
       }
   ]
}
```

RESPONSE

```
201
Content-Type: application/json

{
          “id”: “42”,
         “href”: “http://api/serviceProblem/42”
```



```
        “parentProblem”: [
            {
               “id”: “41”,
               “href”: “http://api/serviceProblem/41”
            }
         ]
}

{
         “id”: “43”,
        “href”: “http://api/serviceProblem/43”
        “parentProblem”: [
            {
               “id”: “41”,
               “href”: “http://api/serviceProblem/41”
            }
         ]
}
```

POST API/SERVICEPROBLEM/UNGROUP

Description:

- This operation used to ungroup one or more child problems from a parent problem.

Behavior:

- Return status codes
  - 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).

Parameter:

Field	Input/Output	M/O	Multiplicity	Description
parentProblem	I	M	1	parent problem
childProblems	I	M	1..*	child problem to ungroup

REQUEST

POST API/serviceProblem/ungroup
Content-type: application/json

```
{
    “parentproblem”: {
          “id”: “41”,
         “href”: “http://api/serviceProblem/41”
    },
   “childproblems”: [
       {
          “id”: “42”,
          “href”: “http://api/serviceProblem/42”
       },
       {
          “id”: “43”,
          “href”: “http://api/serviceProblem/43”
       }
   ]
}
```

RESPONSE

201
Content-Type: application/json

```
{
          “id”: “42”,
         “href”: “http://api/serviceProblem/42”
         “parentProblem”: []
}

{
          “id”: “43”,
         “href”: “http://api/serviceProblem/43”
         “parentProblem”: []
}
```


API NOTIFICATION TEMPLATES

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
 	
{
 	
"event": {
 	
EVENT BODY
},
"eventType": "eventType"
}

RESPONSE
201 Content-Type: application/json

RELEASE HISTORY

Release Number	Date	Release led by:	Description
Release 1.0	04/15/2013	Pierre Gauthier TM Forum pgauthier@tmforum.org	First Release of Draft Version of the Document.
Release 1.1			Updated for use in the Paris Spec Jam – and rebranded.
Release 1.2	12/1/2016	Pierre Gauthier TM Forum pgauthier@tmforum.org	Version issued for Fx16.5

