Frameworx Specification

CATALOG MANAGEMENT API REST SPECIFICATION

TMF620

Date: October, 2013

Latest Update: Frameworx Release 13.5.0	**Member Evaluation**
Version 1.3.0	**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
TM Forum Web Page: www.tmforum.org

TABLE OF CONTENTS

LIST OF TABLES

INTRODUCTION

The following document is the specification of the REST API for Catalog Management. It includes the model definition as well as all available operations. Possible actions are retrieving Product Offerings, Product Specifications and Product Categories. Furthermore the HTTP GET allows filtering.

ISSUES IDENTIFIED DURING ITERATION

Each of the sections below have a parking lot table where changes made to the SID model that were made have been documented and highlighted where necessary for future review.

#	Description	Status
1	The mapping of API resources to the SID model doesn't necessarily require 1:1 mapping.	OPEN
2	Desire to support the ability to exclude specific attributes from a response (attribute filtering) as opposed to including attributes in a response.	OPEN
3	Should an attribute filter be used to GET a resource and no resource match the attribute filter, should a 200 or 204 response be provided. A: Use 200.	CLOSED
4	Move Parking lot items to addendum/appendix A: Done in v1.2 of this document.	CLOSED
5	ProductOffering doesn't contain any attributes/entities for versioning purposes in SID 13.	OPEN
6	The NBN xsd specifies a more detailed VersionDetail entity thank currently supported in SID 13.	OPEN
7	Commercial and Technical eligibility will be defined as part of a future iteration.	OPEN
8	Notifications/Eventing will be defined as part of a future iteration.	OPEN
9	PUT, POST, DELETE, PATCH operations are not expected to be required for the hackathon	OPEN
10	The model followed by the API should be relatively easily	CLOSED

	converted to Service and Resource models.	

RESOURCE MODEL

Managed Entity and Task Resource Models

PRODUCT CATEGORY

The product category resource is used to group product offerings in logical containers. Categories can contain other categories and/or product offerings.

Resource IDs for categories are strings and are defined by the catalog application.

Below is a representation of the Product Category resource in JSON format.

```
{
   "id": "42",
   "parentId": " http://serverlocation:port/catalogManagement/productCategory/41",
   "isRoot": "false",
   "name": "Cloud Services",
   "description": "A category to hold all available cloud service offers"
}}
```

Or for a root category (e.g. no parent)

```
{
   "id": "42",
   "parentId": "",
   "isRoot": "true",
   "name": "Cloud Services",
   "description": "A category to hold all available cloud service offers"
}}
```

PRODUCT OFFERING

The Product Offering resource represents entities that are orderable from the provider of the catalog, this resource includes pricing information.

Resource IDs are numeric and generated by the Catalog application.

Below is a representation of the Product Offering resource in JSON format.

Note: isBundle determines whether a productOffering represents a single productSpecification (false), or a bundle of productOfferings (true).

If false, then a productSpecification will be returned, but the bundledProductOfferings will be absent or empty and vice-versa if isBundle is true.

Below is a representation of the Product Offering resource in JSON format.

```
{
  "id": “42",
  "name": "Virtual Storage Medium",
  "description": "Virtual Storage Medium",
  "isBundle": "false",
  "productCategories": [{
    "id": “http://serverlocation:port/catalogManagement/productCategory/12",
    "name": "Cloud offerings",
    "description": "A group of cloud service offerings"
  }],
  "validFor": {
    "startDateTime": "2013-04-19T16:42:23-04:00",
    "endDateTime": "2013-06-19T00:00:00-04:00"
  },
  "productSpecification": {
    "id": “http://serverlocation:port/catalogManagement/productSpecification/13",
    "name": "specification 1",
    "description": "description 1"
  },
  "productOfferingPrice": [
    {
      "name": "Monthly Price",
      "description": "monthlyprice",
      "validFor": {
        "startDateTime": "2013-04-19T16:42:23-04:00",
        "endDateTime": "2013-06-19T00:00:00-04:00"
      },
      "priceType": "recurring",
      "unitOfMeasure": "",
      "price": {
        "amount": "12",
        "currency": "$"
      },
      "recurringChargePeriod": "monthly"
    },
    {
      "name": "Usage Price",
      "description": "usageprice",
      "validFor": {
        "startDateTime": "2013-04-19T16:42:23-04:00",
        "endDateTime": "2013-06-19T00:00:00-04:00"
      },
      "priceType": "usage",
      "unitOfMeasure": "second",
      "price": {
        "amount": "12",
        "currency": "$"
      },
      "recurringChargePeriod": ""
    }
  ],
  "bundledProductOfferings": {"productOffering": [
    "id": “http://serverlocation:port/catalogManagement/productOffering/15",
    "name": "Offering 1",
    "description": "description 1"
  ]}
}}
```

PRODUCT SPECIFICATION

For every single resource managed by the API provide a JSON based representation of the managed entities and tasks.

You can start with an XML representation but remember that the default representation will be JSON.

Also remember that you representation must be based on the SID at least from a conceptual view point.

Also define the structure of the Resource IDs.

Note: The configurable attribute on the productSpecCharacteristics determines if an instance of the productSpecification can override the value of the attribute. If set to true, the value can be overridden, if set to false, the value is set by the catalog and cannot be changed.

Below is a representation of the Product Specification resource in JSON format.

```
{
  "id": "22",
  "name": "iPhone 42",
  "description": "Siri works on this iPhone",
  "brand": "Apple",
  "validFor": {
     "startDateTime": "2013-04-19T16:42:23-04:00",
     "endDateTime": "2013-06-19T00:00:00-04:00"
  },
  "productSpecCharacteristics": [
     {
        "name": "Screen Size",
        "description": "la dimension de l'ecran",
        "valueType": "number",
        "configurable": "false",
        "ProductSpecCharacteristicValue": [{
           "valueType": "number",
           "default": "true",
           "value": "4.2",
           "unitOfMeasure": "inches",
           "valueFrom": "",
           "valueTo": ""
        }]
     },
     {
        "name": "Colour",
        "description": "La couleur du bidule",
        "valueType": "string",
        "configurable": "true",
        "ProductSpecCharacteristicValue": [
           {
              "valueType": "string",
              "default": "true",
              "value": "Black",
              "unitOfMeasure": "",
              "valueFrom": "",
              "valueTo": ""
           },
           {
              "valueType": "string",
              "default": "false",
              "value": "White",
              "unitOfMeasure": "",
              "valueFrom": "",
              "valueTo": ""
           }
        ]
        }
    ]
}}
```

PRODUCT OFFERING PRICE

This is documented here for reference as a data type, not a resource:

```
{
   "productOfferingPrice": {
      "name": "MonthlyPrice",
      "description": "Monthly Price",
      "validFor": {"startDateTime":"2013-04-19T16:42:23-04:00",
            "endDateTime":"2013-06-19T00:00:00-04:00"
            },
      "priceType": "recurring",
      "unitOfMeasure": "",
      "price": {
         “amount” : "12",
         “currency” : “$”
      },
      "recurringChargePeriod": "month"
   }
}
```

PRODUCT OFFERING PRICE

This is documented here for reference as a data type, not a resource:

```
{
   "productOfferingPrice": {
      "name": "MonthlyPrice",
      "description": "Monthly Price",
      "validFor": {"startDateTime":"2013-04-19T16:42:23-04:00",
            "endDateTime":"2013-06-19T00:00:00-04:00"
            },
      "priceType": "recurring",
      "unitOfMeasure": "",
      "price": {
         “amount” : "12",
         “currency” : “$”
      },
      "recurringChargePeriod": "month"
   }
}
```

API OPERATIONS TEMPLATES

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 /catalogManagement/productCategory

Note that collections can be retrieved via GET /API/<productCategory> with no {ID}

Description :

- This operation retrieves productCategories from a catalog.
- Filtering is enabled on all attributes, particularly due to the hierarchical nature of productCategories, children of a productCategory are obtained by filtering on the parentId attribute of the resource.
- Attribute selection is not enabled for this simple resource
- The resource represents a managed entity or a collection depending on the query pattern.
- The identifier is a string that can consist of numbers, not necessarily alphanumeric.

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

```
GET /api/catalogManagement/productCategory/{ID}/?{fields=attributes}&{filtering expression}
Accept: application/json
```

RESPONSE

```
[{
   "id": “http://serverlocation:port/catalogManagement/productCategory/42",
   "parentId": “http://serverlocation:port/catalogManagement/productCategory/41",
   "isRoot": "false",
   "name": "Cloud Services",
   "description": "A category to hold all available cloud service offers"
}]
```

GET /catalogManagement/productOffering/{ID}

Note that collections can be retrieved via GET /API/< productOffering > with no {ID}

Description :

- This operation returns all productOfferings from the catalog, unless an ID is specified in which case a specific productOffering resource would be returned.
- The resource instance being returned is a productOffering or an array of product offerings if the query returns multiple resources (see earlier specification in this document)
- Filtering is enabled on all productOffering attributes.
  - Note: In order to address the Nice 2013 Catalog Management eHealth Catalyst, a query on the categoryId attribute would be executed.
    - E.g. GET /catalogManagement/productOffering?**productCategories.id=12**
- Attribute selection is enabled.
- The resource is either a managed entity or a collection depending on the query pattern.
- The ID may be a string (or a string containing numbers).

Behavior :

- What status and exception codes are returned.
  - 200 if no productOffering found for supplied categoryId (200) (filter expression)
  - 404 Not found when the supplied ID doesn't match a known productOffering.
- Returns HTTP/1.1 status code 200 if the request was successful.
- Any other special return and/or exception codes.

REQUEST

```
GET /catalogManagement/productOffering/{ID}/?{fields=attributes}&{filtering expression}
Accept: application/json
```

RESPONSE

```
[
{
   "id": "http://serverlocation:port/catalogManagement/productOffering/42",
   "name": "Virtual Storage Medium",
   "description": "Virtual Storage Medium",
   "isBundle": "false",
   "productCategories": [{
      "id": “http://serverlocation:port/catalogManagement/productCategory/12",
      "name": "Cloud offerings",
      "description": "A group of cloud service offerings"
   }],
   "validFor": {
      "startDateTime": "2013-04-19T16:42:23-04:00",
      "endDateTime": "2013-06-19T00:00:00-04:00"
   },
   "productSpecification": {
      "id": “http://serverlocation:port/catalogManagement/productSpecification/13",
      "name": "specification 1",
      "description": "description 1"
   },
   "productOfferingPrice": [
      {
         "name": "Monthly Price",
         "description": "monthlyprice",
         "validFor": {
            "startDateTime": "2013-04-19T16:42:23-04:00",
            "endDateTime": "2013-06-19T00:00:00-04:00"
         },
         "priceType": "recurring",
         "unitOfMeasure": "",
         "price": {
            "amount": "12",
            "currency": "$"
         },
         "recurringChargePeriod": "monthly"
      },
      {
         "name": "Usage Price",
         "description": "usageprice",
         "validFor": {
            "startDateTime": "2013-04-19T16:42:23-04:00",
            "endDateTime": "2013-06-19T00:00:00-04:00"
         },
         "priceType": "usage",
         "unitOfMeasure": "second",
         "price": {
            "amount": "12",
            "currency": "$"
         },
         "recurringChargePeriod": ""
      }
```

```
    ],
    "bundledProductOfferings": {"productOffering": [
        "id": “http://serverlocation:port/catalogManagement/productOffering/15",
        "name": "Offering 1",
        "description": "description 1"
    ]}
}
]
```

In order to address the Nice 2013 Catalog Management eHealth Catalyst (product catalog step 3 and 4), a query on the categoryId attribute would be executed, however only a limited set of attributes is initially returned (id, name, description, productSpecifications and validFor).

The query pattern in this case would be as follows:

GET
/catalogManagement/productOffering/fields=**name,id,description,validFor,product Specification,isBundle,bundledProductOfferings**?productCategories.id=12

RESPONSE for bundled ProductOffering

```
[
{
    "id": "http://serverlocation:port/catalogManagement/productOffering/42",
    "name": "Virtual Storage Medium",
    "description": "Virtual Storage Medium",
    "isBundle": "true",
    "validFor": {
        "startDateTime": "2013-04-19T16:42:23-04:00",
        "endDateTime": "2013-06-19T00:00:00-04:00"
    },
    "productSpecification": {},
    "bundledProductOfferings": [{
        "id": "http://serverlocation:port/catalogManagement/productOffering/15",
        "name": "Offering 1",
        "description": "description 1"
    },
    {}
    ]
}
]
```

RESPONSE for simple ProductOffering

```
[{
    "id": "http://serverlocation:port/catalogManagement/productOffering/42",
    "name": "Virtual Storage Medium",
    "description": "Virtual Storage Medium",
    "isBundle": "false",
    "validFor": {
       "startDateTime": "2013-04-19T16:42:23-04:00",
       "endDateTime": "2013-06-19T00:00:00-04:00"
    },
    "productSpecification": {
       "id": "http://serverlocation:port/catalogManagement/productSpecification/13",
       "name": "specification 1",
       "description": "description 1"
    },
    "bundledProductOfferings": []
}]
```

GET /catalogManagement/productSpecification/{ID}

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

Note that collections can be retrieved via GET /API/<RESOURCE> with no {ID}

Description :

- This operation returns productSpecifications
- The resource instance being returned is a productSpecification or an array of productSpecifications if the query returns multiple specifications (see earlier specification in this document)
- Filtering is enabled on all productSpecification attributes.
- Attribute selection is enabled.
- The resource represents a managed entity or a collection.
- The ID may be a string (or a string containing numbers).

Behavior :

- What status and exception codes are returned.
    - 404 Not found when the supplied ID doesn't match a known productSpecification
- Returns HTTP/1.1 status code 200 if the request was successful.
- Any other special return and/or exception codes.

REQUEST

```
GET /catalogManagement/productSpecification/{ID}/?{fields=attributes}&{filtering expression}
Accept: application/json
```

RESPONSE

```
[
{"ProductSpecification": {
  "id": "http://serverlocation:port/catalogManagement/productSpecification/22",
  "name": "iPhone 42",
  "description": "Siri works on this iPhone",
  "brand": "Apple",
  "validFor": {
    "startDateTime": "2013-04-19T16:42:23-04:00",
    "endDateTime": "2013-06-19T00:00:00-04:00"
  },
  "productSpecCharacteristics": [
    {
      "name": "Screen Size",
      "description": "la dimension de l'ecran",
      "valueType": "number",
      "configurable": "false",
      "ProductSpecCharacteristicValue": [{
        "valueType": "number",
        "default": "true",
        "value": "4.2",
        "unitOfMeasure": "inches",
        "valueFrom": "",
        "valueTo": ""
      }]
    },
    {
      "name": "Colour",
      "description": "La couleur du bidule",
      "valueType": "string",
      "configurable": "true",
      "ProductSpecCharacteristicValue": [
        {
          "valueType": "string",
          "default": "true",
          "value": "Black",
          "unitOfMeasure": "",
          "valueFrom": "",
          "valueTo": ""
        },
        {
          "valueType": "string",
          "default": "false",
          "value": "White",
          "unitOfMeasure": "",
          "valueFrom": "",
          "valueTo": ""
```

```
                }
            ]
        }
    ]
}}
]
```

In order to address the Nice 2013 Catalog Management eHealth Catalyst (product catalog step 5) where the productSpecifications for a productOffering are requested, a query on the productSpecificationId returned in step 4 would be executed.

The query pattern in this case would be as follows:

GET /catalogManagement/productSpecification/**12**

Note, in the case of a complex (isBundle) productOffering, the retrieval of productSpecification would require an iteration though the various specificationIds, or a query on multiple productSpecifications through query filtering.

For example:

GET /catalogManagement/productSpecification?id=**12,13,14**

PUT API/{RESOURCE}/{ID}

No PUT operations have been defined in this iteration.

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.

REQUEST

```
PUT API/{RESOURCE}/{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/{RESOURCE}/{ID}

No PATCH operations have been defined in this iteration.

This Uniform Contract operation is used to partially 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.

REQUEST

```
PATCH API/{RESOURCE}/{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/{RESOURCE}/{ID}

No POST operations have been defined in this iteration.

This Uniform Contract operation is used to create 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.
- Describe what are the mandatory attributes that must be provided when you create the entity.

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.

REQUEST

```
POST API/{RESOURCE}
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/{RESOURCE}/{ID}

No DELETE operations have been defined in this iteration.

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/{RESOURCE}/{ID}
RESPONSE
200

Example see TMF REST Design Guidelines.

API NOTIFICATION TEMPLATES

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.

No NOTIFICATION operations have been defined in this iteration.

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

Example see TMF REST Design Guidelines.

APPENDIX – NBNCO REQUIREMENTS REVIEW

PC-BP001: QUERY PRODUCT CATALOGUE

- Get List of products and version
    - Maps onto productOffering resource in the catalogManagement API.
    - productOffering resource doesn't include version information (version number and detail) in this version of API.
- Get Product Catalogue specification
    - Maps onto productSpecification resource in the catalogManagement API.
    - GET productSpecification, filter on id provided by productOffering resource from GET productOffering.
    - productSpecification doesn't include version information (version number and detail) in this version of the API.

PRODUCTV4.XSD REVIEW

- Version (version number and detail) is not catered for in this iteration of the interface specification. It will be reviewed as part of future iterations.
    - The xsd refers to a version root entity in Product, ProductSpecification and ProductOffering

APPENDIX – PARKING LOT

The parking lot below lists changes made to the SID model for the purpose of simplifying the first cut of the interface, notes have been made where appropriate where entities were thought to be relevant in the future for API enabled updates.

PRODUCT CATEGORY

Parking Lot	
productCategory is linked to productOffering through a relationship to productSpecificationType.	Need to discuss the relationship between category and offering later.
Added Id to productCategory class	
Renamed “type” attribute to “name” on productSpecificationType	
Removed productSpecificationType	

PRODUCT OFFERING

Parking Lot	
Removed productOffering.status	Only dealing with active products for now, obsolete, etc… will be taken care of later
Removed businessInteraction relationship for now	Might be added at a later stage
Removed productCapacity	Might be added at a later stage
Removed productCapacityDemand	
Removed partyRole	
Removed partyProfileType	Might be required for eligibility
Removed partyDemographic	Might be required for eligibility
Removed partyProfileTypeCharacteristic	Might be required for eligibility
Removed productOfferingTerm	Might be required for eligibility

Parking Lot	
Removed product	
Removed Place	Might be required for eligibility
Removed productSpecCharUse	Not required for query, maybe require for update
Removed salesChannel	May be re-introduced at a later stage to enable API to filter by sales channel. Question is whether this should be allowed or managed internally by catalog application and security.
Removed distributionChannel	May be re-introduced at a later stage to enable API to filter by distribution channel. Question is whether this should be allowed or managed internally by catalog application and security.
Removed productOfferingRelationship self-association	Not required for query
Removed marketStrategy	Related to processes internal to the service provider, not required for query
Removed productCatalog	Assuming the consumer of the API knows which catalog to interact with
Removed distChannelProdOffer	
Removed productOfferingStrategy	
Removed prodOfferPricePolicyVariable	May be re-introduced at later stage
Removed PolicySet	May be re-introduced at later stage
Removed resourceCandidate	Not required for Query, may be needed for update
Removed competitorProductCorrelation	
Removed competitorIntelligence	
Removed marketSegment	Internal to SP
Removed marketSegmentCharacteristics	Internal to SP

Parking Lot	
Removed marketStatistics	Internal to SP
Removed serviceLevelSpecification	May be needed in the future for eligibility/filter criteria
Removed serviceCandidate	Not required for Query, may be needed for update
Removed bundledProductOffering relationship from productOffering base class (bundledProductOffering -> productBundle)	But kept association from bundled to offering
Removed simpleProductOffering relationship to productComponent	But kept the opposing relationship
Removed productSpecification1 to simpleProductOffering	
Note: Review productOfferingPrice relation to productSpecCharacteristicValue	
From Pricing. Only kept productOfferingPrice and added it as attributes to the productOffering	
Removed attributes on the association on bundledProductOfferingOptions	
Removed the compositeProductOfferingPrice from productOfferingPrice	Retained multiple productOfferingPrice to productOffering, and also retained productOfferingPriceAlterations
Removed alteredProductOfferingPrice from the entity	Not needed for hackathon, but needs to be revisited in future iterations
Need to converge on having a unique representation independent of the use case (consumer vs administrator)	
Review localization of prices and tax in pricing (SID)	

PRODUCT SPECIFICATION

Parking Lot	
Removed lifecycleStatus attribute from productSpecification	Only dealing with active specifications
Removed businessInteractionItem from productSpecification	
Removed productCapacity from productSpecification	
Remove productCapacityDemand from productSpecification	
Removed productOffering from productSpecification	Only useful for admin purposes, not consumption via API
Removed simpleProductOffering from productSpecification	
Removed customerBillingProductChargesumspec2 from productSpecification	
Removed compositeProductSpecification from productSpecification	
Removed productSpecificationCost from productSpecification	Internal to SP
Removed customerFacingServiceSpec from productSpecification	To be reviewed at later date
Removed resourceSpecification from productSpecification	Not required for digital services
Removed productSpecification relationship to itself (productSpecification1)	Will likely be back in scope later for eligibility checks
Removed productOfferingPrice from productSpecification	
Removed productUsageSpec from productSpecification	Not required
Removed physicalResourceSpec from productSpecification	Not required for digital services

Parking Lot	
Removed serviceLevelSpecification from productSpecification	
Removed atomicProductUsageSpec from atomicProductSpecification	
Removed compositeProductUsageSpec from CompositeProductSpecification	
Removed unique on productSpecCharacteristic	
Removed productSpecificationVersion	Will need to be reviewed post hackathon
Edited productSpecificationType	Removed all attributes except type and description
Note, there may be duplicate "ProductSpecCharacteristicValue" in the resource	

PRODUCT SPEC CHARACTERISTICS

Parking Lot	
Removed minCardinality	
Removed maxCardinality	
Removed derivationFormula	
Removed validFor	During the hackathon, time span will not be sufficient for this data to be useful
Removed rangeInterval	Simplified for the hackathon
???	pricingLogicAlgorithmSpec (TBD)
Removed productSpecCharacteristic and productSpecCharacteristic1 self-association	
Removed productSpecification	
Removed resourceSpecCharacteristics	Not required for Digital services
Removed serviceSpecCharacteristics	Not required for Digital services
Note: _default and _valueType are reserved XML words	

RELEASE HISTORY

Version Number	Date	Release led by:	Description
Version 1.2	27th July 2013	Pierre Gauthier TM Forum pgauthier@tmforum.org	First Release of Draft Version of the Document.

Version 1.3	2nd October 2013	Tina O'Sullivan (TM Forum)	Updates & corrections.

ACKNOWLEDGEMENTS

Release Number	Date	Contributor	Company	Email
1.2	27th July 2013	David Gandy Greg Scullard Josh Salomon Laurent Leboucher Maxime Delon Nancy Jansson Pierre Gauthier	Cisco Cisco Amdocs Orange Orange Ericsson TM Forum	dagandy@cisco.com gscullar@cisco.com joshs@amdocs.com laurent.leboucher@orange.com maxime.delon@orange.com nancy.jansson@ericsson.com pgauthier@tmforum.org