Frameworx Technical Report

Dynamic API Technical Recommendation

Specification & Metadata approach to allow for dynamic payloads and message validation

TR254
Release 15.5.1
April 2016

Latest Update: Frameworx Release 15.5	**TM Forum Approved**
Version 1.1.2	**IPR Mode: RAND**

Notice

this TM FORUM Standards Final Deliverable, can be obtained from the TM FORUM Team Administrator. TM FORUM makes no representation that any information or list of intellectual property rights will at any time be complete, or that any claims in such list are, in fact, Essential Claims.

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

Executive Summary

Emerging technologies (such as SDN, NFV & IoT) have enabled a new breed of services that require a very high degree of collaboration and integration with various partners in order to fulfill and assure these services end to end. In addition, due to the nature of these emerging technologies as well as market research that sees First-Movers gain significant advantages [1], there is a need for service providers to be able to rapidly introduce these services to market. The traditional and longer delivery cycles (in the order of months) are no longer sufficient and a new, more 'agile' method of service delivery is required to deliver these products in the order of weeks.

In order to achieve this, service providers need a way to quickly integrate their systems with each other to enable a truly agile, dynamic marketplace.

As part of the B2B2X body of work it has become apparent that there is a need for both buyer and supplier to be able to exchange data dynamically, which is to say without having to modify their interfaces every time a new product or business interaction changes. IT systems need a way to cope with dynamically changing interface exchanges. These systems need to mimic the semantic web model.

Key considerations necessary to support a dynamic API include:

- Metadata – set of data that describes and gives information about other data
- Policy – manage the definition, change, and configuration of other entities
- Behavior – allow behavior to be added to an individual object, either statically or dynamically, without affecting the behavior of other objects from the same class

This initial technical recommendation looks specifically at the topic of metadata. Future versions will introduce the additional topics of policy and behavior in support of dynamic APIs.

The proposal introduces an approach for defining a dynamic interface that supports returning a definition for a particular entity as well as rules on how to interpret, validate and manage that entity. Which entities this approach is applied to is up to the implementer, but certain SID entities will benefit more than others.

This document aims at introducing a pattern by which the principles of the SID can be used to facilitate a physical implementation (by way of an API). The examples shown employ the use of external schemas (such as JSON schema) to define data primitives, however more work is needed in order to extend the design pattern to include more complex data types. This is planned to be addressed in a future contribution.

1. Basic Premise

The proposal is to use the SID CharacteristicSpec/Characteristic modelling pattern in addition to typical schema based validation to provide a way for a consumer of an interface to interpret and understand how a particular entity can be governed without that having to be coded/known beforehand. This addresses the specific concern of supporting dynamic attributes using industry defined schema (e.g. JSON Schema, XSD)

The typical example is the Product Catalog API[2]. There is an existing TMF REST API that can be used to query Products that a party is allowed to order. An example Product Offering is shown below:


```
{
      "id": "42",
      "href": "http://serverlocation:port/productCatalogManagement/productOffering/42",
      "version": "2.0",
      "lastUpdate": "2013-04-19T16:42:23.0Z",
      "name": "Virtual Storage Medium",
      "description": "Virtual Storage Medium",
      "isBundle": true,
      "lifecycleStatus": "Active",
      "validFor":
      {
            "startDateTime": "2013-04-19T16:42:23.0Z",
            "endDateTime": "2013-06-19T00:00:00.0Z"
      },

      "category":
      [
            {
                  "id": "12",
                  "href": "http://serverlocation:port/catalogManagement/category/12",
                  "version": "2.0",
                  "name": "Cloud offerings"
            }
      ],

      "channel":
      [
            {
                  "id": "13",
                  "href": "http://serverlocation:port/marketSales/channel/13",
                  "name": "Online Channel"
            }
      ],

      "place":
      [
            {
                  "id": "12",
                  "href": "http://serverlocation:port/marketSales/place/12",
                  "name": "France"
            }
      ],
```

```
      "bundledProductOffering":
       [
            {
                  "id": "15",
                  "href":
"http://serverlocation:port/productCatalogManagement/productOffering/15",
                  "lifecycleStatus": "Active",
                  "name": "Offering 15"
            },

            {
                  "id": "64",
                  "href":
"http://serverlocation:port/productCatalogManagement/productOffering/64",
                  "lifecycleStatus": "Active",
                  "name": "Offering 64"
            }
       ],

      "serviceLevelAgreement":
       {
            "id": "28",
            "href": "http://serverlocation:port/slaManagement/serviceLevelAgreement/28",
            "name": "Standard SLA"
       },

      "productSpecification":
            {
                  "id": "13",
                  "href":
"http://serverlocation:port/productCatalogManagement/productSpecification/13",
                  "version": "2.0",
                  "name": "specification product 1"
            },

      "serviceCandidate":
            {
                  "id": "13",
                  "href":
"http://serverlocation:port/serviceCatalogManagement/serviceCandidate/13",
                  "version": "2.0",
                  "name": "specification service 1"
            },

      "resourceCandidate":
            {
                  "id": "13",
                  "href":
"http://serverlocation:port/resourceCatalogManagement/resourceCandidate/13",
                  "version": "2.0",
                  "name": "specification resource 1"
            },

      "productOfferingTerm":
       [
            {
                  "name": "12 Month",
                  "description": "12 month contract",
                  "duration": "12",
                  "validFor":
```

```
                {
                      "startDateTime": "2013-04-19T16:42:23.0Z",
                      "endDateTime": "2013-06-19T00:00:00.0Z"
                }
          }
    ],

    "productOfferingPrice":
    [
          {

                "name": "Monthly Price",
                "description": "monthlyprice",
                "validFor":
                {
                      "startDateTime": "2013-04-19T16:42:23.0Z",
                      "endDateTime": "2013-06-19T00:00:00.0Z"
                },

                "priceType": "recurring",
                "unitOfMeasure": "",
                "price":
                {
                      "taxIncludedAmount": 12,
                      "dutyFreeAmount": 10,
                      "taxRate": 20,
                      "currencyCode": "EUR",
                      "percentage": 0
                },

                "recurringChargePeriod": "monthly"
          },

          {

                "name": "Usage Price",
                "description": "usageprice",
                "validFor":
                {
                      "startDateTime": "2013-04-19T16:42:23.0Z",
                      "endDateTime": "2013-06-19T00:00:00.0Z"
                },

                "priceType": "usage",
                "unitOfMeasure": "second",
                "price":
                {
                      "taxIncludedAmount": 12,
                      "dutyFreeAmount": 10,
                      "taxRate": 20,
                      "currencyCode": "EUR",
                      "percentage": 0
                },

                "recurringChargePeriod": "",
                "productOfferPriceAlteration":
                {
                      "name": "Shipping Discount",
                      "description": "One time shipping discount",
                      "validFor":
```

```
                        {
                              "startDateTime": "2013-04-19T16:42:23.0Z"
                        },

                        "priceType": "One Time discount",
                        "unitOfMeasure": "",
                        "price":
                        {
                              "percentage": 100
                        },

                        "recurringChargePeriod": "",
                        "priceCondition": "apply if total amount of the  order is greater
than 300.00"
                  }
            }
      ]
}
```


If we look at the highlighted sections above, we can see that the Product Offering (which includes not only what can be ordered but ancillary information such as SLAs and pricing) consists of one or more 'specifications'. These specifications describe what the orderable attributes of the product are (as outlined in the SID). The payload above shows a product, service and resource candidate specification. An example Product Specification is shown below:


```
{
      "id": "22",
      "href":
"http://serverlocation:port/productCatalogManagement/productSpecification/22",
      "productNumber": "I42-340-DX",
      "version": "2.0",
      "lastUpdate": "2013-04-19T16:42:23.0Z",
      "name": "iPhone 42",
      "description": "Siri works on this iPhone",
      "isBundle": true,
      "brand": "Apple",
      "lifecycleStatus": "Active",
      "validFor":
      {
            "startDateTime": "2013-06-19T00:00:00.0Z",
            "endDateTime": "2013-04-19T16:42:23.0Z"
      },

      "relatedParty":
      [
            {

                  "id": "1234",
                  "href": "http ://serverLocation:port/partyManagement/partyRole/1234",
                  "role": "Owner"
            }
      ],

      "attachment":
      [
            {
```

```
                "id": "22",
                "href": "http://serverlocation:port/documentManagement/attachment/22",
                "type": "Picture",
                "url": "http://xxxxx"
            }
        ],

        "bundledProductSpecification":
        [
            {
                "id": "15",
                "href":
"http://serverlocation:port/productCatalogManagement/productSpecification/15",
                "lifecycleStatus": "Active",
                "name": "Product specification 15"
            },

            {

                "id": "64",
                "href":
"http://serverlocation:port/productCatalogManagement/productSpecification/64",
                "lifecycleStatus": "Active",
                "name": "Product specification 64"
            }
        ],

        "productSpecificationRelationship":
        [
            {

                "id": "23",
                "href":
"http://serverlocation:port/productCatalogManagement/productSpecification/23",
                "type": "dependency",
                "validFor":
                {
                    "startDateTime": "2013-04-19T16:42:23.0Z"
                }
            }
        ],

        "serviceSpecification":
        [
            {
                "id": "13",
                "href":
"http://serverlocation:port/serviceCatalogManagement/serviceSpecification/13",
                "name": "specification 13",
                "version": "1.1"
            }
        ],

        "resourceSpecification":
        [
            {
                "id": "13",
                "href":
"http://serverlocation:port/resourceCatalogManagement/resourceSpecification/13",
                "name": "specification 13",
```

```
                "version": "1.1"
            }
        ],

        "productSpecCharacteristic":
        [
            {
                "name": "Screen Size",
                "description": "Screen size",
                "valueType": "number",
                "configurable": false,
                "productSpecCharacteristicValue":
                [
                    {
                        "valueType": "number",
                        "default": true,
                        "value": "4.2",
                        "unitOfMeasure": "inches",
                        "valueFrom": "",
                        "valueTo": "",
                        "validFor":
                        {
                            "startDateTime": "2013-04-19T16:42:23.0Z"
                        }
                    }
                ],

                "validFor":
                {
                    "startDateTime": "2013-04-19T16:42:23.0Z"
                }
            },

            {
                "name": "Colour",
                "description": "Colour",
                "valueType": "string",
                "configurable": true,
                "ProductSpecCharacteristicValue":
                [
                    {
                        "valueType": "string",
                        "default": true,
                        "value": "Black",
                        "unitOfMeasure": "",
                        "valueFrom": "",
                        "valueTo": "",
                        "validFor":
                        {
                            "startDateTime": "2013-04-19T16:42:23.0Z"
                        }
                    },

                    {
                        "valueType": "string",
                        "default": false,
                        "value": "White",
                        "unitOfMeasure": "",
                        "valueFrom": "",
                        "valueTo": "",
```

```
                        "validFor":
                        {
                              "startDateTime": "2013-04-19T16:42:23.0Z"
                        }
                  }
            ],

            "validFor":
            {
                  "startDateTime": "2013-04-19T16:42:23.0Z"
            }
      }
   ]
}
```


If we look at the 'Colour' characteristic value above, we can see that the spec defines this attribute as a number that is configurable and that it is an enumeration with two possible values 'black' or 'white'. We also have default values, unitOfMeasure, valueFrom/To (if this was a range), etc. This tells a reader/developer (almost) all they need to know in order to structure the part of the Order that deals with this attribute. Developers can use this information to validate data, render UI components, etc.

However, if we wanted a system to be able to perform this validation or decide which UI component to render without having a person needing to make these decisions (i.e. on the fly), we could combine the specification approach with existing schema validation principles to establish an API integration pattern that would allow for this.

The specification could describe a standardized way of describing how to manage the attributes and relationships of an entity while the schema could describe how each attribute (or groups of attributes) should be validated based on some predefined data type (such as a string, integer, enumeration, range, etc.).

A developer could code a dynamic interaction that does not require pre-existing knowledge of the structure of the entity as long as:

1. The application had both the *Specification* and a *Schema* for every API invocation, which would provide it with the relevant data and metadata required for that invocation.
2. There was a standardized way of how a specification was structured (basically following SID rules for creating specifications, such as following CharacteristicSpec/CharSpecValue dynamic attribute pattern):
    a. This standardized way would need to be governed by TM Forum and rules for how to extend the specification construct should be considered. Members could contribute extensions they see as a standard pattern back to the Forum.
    b. The specification would describe which attributes need to be present, in which order, how they should be structured, etc.
3. The schema should contain a list of all the data types that could be used to specify the type of data object for a particular characteristic (or group of characteristics) that is part of the spec. This can then be used as an instruction set as to how to validate that particular characteristic. The list of all the raw 'data types' that can be used to specify how to validate a characteristic would be

governed by TM Forum; however, specific implementations can insert the required values (such as a custom set of enumerations) as needed.

2. Example

Imagine an API that allows a consumer to order a new vehicle. The consumer doesn't know how to construct the order for the vehicle; they merely know that it is an API that allows them to order a new vehicle. The vehicleAPIs consists of a Catalog and a ProductOrder API. The consumer first queries the Catalog to see what is available.

2.0.1. Query Catalogs

REQUEST

GET http://serverlocation:port/catalogManagement/catalog?type=Product
Accept: application/json

RESPONSE


```
[
{
    "id": "1",
    "href": "http://serverlocation:port/catalogManagement/catalog/1",
    "lastUpdate": "2013-04-19T16:42:23.0Z",
    "version": "2.0",
    "lifecycleStatus": "Active",
    "isRoot": true,
    "name": "Cars",
    "type": "Product",
    "description": "Car Catalog"
},
{
    "id": "2",
    "href": "http://serverlocation:port/catalogManagement/catalog/2",
    "lastUpdate": "2013-04-19T16:42:23.0Z",
    "version": "2.0",
    "lifecycleStatus": "Active",
    "isRoot": true,
    "name": "Trucks",
   "type": "Product",
    "description": "Truck Catalog"
}
]
```


The consumer then decides he wants a car and queries the car catalog and gets the following car product offerings:

2.0.2. Query Specific Catalog

REQUEST

GET http://serverlocation:port/catalogManagement/catalog/1
Accept: application/json

RESPONSE


```
{
    "id":1,
    "href":"http://serverlocation:port/catalogManagement/catalog/1",
    "lastUpdate":"2013-04-19T16:42:23.0Z",
    "version":"2.0",
    "lifecycleStatus":"Active",
    "isRoot":true,
    "name":"Cars",
    "type":"Product",
    "description":"Car Catalog",
    "offerings":[
            {
                    "id":1,
                    "href":"http://serverlocation:port/catalogManagement/offering/1",
                    "version":"1.0",
                    "lastUpdate":"2013-04-19T16:42:23.0Z",
                    "name":"300C",
                    "description":"Chrysler 300C Offering",
                    "lifecycleStatus":"Active",
                    "productSpecification":{
                            "id":"13",
                            "href":
"http://serverlocation:port/catalogManagement/specs/13",
                            "version":"2.0",
                            "name":"Chrysler 300C Specification",
                              "type":"Product"
                    },
                    "productOfferingPrice":[
                            {
                                    "name":"Purchase Price",
                                    "description":"Price of the vehicle",
                                    "price":{
                                            "currencyCode":"AUD",
                                            "price":25000
                                    }

                            }

                    ]

            },
            {
                    "id":"1",
                   "href": "http://serverlocation:port/catalogManagement/offering/2",
                    "version":"1.0",
                    "lastUpdate":"2013-04-19T16:44:23.0Z",
                    "name":"Kia Cerato",
                    "description":"Kia Cerato SP Hatchback",
                    "lifecycleStatus":"Active",
                    "productSpecification":{
                            "id":"15",
                            "href":
```

```
"http://serverlocation:port/catalogManagement/specs/15",
                        "version":"2.0",
                        "name":"Kia Cerato Specification",
                          "type":"Product"
                },
                "productOfferingPrice":[
                        {
                                "name":"Purchase Price",
                                "description":"Price of the vehicle",
                                "price":{
                                        "currencyCode":"AUD",
                                        "price":10000
                                }
                        }
                ]
        }
    ]
}
```


Querying the car catalog yields two available car offerings. If the consumer then decides he/she would like to order the Chrysler, then they would query the Chrysler spec.
The dynamic API concept here would see the interface return the specification (which would contain rules on how to construct and validate the payload, to be described in a future version of this recommendation) as well as the data type schema (that defines how to interpret the metadata):

2.0.3. Query Specific Specification

REQUEST

```
GET http://serverlocation:port/catalogManagement/specs/13
Accept: application/json
```

RESPONSE


```
{
    "id":"13",
    "href":"http://serverlocation:port/catalogManagement/specs/13",
    "version":"1.0",
    "lastUpdate":"2014-04-19T16:42:23-04:00",
    "name":"300C Specification",
    "type":"Product",
    "description":"Chrysler 300C Specification",
    "brand":"Chrysler",
    "lifecycleStatus":"Active",
    "validFor":{
            "startDateTime":"2013-04-19T16:42:23-04:00",
            "endDateTime":"2015-06-19T00:00:00-04:00"
    },
    "metaData":{
            "id":"1",
            "href":"http://serverlocation:port/catalogManagement/schemas/1",
            "version":"2.0",
            "lastUpdate":"2014-04-19T16:42:23-04:00",
            "name":"Schema Object",
            "description":"Schema Object file containing master definition of input and
output types",
            "type":"Schema",
```

```
		"schemaType":"Swagger"
	},
	"productSpecCharacteristic":[
		{
			"name":"Xenon Headlights",
			"description":"Metal-halide lamps with Xenon gas",
			"valueType":"boolean",
			"configurable":"true",
			"productSpecCharacteristicValue":[
				{
					"valueType":"boolean",
					"default":"false",
					"value":"",
					"unitOfMeasure":"",
					"validFor":{
						"startDateTime":"2013-04-19T16:42:23-04:00",
						"endDateTime":""
					}
				}
			]
		},
		{
			"name":"Colour",
			"description":"Car chassis colour",
			"valueType":"enum",
			"configurable":"true",
			"validFor":{
				"startDateTime":"2013-04-19T16:42:23-04:00",
				"endDateTime":""
			},
			"productSpecCharacteristicValue":[
				{
					"valueType":"string",
					"default":"true",
					"value":"Black",
					"unitOfMeasure":"",
					"validFor":{
						"startDateTime":"2013-04-19T16:42:23-04:00",
						"endDateTime":""
					}
				},
				{
					"valueType":"string",
					"default":"false",
					"value":"White",
					"unitOfMeasure":"",
					"validFor":{
						"startDateTime":"2013-04-19T16:42:23-04:00",
						"endDateTime":""
					}
				}
			]
		}
	]
}
```


The schema file reference returned will outline in JSON format how to interpret the various product Specification Characteristic types (i.e. the *valueType* attribute such as 'enum' or 'string'). If this schema file is aligned to an existing standard (such as JSON-

Schema, Swagger, Blueprint or any other JSON standard that supports the meta-schema concept) then the referenced file can be used to enable a GUI, API or Business-to-Business application to validate inputs on the fly.

3. Example

In the example above, the metaData referenced is the Swagger 2.0 schema[3], an excerpt of which is shown below:

```
{
    "schema":{
            "type":"object",
            "description":"A deterministic version of a JSON Schema object.",
            "patternProperties":{
                    "^x-":{
                            "$ref":"#/definitions/vendorExtension"
                    }
            },
            "properties":{
                    "$ref":{
                            "type":"string"
                    },
                    "format":{
                            "type":"string"
                    },
                    "title":{
                            "$ref":"http: //json-schema.org/draft-
04/schema#/properties/title"
                    },
                    "description":{
                            "$ref":"http: //json-schema.org/draft-
04/schema#/properties/description"
                    },
                    "default":{
                            "$ref":"http: //json-schema.org/draft-
04/schema#/properties/default"
                    },
                    "multipleOf":{
                            "$ref":"http: //json-schema.org/draft-
04/schema#/properties/multipleOf"
                    },
                    "maximum":{
                            "$ref":"http: //json-schema.org/draft-
04/schema#/properties/maximum"
                    },
                    "exclusiveMaximum":{
                            "$ref":"http: //json-schema.org/draft-
04/schema#/properties/exclusiveMaximum"
                    },
                    "minimum":{
                            "$ref":"http: //json-schema.org/draft-
04/schema#/properties/minimum"
                    },
                    "exclusiveMinimum":{
                            "$ref":"http: //json-schema.org/draft-
04/schema#/properties/exclusiveMinimum"
                    },
                    "maxLength":{
                            "$ref":"http: //json-schema.org/draft-
04/schema#/definitions/positiveInteger"
                    },
```

```
                    "minLength":{
                            "$ref":"http: //json-schema.org/draft-
04/schema#/definitions/positiveIntegerDefault0"
                    },
                    "pattern":{
                            "$ref":"http: //json-schema.org/draft-
04/schema#/properties/pattern"
                    },
                    "maxItems":{
                            "$ref":"http: //json-schema.org/draft-
04/schema#/definitions/positiveInteger"
                    },
                    "minItems":{
                            "$ref":"http: //json-schema.org/draft-
04/schema#/definitions/positiveIntegerDefault0"
                    },
                    "uniqueItems":{
                            "$ref":"http: //json-schema.org/draft-
04/schema#/properties/uniqueItems"
                    },
                    "maxProperties":{
                            "$ref":"http: //json-schema.org/draft-
04/schema#/definitions/positiveInteger"
                    },
                    "minProperties":{
                            "$ref":"http: //json-schema.org/draft-
04/schema#/definitions/positiveIntegerDefault0"
                    },
                    "required":{
                            "$ref":"http: //json-schema.org/draft-
04/schema#/definitions/stringArray"
                    },
                    "enum":{
                            "$ref":"http: //json-schema.org/draft-
04/schema#/properties/enum"
                    },
                    "type":{
                            "$ref":"http: //json-schema.org/draft-
04/schema#/properties/type"
                    },
                    "items":{
                            "anyOf":[
                                    {
                                            "$ref":"#/definitions/schema"
                                    },
                                    {
                                            "type":"array",
                                            "minItems":1,
                                            "items":{
                                                    "$ref":"#/definitions/schema"
                                            }
                                    }
                            ],
                            "default":{

                            }
                    },
                    "allOf":{
                            "type":"array",
                            "minItems":1,
```

```
                        "items":{
                                "$ref":"#/definitions/schema"
                        }
                },
                "properties":{
                        "type":"object",
                        "additionalProperties":{
                                "$ref":"#/definitions/schema"
                        },
                        "default":{

                        }
                },
                "discriminator":{
                        "type":"string"
                },
                "readOnly":{
                        "type":"boolean",
                        "default":false
                },
                "xml":{
                        "$ref":"#/definitions/xml"
                },
                "externalDocs":{
                        "$ref":"#/definitions/externalDocs"
                },
                "example":{

                }
        }
    }
}
```

New native validation types introduced to the schema will need to be understood by the consumer, however the disruption can be minimized if:

- Provider and Consumer align to one (or more) schema standards and
- Are aware of which version of the standard is in use.

If the consumer is unsure how to validate a particular attribute it can skip the validation and rely on the provider returning an error in the response. This can then be used to display to the user who may be able to manually correct the input based on the error message.

4. Reference Documents

[1] Harvard Business Review, 'The Digital Dividend : First-Mover Advantage', https://hbr.org/resources/pdfs/comm/verizon/18832_HBR_Verizon_Report_IT_rev3_webview.pdf

[2] TM Forum, 'Product Catalog Management API specification', https://www.tmforum.org/resources/standard/tmf620-product-catalog-management-api-rest-specification-r14-5-0/

[3] Swagger Working Group, 'Swagger RESTful API Documentation Specification v2.0', https://github.com/swagger-api/swagger-spec/blob/master/versions/2.0.md#schemaObject

5. Further Reading

- Dynamic Data Specification - http://www.slideshare.net/gtilton/dynamic-data-specification-v2720140118-1729
- Dynamic Modelling using SID patterns - http://www.slideshare.net/gtilton/dynamic-modelling-best-practice-recommendation-for-the-sid
- Dynamic APIs: Negotiating change – http://inform.tmforum.org/strategic-programs-2/agile-business-it/2015/10/dynamic-apis-negotiating-change/
- Dynamic APIs for the Connected Carrier – http://inform.tmforum.org/features-and-analysis/featured/2015/05/merged-catalyst-uses-nfv-to-simplify-partnering/
- EnterpriseWeb TMF Live 2015 Dynamic APIs Catalyst output 071515 – http://projects.tmforum.org/wiki/download/attachments/37391547/EnterpriseWeb_TMF%20Live%202015_Dynamic%20APIs%20Catalyst_output_071515.pdf?version=1&modificationDate=1437315690325&api=v2
- EnterpriseWeb: Management and Operations Platform Metamodel – http://community.tmforum.org/communities/frameworx/contributions.aspx#

6. Administrative Appendix

Document History

6.0.1. Version History

Version Number	Date Modified	Modified by:	Description of changes
1.0	15/11/2015	Pierre Gauthier TM Forum Johnathan Smaller DGIT jsmaller@dgit.biz	Final Draft of Document
1.1	13/11/15	Cliff C Faurer EnterpriseWeb cliff@enterpriseweb.com	Clarify scope of recommendation in this initial release Add to further readings
1.1.1	20/11/15	Alicja Kawecki TM Forum	Updated cover, header; minor formatting/style edits prior to publishing
1.1.2	13/4/16	Alicja Kawecki TM Forum	Updated cover, footer, Notice to reflect TM Forum Approved status

6.0.2. Release History

Release Number	Date Modified	Modified by:	Description of changes
1.0	15/11/2015	Pierre Gauthier TM Forum Johnathan Smaller DGIT jsmaller@dgit.biz	first release of document

Acknowledgments

This document was prepared by members of the TM Forum API Program team.