TMF630 Summary

# Purpose of this document

| What |  | This document summarizes the [TMF630 REST API Guidelines](https://www.tmforum.org/resources/how-to-guide/tmf630-api-design-guidelines-4-0/) version 4.0.1 produced by the TMF, to improve accessibility to TELUS team members |
| --- | --- | --- |
|  |  |  |
| Whom |  | API Providers |
|  |  |  |
| When |  | When designing any new API, regardless of purpose or target consumer |

# Minimum Conformance Checklist

| TMF RESTful implementation conformance checklist | |
| --- | --- |
|  | Documentation is provided in Swagger/OAS 2.0 |
|  | URL is formatted correctly, including [naming and versioning](#_3hb0438e3urm) |
|  | Routing uses [resource/task-based paths](#_9ao03actu4ra) |
|  | [HTTP verbs](#_vbzt80mf41l0) are used to select action against a resource |
|  | Response uses appropriate [HTTP status codes](#_f2aey672nxhk) |
|  | [Response header](#_kmy42imcbpj3) contains details about the response body |
|  | Response body is a [resource/task](#_kiu0ewckm4o) presented using JSON |
|  | Error responses utilize the [Error resource](#_pz3m3tsop4rn) as the response body |
|  | If this is a [TMF Open API](https://projects.tmforum.org/wiki/display/API/Open+API+Table), any mandatory fields in the entity exist |
|  | If this is a [TMF Open API](https://projects.tmforum.org/wiki/display/API/Open+API+Table), any customizations must utilize the TMF [extension pattern](#_26ubnwz22dmg) |
|  | If this is a [TMF Open API](https://projects.tmforum.org/wiki/display/API/Open+API+Table), the Conformance Test Kit must be run and must pass |
| minimum compliance requirement | |
|  | [URL query string](#_b0ls7tco8j8r) can be used by the consumer to modify the response content   * Select (FIELDS) * Sort (SORT) * Match criteria * Page (LIMIT,OFFSET) * Detail (DEPTH,EXPAND) |
|  | [Hypermedia](#_sibmc313p1e8) links to other relevant resources are provided in responses |
|  | The [/home](#_9ao03actu4ra) path provides a JSON-LD [Hypermedia](#_sibmc313p1e8) summary of available functions |
|  | Event subscription is enabled via the [Hub](#_9ao03actu4ra) |
|  | If any resources have versions, they utilize the [TMF version format](#_vb6qgen6yj6j) |
|  | If any operations are asynchronous, they utilize the [TMF monitor pattern](#_szktkw97f8me) |

# Understanding the Foundations

## What is a Resource?

Resources are nouns, such as customer, address, and product.

Standard resources are defined by the TMF and should be used where applicable (see the [TMF Open APIs](https://projects.tmforum.org/wiki/display/API/Open+API+Table)). The TMF resources are generally simpler than what our own TELUS resources might look like, because they contain only attributes that are considered universally applicable in the industry. If a TMF resource doesn’t quite match our needs, it can be [extended](#_26ubnwz22dmg) to provide additional attributes that are specific for us.

If none of the existing TMF resources apply and you need to invent a name for a new resource, it should be a singular, camelCase noun and written out in full without abbreviations or acronyms (e.g., productOffering).

Resources will be the primary component of your URI. You may be used to using a path such as /getCustomer and /setCustomer in order to retrieve/modify the customer information. But for RESTful APIs, instead the path should simply be a resource/noun such as /customer, and then HTTP verbs will indicate the CRUD operation to take against that resource. After all, the URL stands for Universal Resource Locator and so should simply be a path to the location of the resource.

In some cases it is not possible to express what you need as a resource with CRUD operations; in this case, you will use a task.

## 

## What is a Task?

Tasks are verbs, such as validate, check, heal, migrate, clear, close, and transfer. They are for operations that are not easily decomposable to CRUD operations. Tasks will often be asynchronous. Tasks must not contain any HTTP verbs in their name, as that is what the HTTP verbs themselves are for.

Standard tasks are defined by the TMF Open APIs and should be used where applicable. If you need to create a new task, it should use the same naming conventions as for a resource, but will represent an action instead of a thing.

The main operation on tasks is POST to create a new task that initiates the desired operation. Tasks must not be deleted because they are records of work, rather they should be cancelled using PATCH to change the state. Tasks are likely to be long-running; if so, they must use [asynchronous standards](#_etsytxqn4pt7) outlined by TMF to allow the consumer to check-back on the status of their request.

| TMF630 Summary | | |
| --- | --- | --- |
| Documentation must be provided as Swagger ^2.0  API must follow RESTful standards.  API must support HTTPS protocol.  Response body must be provided as JSON by default. | | |
| API Naming & Versioning | | |
| https://*root*/*apiName*/v*majorVersion*/  *ex. https://tsl.telus.com/productCatalogManagement/****v1/*** | | |
| **v1** | .2 | |
| Major | Minor | |
| Change is NOT [backwards compatible](#_w4pvlg2tzd3n)  and requires consumers to make changes. | Change is [backwards compatible](#_w4pvlg2tzd3n), consumers need not change. Minor version is not used in the URL. | |
| URI conventions | | |
| https://…/*resource*/  *ex. https://tsl.telus.com/productCatalogManagement/v1****/catalog*** | All URIs in your API must be based on [resources](#_izt6jqzzplw) or [tasks](#_dt7zmbqwhkr5). You may be used to using URIs like /getCustomer and /setCustomer. Now you will simply use /customer and then use HTTP verbs to select the action. | |
| https://…/*resource*/*id*/  *ex. https://tsl.telus.com/productCatalogManagement/v1****/catalog/123*** | To act on a specific instance of a [resource](#_izt6jqzzplw) or [task](#_dt7zmbqwhkr5), the id of the [resource](#_izt6jqzzplw)/[task](#_dt7zmbqwhkr5) can be specified after the [resource](#_izt6jqzzplw)/[task](#_dt7zmbqwhkr5) name. | |
| Resource  {  “id”: “*unique id for the resource*”  “href”: “/*domain*/*apiname*/*v1*/*resource*/*id*”  ...  } | All [resources](#_izt6jqzzplw) and [tasks](#_dt7zmbqwhkr5) will have an id and an href property at a minimum. These properties must always be part of the resource when it is being returned.  TELUS has decided to use relative hrefs, ie. exclude the API host name. | |
| https://…/hub  POST /hub  DELETE /hub/*id*  POST /*resource*/hub  POST /*resource*/*id*/hub | The [hub](#_6lstatutr0s) URI is reserved for subscribing to [event notifications](#_etsytxqn4pt7). POST is used to subscribe to receive events, and DELETE is used to unsubscribe.  A [hub](#_6lstatutr0s) can be used to subscribe to general API events, or to specific resource events, and query parameters can be applied. | |
| https://…/monitor  GET /monitor/*id*  POST /monitor/*id*/hub | The [monitor](#_szktkw97f8me) URI is reserved for [asynchronous operation](#_etsytxqn4pt7) tracking. The [monitor](#_szktkw97f8me) can be queried to determine the current status of the operation, or the consumer may subscribe to the [monitor](#_szktkw97f8me) in order to receive a completion event notification. | |
| https://…/home  GET /home  200 OK  Body: [*\_links resource*](#_gyhf8nswwh1f) | The home URI is reserved for [Hypermedia / HATEOS](#_sibmc313p1e8): a homepage for your API that acts like a navbar to all operations a consumer could use within your API. It returns a [\_links](#_gyhf8nswwh1f) resource. | |
| [**HTTP Verbs**](https://developers.telus.com/topics/using-http-verbs-tmf630)*migrated to SimplifyHub* | | |
| **GET** /resource  **GET** /resource/*id*  200 OK  Body: *selected resource(s)* | **Select**. Return all matching [resources](#_izt6jqzzplw) or [tasks](#_dt7zmbqwhkr5). If /resource, return the entire set of [resources](#_izt6jqzzplw)/[tasks](#_dt7zmbqwhkr5). If id is specified, return the one resource with that id. Otherwise, the consumer may use [URI Query Parameters](#_dg51jshex3xr) to define a subset.  The id and href fields of all relevant [resources](#_izt6jqzzplw)/[tasks](#_dt7zmbqwhkr5) must always be returned.  If a [resource](#_izt6jqzzplw)/[task](#_dt7zmbqwhkr5) has a [version](#_vb6qgen6yj6j), by default only the latest version will be returned unless otherwise specified in the [URI query parameters](#_dg51jshex3xr). | |
| **POST** /resource  **POST** /task  Body: *full resource/task to create*  201 Created *or*  202 Accepted *(if asynchronous)* *or*  204 No content *(if no body)*  Location: *url for new resource/task*  Body: *created resource/task or empty* | **Create**. Generate a new single resource, kick off a new task, or create a new version of a [resource](#_izt6jqzzplw)/[task](#_dt7zmbqwhkr5).  Response must specify the HTTP Location header unless ?fields=none is in the request URL. | |
| **PUT** /resource/*id*  Body: *full replacement resource*  200 OK  Body: *full replaced resource* | **Replace**. Overwrites the existing resource with the provided one. MUST NOT be used for partial update.  Request body must include a new resource in full, containing at a minimum id and href. | |
| **PATCH** /resource/*id*  **PATCH** /resource  Accept:  *application/json or*  [*application/json-patch+query*](https://tools.ietf.org/html/rfc6902) *or*  [*application/merge-patch+json*](https://tools.ietf.org/html/rfc7386) *or*  [*application/json-patch+json*](https://tools.ietf.org/html/rfc5789)  Body: *partial resource(s)*  200 OK  Content-Type:  *application/json or*  [*application/json-patch+query*](https://tools.ietf.org/html/rfc6902) *or*  [*application/merge-patch+json*](https://tools.ietf.org/html/rfc7396) *or*  [*application/json-patch+json*](https://tools.ietf.org/html/rfc5789)  Body: *full modified resource(s)* | **Update**. Modify existing [resources](#_izt6jqzzplw) or [tasks](#_dt7zmbqwhkr5), or create multiple resources at once.  How to interpret the provided JSON to apply the update is non-trivial. The requester may specify which standard they’d like to use with Accept, and your API must indicate what it did use with Content-type. It is recommended to support [application/json-patch+query](https://sookocheff.com/post/api/understanding-json-patch/) at a minimum.  For batch operations, all operations must be successful otherwise the PATCH is unsuccessful. | |
| **DELETE** /resource/*id*  Body: *empty*  204 No content *or*  202 Accepted *(if asynchronous)*  404 Not Found *(if resource id does not exist)* | **Delete**. Remove the specified [resource](#_izt6jqzzplw). The request must not contain a body.  [Tasks](#_dt7zmbqwhkr5) must not be deleted, rather they must be cancelled using PATCH to change the status accordingly. | |
| [**Common TMF HTTP status codes**](https://developers.telus.com/topics/using-http-status-codes-tmf630) *migrated to SimplifyHub* | | |
| *Please see* [*HTTP status code documentation*](https://en.wikipedia.org/wiki/List_of_HTTP_status_codes) *for further codes and additional information.* | | |
| **200 OK**  Body: *something* | Standard successful response. Response body must have content, otherwise use 204. An empty array counts as content.  Use the most descriptive applicable 2XX code for the response possible. | |
| **201 Created**  Location: *url for new resource/task*  Body: *created resource* | Must be used for successful POST resulting in the creation of a new [resource](#_izt6jqzzplw) or kick-off of a new [task](#_dt7zmbqwhkr5). Response must also specify the HTTP Location header pointing to the new [resource](#_izt6jqzzplw)/[task](#_dt7zmbqwhkr5). | |
| **202 Accepted**  Location: *url for new resource/task*  Link: <https://…/monitor/*id*>;rel=related;title=monitor  Body: [*monitor resource*](#_szktkw97f8me) | Must be used if an asynchronous, long-running operation has been triggered. Response must also specify the HTTP Location and Link headers, and return a [monitor](#_szktkw97f8me) resource in the body. | |
| **204 No content**  Body: *empty* | Operation was successful but there is no content to be sent back in the body. Typically used for DELETE, or POST on a [task](#_dt7zmbqwhkr5), where there is no additional information to be conveyed in the response.  When no results match a valid GET request for multiple resources, the response should be an empty array with status code 200. | |
| **206 Partial content**  X-Total-Count: *# of items in response*  Link:  <https://…/?limit=n>;rel=first,  <https://…/?limit=n,offset=x>;rel=next,  <https://…/?limit=n,offset=y>;rel=prev,  <https://…/?limit=n,offset=z>;rel=last | Used for paging or iterators. Response must include the HTTP X-Total-Count and Link headers, where the Link header provides web-links to access different pages of the result set. | |
| **300 Redirection** | Rarely used to indicate that there is additional action that the consumer must take to complete their request. Use the most appropriate 3XX code. | |
| **400 Bad Request**  Body: [error resource](#_pz3m3tsop4rn) | Standard client error response.  Use the most descriptive applicable 4XX code for the error possible. | |
| **401 Unauthorized** | Used to indicate that there is a problem with the client’s credentials | |
| **403 Forbidden** | Used when the client is authorized to use the API, but does not have authority to complete the specific request they’ve made | |
| **404 Not Found** | Used when the requested [resource](#_izt6jqzzplw) or [task](#_dt7zmbqwhkr5) does not exist. | |
| **405 Method not allowed** | Used if the particular HTTP verb is not implemented for the URI | |
| **500 Internal server error**  Body: [error resource](#_pz3m3tsop4rn) | Standard server failure response. Use the most descriptive applicable 5XX code for the failure possible. | |
| **501 Not Implemented** | Used when a consumer has requested an endpoint that they could reasonably expect from your API, but which has not yet been implemented. | |
| HTTP Headers | | |
| Accepts  Content-Type  values:  application/json  application/xml | Always specify. Consumer indicates preference via Accepts, Provider indicates actual via Content-type, where application/json is the default that must be supported. application/xml is another common option you may choose to support. | |
| Accept-Language  Content-Language  values:  en\_GB  fr\_CA | Always specify. Consumer indicates preference via Accept-Language, Provider indicates actual via Content-Language, where en\_GB is the default that must be supported. At TELUS, we strongly recommend also providing fr\_CA. Supported locales must be documented in your OAS. | |
| Content-Length | Always specify. The length of the body in the request/response. | |
| [ETag](https://en.wikipedia.org/wiki/HTTP_ETag) | Optional, but strongly recommended. Unique identifier for the [resource](#_izt6jqzzplw) that is being returned. | |
| Last-Modified | Situational. Indicates the date that the [resource](#_izt6jqzzplw) being returned was modified. | |
| Location | Situational. Indicates the URI of a newly created [resource](#_izt6jqzzplw), or a [resource](#_izt6jqzzplw) being operated on [asynchronously](#_etsytxqn4pt7). | |
| X-Total-Count | Situational. Indicates how many resources are in the set being returned, like for GET requests with multiple matches. | |
| Cache-Control  Expires  Date | Optional, but strongly recommended. Used to indicate whether or not the consumer could cache the data being returned and for how long. Caching should be encouraged. | |
| X-Rate-Limit-Limit  X-Rate-Limit-Remaining  X-Rate-Limit-Reset | Optional. Used to communicate any request frequency limits you may have built into your API. Use HTTP status code 429 when this limit is exceeded.  Limit: number of requests allowed in the current period.  Remaining: number of requests remaining in the current period.  Reset: seconds left in the current period. | |
| Accept-Range:items  Range:items=i-j  Content-Range:items i-j/n | Optional. For large result sets, Consumer indicates desired range with Accept-Range and Range, Provider indicates actual returned range with Content-Range. i is the starting index for this page, j is the ending index for this page, and n is the total number of items in the complete set. | |
| URI Query String | | |
| The [Query String](https://en.wikipedia.org/wiki/Query_string) at the end of the URL can be used by the requester to have their request operate on a subset of the resource. It is similar to a SQL Query.  Any number of the following instructions can be chained together in the URL using ; or &. | | |
| /?**FIELDS**=f1,…,fn  /?**FIELDS**=none | SELECT. f is the dot-notation name of the field(s) of the [resource](#_izt6jqzzplw) to be returned in the response. id and href are always returned.  When none, only the id and href fields of the [resource](#_izt6jqzzplw)(s) will be returned. | |
| /?**SORT**=f1,**+**f2,**-**f3,…,fn | ORDER BY. f is the dot-notation name of the field(s) to sort the resulting [resource](#_izt6jqzzplw)s with. Order matters.   * Indicates ascending (default) * Indicates descending | |
| /?**DEPTH**=n | TMF [resource](#_izt6jqzzplw)s have many nested relationships to other [resource](#_izt6jqzzplw)s. DEPTH allows you to limit the level of detail, going only as deep as n nested levels. n=0 by default.  - n=0: first level of nested resources will be included with id, href, and @type properties only  - n=1: first level of nested resources shown in full, and second level shows id, href, and @type | |
| /?**EXPAND**=f1,…,fn | Can be used in conjunction with DEPTH to selectively dive into a subset of [resource](#_izt6jqzzplw)s. f is the dot-notation name of the field to expand to the specified DEPTH; other nested [resource](#_izt6jqzzplw)s will not be expanded at all. If DEPTH is unspecified, use DEPTH=1. | |
| /?**LIMIT**=n&**OFFSET**=i  206 Partial Content  X-Total-Count: *n*  Link:  <https://…/?limit=n>;rel=first,  <https://…/?limit=n,offset=i+n>;rel=next,  <https://…/?limit=n,offset=i-n>;rel=prev,  <https://…/?limit=n,offset=max-n>;rel=last | For paging, where n is the maximum number of results to return in the response.  OFFSET is optional, where i=0 if unspecified and i is the index in the set from which to retrieve the n results. | |
| /?f1=v1**&**…**&**fn=vn *(AND)*  /?f1=v1**;**…**;**fn=vn *(OR)* | WHERE. For any field (where f is the dot-notation name of the field), the request may include query parameters much like the WHERE clause of a SQL statement. & is used to AND field statements together, and ; is used to OR field statements together. | |
| /?f**=**v1,…,vn  /?f**.eq**=v1,…,vn  /?f**%3D**v1,…,vn /?f**=**v1**&**…**&**f**=**vn | WHERE field f is equal to any of the values vn. Note that all four of these notations must be supported. f is the dot-notation name of the field.  (In the last case, even though & is being used, it wouldn’t make sense to say one field must be equal to multiple values at the same time, so it is interpreted as OR.) | |
| /?f**>**v  /?f**.gt**=v  /?f**%3E**v | WHERE field f is greater than some value v. All three of these notations must be supported. f is the dot-notation name of the field. | |
| /?f**>=**v  /?f**.gte**=v  /?f**%3E%3D**v | WHERE field f is greater than or equal to some value v. All three of these notations must be supported. f is the dot-notation name of the field. | |
| /?f**<**v  /?f**.lt**=v  /?f**%3C**v | WHERE field f is less than some value v. All three of these notations must be supported. f is the dot-notation name of the field. | |
| /?f**<=**v  /?f**.lte**=v  /?f**%3C%3D**v | WHERE field f is less than or equal to some value v. All three of these notations must be supported. f is the dot-notation name of the field. | |
| /?f**\*=**r1,…,rn  /?f**.regex**=r1,…,rn  /?f**%3D~**r1,…,rn | WHERE field f matches any of the regex statements rn. All three of these notations must be supported. f is the dot-notation name of the field. | |
| Resource Versioning | | |
| Any [resource](#_izt6jqzzplw) may be given a version, in which case each version number for the [resource](#_izt6jqzzplw) must be unique.  https://.../*resource*/*id*:(version=*n*) | | |
| GET/PUT/PATCH/DELETE/*resource* | If the version is unspecified, use the latest version. Typically, non-admins would only have access to the latest version. | |
| GET /resource/?id=*x* | When id is used in a [URI Query String](#_dg51jshex3xr), this would return the list of all versions of the [resource](#_izt6jqzzplw) by the specified id. | |
| GET/PUT/PATCH/DELETE  /*resource*/*id*:(version=*n*)  /*resource*/*id*/?version=*n*  /*resource*/?id=*i*&version=*n* | There is a special syntax to select a specific version, or you can also use the standard [URI Query String](#_dg51jshex3xr) syntax where version is just another field. | |
| POST /*resource*  Body:  {  version: *unique*,  *…*  } | Use POST to create a new version of an existing [resource](#_izt6jqzzplw), where the request body contains a new unique version number. | |

# Resource definitions

Where applicable, you must use the TMF defined [resource](#_izt6jqzzplw)s and [task](#_dt7zmbqwhkr5)s. All of the fields/properties defined in the TMF [resource](#_izt6jqzzplw)/[task](#_dt7zmbqwhkr5) must be used in your API (even if they have no value for now). You may add extra fields to your [resource](#_izt6jqzzplw)/[task](#_dt7zmbqwhkr5) definition, but it’s important to collaborate with other teams using the same [resource](#_izt6jqzzplw)/[task](#_dt7zmbqwhkr5) so that everyone is using the same definition with matching fields/properties.

| Standard Resource/Task attributes | | |
| --- | --- | --- |
| {  “id”: …, | Required. id is mandatory for any [resource](#_izt6jqzzplw)/[task](#_dt7zmbqwhkr5). It must be unique across all instances of the [resource](#_izt6jqzzplw)/[task](#_dt7zmbqwhkr5), not just to a single system. If id is a composite key, each piece of the key should be separated by “-”. | |
| “href”:“/*domain*/*api*/*v1*/*resource*/*id*”, | Required. Link to be able to retrieve this [resource](#_izt6jqzzplw)/[task](#_dt7zmbqwhkr5). TELUS has decided to use relative hrefs, ie. exclude the API host name. | |
| “@type”: …, | The name of the type of [resource](#_izt6jqzzplw)/[task](#_dt7zmbqwhkr5), ex. customer. | |
| “@baseType”: …, | Object oriented parent [resource](#_izt6jqzzplw)/[task](#_dt7zmbqwhkr5) type. If you need to customize a TMF defined [resource](#_izt6jqzzplw)/[task](#_dt7zmbqwhkr5), then that [resource](#_izt6jqzzplw)/[task](#_dt7zmbqwhkr5)’s name should be put here. | |
| “@schemaLocation”: …, | URL where people can find the documentation for this [resource](#_izt6jqzzplw)/[task](#_dt7zmbqwhkr5). | |
| “@referredType”: …, | If passing by reference, indicates the name of the [resource](#_izt6jqzzplw)/[task](#_dt7zmbqwhkr5) type being referenced. | |
| “version”: …,  } | Resources may be versioned, in which case they will use this property to distinguish between versions. Version numbers for various versions of a resource must be unique, such that version can be used in conjunction with id as a key.  Use POST to create a new version of an existing resource. | |
| error *Returned when an HTTP 4XX or 5XX status code has occurred* | | |
| {  “code”: …, | Required. Application error code. | |
| “reason”: …, | Required. Description of why this error occurred. | |
| “message”: …, | Human-friendly message for the consumer. | |
| “status”: *400-2*, | Additional HTTP error code extension.  *At TELUS, SDF always provides HTTP status code 200 even if there was an error, so this field can be used to store the actual status code.* | |
| “referenceError”: …  } | URL pointing to documentation about this specific error. | |
| monitor *A resource used to monitor the status of an* [*asynchronous operation*](#_etsytxqn4pt7) | | |
| GET https://…/monitor/*id*  GET /*resource*/*id*/monitor | A consumer can get the monitor, but cannot modify it in any way. A monitor may be accessed via the monitor collection with the monitor ID, or through the resource that the monitor is attached to with the resource ID. | |
| PUT https://…/monitor  POST https://…/monitor  PATCH https://…/monitor  DELETE https://…/monitor  405 Method not allowed | Consumers may not modify the monitor in any way; it only exists to observe the asynchronous event. The response to any attempt to modify the monitor must be 405. | |
| {  “sourceHref”: “https://…”, | href of the resource being monitored. | |
| “name”: …, | Name of the resource type being monitored. | |
| “state”: “inProgress”, | The state of the operation. Must support inProgress, success, failed; custom states may be added. | |
| “request”: {  “statusCode”: “202”,  “body”: { … },  “header”: [  {  “name”: “Accept”,  “value”: “application/json”  },  …  ]  }, | A representation of the original request that triggered the asynchronous operation. | |
| “response”: {  “statusCode”: “200”,  “body”: { … },  “header”: [  {  “name”: “Content-Type”,  “value”: “application/json”  },  …  ]  }  } | A representation of the response as it would have appeared in a synchronous operation, once the operation has completed. | |
| hub *A resource used to subscribe to receive notifications about* [*events*](#_i4u2jeartogd) | | |
| POST https://…/hub  POST https://…/*resource*/hub  204 No Content  409 Conflict *(if unsuccessful)* | A consumer can use your API to subscribe to events, either at the API level or at a [resource](#_izt6jqzzplw) level in your API. | |
| GET https://…/hub/*id*  PUT https://…/hub/*id*  PATCH https://…/hub/*id*  DELETE https://…/hub/*id* | They can then read or modify their subscription details using the hub id. | |
| {  “callback”: “https://…”, | Where the subscriber wants to receive their notifications. | |
| “query”: “…”  } | Uses the TMF [URL query string](#_dg51jshex3xr) format to modify what kinds of events the subscriber will receive notifications for. | |
| Listener *An endpoint used to publish events* | | |
| **POST /client/listener**  Body: [*event resource*](#_l17und7tmxq) | To publish an event. | |
| Event *The event resource format that must be used for all* [*events*](#_i4u2jeartogd) | | |
| {  “eventId”: …, | ID for the event | |
| “eventTime”: …, | [DateTime](#_vk2sdracbvle) at which the event was created | |
| “eventType”: …, | String category for this kind of event, ex. TroubleTicketStateChangeNotification | |
| “event”: {  “*resource*”: {…}  }, | A representation of the resource this event acted on, such as TroubleTicket | |
| “correlationId”: …, | String | |
| “description”: …, | Human-readable description of the purpose of this event | |
| “domain”: …, | String | |
| “priority”: …, | String. “Normal”, | |
| “timeOccurred”: …, | [DateTime](#_vk2sdracbvle) at which the event occurred | |
| “title”: …, | Human-readable event name | |
| “source”: {}, | Reference to the resource that generated this event | |
| “reportingSystem”: {}, | Reference to a resource, the system that generated this event | |
| “analyticCharacteristic”: {}, | A Characteristic resource | |
| “relatedParty”: {}  } | A RelatedParty resource | |
| importJob & exportJob *An* [*asynchronous*](#_etsytxqn4pt7) *task used to import/export data from/to a file* | | |
| POST /*resource*/exportJob  POST /*resource*/importJob  POST /*resource*/*id*/exportJob  POST /*resource*/*id*/importJob  GET /exportJob/*id*  GET /importJob/*id* | Allowed operations. Trigger the import/export task or query the status of the task | |
| {  “query”: …, | Used to scope the data to be imported/exported, using the same format as the [URL query string](#_dg51jshex3xr). | |
| “path”: “https://…”, | URL of the [resource](#_izt6jqzzplw) to export from / import to | |
| “content-type”: “application/json”, | Format of the exported/imported data, default is application/json | |
| “status”: “notstarted”, | [Status](#_vk2sdracbvle) of the [asynchronous](#_etsytxqn4pt7) import/export task | |
| “url”: “file://…”, | URL of the file to import data from or export data into. | |
| “completionDate”: …, | [DateTime](#_vk2sdracbvle) at which the task was completed. | |
| “creationDate”: …, | [DateTime](#_vk2sdracbvle) at which the task was created. | |
| “errorLog”: …  } | Failure details, if applicable. | |
| \_links resource or Link header *May be provided in any response to provide additional context to the consumer via* [*Hypermedia*](#_sibmc313p1e8)  *Use a* [*JSON-LD*](https://json-ld.org/) *context* | | |
| {  “self”: { … }, | Mandatory. Indicates the current page URI | |
| “home”: { … }, | Home page of the API, which should provide links to all actions available in the API | |
| “related”: { … }, | References other related [resource](#_izt6jqzzplw)s. | |
| “related-*resource*”: { … }, | References a specific kind of related [resource](#_izt6jqzzplw) where necessary. Ex. TroubleTicket has both a related object and a related party, so would use related-object, related-party. | |
| “up”: { … }, | References the parent [resource](#_izt6jqzzplw). | |
| “first”: { … }, | References first subset in a set of [resource](#_izt6jqzzplw)s, used for pagination. | |
| “last”: { … }, | References the last subset in a set of [resource](#_izt6jqzzplw)s, used for pagination. | |
| “next”: { … }, | References the next subset in a set of [resource](#_izt6jqzzplw)s, used for pagination. | |
| “previous”: { … } , | References the previous subset in a set of [resource](#_izt6jqzzplw)s, used for pagination. | |
| “list-*resource*”: { … }, | Link to retrieve all of a specific [resource](#_izt6jqzzplw) collection. | |
| “retrieve-*resource*”: { … }, | Link to retrieve a specific instance of a [resource](#_izt6jqzzplw), using hrefTemplate and hrefVar. | |
| “create-*resource*”: { … }, | Link to create a new [resource](#_izt6jqzzplw). | |
| “update-*resource*”: { … }, | Link to update a [resource](#_izt6jqzzplw), using hrefTemplate and hrefVar. | |
| “delete-*resource*”: { … }, | Link to delete a [resource](#_izt6jqzzplw), using hrefTemplate and hrefVar. | |
| The following are possible properties of the above link types: | | |
| “href”: “https://…/*resource*/*id*”, | Mandatory unless hrefTemplate is present instead. Navigation link for the consumer to get details about the [resource](#_izt6jqzzplw). | |
| “hrefTemplate”: “https://…{*var*}”, | Mandatory unless href is present instead. Navigation link for the consumer but where the consumer must substitute some {parameters}. Requires hrefVar. | |
| “hrefVar”: {  “*var*”: “https://…/schema/*resource*.json”,  …  }, | Mandatory if hrefTemplate is present, list of variables which must be substituted in the hrefTemplate with a link to the schema where more information can be found about the parameters. | |
| “title”: … , | Human-readable identifier for link | |
| “name”: … , | Key to select link | |
| “method”: “GET” | Indicates the allowed HTTP verb: GET, POST, PATCH, PUT, or DELETE | |
| “accepts”: “application/json”, | Indicates the content-type format that will be accepted, typically application/json. | |
| “schemaType”: “json-schema”, | Indicates the format of the schema, typically json-schema. | |
| “schemaUrl”: “https://…/schema/*resource*.json”,  ...  } | Link to the schema describing the expected payload. | |

# Extension Pattern

TMF [resources](#_izt6jqzzplw) and [tasks](#_dt7zmbqwhkr5) are generic so that they are applicable across the entire industry; such a broad definition is unlikely to meet all of our requirements. So, the TMF allows for extending their definitions (ie. adding things) according to the following rules:

* New characteristics or relationships may be added
* Nothing can be removed
* Lists of allowed values (such as state being “inProgress”, “succeeded”, …) can be added to, but existing defined values cannot be removed. Sub-values can be created using dot notation (ex. “inProgress.held”).

Extensions are not meant to be (nor are they allowed to be) created for specific systems. All systems at TELUS that use a [resource](#_izt6jqzzplw)/[task](#_dt7zmbqwhkr5) must collaborate to produce one single definition of that [resource](#_izt6jqzzplw)/[task](#_dt7zmbqwhkr5) that is acceptable for all.

You must configure the following properties in your extended [resource](#_izt6jqzzplw)/[task](#_dt7zmbqwhkr5):

* @baseType: the name of the original [resource](#_izt6jqzzplw)/[task](#_dt7zmbqwhkr5) that your extension is based on
* @type: the name of your extension
* @schemaLocation: the URI where your extension is defined

When adding to a [resource](#_izt6jqzzplw)/[task](#_dt7zmbqwhkr5), avoid freeform name-value characteristic pairs. Freeform data of this nature allows for flexible systems, but creates problems for integration, data management, and security. Aim to always specifically define the characteristics needed by the entity and limit what possible values are allowed for that characteristic as much as possible.

# Pass by Reference or by Value

Any [resource](#_izt6jqzzplw)/[task](#_dt7zmbqwhkr5) may include other [resources](#_izt6jqzzplw)/[tasks](#_dt7zmbqwhkr5) as part of their own definition. When this occurs, the [resource](#_izt6jqzzplw)/[task](#_dt7zmbqwhkr5) may be embedded, or referenced. In the TMF, this is indicated by refOrValue.

In OAS 2 (Swagger), there is no easy way to indicate this, so the schema will be defined as though the [resource](#_izt6jqzzplw)/[task](#_dt7zmbqwhkr5) was being embedded by value, where the implementation may choose to embed or pass a reference. All of the parameters of the embedded [resource](#_izt6jqzzplw)/[task](#_dt7zmbqwhkr5) must therefore be optional (other than *id* and *href*) to allow for this implementation option.

In OAS 3, the schema would use the oneOf relationship to indicate that the [resource](#_izt6jqzzplw)/[task](#_dt7zmbqwhkr5) could be either embedded or referenced, where the options are *resource* and *resource*Ref.

# Standard attribute formats

| Standard Formats | | |
| --- | --- | --- |
| Date | “fieldDate”: “yyyy-MM-dd” | |
| DateTime | “fieldDateTime”: “YYYY-MM-DDThh:mm:ss.sTZD” | |
| *date/datime format legend* | YYYY 4-digit year  MM 2-digit month (01 to 12)  DD 2-digit day (01 to 31)  hh 2-digit 24-hour (00 to 23)  mm 2-digit minute (00 to 59)  ss 2-digit second (00 to 59)  s 1+ digits fraction of a second  TZD time zone (Z or +hh:mm or -hh:mm) | |
| Date Range | {  “startDateTime”: “yyyy-MM-ddTHH:mm”,  “endDateTime”: “yyyy-MM-ddTHH:mm”  } | |
| Money  (float, can optionally be negative) | {  “unit”: “CAD”,  “value”: -##.##  } | |
| Status  (string, must include these options and may add options) | “status”: “*notstarted*”   * notstarted * running * succeeded * failed | |
| State  (string, must include these options and may add options) | “state”: “inProgress”   * InProgress * InError * Completed | |

# Asynchronous Operations Details

Any operation in your API may be asynchronous, or can even offer both asynchronous and synchronous (default) versions of the operation. Most commonly though, tasks are likely to be long running and so utilize asynchronous operations.

Any operation that takes longer than 1 second should be asynchronous. Do not make result paging asynchronous, instead use the [URI query string](#_dg51jshex3xr) paging pattern with LIMIT and OFFSET.

There are 3 typical types of interactions relating to asynchronous operations:

1. [Kick off a new asynchronous operation](#_aqqex93204wz)
2. [Check up on the status of an asynchronous operation](#_5j0m7aog3b06)
3. [Subscribe to receive a notification when the asynchronous operation completes](#_beeclghpjqi)

With respect to event subscriptions (3), see the [Event Hub](#_i4u2jeartogd) section for more details, which are not specific to asynchronous operations and the monitor entity.

| Consumer kicks off a new asynchronous operation | |
| --- | --- |
| **REQUEST** | |
| POST /*task* | Any operation may be asynchronous, though it’s most common for tasks. |
| Header  Expect: 202 Accepted | Optionally used by the Consumer to indicate that they want their request to be executed asynchronously by your API. If this cannot be accommodated by your API, return status 417 Expectation failed.  If you offer both a synchronous and asynchronous version of the operation, and no Expect is specified by the consumer, use the synchronous version by default. |
| Body  {  *task details*  } | Include the [task](#_dt7zmbqwhkr5) details relating to the request in the body, as with any other request. |
| **FAILURE RESPONSE** | |
| 417 Expectation failed | If your API cannot accommodate asynchronous operations, but the consumer has requested it using the Expect header, reject it with this code. |
| **SUCCESS RESPONSE** | |
| 202 Accepted | Indicates this will be executed asynchronously. |
| Header  Content-Type: application/json,  Content-Language: en\_GB,  Content-Length: …,  Last-Modified: …,  **Location:https://…/*resource*/*id,***  **Link:<https://…/monitor/id>;rel=related;title=monitor** | All the standard HTTP headers should be populated.  - Required, [resource](#_izt6jqzzplw) being operated on  - Required, [monitor](#_szktkw97f8me) for the resource |
| Body  {  [*monitor resource*](#_szktkw97f8me)  } | Return the [monitor](#_szktkw97f8me) resource. The consumer may use the provided link to check-up on their [monitor](#_szktkw97f8me) resource at any time. |

| Consumer checks on the status of the asynchronous operation | |
| --- | --- |
| **REQUEST** | |
| GET /monitor/*id*  GET /*resource*/*id*/monitor | Retrieve the [monitor](#_szktkw97f8me) that was linked in the previous response, in order to check on the status. A monitor cannot be modified; GET is the only verb that applies.  The monitor can be accessed via the [monitor](#_szktkw97f8me) collection with the monitor ID, or through the [resource](#_izt6jqzzplw) that the monitor is attached to with the resource ID. |
| **RESPONSE** | |
| 200 OK |  |
| Header  Content-type: application/json,  Content-Language: en\_GB,  Content-Length: …,  Last-Modified: … | All the standard HTTP headers should be populated. |
| Body: [monitor](#_szktkw97f8me)  {  “id”: …,  “state”: “[Completed](#_vk2sdracbvle)”,  “type”: “monitor”,  “href”: “.../monitor/*id*”,  “sourceHref”:“.../*resource*/*id*”,  “request”: {  “method”: “POST”,  “to”: “”,  “body”: {  *…*  },  “header”: [  {  “name”: “Accept”,  “value”: “application/json”  },  …  ]  },  “response”: {  “statusCode”: “202”,  “body”: {  …  },  “header”: [  {  “name”: “Content-Type”,  “value”: “application/json”  },  …  ]  }  } | - Must be InProgress, InError, or Completed  - Relative link to this monitor  - Relative link to [resource](#_izt6jqzzplw) being operated on  - Representation of the original request that kicked-off this operation.  - Representation of the response as it would have appeared in a synchronous operation, once the operation has completed (or failed). |

# 

| Consumer subscribes to receive a notification when the operation completes | |
| --- | --- |
| **REQUEST** | |
| POST https://…/monitor/*id*/hub | The consumer may subscribe to receive a notification in the event the monitor state changes. |
| Body:  {  “callback”: “https://…”,  “query”: “state=Completed,InError”,  “fields”: “*state,sourceHref,response*”  } | - Consumer URL to be called on event  - Select for notification when state changes  - Select certain relevant monitor fields |
| **RESPONSE** | |
| 201 Created |  |
| Header  Content-type: application/json,  Content-Language: en\_GB,  Content-Length: …,  Last-Modified: …,  **Location: https://…/hub/*id*** | All the standard HTTP headers should be populated.  - Location of the new hub instance |
| Body: [hub](#_6lstatutr0s)  {  “id”: …,  “callback”: “https://…”,  “query”: “state=Completed,InError”,  “fields”: “state,sourceHref,response”  } | - ID generated upon creation of this hub  The rest echoes the content of the request. |

# 

# Event Hub

In order to enable choreography between microservices in our future architecture, all APIs should enable event-driven functionality. That is, consumers should be able to subscribe to receive notifications in the case that a certain event occurs in your API. For example, a consumer may want to be made aware when a new trouble-ticket is created. The event hub enables this using a pub/sub model.

| Consumer subscribes to events | |
| --- | --- |
| **REQUEST** | |
| POST /hub  POST /*resource*/hub | A consumer may subscribe to generic API events, or to events relating to a specific resource type. |
| Body:  {  “callback”: “https://…”,  “query”: …,  “fields”: …  } | - Consumer URL to be called on event  - Limits the kinds of events to be notified of  - Limits the response content |
| **RESPONSE** | |
| 201 Created |  |
| Header  Content-type: application/json,  Content-Language: en\_GB,  Content-Length: …,  Last-Modified: …,  **Location: https://…/hub/*id*** | All the standard HTTP headers should be populated.  - Location of the new hub instance |
| Body: [hub](#_6lstatutr0s)  {  “id”: …,  “callback”: “https://…”,  “query”: …,  “fields”: …  } | - ID generated upon creation of this hub  The rest echoes the content of the request. |

| Consumer cancels their subscription | |
| --- | --- |
| **REQUEST** | |
| DELETE /hub/id |  |
| **RESPONSE** | |
| 204 No content |  |

# Definition of Backwards Compatibility

A change is considered backwards compatible if all of the following are true:

* Anything added is optional
* Nothing that was optional has been made mandatory
* Nothing has been removed

# Hypermedia / HATEOS

Hypermedia is like the navigation bar of a website, but for an API. It’s meant for the API to communicate to the consumer what other kinds of requests the consumer might want to make.

The hypermedia information can be included in the [Link header](#_zak7i7ke8qij) in any response, or as a [\_links resource](#_zak7i7ke8qij) in the body (both using the same format).

The [/home URI](#_hl5vj0eayect) for your API is the homepage of your API, and must return [links](#_zak7i7ke8qij) to all operations available within your API.