# ARIS Mobile API

# TECHNICAL INTRODUCTION

ARIS 10.0 - Service Release 3

December 2017

Document content not changed since release 10.0.2. It applies to version 10.0.3 without changes.

This document provides additional information for accessing an ARIS repository via the ARIS API.
It should be considered as a supplement to the official API documentation which is available on every running ARIS Connect/Design Server at **http://<servername:port>/apidocs**.

# TABLE OF CONTENTS

# 0 General principles

- The ARIS RESTful APIs are designed to access an ARIS repository by apps typically running on mobile devices. It is not meant as a replacement for other ways to access an ARIS repository, e.g. the ARIS Report API. Therefore it has various limitations with respect to functionality that you might have expected.
- Every call to the API is atomic: either the operation succeeds or fails.
- Every call to the API requires a valid API cookie/token from ARIS User Management.
- Many calls have obligatory and/or optional URL parameters. All parameter values must be URL-encoded as they may contain special characters.
- If URL parameters are passed that are unknown or not the correct ones for the specific method or misspelled, then they are silently ignored. This may lead to an unexpected outcome of the requested operation.
- Many calls require a database language as parameter. If not given, the fallback language of the current database is used.
- Many operations require an ARIS method filter GUID as parameter. If not given, a specific auto-selection mechanism chooses the right filter, similar to a login on the Connect portal. It is strongly recommended to pass the desired filter as parameter which is also much faster.
- The result objects may contain method data, e.g. `typename.` Method data is delivered in the language from the client's HTTP header ("accept-language"). Alternatively, you can pass an optional URL parameter `methodlanguage` in order to set the method language directly.
- Date parameters must be provided and are returned in UTC in the RFC 3339 Internet format YYYY-MM-DD, and for timestamps in UTC in the RFC 3339 Internet Zulu time format is needed YYYY-MM-DD'T'HH:MM:SS'Z', e.g. 2014-01-22T08:22:55Z.

# 1 Login

For all API calls, a valid token from ARIS User Management (also known as UMC) is required. The token must be obtained by calling a dedicated method from the UMC API. It is not allowed to use a standard UMC token obtained by logging in to the Connect portal.

The token must be included in all subsequent API calls as URL parameter (**`umcsession`**) or added as cookie to the HTTP request.

```
POST
http:// <servername:port>/umc/api/tokens?tenant=default&name=system
                                        &password=manager&key=lsjflskjfsfj
```

**Remarks**:

A typical problem is that the client programmer forgets to urlencode all parameters.

The parameter **key** is the string representation of an X.502 certificate. It must be requested from Software AG for certifying the app: https://www.ariscommunity.com/aris-access-certification-inquiry-form

# 2 Get information on available databases

```
GET
http:// <servername:port>/abs/api/databases
```

This call returns a list of all available databases for the current user. It is the same list that a user would see when opening ARIS Architect.

For each database, its name, main group GUID and the isversioned flag is returned.

Example:

```
{
"kind": "DATABASE",
"name": "United Motor Group",
"isversioned": true,
"maingroup_guid": "4a713de0-5d02-11e3-0fda-fd81e986d7e2"
}
```

**Note**: A licence of type “ARIS Access Full” or “ARIS Access Read-only” is required for this operation.

```
GET
http:// <servername:port>/abs/api/databases/United%20Motor%20Group
```

This call returns more detailed information for the given database name: main-group GUID, isVersioned flag as well as a list of all allowed method filters and db languages. Exactly one database language will have the flag `isalternative = true` which means that it will be used as the fallback database language if a request does not pass the URL parameter `language`.

In case of a versionable database, the existing change lists are returned as well.

**Example for method filter item**

```
{
"kind": "METHODFILTER",
"guid": "dd838074-ac29-11d4-85b8-00005a4053ff",
"name": "Entire method",
"description": "All method content is available."
}
```

**Example for database language item**

```
{
"kind": "DBLANGUAGE",
"language": "en_US",
"isalternative": true
}
```

**Example for changelist item**

```
{
"kind": "CHANGELIST",
"changelist_number": 1,
"user": "internal",
"description": "initial revision after restore from basic archive file.",
"submit_time": "2015-04-07T16:11:13Z"
}
```

# 3 Get an item

The ARIS API offers to retrieve groups, objects (definitions) and models. The desired item must be identified by its ARIS GUID or a full CONNECT item-id (e.g. `c.process.United Motor Group.CibrcP1SEdsnKQALzQzOTg.-1`).

Identifying via group path + name is unsupported as this is possibly ambiguous.

**Example**:

```
GET
http://<servername:port>/abs/api/groups/United%20Motor%20Group/1191ae90-02f7-11dc-2729-000bcd0cce4e
```

By default, only the name attribute (AT_NAME) is included in the response. If you need more attributes, the URL parameter **`attributes`** must be given. Possible values are

- **`all`**: all non-empty attributes
- a comma-separated list of attribute type numbers, API names or type-GUIDs (all of these can be arbitrarily mixed)

**Example**:

**`attributes = 1, AT_DESC, AT_AUTH`** // attributes name, description, author

**Remarks**: Connections are not supported.

**Note:** It is also possible to retrieve HTTP portal links that can be directly used in the browser to access the item via the CONNECT portal. Add URL parameter **`withportallinks = true`**.

**Portal Link Examples (for a model)**:

```
{
"kind": "LINK",
"method": "GET",
"href": "http://<servername:port>/#default/item/c.process.United Motor
Group.CibrcP1SEdsnKQALzQzOTg.-1",
"rel": "ITEM_MODEL"      // CONNECT Item View
},

{
"kind": "LINK",
"method": "GET",
"href": "http://<servername:port>/#default/repository/a.model.United Motor
Group.CibrcP1SEdsnKQALzQzOTg.-1",
"rel": "REPO_MODEL"      // CONNECT Repository View
},

{
"kind": "LINK",
"method": "GET",
"href": "http://<servername:port>/#default/thinclient/c.process.United Motor
Group.CibrcP1SEdsnKQALzQzOTg.-1",
"rel": "TC_MODEL"        // CONNECT Designer View
},

{
"kind": "LINK",
"method": "GET",
"href": "http://
<servername:port>/abs/downloadClient/aris_database.jsp?configuration=ARIS&appserver=myserver&database=United%20Motor%20Group&guid=0a26eb70-fd52-11db-2729-000bcd0cce4e&language=en_US&tenant=default",
"rel": "DC_MODEL"        // CONNECT Download client
}
```

## 4 Get group children

A group usually contains children, i.e., subgroups as well as models and objects.

```
GET
http:// <servername:port>/abs/api/groups/United%20Motor%20Group/4a713de0-5d02-11e3-0fda-fd81e986d7e2/children
```

This call returns all subgroups of the group identified by the given GUID or CONNECT item-iD. If you additionally need the models and/or objects in the group, you can add the URL parameters **`withmodels = true`** or **`withobjects=true`**.

```
GET
http:// <servername:port>/abs/api/groups/United%20Motor%20Group/4a713de0-5d02-11e3-0fda-fd81e986d7e2/children?withmodels=true&withobjects=true
```

**Remark**: It also possible to retrieve an entire subtree by passing the URL parameter **`recursive=true`**. Note that there is an absolute result size limit of 5000[1] items (regardless of the page size you choose). If a group contains more than 5000 children, you will not be able to retrieve them all.

[1] Other size limits may apply depending on the ARIS licence.

# 5 Get content of a model

The content of a model are its occurrences (= model objects, model connections). They can be retrieved in a getModel call by passing the URL parameter **`withcontent=true`**.

```
GET
http://<servername:port>/abs/api/models/United%20Motor%20Group/88ba40a0-cfb6-
11e0-2556-5c260a398437?withcontent=true
```

**Remarks**: Model objects/model connections are a blend of data from occurrence and definition level. They do not contain any graphical information, i.e. no dimension (width, height) and no coordinates of positions and edge paths.

**Example for model object item**

```
{
"kind": "MODELOBJECT",
"occid": "(7wLHfY8btgy:u:L+6PmFOGfzQQ4:x:L+33+c)",
"guid": "d734eb6f-cf14-11e0-2556-5c260a398437",
"link": {
      "kind": "LINK",
      "method": "GET",
      "href": "http://<servername:port>/abs/api/objects/United Motor
Group/d734eb6f-cf14-11e0-2556-
5c260a398437?language=en_US&methodfilter=dd838074-ac29-11d4-85b8-
00005a4053ff",
       "rel": "OBJECT"
       },
"type": 239,
"typename": "Strategy",
"apiname": "OT_STAT",
"symbol": 1627,
"symbolname": "Tactic",
symbol_apiname": "ST_TACTIC",
"attributes": [
       {
       "kind": "ATTRIBUTE",
       "id": "6S4A4i43Hh0:p:L=1=1033:1:s",
       "typename": "Name",
       "type": 1,
       "apiname": "AT_NAME",
       "language": "en_US",
       "value": "Reduce operational costs in Supply Chain"
       }
      ]
 }
```

**Example for model connection item:**

```
{
"kind": "MODELCONNECTION",
"occid": "(7wLHfY8btgy:u:L+-6xMo0A_0oEz:y:L+34+c)",
"type": 67,
"typename": "encompasses",
"apiname": "CT_SUBS_1",
"source_guid": "c7ca78b0-abcf-11e0-7ee8-5c260a398437",
"target_guid": "289b0560-ac64-11e0-7ee8-5c260a398437",
"source_link": {
"kind": "LINK",
"method": "GET",
"href": "http://<servername:port>/abs/api/objects/United Motor Group/c7ca78b0-
abcf-11e0-7ee8-5c260a398437?language=en_US&methodfilter=dd838074-ac29-11d4-
85b8-00005a4053ff",
  "rel": "OBJECT"
},
"target_link": {
       "kind": "LINK",
       "method": "GET",
       "href": "http://<servername:port>/abs/api/objects/United Motor
Group/289b0560-ac64-11e0-7ee8-
5c260a398437?language=en_US&methodfilter=dd838074-ac29-11d4-85b8-
00005a4053ff",
       "rel": "OBJECT"
       },
"source_occid": "(7wLHfY8btgy:u:L+-7cJgUsuG4Td:x:L+33+c)",
"target_occid": "(7wLHfY8btgy:u:L+-2apRUFiN0WT:x:L+33+c)"
}
```

# 6 Create an item

The API offers support for creating groups and objects. For objects, groups and models, it is possible to create attributes (or more correctly formulated in ARIS terminology: it is possible to maintain attributes with a value). It is allowed to add multiple attributes in the same call --> for all of them, the same database language will be used (from URL parameter **`language`**).

Furthermore, it is possible to create an assignment relationship between an existing object and an existing model.

**Note**: The creation of models is not supported!

**Example**:

Create a new object of type 43 (OT_ORG_UNIT) with two attributes (AT_NAME and AT_DESC). Database language = English; parent group = 4a713de0-5d02-11e3-0fda-fd81e986d7e2 (= main group)

POST

```
http://
<servername:port>/abs/api/objects/United%20Motor%20Group?language=en&parent=4a
713de0-5d02-11e3-0fda-fd81e986d7e2
```

Request-Body

```
{
"kind": "OBJECT",
"type": 43,
"attributes": [
      {
      "kind": "ATTRIBUTE",
      "type": 1,
      "value": "My first OrgUnit "
      },
      {
      "kind": "ATTRIBUTE",
      "type": 9,
      "value": "This is the long description"
      }
      ]
}
```

**Remarks**:

The “`kind`” entries in the request body are redundant and not required.

There is a slight difference in the behaviour whether creating an object or a group with a duplicate name.

- If you create a new group with name X and there is already a group X in the desired parent group, then the new group will automatically be renamed to X(1).
- If you create a new object with name X and there is already an object X in the desired parent group, then the new object will have name X, i.e. there will be two objects having the same name X!

# 7 Create model content

Models cannot be created via the ARIS Mobile API, but it is allowed to create model objects and model connections. When creating a new connection, it is possible to use objects that are created in the same request. As they do not have an ID yet, you must define a temporary occid (starting with #) and use these temporary occids when defining source and target of the new connection.

**Example**:
PUT
http:// <servername:port>/abs/api/models/United%20Motor%20Group/42ad5380-3d0e-11e5-6479-22000b630ca4

Request-Body:

```
{
"modelobjects": [
      {
      "kind": "MODELOBJECT",
      "occid": "#1",
      "type" : 22,
      "symbol": 335,
      "attributes": [
            {
            "kind": "ATTRIBUTE",
            "type": 1,
            "value": "new function via model update"
            }
            ]
      },
      {
      "kind": "MODELOBJECT",
      "occid": "#2",
      "type" : 18,
      "symbol": 1,
      "attributes": [
            {
            "kind": "ATTRIBUTE",
            "type": 1,
            "value": "new event via model update"
            }
            ]
      }
      ],
"modelconnections": [
      {
      "kind": "MODELCONNECTION",
      "type": 44,
      "source_occid": "#1",
      "target_occid": "#2"
      }
      ]
}
```

**Remarks**:

- When creating objects as shown above, a new definition object is created. If you want to reuse an already existing object, provide the GUID, e.g.

  ```
  {
  "kind": "MODELOBJECT",
  "occid": "#1",
  "guid" : "4686bd20-3d0e-11e5-6479-22000b630ca4"
  "type" : 22,
  "symbol": 335
  }
  ```

- As model objects and model connections are created without any positional information, they will be located one above the other in the upper left corner of the model. It is possible to enforce an automatic layout of the model when it is opened for the first time in ARIS Business Architect Designer by passing the URL parameter **`layoutonopen=true`**

**Note**: A user with license ARIS Mobile Access can only create one model object or one model connection in a single request.

# 8 Delete model content

Model objects or model connections can be deleted from a model via their occ-ids. If you delete a model object, then all affected model connections are deleted automatically (no need to include them in the request).

DELETE
```
http://<servername:port>/abs/api/models/United%20Motor%20Group/1672a301-3d14-11e5-6479-22000b630ca4/objects
          ?occid=(-7O3yxAzzRgN%3Au%3AL%2B-7RaSdNnA6hr%3Ax%3AL%2B33%2Bc)
```

**Remarks**:

- Occ-IDs often contain special characters --> it is important to ensure properly urlencoded values.
- In the request path, make sure to have /objects or /connections at the end, otherwise the model itself will be deleted!

**Note**: A user with licence ARIS Mobile Access can only delete one model object or one model connection in a single request (automatically deleted model connections are not counted).

# 9 Attributes

Attributes are no top-level items and are tightly bound to their parent item (object, group, model). You can retrieve attributes only along with its parent item to which they belong.
Attributes can be identified by an integer type number (e.g. 1) , the API name (e.g. AT_NAME) or a type-GUID. These type identifiers can be freely mixed in the same request and even the same parameter, for instance in the URL parameter **`attributes`**.
Attributes are maintained in the database language as given by the URL parameter **`language`**. If no such parameter is given, the default language of the current database is used.

**Remarks**:

- Currently, only a limited subset of attribute types is supported, notably integer, float, text, timestamp, time, value attributes, binary
- Binary data must be sent as BASE64 encoded string

## 9.1 Creation

You can create an attribute together with its parent in the same request.
**Example**: Create an object with two attributes “name” and “description”

POST
```
http:// <servername:port>/abs/api/objects/United%20Motor%20Group?language=en
                          &parent=4a713de0-5d02-11e3-0fda-fd81e986d7e2
```

Request-Body:
```
{
"kind": "OBJECT",
"type": 43,
"attributes": [
      {
      "kind": "ATTRIBUTE",
      "type": "AT_NAME",
      "value": "I am an OrgUnit!"
      },
      {
      "kind": "ATTRIBUTE",
      "type": 9,
      "value": "This is the long description"
      }
      ]
}
```

## 9.2 Updating or creation for existing item

If you want to add a new attribute to an already existing item or if you want to update the value of an existing attribute, you must do an update (PUT) operation on the item.

**Example**: Update the name of an existing object and add the author attribute

PUT
http:// <servername:port>/abs/api/objects/United%20Motor%20Group/b7e90c56-00cf-11e2-21d1-5c260a628455?language=en

Request-Body:

```
{
"attributes": [
      {
      "kind": "ATTRIBUTE",
      "type": 1,
      "value": "Just renamed it"
      },
      {
      "kind": "ATTRIBUTE",
      "type": "AT_AUTH",
      "value": "I am the author"
      }
      ]
}
```

## 9.3 Deleting an attribute

Deleting an attribute cannot be done via an update operation on the parent item. Instead, the dedicated DELETE operation must be used. The type numbers of the attributes to be deleted must be passed via the URL parameter **`typenumbers`**

DELETE
http:// <servername:port>/abs/api/objects/United%20Motor%20Group/b7e90c56-00cf-11e2-21d1-5c260a628455/attributes?language=en&typenumbers=AT_DESC, AT_AUTH

**Remarks**:

- Do not forget to have **/attributes** at the end of the URL path, otherwise you do a DELETE on the parent object and it is gone!
- The result of a successful delete operation is simply empty with STATUS = OK, e.g.

```
{
"kind": "RESULT",
"request": "abs#deleteObjectAttributes",
"status": "OK",
"item_count": 0,
"items": []
}
```

**Note:** only one attribute can be deleted in a single request for users with license ARIS Mobile Access.

## 9.4 Styled values for text attributes

In ARIS, text attributes (e.g. AT_DESC) can also have a styled value. If you want to retrieve text attributes with their styled value, you must set the URL parameter **`withstyledtext = true.`** If the text attribute has got a valid styled value, it will then be returned in addition to the standard value field.

A typical response would like as follows:

```
{
"kind": "ATTRIBUTE",
"id": "2Nc1gl0g_LN:p:L=9=1033:1:s",
"typename": "Description/Definition",
"type": 9,
"apiname": "AT_DESC",
"language": "en_US",

 "styled_value":
   "<html><body>
     <p  style=\"margin-left:0;margin-bottom:0;
                margin-top:0\"><b>my&#32;updated</b>&#160;description
     </p>
  </body></html>",

   "value": "my updated description"
}
```

The styled value is HTML starting with tags `<html><body>.`
It is also possible to set a styled value when creating or updating an attribute. Simply pass "styled_value" with a valid HTML string.

**Remarks**:

- The URL parameter **`withstyledtext`** is not required for write-operations (POST, PUT).
- If a write-operation (POST, PUT) contains both `value` and `styled_value`, then `styled_value` has precedence and `value` is basically ignored. According to standard ARIS behaviour, `value` will then contain a plain-text representation of the HTML from `styled_value`.
- Only a basic set of HTML is supported: <b>, <i>, <u>, <strike>, <ul>, <ol>, <li>, <br>, <p>, <span>, <div>.
- For styling information (margin, color, font-family, size) , use an inline style attribute, e.g.
  ```
  <span style=\"font-size:48pt;\">
     <span style=\"font-family:Kokila;\">Hello World</span>
  </span>
  ```
- Unsupported HTML tags are silently ignored (only their text content will be considered).
- Make sure to correctly escape JSON special characters, e.g. \n must be escaped as \\n and " as \"
  .
- If the HTML in `styled_value` does not contain any real formatting, then this will lead to setting a plain text value, e.g.

  <html><body><p>   My name is Bond. James Bond.</p></body></html>

  Result: "value"           : "My name is Bond. James Bond."
          "styled_value" : null

# 10 Assignments

An assignment is simply a link between an existing object and an existing model. With the ARIS Repository API, you can create and delete assignments. It is not possible to retrieve a specific assignment: assignments are always included automatically in the response when you retrieve an object.

**Example**: Create an assignment

POST
http:// <servername:port>/abs/api/objects/United Motor Group/**6aa1b167-fac0-11de-55c7-001a6b3c820f**/assignment/**a39aefb0-fa1c-11db-2729-000bcd0cce4e**
**objectGUID** (source of assignment)
**modelGUID** (target of assignment)

**Example**: Delete an assignment

DELETE
http:// <servername:port>/abs/api/objects/United Motor Group/**6aa1b167-fac0-11de-55c7-001a6b3c820f**/assignment/**a39aefb0-fa1c-11db-2729-000bcd0cce4e**

## 11 Model Graphic

The ARIS API does not provide positional information when retrieving a model with its contents. But it is possible to get a model graphic. It has format PNG (Portable Network Graphics) and is returned as a BASE64-encoded string. There are two optional URL parameters `maxwidth` and `maxheight` that allow you to control the size (in image pixels) of the generated graphic.

**Example**:
GET
http://<servername:port>/abs/api/models/United%20Motor%20Group/0a26eb70-fd52-11db-2729-000bcd0cce4e/graphic

# 12 Finding items in the database

It is possible to search for items based on several criteria which are AND-combined, i.e. they reduce the number of matching result items. You must pass the URL parameter kind in order to declare what to search (model or object). In addition, you can define the desired type (= comma-separated list of object type numbers or model type numbers; API-names or type-GUIDs are also supported) and optionally a filter criterion with respect to an attribute.

**Example**: Find all models of type 13 (MT_EEPC) whose names start with Sale (both small and capital S)
GET
`http://<servername:port>/abs/api/databases/United%20Motor%20Group/find?kind=MODEL&typefilter=13&language=en&attrfilter=AT_NAME=Sale`

**Remarks**:

- Available item kinds: MODEL, OBJECT, GROUP
- For all attributes: + (isMaintained), - (isNotMaintained)
- For numbers, the usual operators are available (=, !=, >, <, >=, <=)
- For Boolean: = and !=
- For text attributes: =
  **Note**: unequal (!=) is not supported because it may lead to full-table scans in the underlying database index.
- The parameter **`attrfilter`** may contain fully parenthesized expressions (parenthesis = { }), combined with operators AND, OR

**Searching for text: examples for the value of `attrfilter`**:

```
AT_NAME   = Jones
9         = Human Resources      // (9 = AT_DESC)
AT_AUTH -                        //  (AT_AUTH not maintained)
{AT_NAME = Jones} AND {AT_DESC = Human Resources}
{{AT_NAME = Jones} OR {AT_DESC = Human Resources}} AND {AT_AUTH -}
```

- **Important**: in case of a parenthesized expression, the search value must not contain `{` or `}`.
  If the search value must contain a curly parenthesis, pass an escaped character, e.g. `\}` when searching for text attributes
- The normal search behaviour is similar to standard search engines: contents of text attributes are scanned on the basis of tokens, i.e., typical separators like blank or dash (-) are ignored. If tokens from the search value are contained in the text attribute (regardless of order/position) this is considered a match.

  **Example:** search expression: `AT_NAME = holder certificate`
  will find "Holder Certificate" and "certificate holder" and "the certificate holder"

- URL parameter **`matchcase`**: if true, searching will be case-sensitive and order of tokens is relevant

- URL parameter **`exactsearch`**: if true, only exact matches are returned; special characters and notably the asterisk ( * ) are interpreted as simple character; in addition, the order of tokens will matter
  **Example**: search expression: `AT_NAME = holder certificate`
  will find "Holder Certificate" but not "certificate holder" and not "The Holder Certificate"

- Special search patterns for text: (parameter `exactsearch` must be false)
  `A*` returns all strings that start with A (A must be a string of length >=2, the first two characters must be letters or digits)
  `A*B` returns all strings that start with A (A must be a string of length >=2, the first two characters must be letters or digits) and end with B

# 13 Paging

Typically when using the database-find operation, many items will be found and returned to the caller. A default page size of 100 is in effect. A different page size can be indicated with the URL parameter **pagesize**.

If there are more items to be returned than the page size allows, the response will contain an entry **next_pagetoken**. This indicates to the client that there is more data to load.

**Example**:
Request:
http://<servername:port>/abs/api/databases/United%20Motor%20Group/find?kind=MODEL&language=en&attrfilter=AT_NAME&wildcards=true&attrcriterion==S*

Response

```
{
"kind": "RESULT",
"request": "abs#find",
"status": "OK",
"item_count": "100",
"next_pagetoken": "1:100:guid",
"items": [
…  100 items here
]}
```

In order to get the next page of results, the client must send the same request again and add the page token:
http://<servername:port>/abs/api/databases/United%20Motor%20Group/find?kind=MODEL&language=en&attrfilter=AT_NAME&wildcards=true&attrcriterion==S*&pagetoken=1:100:guid

Response

```
{
"kind": "RESULT",
"request": "abs#find",
"status": "OK",
"item_count": "35",
"items": [
… remaining 35 items here
]
}
```

The second response contains no entry **next_pagetoken** --> the client knows that there are no more items to load.

**Note**: The maximum allowed page size is 500. If  a page size is given that is too large, it will be silently corrected to the allowed maximum.

## 14 Ordering the result

If more than one item is returned by a request (e.g. database-find), the items are ordered by their GUID in ascending order. Ordering by GUID itself is normally useful but this ensures a stable result, especially with respect to paging.

You can pass the URL parameter **`orderby`** in order to indicate a different sorting, e.g.

```
…&orderby = name     (order by attribute AT_NAME, ascending)
…&orderby = -name    (order by attribute AT_NAME, descending)
```

Other possible values: **`modified`** (ascending), -**`modified`** (descending), **`path`** (group path starting at main group)

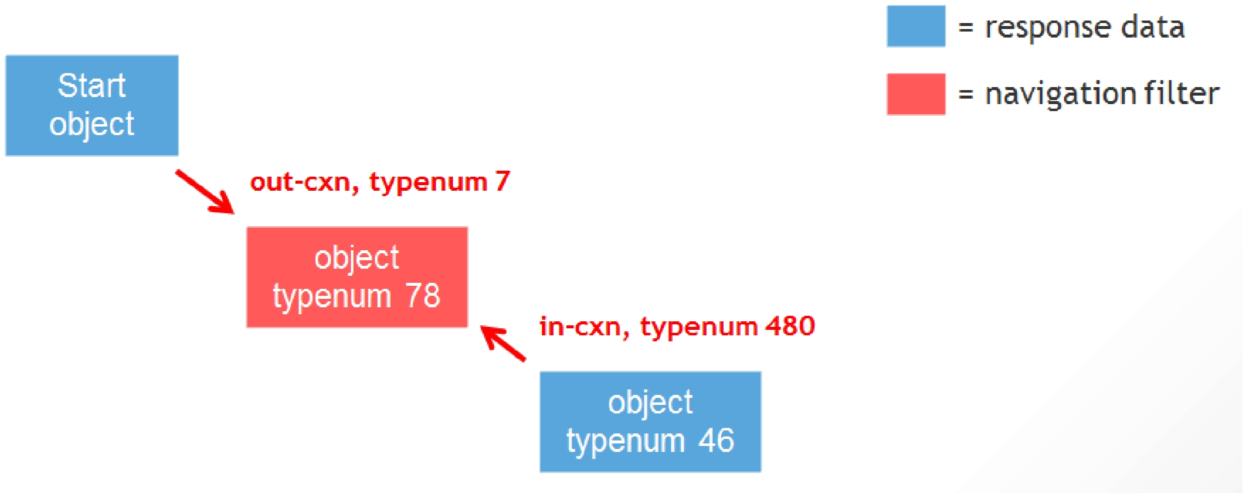
## 15 Generic queries

Generic queries try to offer a kind of graph-based retrieval of items. The syntax and functionality is based on the CONNECT Portal Designer query language which is used for the contents of fact sheets.

Starting from a defined item (object or model identified by GUID/CONNECT item-id or a type number), it is possible to add items that can be reached from the start item via a sequence of filter items.

The general properties of an item definition are as follows:

**type** : MODEL, OBJECT, CONNECTION, ASSIGNMENT, OCCURRENCE, CONNECTIONOCCURRENCE
**function** : FILTER, TARGET (TARGET means that objects that match this item are added to the result)
(FILTER means that this item is only used for filtering; it is the default value
and therefore optional)
**direction**: IN, OUT, BOTH (only allowed for type CONNECTION)
**typenum**: comma-separated list of type numbers, api-names, type-GUIDs



**Remarks**:

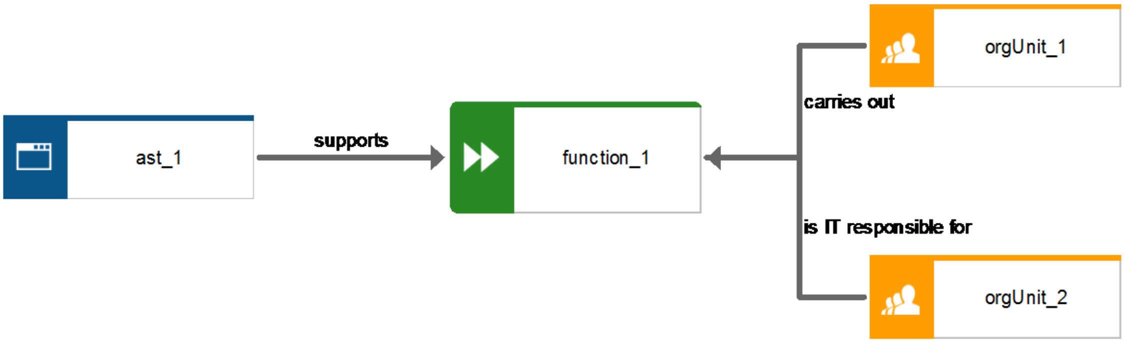
- Not all possible combinations will lead to sensible results! The syntax promises more than what is implemented.
- Item definitions normally contain other item definitions and thus form a nested path of items. Within a single path, only one item of function TARGET is allowed.
- Although a generic query is of read-only nature, it must be sent as a POST request because the query definition must be provided in the request body.
- If the start item is an object, the request URL must be … `/objects/<db>/query`; in case of models, it is …`/models/<db>/query`

The request body must contain the query itself as a nested list of query item definitions.

**Example 1 (starting from object)**:
POST:
`http:// <servername:port>/abs/api/objects/United%20Motor%20Group/query`

Request-Body:

```
{
"start_guids" : "a387cc60-3c34-11e5-216f-782bcb6367e0",
"items" : [
      { "type" : "CONNECTION",
      "direction" : "IN",
      "items" : [
            {
            "type" : "OBJECT",
            "typenum" : "6,43",
            "function" : "TARGET"
            }
            ]
      }
      ]
}
```



**Result:** start function (function_1), ast_1 (type 6) , orgUnit_1 (type 43) , orgUnit_2 (type 43)

**Example 2 (starting from model)**:
POST:
http://<servername:port>/abs/api/models/United%20Motor%20Group/query

Request-Body:

```
{
"start_guids" : "0a26eb70-fd52-11db-2729-000bcd0cce4e",
"items" : [
      {
      "type" : "OCCURRENCE",
      "items" : [
            {
            "type" : "OBJECT",
            "typenum" : "22",
            "function" : "FILTER"
            },
            {
            "type" : "CONNECTIONOCCURRENCE",
            "items" : [
                  {
                  "type" : "CONNECTION",
                  "function" : "FILTER",
                  "direction" : "IN",
                  "items" : [
                        {
                        "type" : "OBJECT",
                        "typenum" : "43,6",
                        "function" : "TARGET"
                        }
                        ]
                  }
                  ]
            }
            ]
      }
      ]
}
```

**Result:** start model, function "Configure vehicle" (type 22), application system type COT (type 6)

.

**ABOUT SOFTWARE AG**