



This PDF is generated from authoritative online content, and is provided for convenience only. This PDF cannot be used for legal purposes. For authoritative understanding of what is and is not supported, always use the online content. To copy code samples, always use the online content.

Developer's Guide

Extensions

Extensions



Purpose: Offers guidelines for managing Extensions provided by the Context Services.

Contents

- [1 Extensions](#)
 - [1.1 About Extensions](#)
 - [1.2 Managing Extension Schema](#)
 - [1.3 Managing Extensions](#)
 - [1.4 Read More](#)

In versions 8.1.000.10 and higher, you need to check the privileges set according to roles prior to using the operations described on this page. See [Role-Based Access Control](#) for additional details.

About Extensions

Extensions are additional information which extend the standard contents of resources such as [Customer Profile](#), [Service](#), [State](#), and [Task](#). An [Extension](#) is a record-a list of attributes-or an array of records, associated with a resource ID.

- You can define as many extension types as you need by creating an [Extension Schema](#) for each of them.
- Extension schema are created through Context Services (see [List of Schema Operations](#)), not through the Configuration Layer (Configuration Manager).

Extension records can be either:

- "single-valued": The extension contains a single record across the resource (for instance, LastName, FirstName, identifiers, etc.)
- "multi-valued": The extension can contain several values (for instance, phone numbers, e-mail addresses, etc.)

Extensions are provided at the same time and at the same level than the attributes of the resource. For instance, the following output presents a profile containing the attributes FirstName, LastName, DOB<ref>DOB: Date Of Birth</ref> and one multi-valued extension *EmailAddress*:

```
{
  "FirstName": "Bruce",
  "LastName": "Banner",
  "DOB": "1962-05-10",
  "EmailAddress": [
    "bruce.banner@marvelous.com",
    "b.banner@hulk.dom"
  ]
}
```

Unique Attributes


In the case of multi-valued extensions, the attributes which are part of the 'unique' list (specified in the [Extension Schema](#)) are used to identify records. The combination of these attributes' values must be unique across the related resource, and this enables UCS to identify a given record in the given extension. For example, consider a 'Bill' extension which includes the attribute *bill_id*. To ensure that a given service does not have two 'Bill' extensions with the same *bill_id*, set the following unique array in the extension schema:

```
unique = ["bill_id"]
```

The attributes of the unique list are mandatory at the extension record's creation. You need to provide values for the 'unique' attributes:

- At the creation of an extension record.

- In operations which update or delete a specific record, such as [Update Record In Profile Extension](#) or [Delete Record From Profile Extension](#).

 Operations which manage extension records are part of the related resource operations. For instance, the operations which manage records of profile extensions are part of the [List of Profile Operations](#).

Limitations

- Once created, you cannot update the schema.
- When you are dealing with extensions or extension schema, make sure that you do not use one of the

[Unauthorized Strings](#) as an attribute name or value.

Managing Extension Schema

Operations and resources in this section are part of

Before you can start using extensions, you must create their schema.

 Once created, you cannot update or remove them.

You can create schema with the following operations:

- [Create Profile Extension Schema](#)
- [Create State Extension Schema](#)
- [Create Task Extension Schema](#)
- [Create Service Extension Schema](#)

Then, you can retrieve extension schema.

- [Query Profile Schema](#)
- [Query Profile Extension Schema](#)
- [Query State Extension Schema](#)
- [Query Task Extension Schema](#)
- [Query Service Extension Schema](#)

Example: Retrieving the schema for profile extensions:

```
GET /metadata/profiles/extensions
```

Result

```
200 OK
[
{
```

```
"name": "Phone",
"type": "multi-valued",
"attributes": [
  {"name": "PhoneType", "type": "integer", "default": 0, "mandatory": "true"},
  {"name": "prefix", "type": "string", "length": 3, "default": "555", },
  {"name": "PhoneNumber", "type": "integer", "length": 15, "mandatory": "true"},
  {"name": "description", "type": "string", "length": 32, "mandatory": "true"},
  {"name": "start_availability", "type": "datetime"},
  {"name": "end_availability", "type": "datetime", "mandatory": "false"}
],
{
  "name": "Address",
  "type": "single-valued",
  "attributes": [
    {"name": "AddressType", "type": "integer", "default": 0},
    {"name": "Address", "type": "string", "length": 256},
    {"name": "City", "type": "string", "length": 32},
    {"name": "County", "type": "string", "length": 32},
    {"name": "PostCode", "type": "string", "length": 10},
    {"name": "Country", "type": "string", "length": 32}
  ]
} ]
```

Managing Extensions

Adding Extensions to a given Resource

You can add extensions when managing the resources with related operations which authorize the `<extension n>` attribute in the operation's body (the following list may not be exhaustive): **Associate Service, Start Service, Complete Service, Start State, Complete State, Perform State Transition, Start Task, Complete Task.**

❗ In that case, if a former value of the extension exists for the given resource, this former extension value is replaced with the new extension value specified in the body.

Let's consider the following multi-valued extension record named 'Satisfaction'. The unique field which identifies records is "place" (the name of the proposed place for the booking. **Example: Records for a 'Satisfaction' extension**

```
[
  {
    "rating": 2,
    "pertinence": 8,
    "usefull": true,
    "place": "Terranova mexico resort"
  },
  {
    "rating": 8,
    "pertinence": 4,
    "usefull": false,
    "place": "Fancy resort Paris"
  }
]
```

The following operation **Complete State** indicates a single value for the 'Satisfaction' extension.

Example: Operation which updates the 'Satisfaction' extension for a given state.

```
POST /services/6739/states/5362/end
{
  "interaction_id": "00001a57JGQ00BVS",
  "disposition": 10,
  "disposition_desc": "SUCCESS",
  "application_type": "customer_online_survey",
  "application_id": 40,
  "resource_type": "html",
  "resource_id": 20,
  "media_type": "webform",
  "Feedback": {
    "FeedbackType": "survey",
    "rating": 7,
    "notes": "warm welcome at frontdesk, thanks for the nice trip"
  },
  "Satisfaction": [
    {
      "rating": 2,
      "pertinence": 6,
      "usefull": true,
      "place": "Marina Porto Vecchio"
    }
  ]
}
```

As a result, the previous records 'Fancy resort Paris ' and 'Terranova mexico resort ' are lost. In this case, to add a new record to the extension, you must specify the whole extension content. For instance, note the following: **Example: Operation which updates the 'Satisfaction' extension without losing records**

```
POST /services/6739/states/5362/end
{
  "interaction_id": "00001a57JGQ00BVS",
  "disposition": 10,
  "disposition_desc": "SUCCESS",
  "application_type": "customer_online_survey",
  "application_id": 40,
  "resource_type": "html",
  "resource_id": 20,
  "media_type": "webform",
  "Feedback": {
    "FeedbackType": "survey",
    "rating": 7,
    "notes": "warm welcome at frontdesk, thanks for the nice trip"
  },
  "Satisfaction": [
    {
      "rating": 2,
      "pertinence": 6,
      "usefull": true,
      "place": "Marina Porto Vecchio"
    },
    {
      "rating": 2,
      "pertinence": 8,
      "usefull": true,
      "place": "Terranova mexico resort"
    }
  ]
}
```

```
{
  "rating":8,
  "pertinence":4,
  "usefull":false,
  "place":"Fancy resort Paris"
}]
}
```

Retrieving Extensions

GET operations which enable to retrieve resources include the "extensions" parameter to specify a list of extensions to retrieve. By default, extensions are not returned. The following list is not exhaustive:

- [Query Customer Profile](#)
- [Query Services](#)
- [Query Service by ID](#)
- [Query States](#)
- [Query State by ID](#)
- [Query Tasks](#)
- [Query Task by ID](#)

Retrieving service with the extensions ClientInfo,relatedOffers

GET /services/3005?extensions=ClientInfo,relatedOffers

Result:

```
{
  "service_id" : 3005,
  "ClientInfo" : {
    "userAgent" : "Mozilla/5.0 (Windows; U; Windows NT 5.1; fr; rv:1.9.2) Gecko/20100115
Firefox/3.6 (.NET CLR 3.5.30729)",
    "clientId" : "192.168.1.1",
    "contentType" : "Content-Type : application/json;charset=UTF-8"
  },
  "service_type" : 100,
  "est_duration" : 300,
  "started" : {
    "timestamp" : "2010-09-07T07:58:16.313Z",
    "application_type" : 400,
    "resource_id" : 10,
    "media_type" : 2,
    "resource_type" : 200,
    "application_id" : 40,
    "interaction_id" : "56"
  },
  "disposition" : 5,
  "completed" : {
    "timestamp" : "2010-06-03T08:51:54.380Z",
    "interaction_id" : "1587"
  }
},
"relatedOffers" : [ {
  "offer_name" : "VIP credit card black ed.",

```

```
    "type" : "9",
    "comments" : "proposed to all client"
  }, {
    "offer_name" : "3 times payment GOLD",
    "type" : "4",
    "comments" : "limited offer"
  }, {
    "offer_name" : "life insurance",
    "type" : "3",
    "comments" : "healt check to be done before approval"
  } ],
  "contact_key" : "bob"
}
```

Managing Extension Records

The Context Services Restful API includes dedicated operations such as update or delete a record, or update the whole extension. They are available in the [List of Service Operations](#), [State Operations](#), and [Task Operations](#) pages. When dealing with these operations, you must provide **unique attributes** to identify the targeted record. For instance, to manage State Extensions, the [State Operations](#) provide the following operations:

- [Update State Extension](#)
- [Update Record In State Extension](#)
- [Delete Record From State Extension](#)

Example: Updating the *Tons of Hill, Paris* record of the extension *relatedOffers* in the service 8389.

```
PUT /services/8389/states/1/extensions/Satisfaction/by/unique
{
  "rating":2,
  "pertinence":8,
  "usefull":true,
  "place":"Tons of Hill, Paris"
}
```

Deleting an Extension

To delete the extension of a given resource, use the related *Update XXX Extension* operation with no attributes in the operation's body.

- [Update Customer Profile](#) and [Update Record In Profile Extension](#)
- [Update Service Extension](#)
- [Update State Extension](#)
- [Update Task Extension](#)

Example: Deleting the *relatedOffers* multi-valued extension of the service 8389

```
PUT /services/8389/extensions/relatedOffers
[]
```

In versions 8.1.000.10 and higher, as explained in [Role-Based Access Control](#),

you need update privileges to clear the extension, as follows:

- Clear Profile Extension—UCS.Customer.updateProfileExtension
- Clear Service Extension—UCS.Service.updateServiceExtension
- Clear State Extension—UCS.States.updateStateExtension
- Clear Task Extension—UCS.Tasks.updateTaskExtension

Read More

- [Profiles and Identification](#)
- [Services, States, and Tasks](#)