

LOCATIONS Model and Services 5.0

Document Status

status: Request for Comment (valid values are < Request for Comment, Preliminary Review, Public Review, Architectural Review, Final Review, Published, Deprecated)

Change Log

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Changes** |
| **0.0.1** |  | * Initial Creation |
| **0.0.2** |  | * Changed to include enhancements observations of working group |
| **3.0** | **10/29/2013** | * Switch to use X-HTTP-METHOD-OVERRIDE standard rather than subMethod non-Standard method for overriding request types. * Create a locationMessage wrapper for every message to increase ability for infrastructure to serialize the data * Versioning and format change with release CUFX 3.0 |
| **3.0** | **12/13/2013** | * Update examples X-API-Version to >=3.0.0 |
| **3.0** | **12/15/2013** | * Edit changes |
| **3.1** | **07/17/2015** | * Updated to release 3.1 |
| **3.2** | **05/10/2016** | * Updated to release 3.2 |
| **3.3** | **02/15/2017** | * Updated to release 3.3 |
| **4.0** | **02/19/2018** | * Updated to release 4.0, Date Range Global Update, Microsoft Global bug fix, changed additionalDataList to customData for consistency, added LocationIdList. Replaced locationId with locationIdList, type with locationTypeList, added servicesList, and isLocationOpen to filter. |
| **4.1** | **12/10/2018** | * Updated to release 4.1 |
| **4.2** | **03/05/2019** | * Updated to release 4.2, \*\*\* Release 4.2 is a breaking fix release. \*\*\* Errors found in App, ArtifactFilter, and BillFilter required a breaking fix to align with the standard and prevent additional implementation difficulties going forward. |
| **4.3** | **10/07/2019** | * Updated to release 4.3, renamed file removing version as proper version control is being used in Github. |
| **4.4** | **10/20/2020** | * Updated to release 4.4 |
| **4.5** | **04/02/2021** | * Updated to release 4.5 |
| **5.0** | **06/07/2021** | * Updated to release 5.0 |
|  |  |  |

Overview of Specification

The Locations specification describes the services and data structures used as part of an offering to allow a user to search for physical locations, such as ATMs and branches. A typical example application in which this service would be used is a mobile ATM locator. In this example, a user would use his device’s GPS coordinates to locate the nearest ATM’s that are served by his financial institution.

This specification describes the data structures and services for the described locator service.

Any known Errors in the document

|  |  |
| --- | --- |
| **Error Description** | Status of Error |
|  |  |

Table of Contents

Document Status 1

Change Log 1

Overview of Specification 2

Any known Errors in the document 2

Table of Contents 2

Document Conventions 2

CUFX API and Documentation Support 3

Release 4.0 Global Update Notes 3

Release 4.4 Global Update Notes 4

Release 5.0 Global Update Notes 4

Definitions related to the specification 4

DATA ELEMENTS 4

DATA ELEMENT: LOCATIONMESSAGE 5

DATA ELEMENT: LOCATION 5

DATA ELEMENT: LOCATIONLIST 5

DATA ELEMENT: LOCATIONFILTER 5

DATA ELEMENT: MESSAGECONTEXT 6

Location Services 7

SERVICE DEFINITIONS 7

SERVICE MESSAGE: GETLOCATIONS 7

Bibliography 10

# Document Conventions

List any document conventions such as what bold and italics mean and how the document is intended to be read.

“Within this specification, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in W3 Working Group (W3C)]. However, for readability, these words do not appear in all uppercase letters in this specification.

At times, this specification recommends good practice for authors and user agents. These recommendations are not normative and conformance with this specification does not depend on their realization. These recommendations contain the expression "We recommend ...", "This specification recommends ...", or some similar wording.”

All formatting in this document utilize Word Styles.

All Citations must utilize Word Citations to automatically show at the end of the document.

All updates after the initial creation must be performed using Tracking Changes turn on and Accepted by the Architecture committee.

# CUFX API and Documentation Support

CUFX is published to SwaggerHub at <https://app.swaggerhub.com/apis/dlacroix01/CUFX> . The latest default version will load automatically.

The purpose of this publication is to demonstrate the full range of CUFX messaging. Provide for complete documentation of the entire message structure and example usage.

Swaggerhub as a normal support feature also translates into several dozen of the most common and in demand client / server formats. This provides a technology specific version of the CUFX standard for essentially all platforms.

The CUFX Specification documents have been written to provide for limited examples of case usage but do not reflect the entirety of elements present in any given message. Please see the latest version of CUFX in Swaggerhub for the complete message and for superior documentation support.

# Release 4.0 Global Update Notes

CUFX Release 4.0 introduces a number modifications that significantly improves the standard and is not backward compatible with prior versions.

Messaging paradigm shift. Prior to CUFX 4.0 a Message Object would be sent and would expect the Object List to be returned or the error message. The response had to be interrogated to determine what was received. With CUFX 4.0, the Object Message that is sent is also expected to be the Object that is returned. Significant improvements have been made to the Message Context to fully support Success, Informational, Warnings and Error responses. End Points may continue to use the prior methods, but use of the Error.xsd is depreciated; all functionality has transitioned into MessageContext.xsd.

Date Range Filtering. A global update was applied across the standard to remove the pairs of date filter elements for any given range and replaced with a single Common.xsd definition DateRange complex type. This makes date range filtering completely uniform across the standard and associates the startDateTime and endDateTime together as an object set.

As example: elements transactionStartDateTime and transactionEndDateTime were replaced in the AccountFilter.xsd with transactionDateRange.

Microsoft Serialization Bug. We discovered the root cause of a serialization error impacting CUFX. A known Microsoft Serialization error from 2006 is present for single element complex types. It causes a naming error of the serialized constructs. If both endpoints are using a Microsoft compilation the error is consistent and does not present itself, the names are both wrong but pass data successfully. When one end point is not using a Microsoft compilation, the field names are in variance and fails. If both end points are using non-Microsoft compilation the serialization would be correct and match.

CUFX 4.0 has applied a global update across all list types throughout the standard. The CUFX list construct was consistently a single element complex type. For all occurrences we have applied an extension base of common:ListBase. ListBase provides pagination support and also resolves the Microsoft serialization error. No longer being a single element complex type, Microsoft compilation now generates the correct names. This will necessitate prior (Microsoft) implementations to remap to the correct serialized names.

# Release 4.4 Global Update Notes

CUFX Release 4.4 introduces a significant enhancement for complex Account identification and filter navigation. The foundational architectural design premise for account navigation is that the CUFX AccountId would be a unique value unto itself within a given institution, or that a composite unique key would be passed. With the direct support of several core system providers it was established that that later case is predominate. The AccountId is generally not a unique value unless in combination with several other values such as AccountType and AccountSubType. Hence passing a unique AccountId meant that the organization had to overload the element value making filtering implementation specific and forcing the endpoints to map overlay the accountId to unpack the value.

CUFX now fully recognizes this architectural paradigm while continuing to support the original architecture.

A global update was applied to provide the elements accountType and accountSubType in all objects that contained accountId.

A new filter list – AccountIdentificationList has been added to all account related filters so that AccountId, AccountType, and AccountSubType can be structured properly for discreet filtering of complex account keys, support filtering by the sub keys and also support inbound and outbound account filtering using the accountToFromIndicator.

By expanding the architectural paradigm to support a non-unique AccountId CUFX is now positioned to better support core adoption of the standard.

# Release 5.0 Global Update Notes

CUFX Release 5.0 Moved PartyId and PartyIdList into the Common.xsd. Namespaces references to Party were removed if there were no other references to Party in the specification. PartyId and PartyIdList type references were updated.

# Definitions related to the specification

**LOCATION**

A physical location that provides banking services to a member. A location can include, but is not limited to, the following:

* ATM owned by the financial institution
* Branch owned by the financial institution
* Shared ATM serving as part of a network (e.g. COOP) used by the financial institution.
* Shared branch serving as part of a shared branching network (e.g. COOP) used by the financial institution.

**NETWORK**

A network for shared ATM’s or for shared branches. Examples of these types of networks are COOP, NYCE, MoneyPass, and Star.

# DATA ELEMENTS

The important complex, container, and enumeration elements making up the CUFX Location service are described below. The full element specifications may be found in **Location.xsd, LocationMessage.xsd**, and **LocationFilter.xsd.** Common elements can be found in **Common.xsd**.

### DATA ELEMENT: LOCATIONMESSAGE

The LocationMessage object, for which the definition can be found in **LocationMessage.xsd,** is a wrapper object that contains a Location for transmission.

**Attributes**

* ***messageContext*** – The CUFX MessageContext for this object. The MessageContext object identifies the sender and provides some level of security or processing information for any given CUFX request.
* ***locationFilter*** – Any LocationFilter object being transmitted in this LocationMessage.
* ***locationList*** – Any LocationList object (*i.e.*, list of zero or more Location objects) being transmitted in this LocationMessage.

### DATA ELEMENT: LOCATION

The Location object, for which the definition can be found in **Location.xsd**, defines the physical location of a banking service (e.g., ATM or branch). Some of the fields are based on a common element and are used across more than one specification. These common elements can be found in **Contact.xsd** and **Common.xsd**.

**Attributes**

* ***locationId*** – The unique identifier of this Location object
* ***name*** – An optional name of the location (e.g., Briargate Branch)
* ***distance*** – The distance between the location and the address that was passed into the location filter used to search for these locations. If no location filter was used, this value would be zero.
* ***type*** - The type of location (e.g., ATM, Branch).
* ***address*** - The address of the ATM or branch. This may be just a latitude and longitude coordinate.
* ***phone*** - The phone number for the location.
* ***mapUrl*** - A map URL that links to a map displaying the location.
* ***network*** – The network of the ATM or Shared Branch (e.g., COOP, NYCE, MoneyPass, Star, etc.).
* ***depositTaking*** - Indicates whether the location accepts deposits.
* ***locationsHoursList*** - Contains the hours for the location.
* ***servicesList*** - The services or features of the location. For example, a branch could have a service of a coin counter, or an ATM could have a feature that it is check deposit taking ATM.
* ***additionalDataList*** - Additional data not taken into account in the other data elements.

### DATA ELEMENT: LOCATIONLIST

Contains a list of locations usually ordered with the closest location first.

### DATA ELEMENT: LOCATIONFILTER

The LocationFilter object, which the definition can be found in **LocationFilter.xsd**, is used to query for locations near an address (which could also just be GPS coordinates), for a given type (e.g., ATM).

**Attributes**

* ***locationId*** – The unique identifier for the location. The LocationId can be used when attempting to retrieve a single location.
* ***type*** – The type of location (e.g., ATM, Branch).
* ***searchFromAddress*** – The address to search for nearby locations.
* ***depositTaking*** – Indicates whether to include only deposit taking locations in the results.
* ***maxNumberOfResults*** - Maximum number of results to include in the response.
* ***maxDistance*** – Maximum distance (e.g., 25 miles) to search for locations from the searchFromAddress.

### DATA ELEMENT: MESSAGECONTEXT

Like many of the other CUFX service, a MessageContext object must be passed for each request. Refer to the Security Services documentation for more information.

# Location Services

## SERVICE DEFINITIONS

The location data model currently support read operations. Create, update, and delete operations for locations are currently beyond the scope of the CUFX locations services.

### SERVICE MESSAGE: GETLOCATIONS

When provided a locationFilter containing:

* filter elements such as the type of location that the user is trying to retrieve (e.g., ATM, SharedBranch),
* the address from which to search from (the address could be a street address, or the latitude and longitude coordinates obtained from a GPS),
* and the number of results to return,

the GetLocations service returns a list of locations.

|  |  |
| --- | --- |
| **INPUTS** | cufx:locationMessage (which includes)   * [cufx:messageContext](file:///\\files2\users\CMarjaniemi\Projects\CUFX\MessageContext.html) * [cufx:location](file:///\\files2\users\CMarjaniemi\Projects\CUFX\Generated%20HTML%20Docs\Loan.html)Filter (for read, update) * cufx:locationList (for create, update, delete) |
| **OUTPUTS** | cufx:locationMessage (which includes)   * [cufx:messageContext](file:///\\files2\users\CMarjaniemi\Projects\CUFX\MessageContext.html) * cufx:locationList |
| **RETURN VALUES** | cufx:locationMessage (which includes)   * [cufx:messageContext](file:///\\files2\users\CMarjaniemi\Projects\CUFX\MessageContext.html)   + statusList |
| **SIDE EFFECTS** | No data is being manipulated, only a lookup is being performed. |
| **DEPENDENCIES** | None. |
| **CUFX REST LINK** | https://api.dataprovider.com/locationmessage |

REST-JSON Example

The following example illustrates how to return the top 50 within 25 miles of a given postal code (e.g., “80920”).

**REQUEST:**

Headers:

**<security related header parameters... see Security Services>**

Accept: application/json

Accept-Charset: utf-8

Accept-Language: en-us (IANA – language codes)(W3C, HTTP Protocols)

Content-type: application/json; charset=utf-8

**X-HTTP-Method-Override: GET**

00-Version: >=5.0.0

**POST h**ttps://api.dataprovider.com/locationmessage

{

“locationMessage”:{

"messageContext" : {<SeeMessageContext.xsd>

},

"locationFilter": {

"searchFromAddress" : {

"address" : {

"postalCode" : [ "78704" ]

}

},

"depositTakingSpecified": true,

"maxNumberOfResults": 50,

"maxDistance": {

"unit": "mi",

"value": 25.0

}

}

}

**RESPONSE:**

Headers:

Status Code: 200 Ok

Content-type: application/json; charset=utf-8

Content-Language: en-us

Payload:

{

“locationMessage”:{

"messageContext" : {<SeeMessageContext.xsd>

},

"locationList": [

{

"locationId": "12345",

"name": "Briargate ATM",

"distance": {

"unit": "mi",

"value": 0.4

},

"type": "ATM",

"typeSpecified": true,

"address": {

"line1": "6140 Austin Bluffs Pkwy",

"city": "Colorado Springs",

"stateProvince": "CO",

"postalCode": "80920",

"coordinate": {

"latitude": 38.8234323,

"longitude": -104.2995991

}

},

"network": "COOP",

"depositTaking": true,

"depositTakingSpecified": true

},

{

"locationId": "23456",

"name": "Glendale Branch",

"distance": {

"unit": "mi",

"value": 1.2

},

"type": "Branch",

"typeSpecified": true,

"address": {

"line1": "123 Terrace Cove",

"city": "Colorado Springs",

"stateProvince": "CO",

"postalCode": "80920",

"coordinate": {

"latitude": 38.8987098,

"longitude": -104.7651091

}

},

"network": "COOP",

"depositTaking": true,

"depositTakingSpecified": true

}

]

}

The following example illustrates how to return the top 25 ATM locations within 25 miles of a given GPS location.

**REQUEST:**

Headers:

**<security related header parameters... see Security Services>**

Accept: application/json

Accept-Charset: utf-8

Accept-Language: en-us (IANA – language codes)(W3C, HTTP Protocols)

Content-type: application/json; charset=utf-8

**X-HTTP-Method-Override: GET**

X-API-Version: >=5.0.0

**POST h**ttps://api.dataprovider.com/locationmessage

{

“locationMessage”:{

"messageContext" : <SeeMessageContext.xsd>,

"type": "ATM",

"searchFromAddress": {

"coordinate": {

"latitude": 38.8256561,

"longitude": -104.6995991

}

},

"depositTaking": true,

"depositTakingSpecified": true,

"maxNumberOfResults": 50,

"maxDistance": {

"unit": "mi",

"value": 25.0

}

}

}

**RESPONSE:**

Headers:

Status Code: 200 Ok

Content-type: application/json; charset=utf-8

Content-Language: en-us

Payload:

{

“locationMessage”:{

"messageContext" : {<SeeMessageContext.xsd>

},

"locationList": {

"location": [

{

"locationId": "12345",

"name": "Briargate ATM",

"distance": {

"unit": "mi",

"value": 0.4

},

"type": "ATM",

"typeSpecified": true,

"address": {

"line1": "6140 Austin Bluffs Pkwy",

"city": "Colorado Springs",

"stateProvince": "CO",

"postalCode": "80920",

"coordinate": {

"latitude": 38.8234323,

"longitude": -104.2995991

}

},

"network": "COOP",

"depositTaking": true,

"depositTakingSpecified": true

}

]

}

}

# Bibliography

W3C. (n.d.). *Key words for use in RFCs to Indicate Requirement Levels [RFC2119].* Retrieved Sept. 8th, 2011, from W3C.