

Eligibility Requirement Data 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 |
| 0.0.01 |  | * Initial Release |
| 0.0.02 |  | * Updated Overview of Specification |
| 0.0.03 |  | * Updated XSD filenames to start PascalCase |
| 0.0.04 |  | * Update id lists to be correct to XSD |
| 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 eligibilityRequirementMessage 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 |
| 3.1 | **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 |
| 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

Defines the eligibility requirement Data object for use by all specifications. Eligibility requirement reference information is used by a financial institution to identify if a potential party is eligible to use their services. Eligibility might be related to a particular company, geographical region or other community. In addition to a base set of eligibility requirements, the model should support data input by the user. See the party data and services for eligibility data that can be stored for the party.

# Any know Errors in the document

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

# Table of Contents

[Document Status 1](#_Toc73693181)

[Change Log 1](#_Toc73693182)

[Overview of Specification 2](#_Toc73693183)

[Any know Errors in the document 2](#_Toc73693184)

[Table of Contents 2](#_Toc73693185)

[Document Conventions 2](#_Toc73693186)

[CUFX API and Documentation Support 3](#_Toc73693187)

[Release 4.0 Global Update Notes 3](#_Toc73693188)

[Release 4.4 Global Update Notes 3](#_Toc73693189)

[Release 5.0 Global Update Notes 4](#_Toc73693190)

[Definitions related to the specification 4](#_Toc73693191)

[Data Elements 4](#_Toc73693192)

[Variables used when accessing the eligibilityRequirement object (but not part of the object) 4](#_Toc73693193)

[EligibilityRequirement Object attributes 4](#_Toc73693194)

[Services 5](#_Toc73693195)

[eligibilityRequirement 5](#_Toc73693196)

[Overview 5](#_Toc73693197)

[Party Resource based create, read, update, delete services 5](#_Toc73693198)

[Service Message: Read eligibility Requirements 5](#_Toc73693199)

[REST-JSON READ REQUEST-RESPONSE (example: Know requirement Id’s) 5](#_Toc73693200)

[REST-JSON READ REQUEST-RESPONSE (example: Know Affinity Brands) 6](#_Toc73693201)

[General Error handling For All Services 8](#_Toc73693202)

[Bibliography 8](#_Toc73693203)

# 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

EligibilityRequirement Data

List of possible limits that enable a potential party make use of a financial institutions products and services.

Affinity Brand

A marketing presence that groups one or more communities together for the purposes of create a better relationship with those communities. Typically an affinity brand may include a combination of compaigns, logos, products, and services that may be specific to that community.

# Data Elements

## Variables used when accessing the eligibilityRequirement object (but not part of the object)

Refer to Security Services documentation to understand what may be contained the header and processed by security procedures. When accessing the data include **MessageContext.xsd** so that the service can determine the scope of the request. Include any filter variables related to the request. See **EligibilityRequirementFilter.xsd.** .

The filters are used to filter based on associated data type for the eligibility Requirement such as eligibility Requirement Id and affinity brand. etc.. Combining filters can be used to get list of eligibility requirements for a affinity brand, etc.

## EligibilityRequirement Object attributes

All CUFX fields related to a preference are defined in **EligibilityRequirement.xsd**.

Note: Fields not listed in the calling specification are not to be returned to the calling specification. i.e. If the field affinity brand is not listed in the calling specification, then do not return the data field to alleviate issues with unexpected information and bloat of information being returned to light weight applications.

# Services

## eligibilityRequirement

To READ high level eligibilityRequirement data, clients access the eligibilityRequirement Data services..

* Specifications depend on the Security service for single-sign-on and/or authentication.

## Overview

|  |  |
| --- | --- |
| Definition | Collection of services to read eligibilityRequirement reference data. |
| Overview of Capabilities | Read an eligibility requirement. |
| Dependencies | Security Services, messageContext, party |
| Sample CUFX REST LINK | https://api.dataprovider.com/eligibilityrequirementmessage |
| CUFX SOAP LINK |  |
| CUFX WaDL LINK |  |

## Party Resource based create, read, update, delete services

|  |  |
| --- | --- |
| INPUTS | cufx:eligibilityRequirementMessage (which includes)   * [cufx:MessageContext](file:///\\files2\users\CMarjaniemi\Projects\CUFX\MessageContext.html) * cufx:EligibilityRequirementFilter (for read, update) * cufx:EligibilityRequirementList (for create, update, delete) |
| Outputs | cufx:eligibilityRequirementMessage (which includes)   * [cufx:MessageContext](file:///\\files2\users\CMarjaniemi\Projects\CUFX\MessageContext.html) * cufx:EligibilityRequirementList) |
| Return Values | cufx:eligibilityRequirementMessage (which includes)   * cufx:MessageContext   + statusList |
| Side Effects | Read has no side effects |
| Dependencies | Security Services for authentication and security. |
| Fields used | Message Headers : See security services  messageContext: See MessageContext.xsd  Filters: See EligibilityRequirementFilter.xsd  Attributes: eligibilityRequirement**:** See EligibilityRequirement**.**xsd |
| Testing Procedures for Certification | Access the CUFX eligibility requirement data test service and verify that test data is returned appropriately. |

### Service Message: Read eligibility Requirements

### REST-JSON READ REQUEST-RESPONSE (example: Know requirement Id’s)

Example: Read a list of requirements based on known eligibility requirement Id’s. Note that if the user selected one a requirement with a nextActionRequiredId, that that additional information that was gathered might be stored by the calling application in the party data eligibility requirement met reference description field.

Clients submit an eligibility requirement data request to the Data Provider’s URL. For all CUFX RESTful-JSON requests, the client must set the appropriate headers. This example may not include all fields and is for example only.

REQUEST:

Headers:

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

Accept: application/json

Accept-Charset: utf-8

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

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

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

X-API-Version: >=5.0.0

**POST** https://api.dataprovider.com/eligilityrequirementmessage

{

“eligibilityRequirementMessage”:{

“messageContext”: { <see MessageContext.xsd>

},

“eligibilityRequirementFilter”: {

“eligibilityRequirementIdList”:[

”151321”,

”151355”

]

}

}

}

The Data Provider responds to requests with a JSON-formatted message:

**RESPONSE:**

Headers:

Status Code: 200 Ok

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

Content-Language: en-us

Payload:

{

“eligibilityRequirementMessage”:{

“messageContext”: { <see MessageContext.xsd>

},

“eligibilityRequirementList”: [

{

“requirementId”: “151321”,

“description”: “Primary Residence in Wayne County”,

“nextActionRequiredId”: “10” *<example: ask for years at residence>*

},

{

“requirementId”: “151355”,

“description”: “Primary Residence in Logan County”

}

]

}

}

### REST-JSON READ REQUEST-RESPONSE (example: Know Affinity Brands)

Example: Read a list of requirements based on known affinity brands.

Clients submit an eligibility requirement data request to the Data Provider’s URL. For all CUFX RESTful-JSON requests, the client must set the appropriate headers. This example may not include all fields and is for example only.

REQUEST:

Headers:

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

Accept: application/json

Accept-Charset: utf-8

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

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

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

X-API-Version: >=5.0.0

**POST** https://api.dataprovider.com/eligilityrequirementmessage

{

“eligibilityRequirementMessage”:{

“messageContext”: { <see MessageContext.xsd>

},

“eligibilityRequirementFilter”: {

“affinityBrandList”:[

“Wonderful Credit Union”,

“Wonderful Financial Services” ]

}

}

}

The Data Provider responds to requests with a JSON-formatted message:

**RESPONSE:**

Headers:

Status Code: 200 Ok

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

Content-Language: en-us

Payload:

{

“eligibilityRequirementMessage”:{

“messageContext”: { <see MessageContext.xsd>

},

“eligibilityRequirementList”: [

{

“requirementId”: “5187411”,

“description”: “Residence in Twin Cities”,

“nextActionRequiredId”: “20” *<example: ask for residence zip code>*

“affinityBrandList”: [

”Wonderful Credit Union”,

”Wonderful Financial Services”

]

},

{

“requirementId”: “484121”,

“description”: “Employee at Big City Corp”,

“affinityBrandList”: [

”Wonderful Financial Services”

]

},

{

“requirementId”: “8798421”,

“description”: “Spouse of Employee at Big City Corp”,

“nextActionRequiredId”: “12”, *<example: ask for employee name or SSN>*

“affinityBrandList”: [

”Wonderful Financial Services”

]

}

]

}

}

# General Error handling For All Services

Refer to latest CUFX documentation *Error Mapping*.

# Bibliography

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