

MessageContext Object 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 Creation |
| **0.0.02** |  | * Clarified that JSON assumes complex types within a list. Thus why user isn’t listed in example code. |
| **0.0.03** |  | * Updated to use data definitions from messageContext.xsd. * Updated UserType to Pascal case in examples |
| **0.0.04** |  | * Updated Overview of Specification |
| **0.0.05** |  | * Added use of customData example. |
| **0.0.06** |  | * Adding field to identify to return all, only updated fields or no data in the create and update |
| **3.0** | **12/16/2013** | * Versioning and format change with release CUFX 3.0 |
| **3.1** | **07/17/2015** | * Updated to release 3.1 |
| **3.2** | **05/10/2016** | * Updated to release 3.2 |
| **3.3** | **02/17/2017** | * Updated to release 3.3 |
| **4.0** | **02/19/2018** | * Updated to release 4.0, Microsoft Global bug fix, updated cufxVersion to use common:CufxVersion. Added complex type StatusList, Status. Status contains statusType, statusCode, statusSubCode, statusMessage and substitutionList to provide full capability message support. Use of error.xsd is depreciated. |
| **4.1** | **12/10/2018** | * Updated to release 4.1, Global update applied for minOccurs 0 maxOccurs 1. Xsd had 1 or more elements corrected to be consistent with the standard definition. * Added configurationid to messageContext, * Added sendingNetworkNodeId and receivingNetworkNodeId * Moved Id definitions to common added reference. Add reference to xsd User, App, Moved type Environment to common, * Added sessionId, Moved User, UserList, User Type definition to User.xsd, StatusCode minOcurrs set to zero * Moved networknodeid to common, applied reference by to networknode, and messagecontext. |
| **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. Added referenceId and referenceIndex to correctly associate the status messaging response to the original input record. Corrected documentation spelling errors. |
| **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 CUFX Message Context Data object specification describes the object that can be included in all data requests to aid in the applying any additional discovery of who/what made the request. This is not provided for security purposes. See Security Services for more detail on how this fits into security processes.

# Known Errors in the document

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

Table of Contents

[Document Status 1](#_Toc73693880)

[Change Log 1](#_Toc73693881)

[Overview of Specification 2](#_Toc73693882)

[Known Errors in the document 2](#_Toc73693883)

[Document Conventions 3](#_Toc73693884)

[CUFX API and Documentation Support 3](#_Toc73693885)

[Release 4.0 Global Update Notes 3](#_Toc73693886)

[Release 4.4 Global Update Notes 4](#_Toc73693887)

[Release 5.0 Global Update Notes 4](#_Toc73693888)

[Definitions related to the specification 5](#_Toc73693889)

[REST-JSON Examples: 5](#_Toc73693890)

[Bibliography 5](#_Toc73693891)

# Document Conventions

“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 with Track Changes **On** and Accepted by the Architecture Committee.

Any Quotes in code examples below should typically be coded as ASCII character decimal 034 or 022 hex.

# 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

See each use object in the CUFX spec on how to include in the individual messages. See Security Services for overall conceptual use. See messageContext.xsd and messageContext.html for definitions of data elements.

## REST-JSON Examples:

In CUFX release 4.0 MessageContext is part of every object message container for all messages sent and received. Please see the specific specification document for detail examples and usage of MessageContext.

# Bibliography

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