

Fee Data 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 |
| 0.0.3 |  | * Initial Creation and minor revisions |
| 3.0 | **10/29/2013** | * Versioning and format change with release CUFX 3.0 * Renamed FeeFilter to FeeScheduleFilter |
| 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, Removed and normalized common:IntervalFrequencyType into common:FrequencyType |
| 4.1 | **12/10/2018** | * Updated to release 4.1, Global removal of choice constructs to eliminate cross language serialization issues. * FeeSchedule - Added feeEffectiveDateRange * FeeSchedule, FeeScheduleFilter - Added feeIdList, feeTypeList, feeEffectiveDateRange. |
| 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, FeeSchedule - Added customData to Fee * Added accountType and accountSubType to RelatedTo * FeeScheduleFilter - Added accountIdentificationList to filter. |
| 4.5 | **04/02/2021** | * Updated to release 4.5 |
| 5.0 | **06/07/2021** | * Updated to release 5.0, FeeSchedule - Update PartyId to reference Common.xsd. removed deprecation element type. Removed deprecation type ExtendedFeeType. * FeeSchedule - Update PartyId to reference Common.xsd. removed deprecation element taxIdList. |

Overview of Specification

The Fee specification describes the services used as part of a new membership application process, for creating a loan, funding the loan, and disbursing the funds.

Any know Errors in the document

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

Table of Contents

Document Status 1

Change Log 1

Overview of Specification 2

Any know Errors in the document 2

Table of Contents 2

Document Conventions 3

CUFX API and Documentation Support 3

Release 4.0 Global Update Notes 4

Release 4.4 Global Update Notes 4

Release 5.0 Global Update Notes 4

Definitions related to the specification 5

High level use cases 6

Use Case 1: RETURN a fee schedule for a member 6

Data Elements 7

FEE SChEDULE Data attributes 7

Data Element: FeeSchedule 7

Data Element: FeeScheduleFilter 7

Fee Schedule filter 7

Data Element: FeeScheduleFilter 7

Data Element: partyIdlist 7

Data Element: relationshipIdlist 7

Data Element: contactIdlist 7

Data Element: accountIdlist 8

Data Element: partyTypelist 8

fee schedule 8

Data Element: feeList 8

Data Element: FeeList 8

Data Element: fee 8

Data Element: feeId 8

Data Element: ExtendedFeeType 8

Data Element: relatedTo 9

Data Element: ExtendedFeeType 9

Data Element: frequency 9

Data Element: minimumFrequency 9

Data Element: maximumFrequency 9

Data Element: priceList 9

Data Element: description 9

Data Element: price 9

Data Element: CustomFeeType 10

Data Element: FeeType 10

Data Element: feeCategory 10

Data Element: Unpaid 10

Data Element: Overdraft 10

Data Element: FeeId 11

Data Element: RelatedTo 11

Data Element: partyID 11

Data Element: relationshipID 11

Data Element: accountID 11

Services 11

Overview 11

FeeSchedule Resource Based Create, Read, Update, Delete Services 12

REST-JSON Read Fee Schedule of a Specific Account ID, relationship and Party 12

Bibliography 14

# 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

System of record

Authoritative data source for information such as the loans for which this specification will be creating.

FEE

Examples of types of fees are overdraft (NSF), Stop, Bill Payment, Over Limit, Payment Skip, and so on. Extended fees are also possible to enable institutions to describe fees unique to their institution and products.

FEE SCHEDULE

All the possible fees for the user’s accounts and products

FEE Filter

A list of options to filter fees in a query. The options can be used in combination or by themselves and it is up to the data source to either return or affect the data requested or return a standard error identifying why the request was rejected.

# High level use cases

## Use Case 1: RETURN a fee schedule for a member

This use case assumes that the calling application has one or more accounts with the institution.

CUFX Consumer (e.g. mobile, PFM)

1

Request Fee Schedule for Member

2

Respond with Fee Schedule

CUFX Core Data Provider

# Data Elements

## FEE SChEDULE Data attributes

The following CUFX fields referenced in the services defined below are utilized for fees:

### Data Element: FeeSchedule

The FeeSchedule object, which the definition can be found in **FeeSchedule.xsd**, contains all the fields necessary to communicate the relevant fees for a user and their related products. Institutions can communicate unique fees using CustomFeeType.

### Data Element: FeeScheduleFilter

The FeeFilter object, which the definition can be found **FeeScheduleFilter.xsd**, contains all the fields necessary for querying fees related to a user’s accounts and products.

# Fee Schedule filter

### Data Element: FeeScheduleFilter

The list of options for filters for Fees. The options can be used in combination or by themselves and it is up to the data source to either return or affect the data requested or return a standard error identifying why the request was rejected.

### Data Element: partyIdlist

The list of partyId's that should be returned or affected by a read, update or delete party request.

### Data Element: relationshipIdlist

The list of relationshipId's that should have their associated parties either returned or affected by a read, update or delete party request.

### Data Element: contactIdlist

The list of contact Id's that should have their associated parties either returned or affected by a read, update or delete party request.

### Data Element: accountIdlist

The list of Account Id's that should have their associated parties either returned or affected by a read, update or delete party request.

### Data Element: partyTypelist

The list of Party Types that should have their associated parties either returned or affected by a read, update or delete party request.

# fee schedule

### Data Element: feeList

The accounts collection/array contains all the fees for instance that the credential set is authorized to access at the Institution based on applied filters.

### Data Element: FeeList

The accounts collection/array contains all the fees for instance that the credential set is authorized to access at the Institution based on applied filters.

### Data Element: fee

### Data Element: feeId

The ID for the fee.

### Data Element: ExtendedFeeType

The fee type is the textual representation the kind of fee.

### Data Element: relatedTo

Party, relationship, or account related to this Fee.

### Data Element: ExtendedFeeType

Allows a fee type unique to the institution or product using the "x:" prefix. CUFX-compatible client may ignore unrecognized CustomFeeTypes.

### Data Element: frequency

A value that indicates how this price is applied. For instance, this price may per stop payment, for a stop payment action, which would equate to PerUse. This value may be a recurring monthly charge, regardless of usage, which would be the Monthly frequency.

### Data Element: minimumFrequency

A value that indicates to what frequency this price applies. This element shall only exist for the PerUse frequency. It can be used to denote tiered pricing. For instance, one party-action coupling may return four Price elements, one for when the function is used between 1 and 10 times, between 11-99 times, 100-999 times, and 1000 times to 10000000 times.

### Data Element: maximumFrequency

The upper bound of a tiered pricing amount.

### Data Element: priceList

### Data Element: description

A description for the tier of pricing.

### Data Element: price

The fee for the given tier.

### Data Element: CustomFeeType

A fee unique to the institution or product. A string value that must contain a "custom:" prefix.

### Data Element: FeeType

### Data Element: feeCategory

The feeCategory shows the family or category of a fee.

### Data Element: Unpaid

Unpaid Item Fee Due to Draft or Due to other: for NSF Fee

### Data Element: Overdraft

An overdraft is an extension of credit from a lending institution when an account balance reaches zero or an act of overdrawing a bank account. Overdraft fees are directly related to overdraft items due to draft and related transactions. These may include courtesy pay fees, overdraft item fee force paid, transfer courtesy fees, below minimum withdrawal fees, below minimum advance fees, and overdraft transfer fees.

<xs:enumeration value="Withdrawal">

Withdrawal Fee: for W/D Fee

<xs:enumeration value="Advance">

Advance Fee: for Adv Fee

<xs:enumeration value="Deposit">

Deposit Fee.

<xs:enumeration value="Payment">

Payment Fee.

<xs:enumeration value="StopFee">

Stop Fee.

<xs:enumeration value="Overlimit">

Overlimit Fee Due to Draft: for OVL Fee; not covered by Reg DD

<xs:enumeration value="BillPayment">

Bill Payment Fee: not covered by Reg DD

<xs:enumeration value="Transfer">

Transfer Fee: for XFR Fee; not covered by Reg DD

<xs:enumeration value="WithdrawaltoBelowBase">

Withdrawal to Below Base Fee: not covered by Reg DD

<xs:enumeration value="ClosingWithinBaseCountDays">

Closing Within Base Count Days Fee: not covered by Reg DD

<xs:enumeration value="WithdrawalWhenBalanceBelowBase">

Withdrawal When Balance Below Base Fee: not covered by Reg DD

<xs:enumeration value="TellerUsage">

Fees related to interaction with tellers. This could include a lobby visit fee.

<xs:enumeration value="CheckPrinting">

Check Printing Fee: for Check Fees; not covered by Reg DD

<xs:enumeration value="CheckOther">

Check Other Fee: for Check Fees; not covered by Reg DD

<xs:enumeration value="NewShare">

New Share Fee: not covered by Reg DD

<xs:enumeration value="DocumentStamp">

Document Stamp Fee: applies to Loan Tax (LT) teller transactions; not covered by Reg DD

<xs:enumeration value="ATMCustom">

ATM Custom Fee: for Custom ATM Fee Edits; not covered by Reg DD

<xs:enumeration value="AudioCustom">

Audio Custom Fee: for Custom Audio Fee Edits; not covered by Reg DD

<xs:enumeration value="OnlineServices">

Fees related to the delivery of online services.

An example would be charging member for custom data feed to desktop personal financial management products.

<xs:enumeration value="InternationalTransaction">

International Transaction Fee: for Custom Card Fee Edits for International

Transaction Fees; not covered by Reg DD

<xs:enumeration value="PaymentSkip">

Payment Skip Fee: only for open end loans and lines of credit; covered by Regulation Z

<xs:enumeration value="OtherRegulationZ">

Other Regulation Z Fee: only for open end loans and lines of credit

<xs:element name="IsRegDApplicable" type="xs:boolean" minOccurs="0" maxOccurs="1">

This element indicates if the fee is RegulationD applicable.

Default is false. If element is missing; false is assumed.

### Data Element: FeeId

The Fee ID is a persistent and unique identifier for the Fee. It must be unique for all users across the institution and remain the same over the life of the fee.

### Data Element: RelatedTo

Party, relationship, or account related to the Fee.

### Data Element: partyID

Id of party related to the Fee, such as a person or a business.

### Data Element: relationshipID

Id of relationship related to the Fee

### Data Element: accountID

Id of account related to the Fee

# Services

## Overview

|  |  |
| --- | --- |
| Definition | Collection of services to manage an FeeSchedule |
| Overview of Capabilities | Create, read, update and delete a FeeSchedule. The following scenarios may exist. The FeeSchedule may be connected to an existing relationship or parties |
| Dependencies | Party, relationship data, account data and security services |
| Sample CUFX REST LINK | https://api.dataprovider.com/feeschedulemessage |
| CUFX WaDL LINK |  |

## FeeSchedule Resource Based Create, Read, Update, Delete Services

|  |  |
| --- | --- |
| INPUTS | cufx:feeScheduleMessage (which includes)   * cufx:MessageContext * cufx: feeScheduleFilter (for read, update) * cufx: feeScheduleList (for create, update, delete) |
| Outputs | cufx:a feeScheduleMessage (which includes)   * cufx:MessageContext * cufx: feeScheduleList |
| Return Values | cufx: feeScheduleMessage (which includes)   * cufx:MessageContext   + statusList |
| Side Effects | Creation, update or deletion of application. 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 feeScheduleFilter.xsd  **Attributes:** feeScheduleList : See FeeSchedule.xsd  **Errors:** See MessageContext.xsd |

### REST-JSON Read Fee Schedule of a Specific Account ID, relationship and Party

Note: Not all fields are listed for simplicity of an example to create an application.

**Required**: messageContext, **at least one filter in feeScheduleFilter.**

**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 https://api.datasource.com/feeschedulemessage

("feeScheduleMessage": {

"@xmlns:xsi": "http://www.w3.org/2001/XMLSchema-instance",

"messageContext": {

"cufxVersion": "5.0.0",

"requestId": "requestId1",

"vendorId": "vendorId1",

"appId": "appId1",

"fiId": "fiId1",

"dataSourceId": "dataSourceId1",

"environment": "Development",

"returnDataFilter": "All",

"includeBlankFields": "true",

"includeZeroNumerics": "true",

"user": {

"userId": "userId1",

"processorSessionId": "processorSessionId1",

"userType": "EmployeeId"

}

},

"feeScheduleFilter": {

"partyIdList": {

"partyId": "partyId1"

},

"relationshipIdList": {

"relationshipId": "relationshipId1"

},

"contactIdList": {

"contactId": "contactId1"

},

"accountIdList": {

"accountId": "accountId1"

}

}

}

**RESPONSE**:

**Headers:**

Status Code: 200 Ok

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

Content-Language: en-us

Payload:

{“feeScheduleMessage": {

"@xmlns:xsi": "http://www.w3.org/2001/XMLSchema-instance",

"messageContext": {

"cufxVersion": "5.0.0",

"requestId": "requestId1",

"vendorId": "vendorId1",

"appId": "appId1",

"fiId": "fiId1",

"dataSourceId": "dataSourceId1",

"environment": "Development",

"returnDataFilter": "All",

"includeBlankFields": "true",

"includeZeroNumerics": "true",

"user": {

"userId": "userId1",

"processorSessionId": "processorSessionId1",

"userType": "EmployeeId"

},

"statusList": {

"status": {

"statusType": "Success"

}

}

},

"feeList": {

"currentPage": "1",

"totalPages": "1",

"pageSize": "1",

"positionIndex": "positionIndex1",

"returnPage": "1",

"returnPositionIndex": "returnPositionIndex1",

"fee": {

"feeId": "feeId1",

"relatedTo": {

"partyId": "partyId1"

},

"frequency": "AmortizedSchedule",

"minimumFrequency": "1",

"maximumFrequency": "1",

"priceList": {

"description": "description1",

"price": {

"value": "1",

"currencyCode": "USD",

"exchangeRate": "1"

}

}

}

}

}

}

# Bibliography

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