

Product Service Request Model and Services 4.4

# Document Status

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

# Authors and Change Log

|  |  |  |
| --- | --- | --- |
| Version | Date | Changes |
| 0.0.01 |  | * Initial Creation |
| 0.0.02 |  | * Removed required Yes/No field in data dictionary and moved to service * Fix type of Fulfillment Note |
| 0.0.03 |  | * Cleaned up formatting * Added leads and lead collection * Added filters definitions for the service * Added party details * Reformatted services for above changes |
| 0.0.4 |  | * Remove definitions and moved to lead.xsd * Remove definitions for message context fields and include reference to messageContext.xsd * Add leadFilter.xsd for filtering * Remove hyperlink in service definition |
| 0.0.5 |  | * Replace error definitions with references to error documentation and Error.xsd * Add delete and update examples. * Fix messageContext brackets. |
| 0.0.6 |  | * Change plural to List form |
| 0.0.7 |  | * Add contact and party id’s in the standalone response for create. |
| 0.0.8 |  | * Updated Overview of Specification |
| 0.0.9 |  | * Changed XSD filenames to PascalCase |
| 0.1.0 |  | * Changed from lead to Product Service Request * Add ability to relate to an account * Add ability to have a question on service * Updated structures to new productServiceRequest tiered structure |
| 0.1.1 |  | * Clarify specification overview |
| 0.1.2 |  | * Released for comments |
| 0.1.3 |  | * Included secureMessageId as list, updated some annotations to be more descriptive. Added verbiage about use of product service request. |
| 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 productServiceRequestMessage wrapper for every message to increase ability for infrastructure to serialize the data * Versioning and format with release CUFX 3.0 |
| 3.0 | **12/16/2013** | * 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, Split out activity objects from product service request creating new activity.xsd |
| 4.1 | **12/10/2018** | * Updated to release 4.1, * Global removal of choice constructs to eliminate cross language serialization issues. * Global update applied for minOccurs 0 maxOccurs 1. Xsd had 1 or more elements corrected to be consistent with the standard definition. * ProductServiceRequest - Updated description. |
| 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, ProductServiceRequestFilter - Added accountIdentificationList to filter |

# Overview of Specification

This CUFX Product Service Request Data and Services specification enables an application to submit details about products and services that the party is interested in that go beyond a loan or deposit product. Product or Service requests are pre- stage information requests to help internal staff identify additional products and services that a party has requested. A product request might be an Investment account that the party is interested in but can’t be supported through the Deposit services. A service request example is requesting a box of checks, or stopping payment on a check, notification that a card has been lost, etc. This entity can be standalone (in the case of a CRM like system) or related to party, account, contact and relationship data previously created on an account. See capabilities for the service for more details. In addition, the status of the events and tasks related to fulfillment of the request(s) can be accessed via this service.

# Any know Errors in the document

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

Table of Contents

[Document Status 1](#_Toc54094256)

[Authors and Change Log 1](#_Toc54094257)

[Overview of Specification 2](#_Toc54094258)

[Any know Errors in the document 2](#_Toc54094259)

[Document Conventions 3](#_Toc54094260)

[CUFX API and Documentation Support 3](#_Toc54094261)

[Release 4.0 Global Update Notes 4](#_Toc54094262)

[Release 4.4 Global Update Notes 4](#_Toc54094263)

[Definitions related to the specification 4](#_Toc54094264)

[Data Elements 5](#_Toc54094265)

[Filters used when accessing the product service request data 5](#_Toc54094266)

[ProductServiceRequest Data attributes 5](#_Toc54094267)

[ProductServiceRequest Services 5](#_Toc54094268)

[Overview 5](#_Toc54094269)

[ProductServiceRequest Resource based create, read, update, delete services 6](#_Toc54094270)

[REST-JSON CREATE ProductServiceRequest 6](#_Toc54094271)

[REST-JSON READ ProductServiceRequest Example 8](#_Toc54094272)

[REST-JSON UPDATE ProductServiceRequest Example 9](#_Toc54094273)

[REST-JSON DELETE ProductServiceRequest Example 10](#_Toc54094274)

[REST-JSON CREATE ProductServiceRequest (crm style submission) 11](#_Toc54094275)

[General Error handling For All Services 13](#_Toc54094276)

[Bibliography 13](#_Toc54094277)

# 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.

# Definitions related to the specification

PRODUCT SERVICE REQUEST

An existing or potential party is interested in a product, products, service or services or just has a question on a product or service. The request can have several statuses as to the progress of the request since it may take several steps to complete the request(s)

# Data Elements

## Filters used when accessing the product service request data

Refer to Security Services documentation to understand what may be contained in 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. Refer to recent CUFX messageContext Data and CUFX Security Services for use of MessageContext.xsd. Include any filter variables related to the request. See **ProductServiceRequestFilter.xsd**

## ProductServiceRequest Data attributes

Refer to repository **ProductServiceRequest.xsd** for attribute definitions of the productServiceRequest object.

# ProductServiceRequest Services

## Overview

|  |  |
| --- | --- |
| Definition | Collection of services to manage a productServiceRequest |
| Overview of Capabilities | Create, read, update and delete a productServiceRequest. The following scenarios may exist. The productServiceRequest may be connected to an existing party, relationship and contact. In addition, a productServiceRequest may exist without an existing party, relationship or contact. The productServiceRequest if created without connection to existing party, account, contact or relationship could be converted to be connected to an existing party, account, contact and relationship or aid in creating party, account, contact or relationship within the endpoint system. If created without connection to existing party, etc. then it might be sent to a CRM system for follow up with the contact. Thus the reason for the two different examples. In addition, the status of the events related to fulfillment of the request(s) can be accessed via this service.  The Product Service Request service is of a generic nature. This service can be used as a catch all for processes that are not explicitly outlined with other services. This service is engineered to support a wide variety of generic business processes. An example of a business process that does not have a dedicated service, but could be supported by the product service request is disputing a charge. You could use the Product Service Request service to support disputing an invalid debit charge, stopping a check, or stopping a range of checks. |
| Dependencies | Party , contact, and relationship data and services |
| Sample CUFX REST LINK | (connected version: related to existing relationship)  https://api.dataprovider.com/productservicerequestmessage  or  (stand-alone version: CRM type of a system)  https://api.**crm**dataprovider.com/productservicerequestmessage |
| CUFX SOAP LINK |  |
| CUFX WaDL LINK |  |

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

|  |  |
| --- | --- |
| INPUTS | cufx:productServiceRequestMessage (which includes)   * [cufx:MessageContext](file:///\\files2\users\CMarjaniemi\Projects\CUFX\MessageContext.html) * cufx:productServiceRequestFilter (for read, update, delete) * cufx:productServiceRequest (for create, update) |
| Outputs | cufx:productServiceRequestMessage (which includes)   * [cufx:MessageContext](file:///\\files2\users\CMarjaniemi\Projects\CUFX\MessageContext.html) * cufx:productServiceRequest |
| Return Values | cufx:productServiceRequestMessage (which includes)   * [cufx:MessageContext](file:///\\files2\users\CMarjaniemi\Projects\CUFX\MessageContext.html)   + statusList |
| Side Effects | Creation of productServiceRequest, update of existing productServiceRequest, or deletion of productServiceRequest. Read has no side effects. |
| Dependencies | Security Services for authentication and security. |
| Fields used | Message Headers : See security services  messageContext: See MessageContext.xsd  Filters: productServiceRequestFilter.xsd  Attributes: productServiceRequestList: See ProductServiceRequestList.xsd |

### REST-JSON CREATE ProductServiceRequest

This example shows where the relationship is known and the contact is already known.

**Required**: messageContext, at least one productOfInterest, source

**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-API-Version: >=4.3.0

**POST h**ttps://api.datasource.com/productservicerequestmessage

{

“productServiceRequestMessage”:{

“messageContext”: { <see MessageContext.xsd>

},

“productServiceRequestList”: [

{

“requestList”:

[

{“requestItem”:[{“productOfInterest”:”Mortgage”}],

“requestStatus”:”New”,

“requestedDateTime”:”2013-03-28T11:31Z”,

“dueDateTime”:”2013-03-29T11:31Z”,

“requestNoteList”:[

{“noteText”:”Need to talk ASAP!”

“noteCreator”:”John Bell”

}

],

“requestRelatedTo”:{

{“contactIdList”: [”164879984”]

},

“activityList”:[

{“activityName”:”Create Mortgage App”,

“activityStatus”:”New”,

“creator”:”SysId3534”,

“requestedDateTime”:”2013-03-28T11:31Z”,

“dueDateTime”:”2013-03-29T11:31Z”,

“activityNoteList”:[

“noteText”:”High Priority Request”

“noteCreator”:”SysId3534”

]

}

]

}

],

“secureMessageList”:[

{“secureMessageId”:”2346532664”}],

“documentList”:{[

{“documentId”:”234454756345”}],

},

“source”:”Website”,

“status”:”New”,

“createDateTime”:”2013-03-28T1:31Z”,

“productServiceRequestRelatedTo”:{

“contactIdList”: [ ”12345” ],

“relationshipIdList”: [ ”105949495” ],

}

“productServiceRequestNoteList”:[

{“noteText”:”I heard you are awesome!”

“noteCreator”:”John Bell”

}

]

}

]

}

}

**RESPONSE:**

Headers:

Status Code: 200 Ok

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

Content-Language: en-us

Payload:

{

“productServiceRequestMessage”:{

“messageContext”: { <see MessageContext.xsd>

},

“productServiceRequestList”: [

{

“productServiceRequestId”:”1514asd4151230a”,

“requestList”:

[

{“requestId”:”159609633”,

“requestItem”:[{“productOfInterest”:”Mortgage”}],

“requestStatus”:”New”,

“requestedDateTime”:”2013-03-28T11:31Z”,

“dueDateTime”:”2013-03-29T11:31Z”,

“requestNoteList”:[

{“noteText”:”Need to talk ASAP!”

“noteCreator”:”John Bell”

}

],

“requestRelatedTo”:{ “contactIdList”: [ ”164879984” ] },

},

“secureMessageList”:[

{“secureMessageId”:”2346532664”}],

“documentList”:{[

{“documentId”:”234454756345”}],

},

],

“source”:”Website”,

“status”:”New”,

“createDateTime”:”2013-03-28T1:31Z”,

“productServiceRequestRelatedTo”:{

“contactIdList”: [ ”12345” ],

“relationshipIdList”: [ ”105949495” ],

}

“productServiceRequestNoteList”:[

{“noteText”:”I heard you are awesome!”

“noteCreator”:”John Bell”

}

]

}

]

}

### REST-JSON READ ProductServiceRequest Example

**Required**: messageContext and at least one filter criteria in the productServiceRequestFilter object.

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: >=4.3.0

**POST h**ttps://api.datasource.com/productservicerequestmessage

{

“productServiceRequestMessage”:{

“messageContext”: { <see MessageContext.xsd>

},

“productServiceRequestFilter”: {

“relationshipIdList”:[ ”105949495” ]

}

}

}

**RESPONSE:**

Headers:

Status Code: 200 Ok

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

Content-Language: en-us

Payload:

{

“productServiceRequestMessage”:{

“messageContext”: { <see MessageContext.xsd>

},

“productServiceRequestList”: [

{

“productServiceRequestId”:”1514asd4151230a”,

“requestList”:

[

{“requestId”:”159609633”,

“requestItem”:[{“productOfInterest”:”Mortgage”}],

“requestStatus”:”New”,

“requestedDateTime”:”2013-03-28T11:31Z”,

“dueDateTime”:”2013-03-29T11:31Z”,

“requestNoteList”:[

{“noteText”:”Need to talk ASAP!”

“noteCreator”:”John Bell”

}

],

“requestRelatedTo”:{ [“contactIdList”: ”164879984” ] },

},

“secureMessageList”:[

{“secureMessageId”:”2346532664”}],

“documentList”:{[

{“documentId”:”234454756345”}],

]

}

],

“source”:”Website”,

“status”:”New”,

“createDateTime”:”2013-03-28T1:31Z”,

“productServiceRequestRelatedTo”:{

“contactIdList”: [ ”12345” ],

“relationshipIdList”: [ ”105949495” ],

}

“productServiceRequestNoteList”:[

{“noteText”:”I heard you are awesome!”

“noteCreator”:”John Bell”

}

]

}

]

}

### REST-JSON UPDATE ProductServiceRequest Example

**Required**: messageContext and at least one productServiceRequestId in the productServiceRequestFilter

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-API-Version: >=4.3.0

**PUT h**ttps://api.datasource.com/productservicerequestmessage

{

“productServiceRequestMessage”:{

“messageContext”: { <see MessageContext.xsd>

},

“productServiceRequestFilter”: {

“productServiceRequestIdList”:[ ”1514asd4151230a” ]

},

“productServiceRequestList”: [

{ “**productServiceRequestId**”:”1514asd4151230a”

“status”:”InDiscussion”

“productServiceRequestNoteList”:[

{“noteText”:”Interested in a $100,000 Mortgage!”

“noteCreator”:”UserId353434”

}

]

}

]

}

}

**RESPONSE:**

Headers:

Status Code: 200 Ok

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

Content-Language: en-us

Payload:

{

“productServiceRequestMessage”:{

“messageContext”: { <see MessageContext.xsd>

},

“productServiceRequestList”: [

{

“productServiceRequestId”:”1514asd4151230a”,

“requestList”:

[

{“requestId”:”159609633”,

“requestItem”:[{“productOfInterest”:”Mortgage”}],

“requestStatus”:”New”,

“requestedDateTime”:”2013-03-28T11:31Z”,

“dueDateTime”:”2013-03-29T11:31Z”,

“requestNoteList”:[

{“noteText”:”Need to talk ASAP!”

“noteCreator”:”John Bell”

}

],

“requestRelatedTo”: { “contactIdList”: [ ”164879984” ] },

}

],

“source”:”Website”,

“status”:”InDiscussion”,

“createDateTime”:”2013-03-28T1:31Z”,

“productServiceRequestRelatedTo”:{

“contactIdList”:[ ”12345” ],

“relationshipIdList”:[ ”105949495” ]

}

“productServiceRequestNoteList”:[

{“noteText”:”I heard you are awesome!”

“noteCreator”:”John Bell”

},

{“noteText”:”Interested in a $100,000 Mortgage!”

“noteCreator”:”UserId353434”

}

]

}

]

}

### REST-JSON DELETE ProductServiceRequest Example

**Required**: messageContext and at least one productServiceRequestId in the productServiceRequestFilter

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: DELETE**

X-API-Version: >=4.3.0

**PUT h**ttps://api.datasource.com/productservicerequest

{

“productServiceRequestMessage”:{

“messageContext”: { <see MessageContext.xsd>

},

“productServiceRequestFilter”: {

“relationshipIdList”:[ ”105949495” ]

}

}

}

**RESPONSE:**

Headers:

Status Code: 200 Ok

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

Content-Language: en-us

Payload:

{

“productServiceRequestMessage”:{

“messageContext”: { <see MessageContext.xsd>

}

}

### REST-JSON CREATE ProductServiceRequest (crm style submission)

This example shows where the relationship is UNKNOWN therefore contact is also unknown. For example, this is hitting a data source other than the core end-point that might queue up similar requests for information, products and services that are not ready to be submitted to a final end-point for the information.

**Required**: messageContext and at least one lead in the productServiceRequestList collection and at least one productOfInterest, source.

**REQUEST:**

Headers:

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-API-Version: >=4.0.0

**POST h**ttps://api.crmdatasource.com/productservicerequest

{

“productServiceRequestMessage”:{

“messageContext”: { <see MessageContext.xsd>

},

“productServiceRequestList”: [

{

“requestList”:

[

{“requestItem”:[{“question”:”What are your mortgage rates?”}],

“requestStatus”:”New”,

“requestedDateTime”:”2013-03-28T11:31Z”,

“dueDateTime”:”2013-03-29T11:31Z”,

“requestRelatedTo”:{

“partyList”:

[

{

{“entity”:{

“individual”:{

formattedName:”Tom Thumb”

}

},

“contactList:[

{“type”:”PHONE”,

“phone”:{

“type”:”WORK”,

“number”:”1-847-555-1212”

“planFormat”:”NANP”

}

},

{“type”:”EMAIL”,

“email”:{

“type”:”HOME”,

“address”:”tt@cufx.org”

}

}

]

}

],

},

“source”:”Website”,

“status”:”New”,

]

}

]

}

**RESPONSE:**

Headers:

Status Code: 200 Ok

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

Content-Language: en-us

Payload:

{

“productServiceRequestMessage”:{

“messageContext”: { <see MessageContext.xsd>

},

“productServiceRequestList”: [

{

“productServiceRequestId”:”1514asd4151230a”,

“requestList”:

[

{“requestItem”:[{“question”:”What are your mortgage rates?”}],

“requestStatus”:”New”,

“requestedDateTime”:”2013-03-28T11:31Z”,

“dueDateTime”:”2013-03-29T11:31Z”,

“requestRelatedTo”:{

“partyList”:

[

{

“partyId”:”13534”,

{“entity”:{

“individual”:{

formattedName:”Tom Thumb”

}

},

“contactList:[

{“contactId”:”38322”,

“type”:”PHONE”,

“phone”:{

“type”:”WORK”,

“number”:”1-847-555-1212”

“planFormat”:”NANP”

}

},

{“type”:”EMAIL”,

“email”:{

“type”:”HOME”,

“address”:”tt@cufx.org”

}

}

]

}

],

},

“source”:”Website”,

“status”:”New”,

]

}

]

}

# 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.