

Party Data and Services 3.0

# Document Status

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

This version: **Assembla**.com. Files Tag = CUFX\_3.0\_RFC\_Active

Previous Version: **Assembla**.com. Files Tag = CUFX\_3.0\_RFC\_Archive

# Change Log

|  |  |  |
| --- | --- | --- |
| Version | Date | Changes |
| 0.0.01 |  | * Initial Draft |
| 0.0.02 |  | * Cleaned up formatting * Added Party Credentials data structure for storing information for accessing different data stores |
| 0.0.03 |  | * Updated logo and version information in document status * Move Required/Not required to service definition * Added ArtifactId to the Identification Document section so image, etc. could be related to the identification information * Added identificationDocument container around document fields * Reordered fields for better readability * Added missing State, City, etc. in address data structure * Moved the credentials list to party data from the preferences data * Shortened many of the value names due to length getting too long |
| 0.0.04 |  | * Extended list of possible credential types * Moved Codeword into credentials item * Added PostalCode in address contact information * Changed CredentialGroupId to fiUserId to match security model |
| 0.0.05 |  | * Added CTR Address as contact type * Added IRS Verified TaxId Flag * Added the IRS withholding fields * Added IRS TaxId Warning Count * Added additional document types for individual (photo, signature card) * Added display order for documentation * Changed eligibility requirement key values * Added phone number format identifiers * Added income amounts |
| 0.0.06 |  | * Added temporary credential field and expiration date/time for credentials * Reformatted domain constants with underscores for better readability * Added date formatting for each date field |
| 0.0.07 |  | * Added Doing Business As fields (DBA) |
| 0.0.08 |  | * Minor fixes when converting to XSD in capitalization and copy paste errors |
| 0.0.09 |  | * Removed definitions for field level data as it’s been moved to party.xsd |
| 0.0.10 |  | * Add party create, read, update, delete examples * Replace parameters with messageContext.xsd and partyFilter.xsd definitions |
| 0.0.11 |  | * Converted Plural Form to List Form |
| 0.0.12 |  | * Fixed typos in service examples |
| 0.0.13 |  | * Continued cleaning up examples |
| 0.0.14 |  | * Updated Overview of Specification |
| 0.0.15 |  | * Update references to party.xsd to Party.xsd |
| 0.0.16 |  | * Moved credentials to CredentialGroup.xsd and its own spec |
|  |  | * Added householdId to party object on read (calculated by back end systems) * Remove comment about returning minimum amount of data. This is not currently supported by the spec. Reviewing oData as method for limiting amount of data. * Renamed entity to characteristics |
| 0.0.17 |  | * Updated Id fields to be based on XSD’s |
| 0.0.18 |  | * Switch to use X-HTTP-METHOD-OVERRIDE standard rather than subMethod non-Standard method for overriding request types * Create a partyMessage wrapper for every message to increase ability for infrastructure to serialize the data |
| 3.0 | **10/22/2013** | * Versioning and format change with release CUFX 3.0 |
| 3.0 | **12/16/2013** | * Update examples X-API-Version to >=3.0.0 |
| 3.0 | **12/20/2013** | * Fixed typos and consistent formatting. |

# Overview of Specification

The Party Data Model and Services define how a party is created, read, updated and deleted on the core. A party is an individual, organization, club or not-for-profit that uses or is associated to the financial institution’s products and services.

This service is the method for creating, modifying, reading or deleting party data. In addition, when creating a party, if the contact data is provided (address, email, phone), then it will also add that data related to the party to the core data.

# Any knowN Errors in the document

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

# Table of Contents

[Document Status 1](#_Toc349046099)

[Authors and Change Log 1](#_Toc349046100)

[Overview of Specification 2](#_Toc349046101)

[Any Known Errors in the document 2](#_Toc349046102)

[Table of Contents 3](#_Toc349046103)

[Document Conventions 3](#_Toc349046104)

[Definitions Related to the Specification 3](#_Toc349046105)

[Data Elements 3](#_Toc349046106)

[Filters used when accessing the Party data 3](#_Toc349046107)

[Party data attributes 4](#_Toc349046108)

[Party Services 4](#_Toc349046109)

[Overview 4](#_Toc349046110)

[Party resource based create, read, update, delete services 4](#_Toc349046111)

[REST-JSON CREATE Party example 5](#_Toc349046112)

[REST-JSON READ Party example 7](#_Toc349046113)

[REST-JSON UPDATE Party example 8](#_Toc349046114)

[REST-JSON DELETE Party example 9](#_Toc349046115)

[General Error Handling for All Services 10](#_Toc349046116)

[**Bibliography** 10](#_Toc349046117)

# 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 utilizes Word Styles.

All Citations must utilize Word Citations so that it automatically shows at the end of the document.

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

# Definitions related to the specification

**Party**

Any person or entity which may be in the process of researching products and services at a financial institution or may have an existing account relationship with the financial institution and stored in a financial services platform. Typically this is limited to entities that have an SSN or TIN created for them.

# Data Elements

## Filters used when accessing the Party 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 **PartyFilter.xsd.**

## Party Data attributes

All CUFX fields related to a party are defined in Party.xsd.

# Party Services

## Overview

|  |  |
| --- | --- |
| Definition | Collection of services to manage a party |
| Overview of Capabilities | Create, read, update and delete a party. The following scenarios may exist. The party may be connected to an existing relationship (by adding the partyId using the relationship service), account (by using the loan or deposit service) or contact (by using the contact service or explicitly through the use of this service). This service only creates the parties/contacts and does not modify the relationship data directly. In addition, a party may exist without an existing relationship or account in preparation to be connected to a relationship or account. Some systems may need to place the party in a temporary location until the relationship or accounts are made available. |
| Dependencies | Security Services, messageContext, party |
| Sample CUFX REST LINK | https://api.dataprovider.com/party/ |
| CUFX SOAP LINK |  |
| CUFX WaDL LINK |  |

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

|  |  |
| --- | --- |
| INPUTS | cufx:partyMessage (which includes)   * [cufx:messageContext](file:///\\files2\users\CMarjaniemi\Projects\CUFX\MessageContext.html) * cufx:partyFilter (for read, update, delete) * cufx:partyList (for create, update) |
| Outputs | cufx:partyList |
| Return Values | cufx: Error |
| Side Effects | Creation, update or deletion of party, contact; read has no side effects |
| Dependencies | Security Services for authentication and security |
| Fields used | **Message Headers** : See security services  **partyMessage: which includes**   * **messageContext**: See MessageContext.xsd * **Filters**: See PartyFilter.xsd * **Attributes: party : See Party.xsd** |

### REST-JSON CREATE Party example

This example shows where the relationship is not known and the contact is not known. This service only creates the parties/contacts and does not modify the relationship data.

This example creates an entity that is an individual (as opposed to an organization, trust or estate).

**Required**: messageContext, at least one party within partyList

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: >=3.0.0

**POST** [**h**ttps://api.datasource.com/party](https://api.datasource.com/party)

{

“partyMessage”:{

“messageContext”: { <see MessageContext.xsd> }

“partyList”: [

{“irs”:

{“taxId”:”123456789”,

“taxIdType”:”SocialSecurityNumber”,

“reportingFlag”:”true”,

"backupWithholdingExemptionReason":"ExceptFromWithholding"

},

“type”:”Individual”,

“characteristics”:

{“individual”:

{“title”:”General”,

“firstName”:”Thomas”,

“middleName”:”R”,

“lastName”:”Thumb”,

“mothersMaidenName”:”Toe”,

“nickname”:”Tom”,

“birthdate”:”1838-01-04”

“citizenshipList”:[

{“citizenship”:”USA”}

},

“employmentStatus”=”Contract”,

“employmentList:[

{“employerName”:”Barnum Circus”

“employmentStartDate”:”1843-04-01”,

“employeeOccupation”:”Entertainer”,

“incomeDetail”:{

“grossIncomeData”:{

“amount”:{“value”:15.00},

“frequency”:“Weekly”

}

}

]

}

}

“contactList”: [

{“contactType”:”Email”,

“email”:{

“type”:”Home”,

“address”:”tom@barnum.com”

}

}

]

}

]

}

}

**RESPONSE:**

Headers:

Status Code: 200 Ok

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

Content-Language: en-us

{

“partyList”: [

{**“**id”:”151349885h348870ag”,

“irs”:

{“taxId”:”123456789”,

“taxIdType”:”SocialSecurityNumber”,

“reportingFlag”:”true”,

"backupWithholdingExemptionReason":"ExceptFromWithholding"

},

“type”:”Individual”,

“characteristics”:

{“individual”:

{“title”:”General”,

“firstName”:”Thomas”,

“middleName”:”R”,

“lastName”:”Thumb”,

“mothersMaidenName”:”Toe”,

“nickname”:”Tom”,

“birthdate”:”1838-01-04”

“citizenshipList”:[

{“citizenship”:”USA”}

},

“employmentStatus”=”Contract”,

“employmentList:[

{“employerName”:”Barnum Circus”

“employmentStartDate”:”1843-04-01”,

“employeeOccupation”:”Entertainer”,

“incomeDetail”:{

“grossIncomeData”:{

“amount”:{“value”:15.00},

“frequency”:“Weekly”

}

}

]

}

}

“contactList”: [

{“contactType”:”Email”,

“email”:{

“type”:”Home”,

“address”:”tom@barnum.com”

}

}

]

}

]

}

### REST-JSON READ Party example

This example reads an entity that is an individual (as opposed to an organization, trust or estate). This service only reads the parties/contacts and does not read the relationship data.

**Required**: messageContext, at least one valid filter in partyFilter

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: >=3.0.0

**POST** [**h**ttps://api.datasource.com/party](https://api.datasource.com/party)

{

“partyMessage”:{

“messageContext”: { <see MessageContext.xsd> },

“partyFilter”:{

“partyIdList”:[ ”151349885h348870ag” ]

}

}

}

**RESPONSE:**

Headers:

Status Code: 200 Ok

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

Content-Language: en-us

{

“partyList”: [

{**“**id”:”151349885h348870ag”,

“irs”:

{“taxId”:”123456789”,

“taxIdType”:”SocialSecurityNumber”,

“reportingFlag”:”true”,

"backupWithholdingExemptionReason":"ExceptFromWithholding"

},

“type”:”Individual”,

“characteristics”:

{“individual”:

{“title”:”General”,

“firstName”:”Thomas”,

“middleName”:”R”,

“lastName”:”Thumb”,

“mothersMaidenName”:”Toe”,

“nickname”:”Tom”,

“birthdate”:”1838-01-04”

“citizenshipList”:[

{“citizenship”:”USA”}

},

“employmentStatus”=”Contract”,

“employmentList:[

{“employerName”:”Barnum Circus”

“employmentStartDate”:”1843-04-01”,

“employeeOccupation”:”Entertainer”,

“incomeDetail”:{

“grossIncomeData”:{

“amount”:{“value”:15.00},

“frequency”:“Weekly”

}

}

]

}

},

“contactList”: [

{“contactType”:”Email”,

“email”:{

“type”:”Home”,

“address”:”tom@barnum.com”

}

}

]

}

]

}

### REST-JSON UPDATE Party example

This example updates an entity that is an individual (as opposed to an organization, trust or estate).

**Required**: messageContext, at least one valid filter in partyFilter, the specific id of the party to be updated in the message

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: >=3.0.0

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

{

“partyMessage”:{

“messageContext”: { <see MessageContext.xsd> },

“partyFilter”:{

“partyIdList”:[”151349885h348870ag”}]

}

“partyList”: [

**{“**id”:”151349885h348870ag”,

“contactList”: [

{“contactId”:”1507103209349”

“email”:{

“address”:”iquit@bigandtall.com”

}

},

“fiUserIdList”:[

”84541abs38375443”,

”183563pijef”

],

“householdId”:”13534313fa3534”

]

}

}

**RESPONSE:**

Headers:

Status Code: 200 Ok

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

Content-Language: en-us

{

[

{**“**id”:”151349885h348870ag”,

“irs”:

{“taxId”:”123456789”,

“taxIdType”:”SocialSecurityNumber”,

“reportingFlag”:”true”,

"backupWithholdingExemptionReason":"ExceptFromWithholding"

},

“type”:”Individual”,

“characteristics”:

{“individual”:

{“title”:”General”,

“firstName”:”Thomas”,

“middleName”:”R”,

“lastName”:”Thumb”,

“mothersMaidenName”:”Toe”,

“nickname”:”Tom”,

“birthdate”:”1838-01-04”

“citizenshipList”:[

{“citizenship”:”USA”}

},

“employmentStatus”=”Contract”,

“employmentList:[

{“employerName”:”Barnum Circus”

“employmentStartDate”:”1843-04-01”,

“employeeOccupation”:”Entertainer”,

“incomeDetail”:{

“grossIncomeData”:{

“amount”:{“value”:15.00},

“frequency”:“Weekly”

}

}

]

}

},

“contactList”: [

{“contactId”:”1507103209349”

“email”:{

“address”:”iquit@bigandtall.com”

},

“fiUserIdList”:[

”84541abs38375443”,

”183563pijef”

],

“householdId”:”13534313fa3534”

}

]

}

### REST-JSON DELETE Party example

Note: This example deletes an entity that is an individual (as opposed to an organization, trust or estate). This service only deletes the parties and does not delete the relationship data or contact data.

**Required**: messageContext, at least one valid filter in partyFilter, the specific id of the party to be updated in the message

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: >=3.0.0

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

{

“partyMessage”:{

“messageContext”: { <see MessageContext.xsd>},

“partyFilter”:{

“partyIdList”:[ ”151349885h348870ag” ]

}

}

}

**RESPONSE**:

Headers:

Status Code: 200 Ok

# General Error handling For All Services

Refer to latest CUFX documentation *Error Mapping*.

**Bibliography**

*E.164.* (n.d.). Retrieved 06 28, 2012, from International PUblic Telecommunications Number Plan: http://www.itu.int/rec/T-REC-E.164/en

*North American Number Plan Administration.* (n.d.). Retrieved 06 28, 2012, from North American Number Plan Administration: http://www.nanpa.com/

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