OpenTransact	P. Braendgaard
Internet-Draft	PicoMoney Inc
Intended status: Informational	N. Matake
Expires: July 13, 2012	Cerego Japan Inc
	T. Brown
	D. Nicol
	TipJar
	January 10, 2012

TOC

OpenTransact Core
draft-pelle-opentransact-core-02

Abstract

This document specifies the OpenTransact standard for requesting and performing transfers of assets over the http protocol.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as “work in progress.”

This Internet-Draft will expire on July 13, 2012.

Table of Contents

TOC

1. Status of Document

This document is in draft and reflects the current designs from the OpenTransact mailing list.

TOC

2. Introduction

The goal is to develop a very simple low level standard for transferring an amount of an asset from one account to another.

Most payment systems and existing standards use the messaging paradigm for historical reasons. OpenTransact specifically rejects the message paradigm and prefers the RESTful (REpresentational State Transfer) resource approach as used on the web with URL's and HTTP at it's core.

We aim to create a new standard from scratch ignoring all legacy systems, while leaving it flexible enough to allow applications built on top of it to deal with legacy systems.

The standard is designed to follow standard RESTful practices and be concise and human readable.

2.1. Assets

Within OpenTransact we use the accounting definition of the word "Asset" to mean anything fungible about which one could make an accounting book entry.

For example:

- money
- shares
- bonds
- mobile phone minutes
- hours of work
- property
- domain names
- tons of sand
- general admission tickets to an event
- et cetera

2.1.1. Asset Service

An Asset Service is a service maintained by an organization to manage accounts of one asset. For money and other financial assets the Asset Service would normally be run by a Financial Service Provider of some type. Other types of assets are offered by non-financial services. The regulatory definition of "financial" is of course out of the scope of this document.

2.1.2. Transaction URL

Each Asset Service has a unique transaction URL. An informal convention is developing, but is out of the cope of this document. Guidance concerning the form of these URLs with regard to specific details like currency, card type, size, color MAY become available in a future Best Current Practices document.

The service available at the transaction URL MUST follow basic REST practices:

- A transaction URL such as https://paymentsarewe.example.com/prwpoints MUST NOT contain query or fragment part.
- Loading the URL into a normal web browser should present a human readable description of the asset.
- A POST to the URL is used for creating a transaction transferring value.
- A GET to the URL from a normal web browser with specific parameters becomes a payment request that the user can authorize.
- A GET to the URL in a machine readable form such as json returns meta data about the asset and optionally a list of transactions that the current user is allowed to see.
- Each transaction has a unique URL. This SHOULD be formed by appending a unique transaction identifier to the transaction URL, such as https://paymentsarewe.example.com/prwpoints/aaf4c61ddcc5e8a2dabede0f3b482cd9aea9434d.

2.1.3. Example of Asset Services

Lets say it's an imaginary electronic currency asset provider eepay.example.net they only offer one asset type, their currency. So they would only have one transaction url:

- http://eepay.example.net/transactions

If the asset provider offered multiple currencies it should have a url for each currency as they are really separate asset types:

- http://eepay.example.net/transactions/usd
- http://eepay.example.net/transactions/euro

A bank offering OT as an alternative to ACH service could have a tranaction URL for each of the currencies in which it offers checking accounts. The details of interbank settlements are out of the scope of this document.

- http://coolbank.example.com/current
- http://coolbank.example.com/savings
- http://coolbank.example.com/bonds/mortgage

A mutual fund company would have a url for each of their funds.

- http://fidelify.example.com/funds/sp500
- http://fidelify.example.com/funds/emergingmarkets

With relaxation of current regulations on brokerage transactions all going through markets, a broker (rather, a stock market) could implement an OpenTransact API and have a different

URL for each symbol.

- http://nyex.example.com/trade/AAPL
- http://nyex.example.com/trade/EBAY

This would ease the barrier to registered securities being used directly as money.

If we let the URL do the describing, there are many different possibilities. This allows support for all manners of asset types.

All the above examples are fungible assets. In general it is best practice that one item of value for one asset is fungible for one another.

For unique items such as domain names, property titles and diamonds that are unique and infungible, we can still create asset urls for each item, but only accept transfer amounts of 1.

All the above examples are fungible assets.

For unique items such as domain names, property titles and diamonds that are unique and infungible, we may define a separate currency for the item representing partial ownership. The legal framework for defining the resulting partnerships and their governance is out of scope of this document.

TOC

2.1.4. Derivative Assets

A Derivative asset is an asset that is based on the current asset but provides different semantics and business rules.

Examples:

- Reserves/Escrows
- Subscription services providing recurring billing
- Exchanges providing exchange of asset with another asset

It is out of scope to define here exactly how these would work, but the OpenTransact community will build a list of recipes for how to implement these and may publish further drafts outlining specifics in the future.

TOC

2.2. Roles

OpenTransact defines 4 roles:

- Asset provider
  - The entity providing or issuing the asset
- Transferer
  - The entity transfering an amount of the asset
- Transferee
  - The entity receiving an amount of the asset
- 3rd party application
  - An application performing a transfer on behalf of the Transferer

---

TOC

2.3. Transfer

We will use the term “Transfer” as it is more widely applicable than “Payment”.

A Transfer is legally a transfer in ownership of some amount of the Asset from the Transferer to the Transferee.

Eg. A Payment of $12 from Bob to Alice is a Transfer of $12 with Bob being the Transferer and Alice the Transferee.

TOC

2.4. Transfer Request

A Transfer request is when the Transferee requests a transfer of a given amount of an asset from the Transferer.

Eg. Alice sends an invoice to Bob for $12. This is a Transfer Request from Alice to Bob where Alice is the Transferee and Bob the Transferer.

A Transfer Request is simply a specially formatted payment link that takes Bob to Asset Service if followed. The Asset Service shall present the Transfer Request to Bob who can authorize or decline it.

TOC

2.5. Transfer Authorization

A Transfer Authorization is when the Transferee or a third party application requests an authorization to transfer of a given amount of an asset from the Transferer.

Eg. Bob wants to hire someone on an online job board to edit a document for $33. The Job Board creates a Transfer Authorization link. Bob follows this link to the Asset Service and authorizes the Job Board to transfer $33 from his account to some one else within a time period.

A Transfer Request is simply a specially formatted payment link that takes Bob to Asset Service if followed. The Asset Service shall present the Transfer Request to Bob who can authorize or decline it. If Bob authorizes it the Asset Service issues an authorization code to the Job Board that they can use to exchange for an OAuth token.

TOC

2.6. Exchange Transaction

A Transfer is often but not always part of an exchange of value between 2 types of assets. Eg.:

- 10 consulting hours exchanged for 1100EUR
- 10 USD exchanged for 8 EURO
- 0.99 USD exchanged for an mp3 song

There are as many different exchange mechanisms for creating exchanges as there are exchange types.

- Invoicing system
- App Store
- Web shop
- Auction site
- Stock Exchange

It is outside the scope for OpenTransact to define every single type of exchange that is possible. OpenTransact provides a fundamental building block in building such systems. It integrates well with exchange systems that don't yet understand OpenTransact.

TOC

2.6.1. Exchanged item URI

In an Exchange Transaction we can include a url to the exchanged item. This URI could either be a link to the exchanged item itself or the unique URI identifying the transaction itself.

TOC

2.6.2. Authorization

A useful building block for creating Exchange services is the Transfer Authorization flow.

TOC

2.7. Vocabulary

OpenTransact includes a standard list parameter names used in the GET and POST requests.

- *asset* - Transaction URL
- *from* - Transferer account identifier
- *to* - Transferee account identifier
- *amount* - Amount of asset
- *note* - Textual description of transfer
- *for* - Identifier of exchanged item. This SHOULD be a URI
- *redirect_uri* - URI to redirect User Agent to
- *callback_uri* - URI for performing callback after request
- *client_id* - Identifier of 3rd party application
- *expires_in* - Request that a token expires in given seconds

They are designed to be small and semantically correct. eg

> A transfer of 10 USD from Bob to Alice for consulting.

Would become the following:

- amount=10
- asset=http://pay.me/usd
- from=bob@test.example..com
- to=alice@test.example..com
- note=Consulting on XYZ project
- for=http://myinvoice.test/invoices/123123

TOC

2.8. Authentication

OpenTransact does not define any new authentication mechanisms, but relies on the Asset Provider's existing mechanisms for authenticating the Transferer and OAuth 2.0 **[I-D.ietf-oauth-v2-bearer]** for authenticating 3rd party applications on behalf of the Transferer.

TOC

2.9. Parameter encoding

Since OpenTransact is designed to be simple to implement, the basic parameter encoding is URL form encoding.

Data responses should be in JSON format.

OpenTransact includes a standard for name strings and value ranges for use with the JSON objects returned from a request made to a compliant OT transaction URL.

TOC

2.10. Extensions

OpenTransact is designed to be extensible, either through proprietary extensions, conventions or futher standards.

For example it may be useful to follow the lat/lon convention allowing geotagging of data.

TOC

2.11. Notational Conventions

The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL', 'SHALL NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED', 'MAY', and 'OPTIONAL' in this specification are to be interpreted as described in **[RFC2119]**.

This specification uses the Augmented Backus-Naur Form (ABNF) notation of **[RFC5234]**.

Certain security-related terms are to be understood in the sense defined in **[RFC4949]**. These terms include, but are not limited to, 'attack', 'authentication', 'authorization', 'certificate', 'confidentiality', 'credential', 'encryption', 'identity', 'sign', 'signature', 'trust', 'validate', and 'verify'.

Unless otherwise noted, all the protocol parameter names and values are case sensitive.

3. Transfer Request

A Transfer Request consists of a GET to the Asset URL.

```
GET /usd?to=bill@example.com&amount=100.00&note=Milk&redirect_uri=http://site.example.com/callback HTTP/1.1
Host: pay.me
```

Figure 1

We use the following parameters from our common vocabulary. All fields are optional:

- *to* Account identifier of Transferee. If left out it defaults to the 3rd party applications own account on Asset Service or a predefined account as specified when authorizing the access token.
- *amount* Amount as a number with decimal points. Symbols are allowed but SHOULD be ignored. If left out it defaults to the Asset's minimum transfer, 1 or an amount predefined when authorizing the access token.
- *note* Textual description, which can include hash tags. Asset Service may truncate this. No default.
- *from* Account identifier of Transferer. This should normally be left out as it is implied by the authorizer of the Access Token. The Asset Service MUST verify that the Access Token is authorized to transfer from this account. This could be useful for Asset providers charging their customers accounts.
- *for* URI identifying the exchanged item.
- *redirect_uri* URI for redirecting client to afterwards
- *callback_uri* URI for performing a web callback

When a user follows this link, the Asset Service should present the user with a form authorizing the payment.

Note: Client can include OpenID Connect parameters.

3.1. Response

The user is redirected back to the clients redirect uri with the following url encoded parameters:

- *txn_url* A url identifying the transaction.

Asset provider can include an access_token in the query string of txn_url.

3.2. Errors

Error types use OAuth 2.0's error codes. **[I-D.ietf-oauth-v2]**

4. Transfer Authorization

A Transfer Authorization consists of a GET to the Asset URL with a client_id.

```
GET /usd?to=bill@example.com&amount=100.00&note=Milk&redirect_uri=http://site.example.com/callback&client_id=1234 HTTP/1.1
Host: pay.me
```

Figure 2

A Transfer Authorization is really a OAuth2 Authorization **[I-D.ietf-oauth-v2]** with a few extra payment related parameters.

We use the following parameters from our common vocabulary. All fields are optional:

- *to* Account identifier of Transferee. If left out it defaults to the 3rd party applications own account on Asset Service or a predefined account as specified when authorizing the access token.
- *amount* Amount as a number with decimal points. Symbols are allowed but SHOULD be ignored. If left out it defaults to the Asset's minimum transfer, 1 or an amount predefined when authorizing the access token.
- *note* Textual description, which can include hash tags. Asset Service may truncate this. No default.
- *from* Account identifier of Transferer. This should normally be left out as it is implied by the authorizer of the Access Token. The Asset Service MUST verify that the Access Token is authorized to transfer from this account. This could be

useful for Asset providers charging their customers accounts.
- *for* URI identifying the exchanged item.

OAuth2 related parameters. See **[I-D.ietf-oauth-v2]** section 5 for full details

- *client_id* OAuth2 client id
- *redirect_uri* URI for redirecting client to afterwards
- *callback_uri* URI for performing a web callback
- *response_type* token or code REQUIRED
- *expires_in* Request the amount of time in seconds this token should be valid

When a user follows this link, the Asset Service should present the user with a form authorizing the payment.

Note: Client can include OpenID Connect parameters.

TOC

4.1. Response

Follows OAuth 2 response depending on response_type requested. **[I-D.ietf-oauth-v2]**

TOC

4.2. Errors

Error types use OAuth 2.0's error codes. **[I-D.ietf-oauth-v2]**

TOC

5. Transfer

A transfer consists of a HTTP POST to the asset url by a 3rd party application on behalf of the Transferer.

The Application MUST have an OAuth 2.0 access token issued as defined in the **[I-D.ietf-oauth-v2]** section 7.

The asset provider SHOULD support the **[I-D.ietf-oauth-v2-bearer]** Access Token type and can support other access token such as **[I-D.ietf-oauth-v2-http-mac]**.

The Transfer MUST be created using HTTPS when using **[I-D.ietf-oauth-v2-bearer]** and other unsigned access tokens.

```
POST /usd HTTP/1.1
Host: pay.me
Authorization: Bearer vF9dft4qmT

to=bill@example.com&amount=100.00&note=Milk
```

Figure 3

We use the following parameters from our common vocabulary in 1.6. All fields are optional:

- *to* Account identifier of Transferee. If left out it defaults to the 3rd party applications own account on Asset Service or a predefined account as specified when authorizing the access token.
- *amount* Amount as a number with decimal points. Symbols are allowed but SHOULD be ignored. If left out it defaults to the Asset's minimum transfer, 1 or an amount predefined when authorizing the access token.
- *note* Textual description, which can include hash tags. Asset Service may truncate this. No default.
- *from* Account identifier of Transferer. This should normally be left out as it is implied by the authorizer of the Access Token. The Asset Service MUST verify that the Access Token is authorized to transfer from this account. This could be useful for Asset providers charging their customers accounts.
- *for* URI identifying the exchanged item.

TOC

5.1. Response

http 201 with Receipt json.

TOC

6. Receipt

The receipt is returned when creating a transaction as well as when accessing a transaction url. It can also be used for creating a transaction list by the asset provider.

The receipt is a JSON object consisting of the following fields:

- *txn_url*
- *to*
- *from*
- *amount*
- *note*
- *for*
- *asset_url*
- *timestamp*

TOC

7. Asset Meta data

A client can find out information about an asset by accessing the asset url directly with a http Accept header of application/json:

```
GET /usd HTTP/1.1
Host: pay.me
Accept: application/json
```

Figure 4

This returns a json hash of meta information about the asset.

The minimum required data would be:

- name - Short name of asset

The minimal asset meta data is:

```
{
  "name":"Pay Me"
}
```

Figure 5

Further OpenTransact specific parameters could be:

- default_amount - The default amount transfered if an amount is not specified in a transfer
- provider_uri - The provider of the asset's home page
- description - Short description
- logo_uri - Image url for Assets logo
- unit - ISO currency unit of asset if monetary or other such as (minute, gram, point etc)
- derivatives - a list of derivative assets supported by this server (see below)

Example:

```
{
  "name":"Pay Me",
  "default_amount":1.0,
  "provider_uri":"http://pay.sample.com",
  "logo_uri":"http://pay.sample.com/logo.png",
  "unit":"USD",
  "derivatives": [
    {"reserve":"http:/pay.sample.com/reserves"},
    {"subscription":"http:/pay.sample.com/subscriptions"},
    {"exchange":"http:/pay.sample.com/exchange"}
  ]
}
```

Figure 6

Asset services can provide further information more specific to their particular asset type.

If an OAuth Access Token is provided the Asset Service should provide information related to the capabilities of the token.

- balance - balance of account
- available_balance - available balance of account

For tokens obtained through a Transfer Authentication this should reflect the remaining balance of the authorized amount.

TOC

7.1. Transaction history

If tokens scope allows access to accounts transaction history a url to the transaction history or a transaction history SHOULD be included here:

- transactions_uri - URI to list of transactions
- transactions - array of receipts

TOC

7.2. Derivative assets

If an asset has derivative assets they SHOULD be listed in the optional derivatives list:

```
{
  "name": "Pay Me",
  "derivatives": [
    {"reserve":"http:/pay.sample.com/reserves"},
    {"subscription":"http:/pay.sample.com/subscriptions"},
    {"exchange":"http:/pay.sample.com/exchange"}
  ]
}
```

Figure 7

The OpenTransact group will maintain a list of recipes that may be further standardized.

8. Security Considerations

TOC

The security of OpenTransact is achieved by it's use of **[I-D.ietf-oauth-v2]**. Please see its **security discussion** and a more thorough discussion in **[I-D.ietf-oauth-v2-threatmodel]**.

9. Normative References

TOC

[I-D.ietf-oauth-v2] Hammer-Lahav, E., Recordon, D., and D. Hardt, "**The OAuth 2.0 Authorization Protocol**," draft-ietf-oauth-v2-22 (work in progress), September 2011 (**TXT**, **PDF**).

[I-D.ietf-oauth-v2-bearer] Jones, M., Hardt, D., and D. Recordon, "**The OAuth 2.0 Authorization Protocol: Bearer Tokens**," draft-ietf-oauth-v2-bearer-15 (work in progress), December 2011 (**TXT**, **PDF**).

[I-D.ietf-oauth-v2-http-mac] Hammer-Lahav, E., Barth, A., and B. Adida, "**HTTP Authentication: MAC Access Authentication**," draft-ietf-oauth-v2-http-mac-00 (work in progress), May 2011 (**TXT**, **PDF**).

[I-D.ietf-oauth-v2-threatmodel] Lodderstedt, T., McGloin, M., and P. Hunt, "**OAuth 2.0 Threat Model and Security Considerations**," draft-ietf-oauth-v2-threatmodel-01 (work in progress), October 2011 (**TXT**).

Authors' Addresses

TOC

Pelle Braendgaard
PicoMoney Inc
Email: **pelle@picomoney.com**

Nov Matake
Cerego Japan Inc
Email: **nov@matake.jp**

Tom Brown
Email: **herestomwiththeweather@gmail.com**

David Nicol
TipJar
Email: **davidnicol@gmail.com**