`

New Payments Platform

API Framework

Publication Version 2.0

22 May 2019

NPP Australia Limited and SWIFT SCRL

Contents

TERMS OF USE

The NPP API Framework is provided 'AS IS' without warranties of any kind, and NPP Australia Limited (NPPA) neither assumes nor accepts any liability for any errors or omissions contained in this document. **NPPA disclaims all representations and warranties, express or implied, including without limitation, implied warranties of merchantability, fitness for a particular purpose, title and non-infringement, as to the specifications set out in this document.**

NPPA makes no representations or warranties with respect to intellectual property rights of any third parties in or in relation to the NPP API Framework. NPPA has no responsibility to determine whether any implementation of the NPP API Framework may violate, infringe, otherwise exercise the patent, copyright, trademark, trade secret, know-how other intellectual property rights of any third party, and thus any person who implements any part of the NPP API Framework should make their own inquiries and obtain independent advice before implementing any part of this Framework.

Without limiting the foregoing, the NPP API Framework may provide for the use of technology which may be the subject matter of patents in several countries. Any person seeking to implement a solution based on the NPP API Framework is solely responsible for determining whether its activities require a licence to such technology. NPPA shall not be liable for any person's infringement of intellectual property rights in connection with the NPP API Framework.

Third parties wishing to use NPPA intellectual property including the PayID trademark, should visit **https://payid.com.au/payid-for-business/**.

Pay ID is a registered trademark of NPP Australia Limited.

Important Note:
Designed to promote inter-operability and standardisation, the NPP API Framework defines the key technical approach and mandatory data attributes for NPP APIs, aligned to ISO 20022 standards. The NPP API Framework includes sample APIs which are included for the purposes of illustrating how the framework could be used by participating financial institutions. For more information regarding what financial institution APIs are available for third party use, please contact your financial institution

Please note that NPPA does not host an NPP API service or offer NPP APIs for third party use on the platform. Participating financial institutions may make their proprietary NPP APIs available for use by third parties.

1 Introduction

The New Payments Platform (NPP) is national infrastructure for fast, flexible, data rich payments in Australia. NPP Australia Limited (NPPA) has engaged SWIFT to design, build and operate the NPP basic infrastructure.

This document has been jointly developed by NPPA and SWIFT to promote development of the NPP ecosystem and to assist NPP Participants, Third Party Service Providers and software developers with the development of API solutions for NPP transactions.

In order to maximise standardisation and interoperability, and to provide a consistent NPP experience, NPP Participants, Third Party Service Providers and software developers are encouraged to use the NPP API Framework as a guide for the development of their own API solutions for NPP. This document provides a consolidated point of reference and provides an expanded view for NPP Participants, Third Party Service Providers and software developers to further their adoption of API technologies for NPP.

NPPA does not mandate use of this Framework by participating financial institutions. NPP Participants that utilise this Framework for the development of open APIs or partner-based APIs are free to use this Framework, and to publicise such use, as they see fit.

NPPA does not host an NPP API service or offer NPP APIs for third party use on the platform. Participating financial institutions may make their proprietary NPP APIs available for use by third parties.

1.1 Purpose & Scope

The purpose of the NPP API Framework document is to provide guidance relevant to the design of APIs in the context of NPP. The Framework is designed to support and facilitate NPP Participants' open NPP APIs and partner-based NPP APIs, and to enable Third Party Service Providers and software developers to design NPP payment services. The NPP API Framework is intended to encourage innovation by establishing a set of minimum standards primarily for the benefit of Third Party Service Providers and software developers, which may obviate the need to build multiple customised APIs for each NPP Participant and Overlay Service product.

Overlay Service Providers (**OSPs**) are encouraged to make available specific data usage or mandatory elements for use with the API framework. These elements will be defined and maintained by each OSP and published by NPPA together with this framework.

NPP Participants, Third Party Service Providers and software developers may identify different or additional considerations for their API design at a more detailed level which are not covered by this document.

The specific purpose of this document is to:

- Identify a common set of design principles and best practices that could be leveraged by the NPP community to reduce interoperability variation across core data exchanges when implementing API based technologies;
- At a high level, outline considerations to ensure the NPP community has a common reference of key API design considerations and how they may be approached by the industry; and
- Identify applicable standards for NPP data transfer, data constructs and security

1.2 Initial sample APIs

To provide Third Party Service Providers and software developers with an understanding of how to utilise the NPP API Framework, seven sample NPP APIs have been included as part of the framework documentation. The sample NPP APIs include usage guidelines in .pdf and excel format as well as a JSON example. The samples include the following:

- Look up of a PayID which will validate that the PayID exists and if valid return the associated short name and status - API: pain.a11.001.03 (Get Account Servicer by Alias);
- Submit a payment initiation request to process a payment - API: pain.a09.001.02 (Submit Payment);
- Confirm completion of payment - API: pain.a10.001.02 (Get Payment Status);
- Submit a request to return a previously submitted payment – API: camt.a09.001.01 (Cancel Payment);
- Submit a request to return a previous received payment – API: camt.a10.001.01 (Return Payment);
- Fetch payment details of a debit or credit from/to their account – API: camt.a11.001.01 (Payment Event Details); and
- Send notification of a payment event, debit or credit, from/to an account – API: camt.a12.001.01 (Payment Event Notification).

1.3 Approach

This document describes a development framework and refers to other related reference material.

To facilitate use of the document, each section has sub-sections to highlight key themes or processes NPP Participants and third parties should consider in their development of API solutions for NPP. It is intended to be of use to both existing and prospective NPP Participants, Third Party Service Providers and software developers.

The use of the API Framework document as a source of information does not affect or alter:

(a) any rights or obligations of NPP Participants under the NPPA Regulations and NPP Procedures. For the avoidance of doubt, where there is any inconsistency between this document and the NPP Regulations and Procedures, or an NPPA-approved NPP design document, those documents prevail to the extent of the inconsistency; or

(b) the rights or obligations of NPP Participants as data controllers, to comply with privacy laws and to establish their own permission frameworks and requirements for secure data transfer.

1.4. API Sandbox

NPP Australia in conjunction with SWIFT have developed an API sandbox to help Third Party Service Providers and software developers to learn and test the NPP's capabilities via the available sample NPP APIs. External parties can build and test NPP based solutions in this independent environment.

To request access to the API Sandbox, please complete this **form** and return it by email to **info@nppa.com.au**.

1.5 Glossary & abbreviations in this document

Term	Description
Addressing Service	Component of the NPP Basic Infrastructure that enables registration of customer account information and PayID (account proxy) information
API	Application Program Interface
BI	NPP Basic Infrastructure
Connected Institutions	Connect to the Basic Infrastructure solely for the purposes of sending and receiving non-value messages such as payment initiation messages
JSON	JavaScript Object Notation
NPP Participant	Connects to the Basic Infrastructure for the purposes of clearing and settling NPP Payments
Overlay Service	Refers to a payment service, or payment-related service, using the NPP Basic Infrastructure
Overlay Service Provider	Overlay Service Providers provide Overlay Services to Overlay Service Subscribers. Overlay Service Providers may also be Connected Institutions
PayID	Refers to an alias record in the Addressing Service. The Addressing Service provides a mechanism that allows a registered PayID (acting as a unique identifier for a Customer Account) to be resolved to an Account. A PayID can be one of four alias types; phone number, email address, ABN or Organisational Identifier
RESTful	Representational state transfer (REST) or RESTful web services
Third Party Service Provider	Payment service providers that are third parties (i.e. not any of the following: owner of the account, the account servicer or the account servicer's sponsoring Participant)

2 NPP Open API Design Principles

2.1 Basics

The following key design principles incorprate both RESTful concepts and ISO20022 as the data standard and take into consideration the flexibility of the NPP Overlay Service concept.

2.1.1 RESTful APIs

Each NPP API should adhere to the RESTful API concepts as *the transfer standard* of choice.

A RESTful API is a set of Hypertext Transfer Protocol (HTTP) request messages, along with a definition of the structure of response messages, which is in a JavaScript Object Notation (JSON) format.

Overall, the priority should be to have an API that is simple to understand and easy to use. In instances where following RESTful principles would be convoluted and complex, the principles have not been followed.

References:

- The highest level Data Description Language used is the JSON Schema: **http://json-schema.org/**
- Best Practice has also been taken from the Data Description Language for APIs; JSON API: **http://jsonapi.org/**
- The Interface Description Language used is the Swagger Specification version 2.0 (also known as Open API): **http://swagger.io/**

2.1.2 ISO 20022

NPP API payloads should be designed based on existing ISO 20022 message elements and components, where available, as *the data standard*. Intended to decrease implementation time for developers to consume, these APIs support interoperability with the asynchronous NPP message formats currently used across the platform.

The principles applied to re-use of ISO message elements and components are:

- Where relevant – the API payloads should be flattened so that they are more developer friendly.
- Only elements that are required for the functioning of the API endpoint should be included in the API payload. API endpoints are defined for specific use-cases (not to be generically extensible for all use-cases). For example - only elements that are required for a single immediate payment initiation would be included in the Payment API payload.
- Support modification of ISO 20022 elements where the existing standard does not cater for an API context (such as filtering, pagination etc.). For example, latitude and longitude in decimal format - as this is how developers will work with latitude and longitude; or using simple types (e.g. a single date-time field) instead of a complex type (e.g. a choice field with a nesting of date and time).
- Consideration of *ISO 20022 and JSON: An Implementation Best Practices.* Includes pertinent modelling guides

2.1.3 Security Standards

Each NPP Participant will be responsible for setting the security standards for third party service providers connecting to the NPP Participant via APIs.

NPPA recommends the use of global best practice in security standards where possible.

2.1.4 NPP Overlay Service Identification

In order to support in each common API request call, a mechanism to vary the API call attributes according to Overlay Service, the Service Level element (from <SvcLvl/Prtry> element in ISO 20022> is defined as the first element in the API request. The Service Level element is used to distinguish the specific messaging or overlay service under which an API (or XML message) is being used e.g.: npp.msg.01-x2p1.01 (example of an overlay service for basic messaging).

When available, NPPA intends to publish (with the Overlay Service Provider's permission) a list of Overlay Service specific attributes in conjunction with the API framework e.g. Osko Service X2P1 uses the 2x140 characters of unstructured remittance information.

2.1.5 Status Codes

Each API may need to consider three status codes that serve different purposes:

- The HTTP Status Code reflects the outcome of the API call (the HTTP operation on the resource).
  E.g. 200 - OK , 400 - Bad Request, 405 - Method Not Allowed
- In the API content, where a status could be returned to reflect the outcome of the request. For example, the Status field in a Payment API payload could reflect the status of a specific payment that makes use of the ISO 20022 PaymentStatusCode code-list enumeration (external code list) to report status.
  E.g. CH11 - Creditor Identifier Incorrect
- Participant specific status codes. Each NPP Participant may have a specific set of status codes that it has defined for API connectivity to its own infrastructure.

3 NPP Open API Use Cases

3.1 Payment Initiation

The Usage example (figure 1) and payment flow (figure 2 below) illustrates an NPP payment initiation process that *could* be realised as NPP real- time Credit Transfer through the use of a series of common open APIs made available by NPP Participants and associated institutions. APIs are foreseen as complementary to other channels such as asynchronous messaging or web forms / mobile phone apps.

3.1.1 Usage Example

Figure 1 below illustrates the example of a corporate paying their supplier and the associated interactions and work flow that might take place:

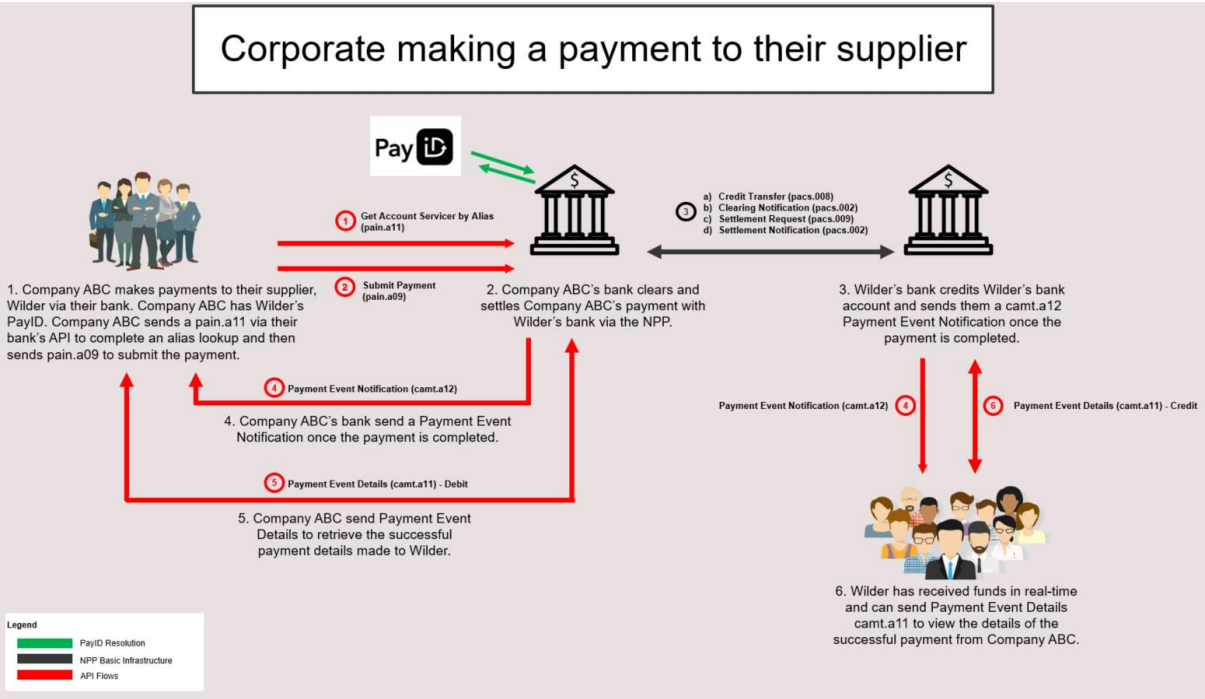

Figure 1. – API Usage Example

3.1.2 Payment Flow Processing Steps

The sequence of processing steps in this section is provided as an illustrative example of a possible payment flow using APIs; it is a guide only and is not meant to be prescriptive.

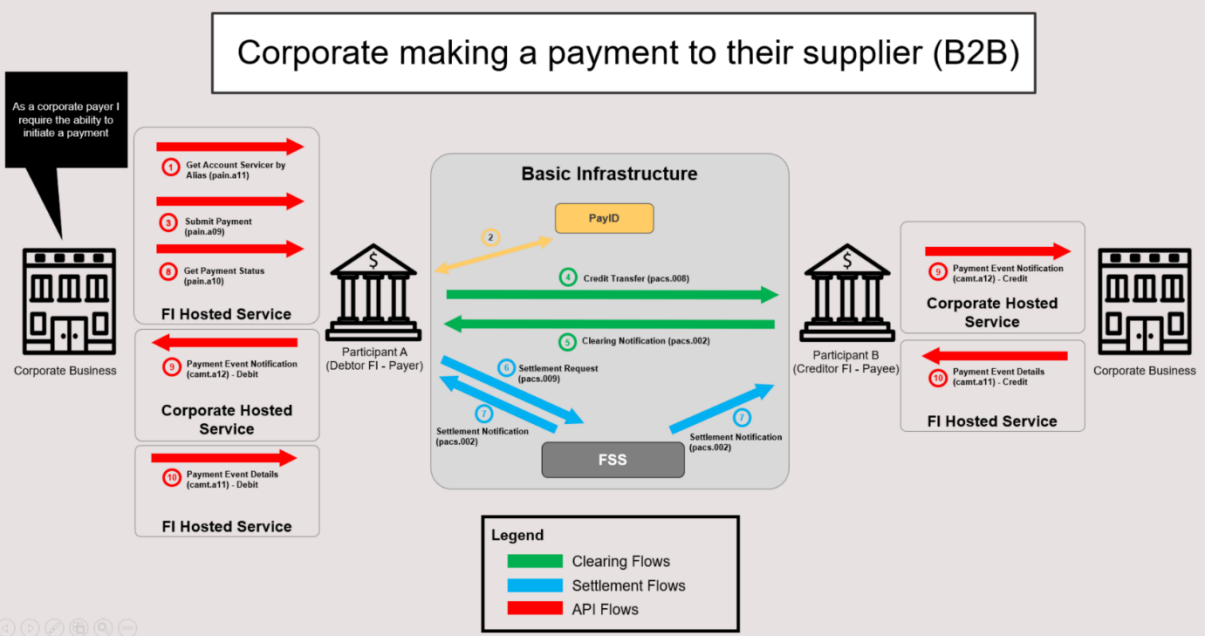

Figure 2. – Payment Initiation payment flow

Step	Description
1	Payer business initiates a check for a valid PayID through its NPP Participant before submitting a payment initiation
2	Payer business submits a payment initiation request to its NPP Participant
3	Payer's NPP Participant uses details in payment initiation message and PayID to create a NPP clearing request to the Payee's NPP Participant.
4	Payer's NPP Participant creates a clearing request with details from the payment initiation request and PayID which is routed via the NPP Basic Infrastructure to the Payee's NPP Participant
5	Payee's NPP Participant accepts clearing request and responds with a clearing notification
6-7	NPP Basic Infrastructure and FSS complete settlement processing and sends confirmations to NPP Participants
8	Payer's NPP Participant responds to Payment Status (successful / unsuccessful) to Payer
9	Payer NPP Participant to notify Payer's business of a payment event in their account. Payee NPP Participant to notify Payee's business of a payment event in their account.

Step	Description
10	Payer's business to fetch from the Payer's NPP Participant details of a debit from their account. Payee's business to fetch from the Payee's NPP Participant details of a credit to their account.

3.2 Cancel Payment

3.2.1 Usage Example

The figure below illustrates the example of a corporate requesting the return of an incorrect payment and the associated interactions and work flow that might take place:

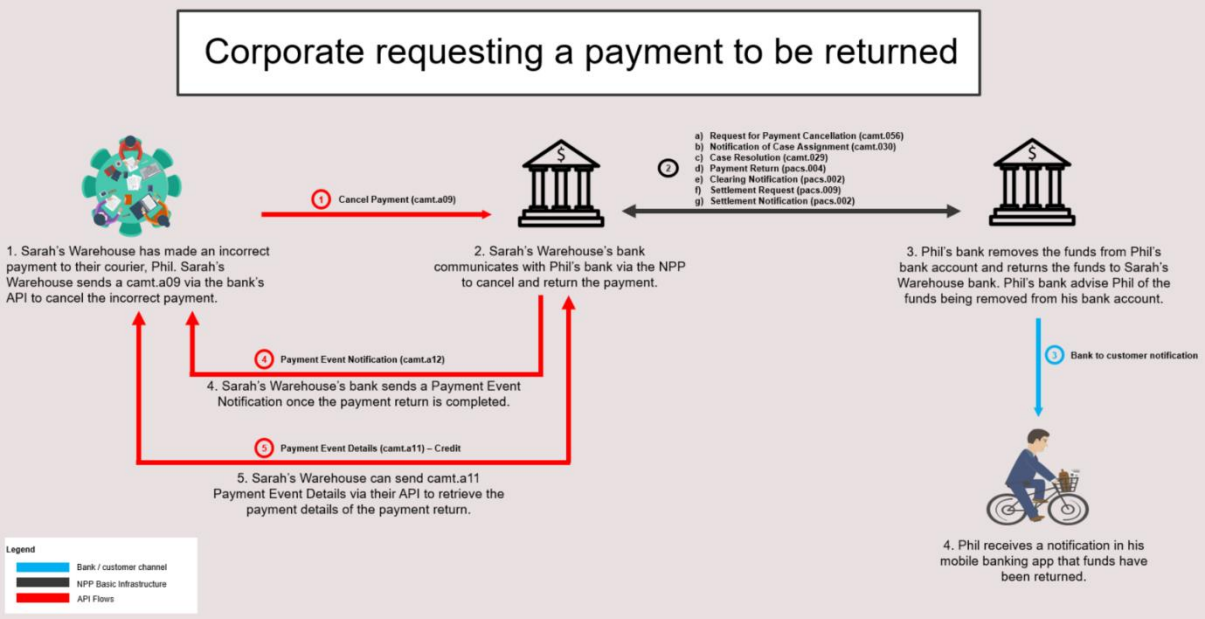

Figure 3. – API Usage Example

3.2.2 Payment Cancellation Flow Processing Steps

The sequence of processing steps in this section is provided as an illustrative example of a possible payment flow using APIs; it is a guide only and is not meant to be prescriptive.

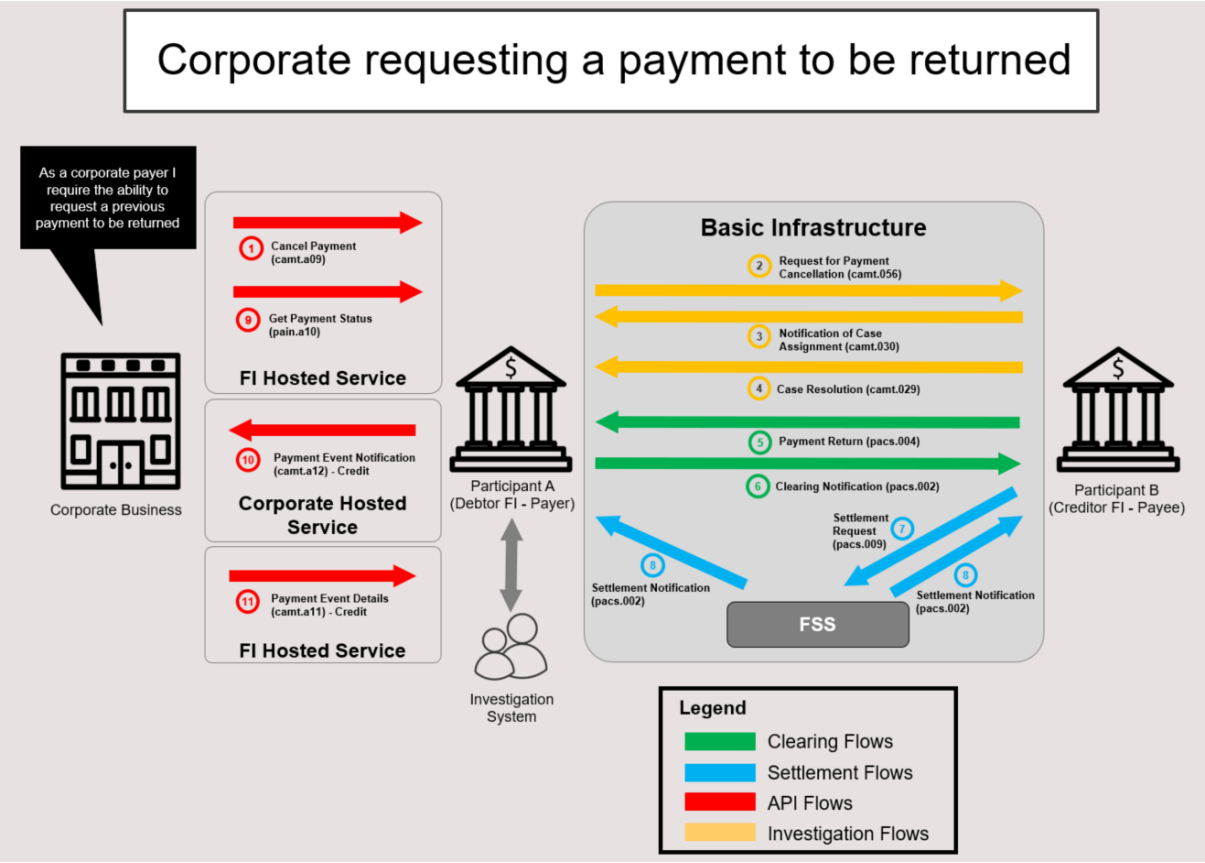

Figure 4. – Cancel Payment Use Case Overview

Step	Description
1	Payer business submits a request to return a previously submitted payment to its NPP Participant
2	Payer's NPP Participant creates a payment cancellation request with details from the cancel payment request which is routed via the NPP Basic Infrastructure to the Payee's NPP Participant
3-6	Payer NPP Participant and Payee NPP Participant will use Investigation messages and Payment return messages to process the Payment cancellation request
7-8	NPP Basic Infrastructure and FSS complete settlement processing and sends confirmations to NPP Participants
9	Payer's NPP Participant responds to Payment Status (successful / unsuccessful) to Payer

Step	Description
10	Payer NPP Participant to notify Payer's business of a payment event in their account.
11	Payer's business to fetch from the Payer's NPP Participant details of a credit to their account.

3.3 Return Payment

3.3.1 Usage Example

The figure below illustrates the example of a corporate returning an extra payment from their customer and the associated interactions and work flow that might take place:

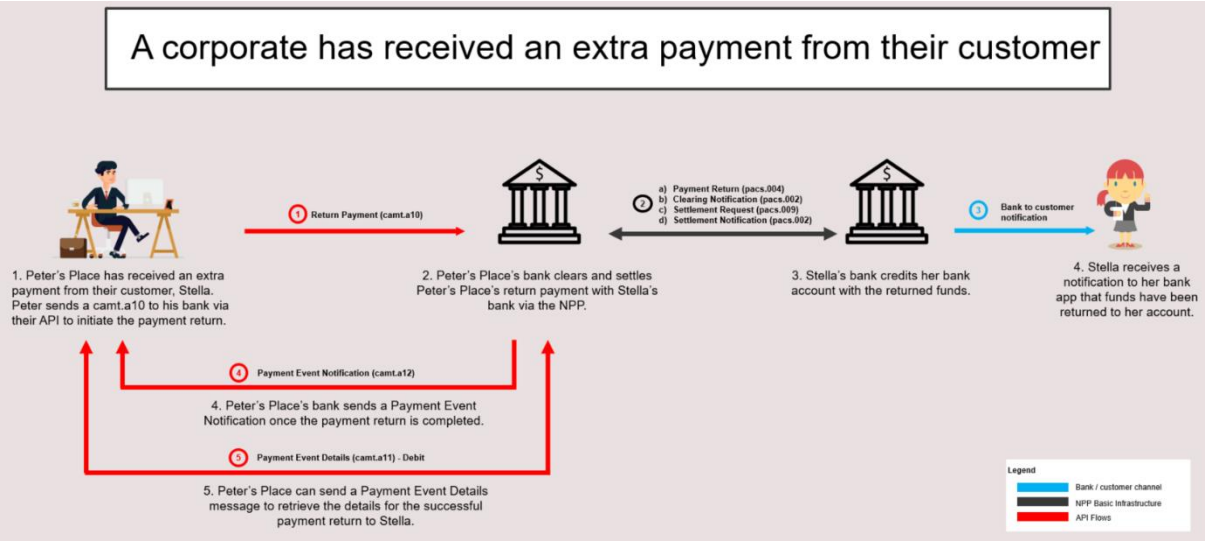

Figure 5. – API Usage Example

3.3.2 Return Payment Flow Processing Steps

The sequence of processing steps in this section is provided as an illustrative example of a possible payment flow using APIs; it is a guide only and is not meant to be prescriptive.

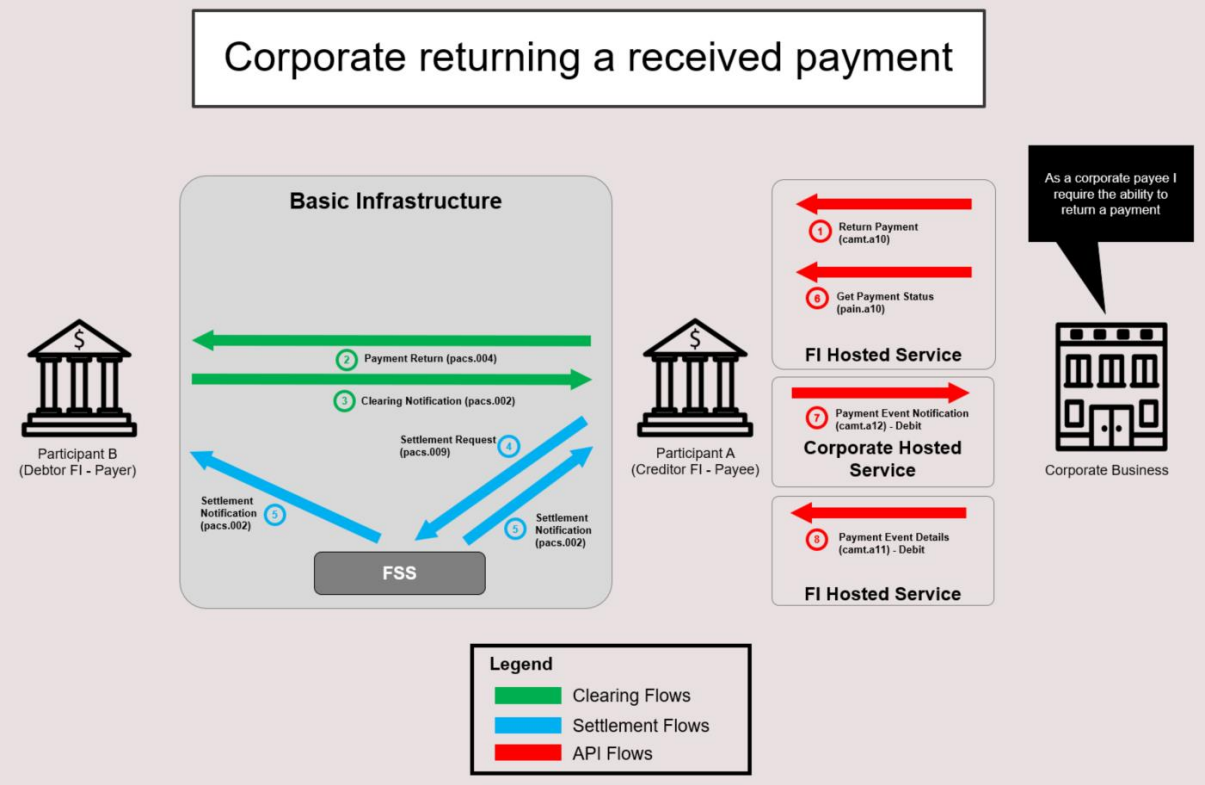

Figure 6. – Return Payment Use Case Overview

Step	Description
1	Payee business submits a request to return a previously received payment to its NPP Participant
2-3	Payee NPP Participant will send a Payment return request to the Payer NPP Participant and Payer NPP Participant will send a clearing notification for return to the Payee NPP Participant.
4-5	NPP Basic Infrastructure and FSS complete settlement processing and sends confirmations to NPP Participants
6	Payee's NPP Participant responds to Payment Status (successful / unsuccessful) to Payee
7	Payee NPP Participant to notify Payee's business of a payment event in their account.
8	Payee's business to fetch from the Payee's NPP Participant details of a debit from their account.

3.4 Sample API Documentation

The associated zip file “NPP API Sample Schema and Documents V2.0” contains the following for each of the 7 sample APIs and the NPP Swagger API Contracts (for Open API Specification 2.0 and 3.0):

- **API: pain.a09.001.02 (Submit Payment)**
  pain.a09.001.02.pdf
  pain.a09.001.02.xls
  pain.a09.001.02.schema.json

- **API: pain.a10.001.02 (Get Payment Status**)
  pain.a10.001.02.pdf
  pain.a10.001.02.xls
  pain.a10.001.02.schema.json

- **API: pain.a11.001.03 (Get Account Servicer by Alias)**
  pain.a11.001.03.pdf
  pain.a11.001.03.xls
  pain.a11.001.03.schema.json

- **API: camt.a09.001.01 (Cancel Payment)**
  camt.a09.001.01.pdf
  camt.a09.001.01.xls
  camt.a09.001.01.schema.json

- **API: camt.a10.001.01 (Return Payment**)
  camt.a10.001.01.pdf
  camt.a10.001.01.xls
  camt.a10.001.01.schema.json

- **API: camt.a11.001.01 (Payment Event Details)**
  camt.a11.001.01.pdf
  camt.a11.001.01.xls
  camt.a11.001.01.schema.json

- **API: camt.a12.001.01 (Payment Event Notification)**
  camt.a12.001.01.pdf
  camt.a12.001.01.xls
  camt.a12.001.01.schema.json

- **SwaggerHub Contract (OAS 2.0)**
  NPP-API-iso-20022_npp_api-1.0.2-swagger.json
  NPP-API-iso-20022_npp_api-1.0.2-swagger.yaml
  NPP-API-iso-20022_npp_api_notification-1.0.1-swagger.json
  NPP-API-iso-20022_npp_api_notification-1.0.1-swagger.yaml

- **SwaggerHub Contract (OAS 3.0)**
  NPP-API-iso-20022_npp_api-1.0.2-oas3-swagger.json
  NPP-API-iso-20022_npp_api-1.0.2-oas3-swagger.yaml
  NPP-API-iso-20022_npp_api_notification-1.0.1-oas3-swagger.json
  NPP-API-iso-20022_npp_api_notification-1.0.1-oas3-swagger.yaml