

1 Promise to Pay Data Contract

1.1 Overview

This section provides information on the flow of data between the Promise to Pay product and the web service. The purpose of this REST-ful web service is to allow a back-end system to store information that the caller has acknowledged to pay their bill.

1.2 Request

Requests are made using HTTP POST requests, which pass the required caller details to the web service. For the Promise to Pay product, the caller is identified using an Identification Module at the start of the callflow. The following parameters are then passed into the automatic query web service:

Table 1 Promise to Pay web service request parameters

Parameter	Description	Example Values
AccountNumber	An arbitrary value retrieved by the Identification Module, to be passed to the web service to retrieve caller's account details and perform the task.	"ABC12345", "123456789"

A typical URL might look like this:

```
http://localhost:8080/fish-services/test/PromiseToPay.jsp
```

And a typical HTTP POST body might look like this:

```
cli=02890571100&dnis=7896&sessionid=1234%2D3AAF%2D3372&AccountNumber=12345678
```

1.3 Response

The XML response specifies the overall status of the lookup, i.e. "success", or some other return code such as "agent", and provides a mechanism to set arbitrary variables in the call session.

When specifying variables in the response, you can cause some or all key-value pairs to be attached to the call via the CTI (where the platform supports it) by including an optional "attach" attribute with a value of "true" or to set them as the CLI data by including an optional "remember" attribute with a value of "true".

A typical XML response looks like this:

```
<promiseToPayResults>
  <status>success</status>
</promiseToPayResults>
```

An promise to pay request attempt where the caller's account is prohibited might look like this:

```
<promiseToPayResults>
  <status>failure</status>
</promiseToPayResults>
```

A promise to pay request attempt where an error occurs might look like this:

```
<promiseToPayResults>
  <status>error</status>
</promiseToPayResults>
```



HTTP response codes other than "200" will be treated as an error.

1.3.1 Statuses

The <status> element is the only mandatory element. The following statuses can be returned:

- “success” – indicates that the query was successful
- ‘failure’ – this status would indicate that the promise to pay request was not possible for the caller
- ‘error’ – the application’s error handling path will be followed in the callflow
- any other status – causes the Module to return with that status.