mojaloop

The Duplicate Problem: where are we now?

What's the duplicate problem?

- It's a problem about identifying the DFSP to send a payment request to.
- Mojaloop does this by:
  - Asking the sending customer for some information about the entity they want to pay.
  - Identifying a DFSP who recognizes that piece of information.
  - Asking the DFSP directly if it will commit to crediting one of its customers based on the information.
- The problem arises when more than one DFSP recognizes that piece of information:
  - Which DFSP should the payment be addressed to?
- What happens if one DFSP could map that information onto more than one account?
  - Mojaloop says that's an issue for the DFSP, not Mojaloop.

Examples of the duplicate problem

- A customer joins more than one Mobile Money System
- A customer has an account at a Mobile Money System and an account at a bank where she has registered her MSISDN as an alias.
- A customer has an account at one Mobile Money System; her husband, with whom she shares a phone, has an account at a different Mobile Money System.
- A DFSP registers a merchant and gives it a merchant code; another DFSP registers a different merchant and gives it the same merchant code.
- Two banks use the same core banking system, and therefore the same account numbering system.

What circumstances are we talking about?

	One entity	Multiple entities
One DFSP	Unique match	Should never happen
Multiple DFSPs	Customer has an account at multiple DFSPs Or… Multiple customers are using the same identifier	Identifier is non-unique Merchant IDs, Account numbers

Who can solve the problem?

- The payer DFSP (or its customer) must solve the problem.
  - The switch can help by presenting the alternatives.
  - The ITK can shoulder some of the responsibility.
- First, some UIs require their users to select the target DFSP in advance.
  - Where this is the case, a payer DFSP can simply route the request directly to the DFSP selected.
- Otherwise:

Collecting and ordering duplicates

- If the oracle returns multiple matches for an identifier:
  - Send a resolution request to each matching DFSP
  - Discard any negative responses.
  - Wait till all responses have been received.
- If there are still duplicates
  - Apply ordering algorithms
- Return all matches to the selection component.

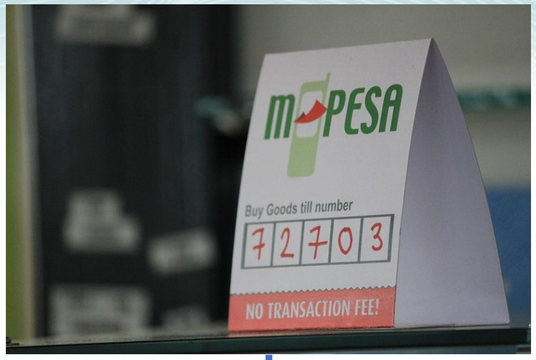
Ordering algorithms

- We can imagine the use of one or more ordering algorithms.
- For instance:
  - Order in descending date of registration.
  - Order according to an externally expressed preference.
- The purpose is to ensure that the most likely identification is received first.
- Ordering algorithms should be specific to oracles.
- They should be like oracles:
  - Mojaloop provides templates.
  - Implementers can use or modify the templates, or construct their own.

The selection component

- Resides either at the switch or at the DFSP.
- Manages choices relating to duplicate identifiers.
- Allows a DFSP to specify whether it will accept multiple address resolution responses or not.
- If a DFSP accepts multiple responses:
  - Return the responses to the DFSP
- Otherwise:
  - Return the first response to the DFSP.
  - If the DFSP asks again, return the next response to the DFSP.

Example 1: different identities



ID	DFSP
72703	M-Pesa
72703	Airtel

Payer DFSP

Who owns Merchant ID 72703?

Mojaloop Switch

Merchant ID Oracle

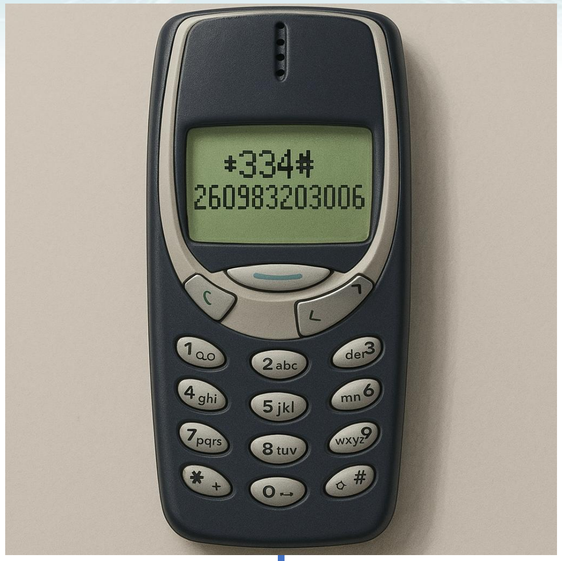
Name:
Miller's Burgers
Currency: KES

Name:
Matt's Pizza
Currency: KES

M-Pesa

Airtel

Example 2: different accounts



ID	DFSP
260983203006	M-Pesa
260983203006	Airtel

Payer DFSP

Who owns MSISDN 260983203006?

Mojaloop Switch

MSISDN Oracle

Name: Rebeka Nyanjega Currency: KES

M-Pesa

Name: Rebeka Nyanjega Currency: KES

Airtel

Tracking and storing preferences

- Under the current structures, there is no way of:
  - Tying a request for the agreement of terms to the address resolution request which (may have) preceded it.
  - Understanding which customer issued an address resolution request.
- If we could do that, we could build an *ad hoc* system for tracking customer preferences in relation to accounts.

How might this work?

- Using ISO 20022, we could add an endpoint which included an acmt.023 as its body.
  - This message allows the payer DFSP to identify the customer who is asking for the resolution of the address.
  - We would need to extend the FSPIOP API correspondingly.
- When we receive a request for the agreement of terms, this makes an assertion of the form: “For customer *x*, DFSP *y* is the correct assignment of identifier *z*.
- If we store this assertion, we could use it as input to the selection component.
  - This increases the chances of the user getting the right DFSP first time…
  - … while allowing them to change their mind if they want to

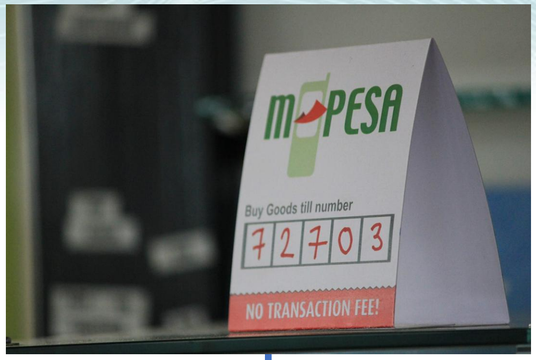
Who signs?

- At present, the creditor DFSP signs its response.
  - This means: “I warrant that I hold an account corresponding with the identifier you sent me, and that it is able to receive funds.”
- But now we have multiple responses, and only one space for a non-repudiation signature.
- Possible solutions:
  - The switch signs the composite response.
  - The switch composes the response, and adds the signatures received from the creditor parties to the **FSPIOP-Signature** field in the message header as well as signing the whole message.
  - For ISO 20022 messages, the Business Application Header can include multiple copies of the *Related* element.
    - Each *Related* element is a Business Application Header with a signature
    - So we can include the Business Application Headers of the component messages, and their signatures.

How does the payer choose?

- The switch returns multiple responses in an agreed format…
- … but these can only be actioned by the user interface of the sender's DFSP.
- So:
  - *Either* the sender's DFSP will have to modify its UI to allow the choice to be made.
  - *Or* the selection component feeds the results to the DFSP one at a time.
    - If the DFSP re-sends a request for the same identifier, the selection component:
      - *Either* responds with the next item in its list….
      - … Or returns a *Not Found* if there are no more items.

Using the selection component



Merchant ID Oracle

ID	DFSP
72703	M-Pesa
72703	Airtel

Payer DFSP

Who owns Merchant ID 72703?

Please agree terms for payment to Miller's Burgers

Store preference for this customer

Mojaloop Switch

Compose response

Name: Miller's Burgers Currency: KES

M-Pesa

Name: Matt's Pizza Currency: KES

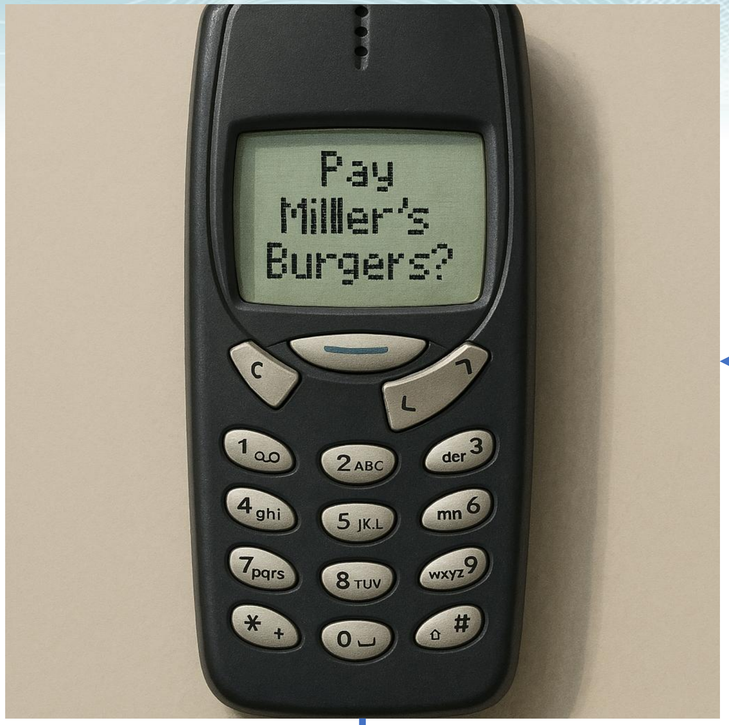
Airtel

DFSP: M-Pesa Name: Miller's Burgers

Selection component

Composite response: Name: 1) Miller's Burgers 2) Matt's Pizza Currency: KES DFSP: 1) M-Pesa, 2) Airtel

Re-using the selection component



Is this the correct name?

No, it's not

Payer DFSP

Who owns Merchant ID 72703?

Selection component

DFSP: Airtel
Name: Matt's Pizza

Discussion