mojaloop

Crossing borders

Currency conversion and cross-instance transfers in Mojaloop

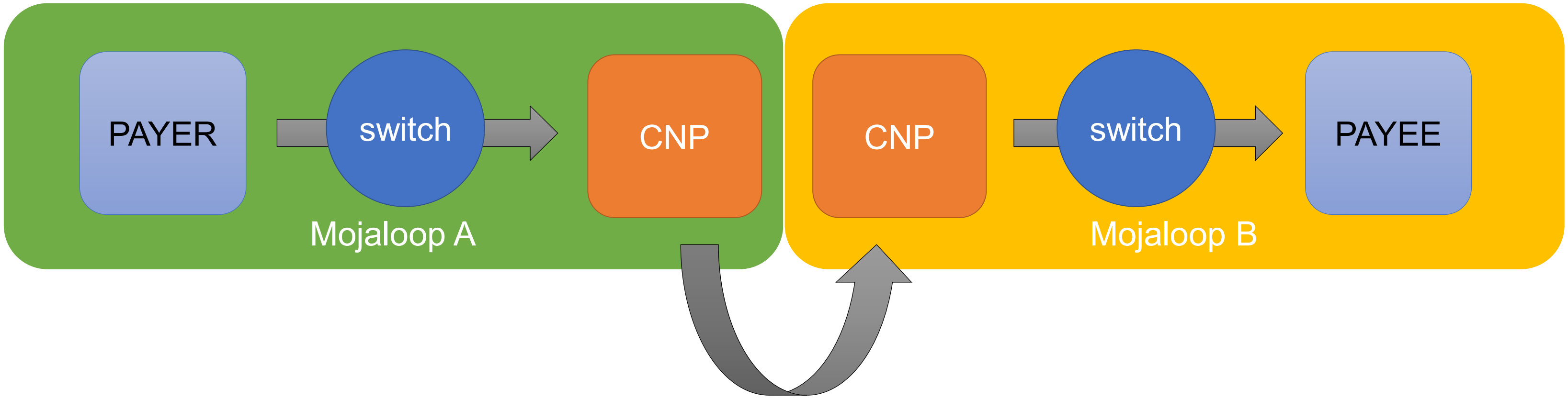
The story so far…

Updates

- Workshop in London – 29 and 30 October
- Scoped work to “Payer Initiated” generic pattern
- Designed flows for lookup, quote and transaction phases
- Define detailed API changes required to support use cases

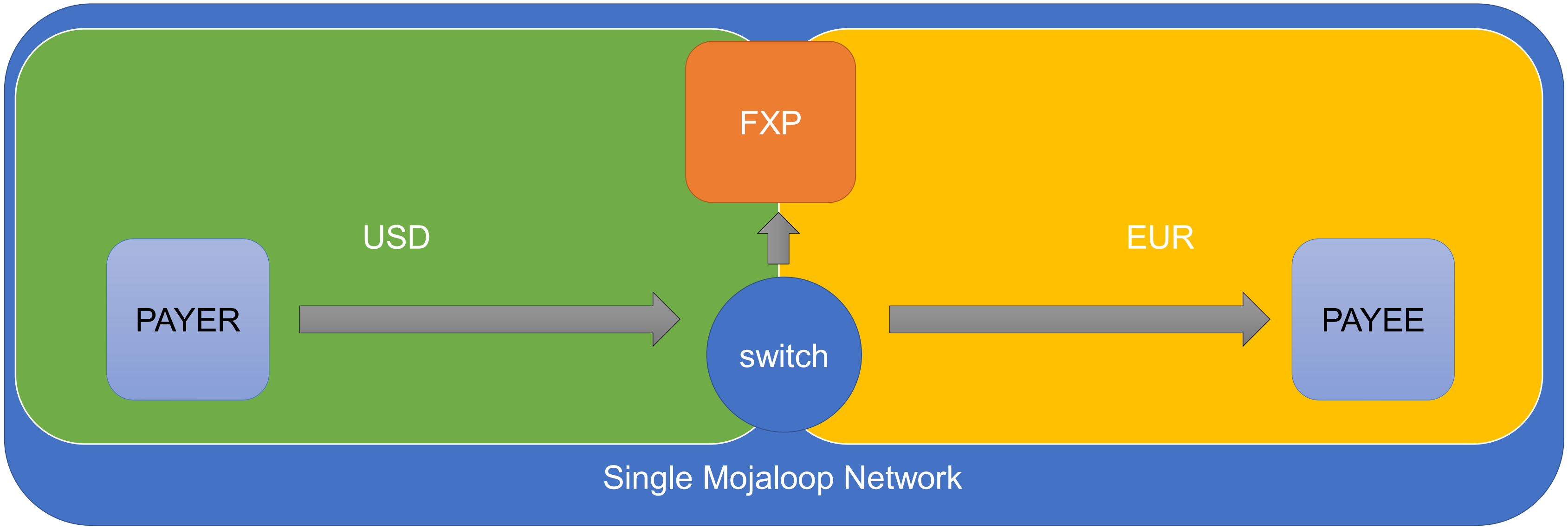
Cross-Network Providers

- A cross-network provider is split between two physical networks
- Accepts a ***transfer*** on network A and makes a ***transfer*** on network B
- This is a single ***transaction***



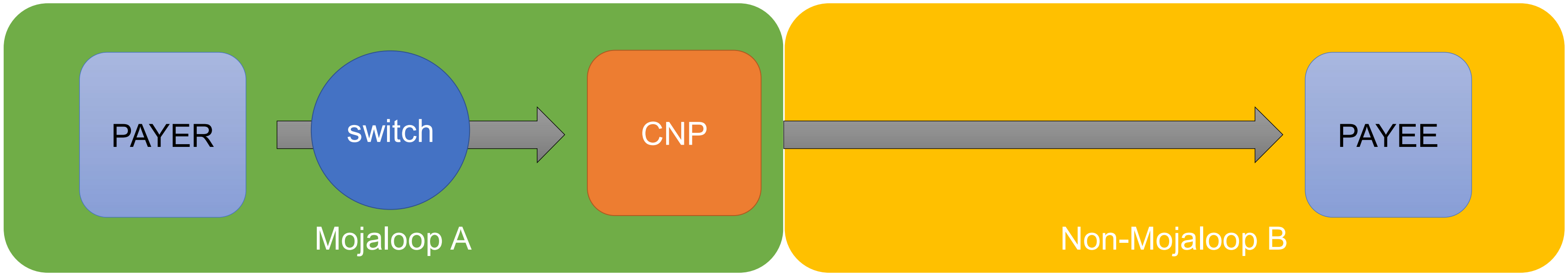
Foreign Exchange Provider

- An FXP facilitates FX inside the same network

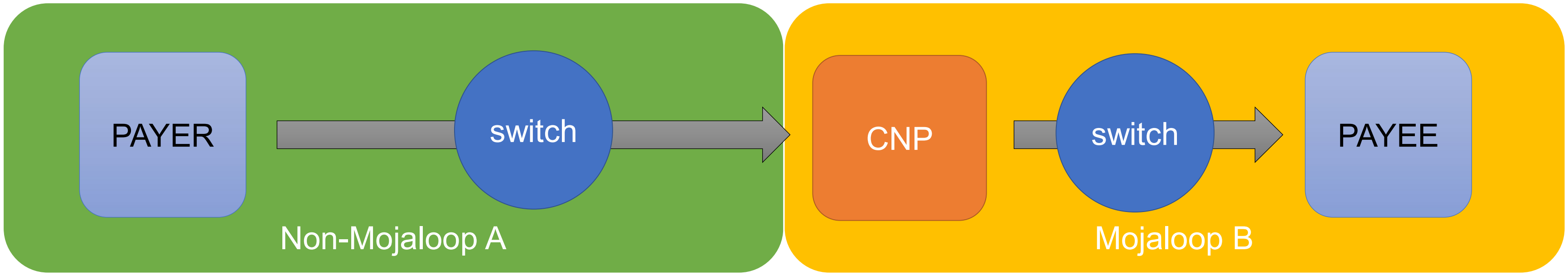


Connecting to non-Mojaloop networks

- Payer is in a Mojaloop network



- Payee is in a Mojaloop network



Lookup Phase

- Two deployment scenarios we need to consider
  1. Switch provides a list of CNPs to payer FSP
     1. Payer FSP sends party lookup to switch
     2. Switch identifies party as being “out of network”
     3. Switch sends list of all CNPs to payer FSP
     4. Payer FSP sends a lookup request to each CNP
  2. Switch interrogates CNPs directly to Lookup Party Information
     1. Payer FSP sends party lookup to switch
     2. Switch identifies party as being “out of network”
     3. Switch forwards lookup to each CNP
     4. Switch returns aggregated result back to Payer FSP
- Party data model supplemented with account info (see next slide)

Party (part of Lookup response)

<table>
<tr><th>Data Element</th><th>Cardinality</th><th>Type</th><th>Description</th></tr>
<tr><td>partyIdInfo</td><td>1</td><td><table><tr><th>Name</th><th>Description</th></tr><tr><td>partyIdType</td><td>The type of the identifier.</td></tr><tr><td>partyIdentifier</td><td>An identifier for the Party.</td></tr><tr><td>partySubIdOrType</td><td>A sub-identifier or sub-type for the Party.</td></tr><tr><td>fspId</td><td>The FSP ID (if known)</td></tr></table></td><td>Party Id type, id, sub ID or type, and FSP Id.</td></tr>
<tr><td>merchantClassificationCode</td><td>0..1</td><td>ntClassificationCode</td><td>Used in the context of Payee Information, where the Payee happens to be a merchant.</td></tr>
<tr><td>name</td><td>0..1</td><td></td><td>Display name of the Party, could be a real name or nickname.</td></tr>
<tr><td>personalInfo</td><td>0..1</td><td></td><td>Personal information used to verify identity of Party such as name and date of birth.</td></tr>
<tr><td>accounts</td><td>0..1</td><td><table><tr><td>currency</td><td>The currency of the account</td></tr><tr><td>address</td><td>The address of the account.</td></tr></table></td><td>A list of accounts that can accept transfers for the party.</td></tr>
</table>

This is the identifier of the final destination FSP. It is different to the identifier of the destination participant in the message header

Observations

NOTE: A single MSISDN could resolve to accounts at multiple FSPs…

- We need to support multiple parties per lookup response (a list)
- Can we use BIC codes or similar to identify FSPs across networks? (Being explored)

Quote Phase

- Accommodate non-Mojaloop integrations for payments delivered outside Mojaloop ecosystem
  - Add optional “maxValueDate” to quote request
  - Add “valueDate” to quote response
- Add a list of Participants to share rates/fees and compliance data
- Add “account address” to ensure the quote is routed to the correct account
- Add transaction object and echo data to response to accommodate deprecating OER encoded packet

Quote

Name		Type	Description
quoteId	1	CorrelationId	Common ID between the FSPs for the quote object, decided by the Payer FSP.
transactionId	1	CorrelationId	Common ID between the FSPs for the future transaction object.
transactionRequestId	0..1	CorrelationId	Identifies an optional previously-sent transaction request.
payee	1	Party	Information about the Payee in the proposed financial transaction.
payer	1	Party	Information about the Payer in the proposed financial transaction.
amountType	1	AmountType	SEND for send amount, RECEIVE for receive amount.
amount	1	Money	The amount the Payer would like to send or the amount the Payee should receive.
fees	0..1	Money	Fees in the transaction.
transactionType	1	TransactionType	Type of transaction for which the quote is requested.
geoCode	0..1	GeoCode	Longitude and Latitude of the initiating Party. Can be used to detect fraud.
note	0..1	Note	A memo that will be attached to the transaction.
expiration	0..1	DateTime	Expiration is optional.
accountAddress	0..1	AccountAddress	The address of the payee account
participants	0..16	Participant	The participants in the transaction.
maxValueDate	0..1	DateTime	The maximum Value Date for this transaction to clear in the payee’s account
extensionList	0..1	ExtensionList	Optional extension, specific to deployment.

Quote (Response)

Name		Type	Description
transferAmount	1	Money	The amount of Money that the Payer FSP should transfer to the Payee FSP.
payeeReceiveAmount	0..1	Money	The amount of Money that the Payee should receive in the end-to-end transaction.
payeeFspFee	0..1	Money	Payee FSP’s part of the transaction fee.
payeeFspCommission	0..1	Money	Transaction commission from the Payee FSP.
expiration	1	DateTime	Date and time until when the quotation is valid and can be honored when used in the subsequent transaction.
geoCode	0..1	GeoCode	Longitude and Latitude of the Payee. Can be used to detect fraud.
transaction	1	Transaction	The end-to-end transaction.
echoData	0..1	String(1..2048)	Opaque data provided by the payee that must be echoed back unchanged in the transfer.
condition	1	IlpCondition	The condition that must be attached to the transfer by the Payer.
participants	0..16	Participant	The participants in the transaction.
extensionList	0..1	ExtensionList	Optional extension, specific to deployment.

Participant

Name	Cardinality	Type	Description
fspId	1	FspId	The identity of the participant
transferCurrency	1	Currency	The currency of the transfer that will be made by this participant.
fee	1	Money	The fee that will be charged by the participant.
rate	0..1	Amount	The rate of exchange that will applied by this participant.
expiration	1	DateTime	Date and time until when the quotation is valid and can be honored when used in the subsequent transaction.
dataRequired	0..32	String(1..128)	List of required
dataProvided	0..32	EncryptedData	The data provided by other participants for use by this participant.

Optimizations

- If the CNP rates are known/published the switch can pre-select the best CNP for a quote

Challenges / Next Steps

- Lack of real-world examples to test design against
- Need more input from implementors
- Need to focus on non-Mojaloop integration until Mojaloop is more ubiquitous

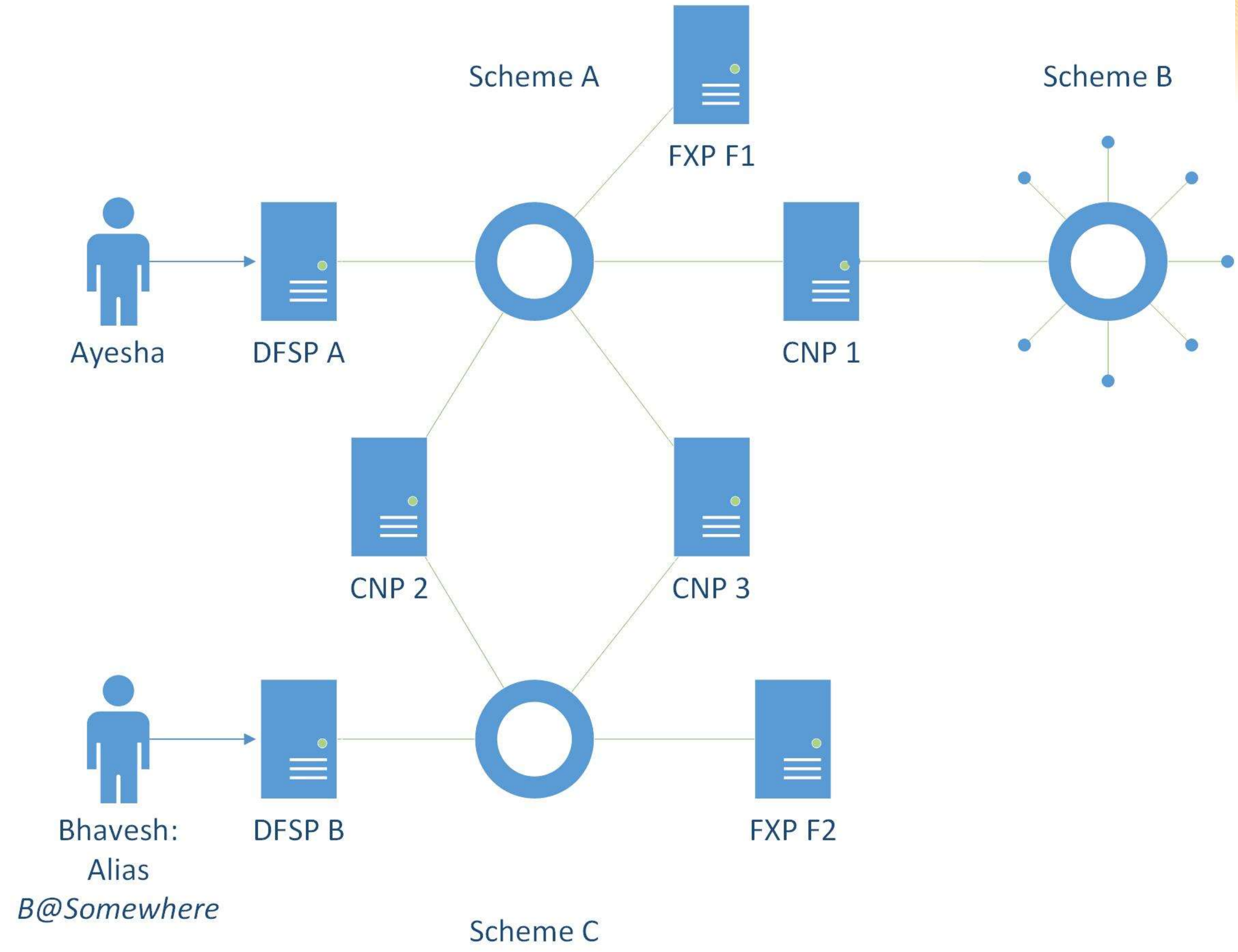
Looking in detail

The discovery phase in a cross-network environment

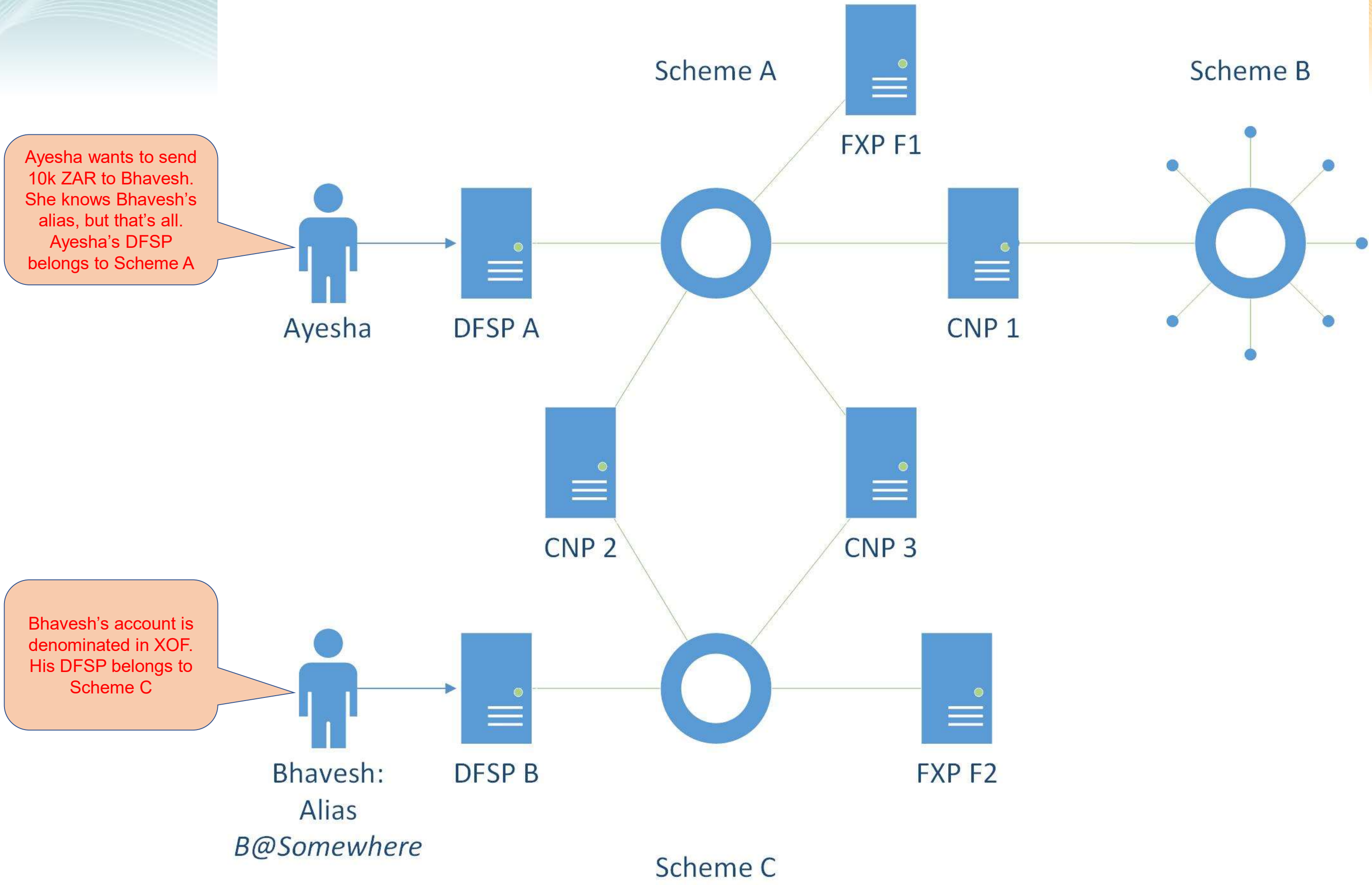
**“The World was all before them, where to choose
Thir place of rest, and Providence thir guide:
They hand in hand with wandring steps and slow,
Through Eden took thir solitarie way.”**

John Milton,
Paradise Lost, XII, 646-9

An example cross-instance context



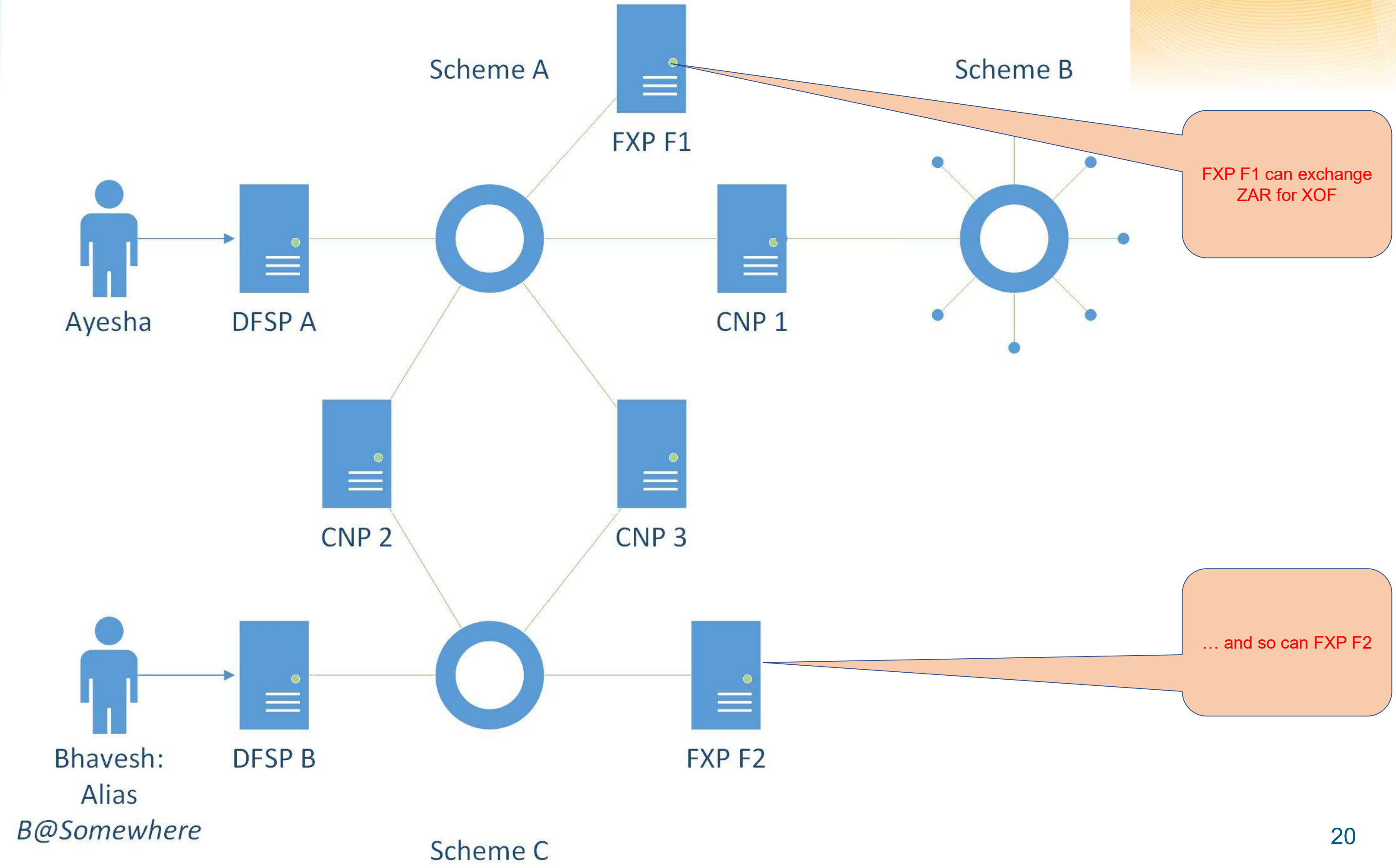
An example cross-instance context



Ayesha wants to send 10k ZAR to Bhavesh. She knows Bhavesh's alias, but that's all. Ayesha's DFSP belongs to Scheme A

Bhavesh's account is denominated in XOF. His DFSP belongs to Scheme C

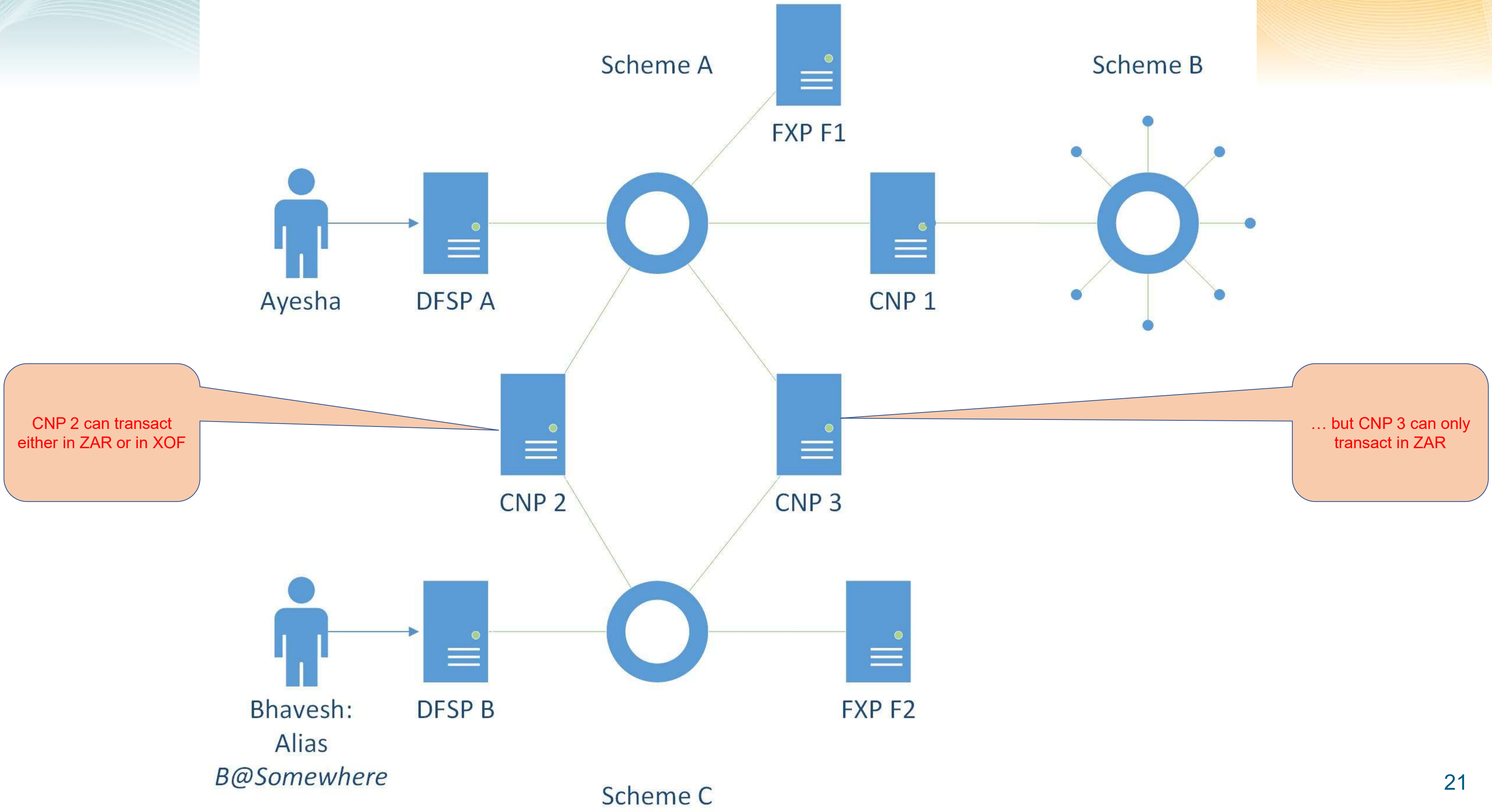
An example cross-instance context



FXP F1 can exchange ZAR for XOF

… and so can FXP F2

An example cross-instance context

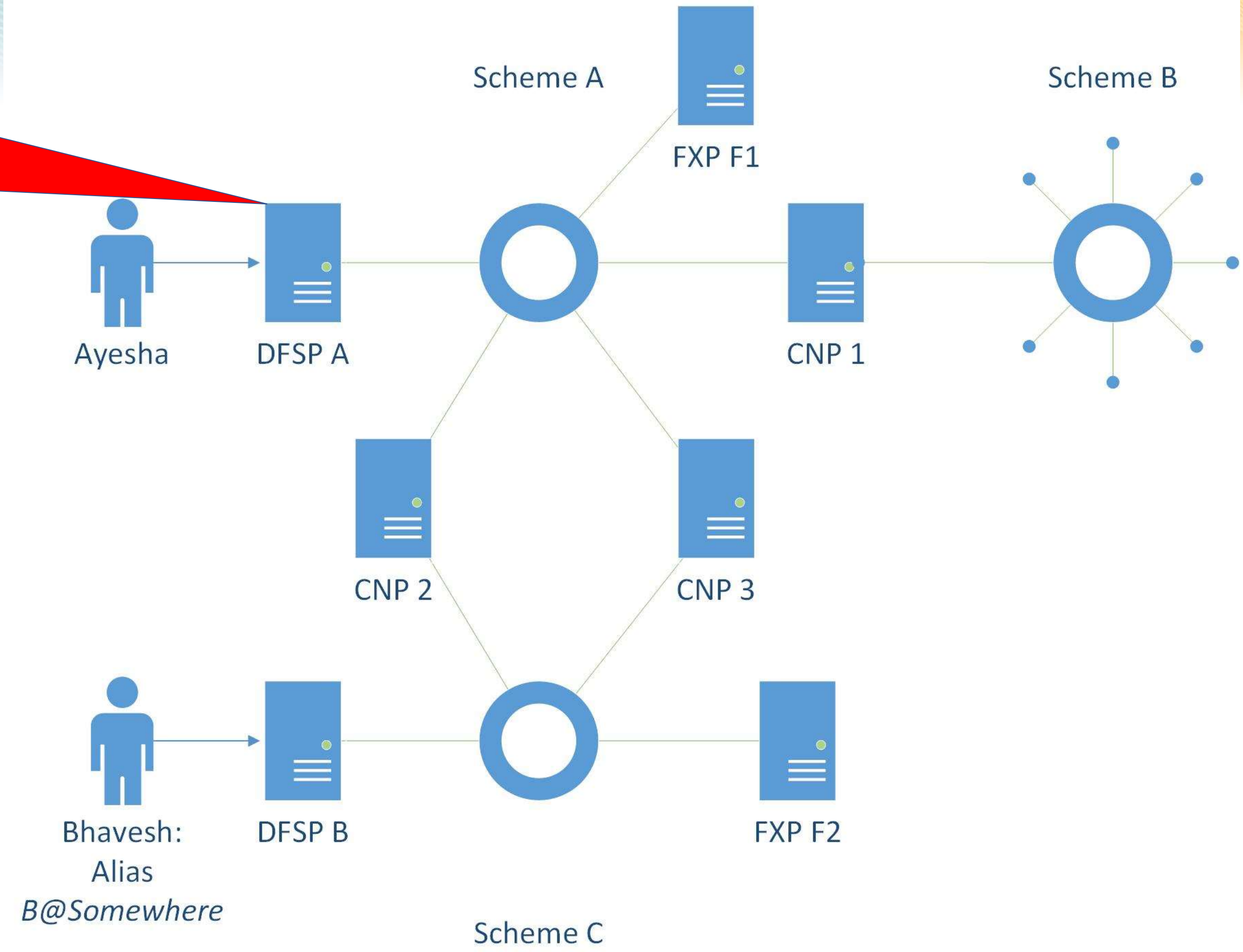


CNP 2 can transact either in ZAR or in XOF

… but CNP 3 can only transact in ZAR

Constructing the transfer routes

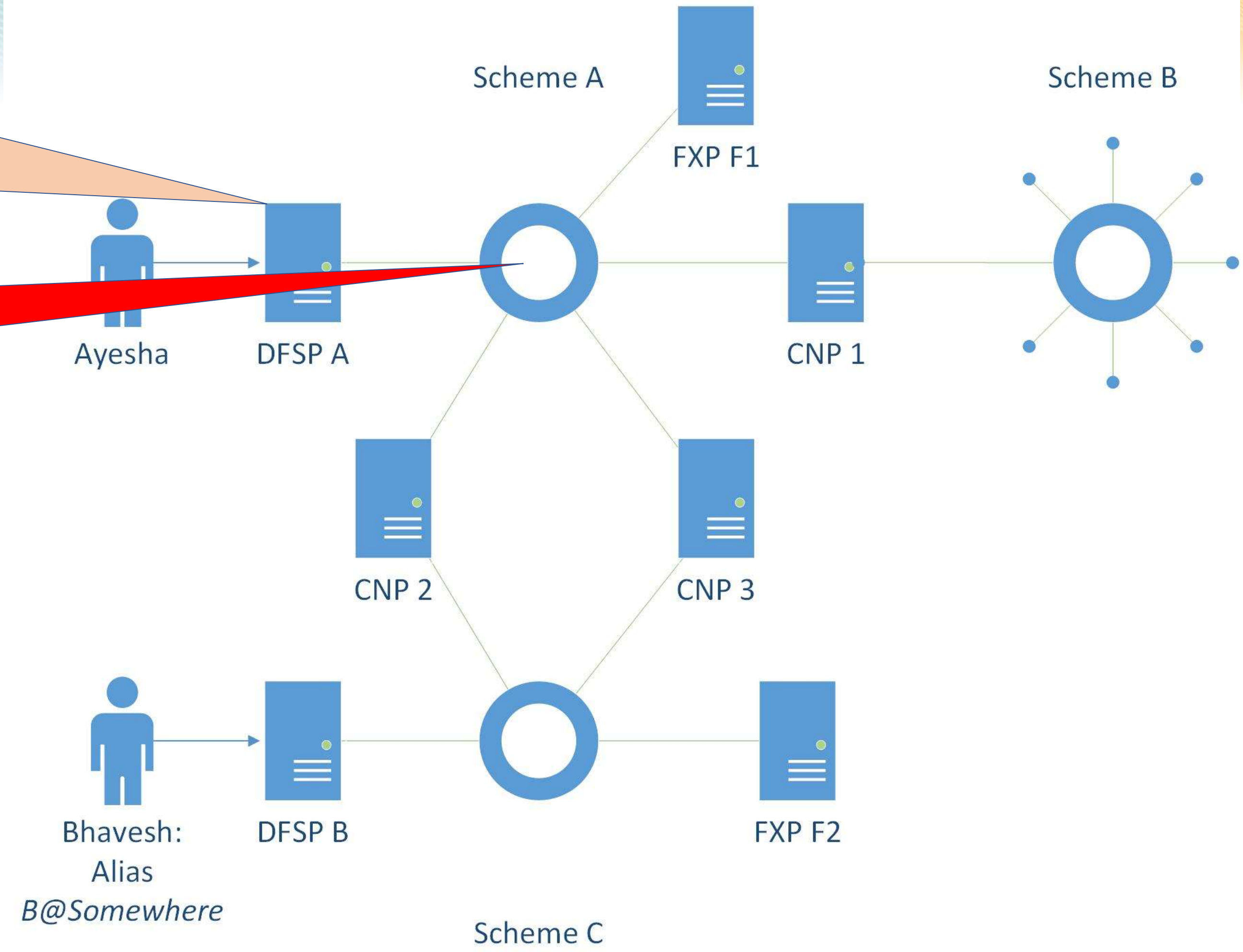
Ayesha starts with a request in the normal way. Her DFSP adds the currency.



Constructing the transfer routes

Ayesha starts with a request in the normal way. Her DFSP adds the currency

Scheme A's switch asks the ALS if it knows about Bhavesh's alias. It doesn't, so the switch asks the ALS if it knows about any CNPs



Digression 1: cross-network service advertisement

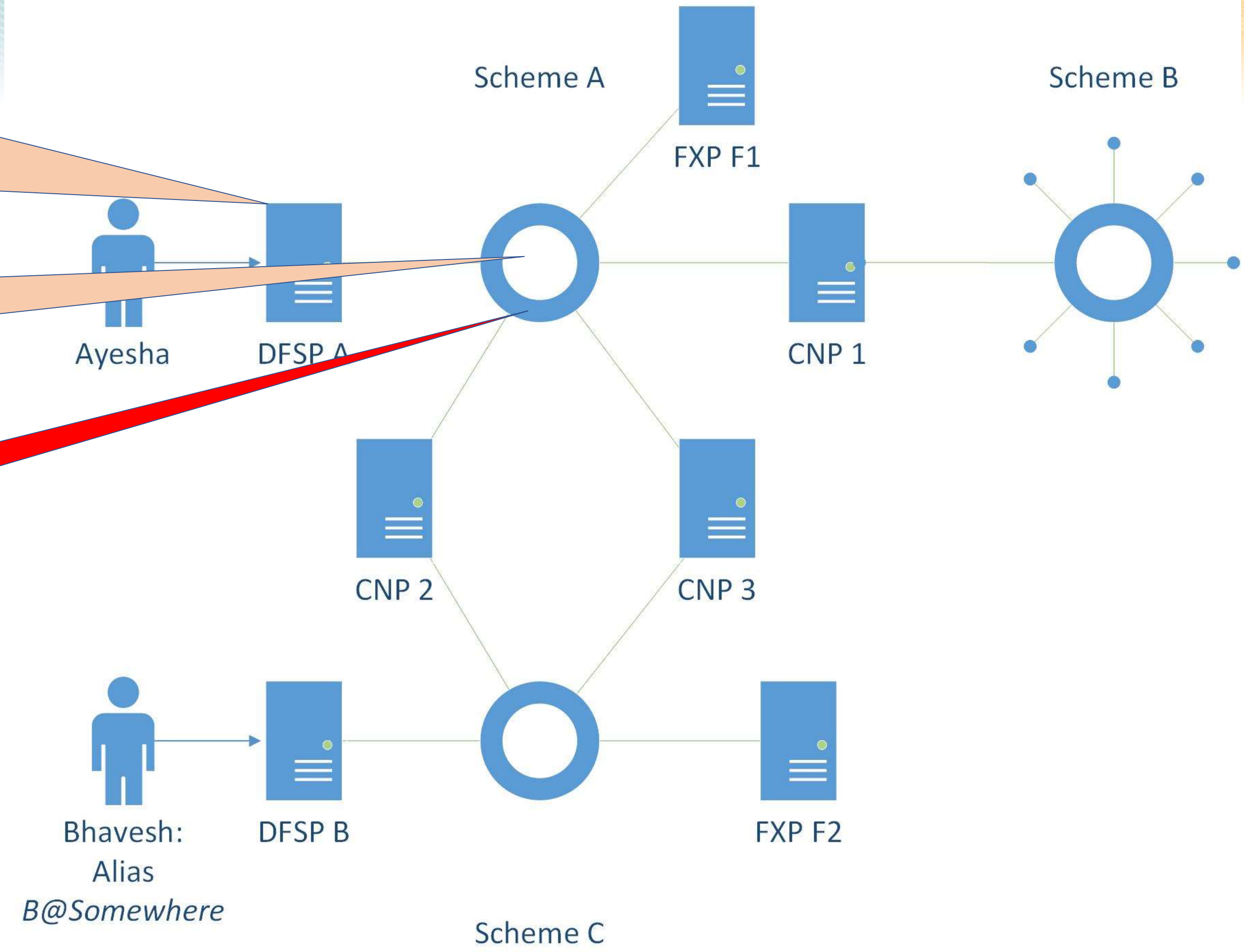
- We envisage two services: FXP and CNP
- A DFSP can provide either or both services, and can also be an account-holding participant in the scheme
- A DFSP advertises its services to the scheme using the **/participants** resource, with an identifier type **FXP** or **CNP**
- The switch can request the names of all DFSPs who provide a given service using the **GET /parties** resource and the required type (e.g. **GET /parties/CNP**)
- The ALS will return the DFSP ids of all the participants who have advertised their participation in the requested service

Constructing the transfer routes

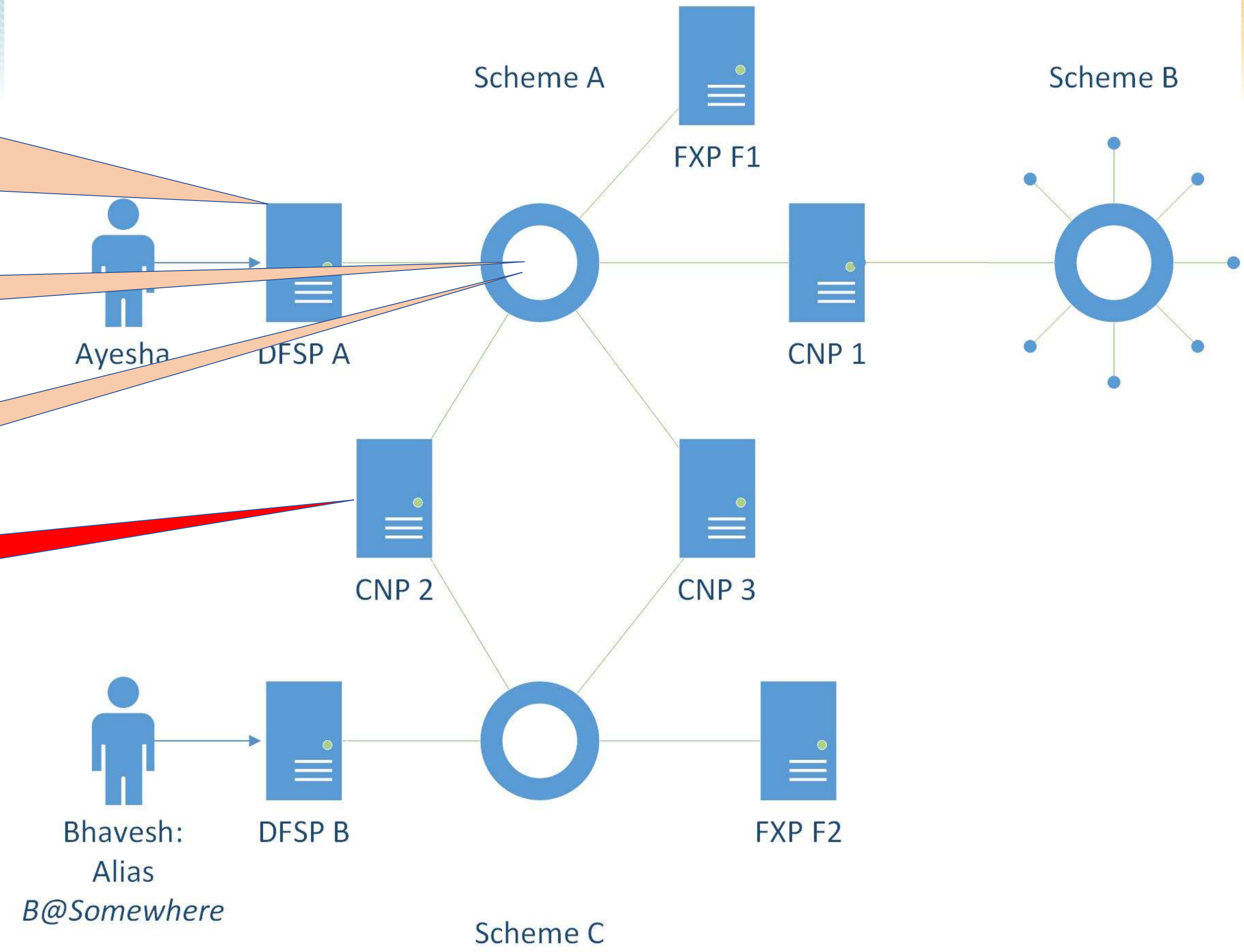
Ayesha starts with a request in the normal way. Her DFSP adds the currency

Scheme A’s switch asks the ALS if it knows about Bhavesh’s alias. It doesn’tALS if it knows about any CNPs.

So the switch asks the ALS for the names of all its CNP providers. There are 3



Constructing the transfer routes



Ayesha starts with a request in the normal way. Her DFSP adds the currency

Scheme A's switch asks the ALS if it knows about Bhavesh's alias. It doesn't.

So the switch asks the ALS for the names of all its CNP providers. There are 3

The switch forwards the request to CNP 2

Constructing the transfer routes

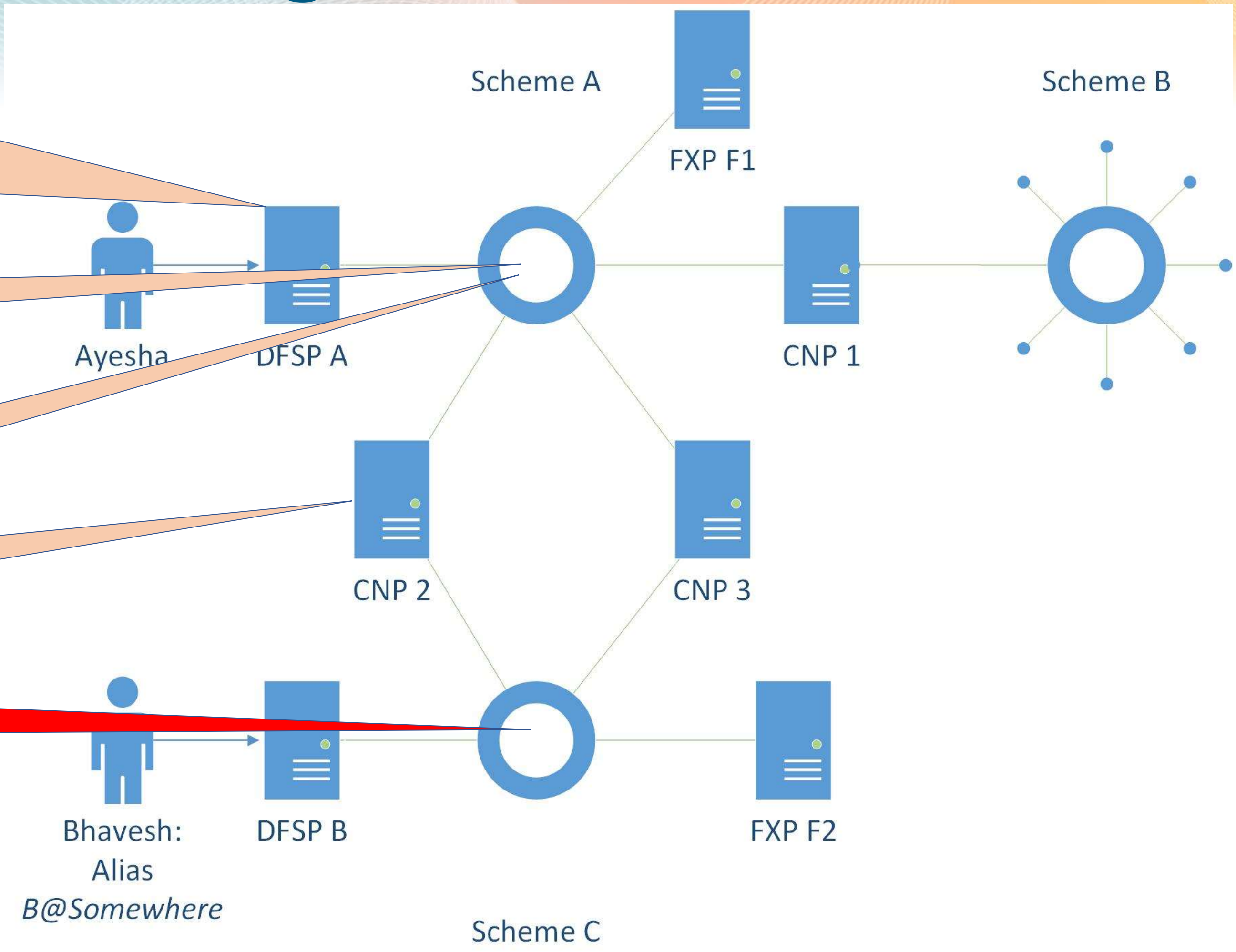
Ayesha starts with a request in the normal way. Her DFSP adds the currency

Scheme A's switch asks the ALS if it knows about Bhavesh's alias. It doesn't.

So the switch asks the ALS for the names of all its CNP providers. There are 3

The switch forwards the request to CNP 2

… and CNP 2 asks the switch in Scheme 3 if it recognises Bhavesh's alias.



Constructing the transfer routes

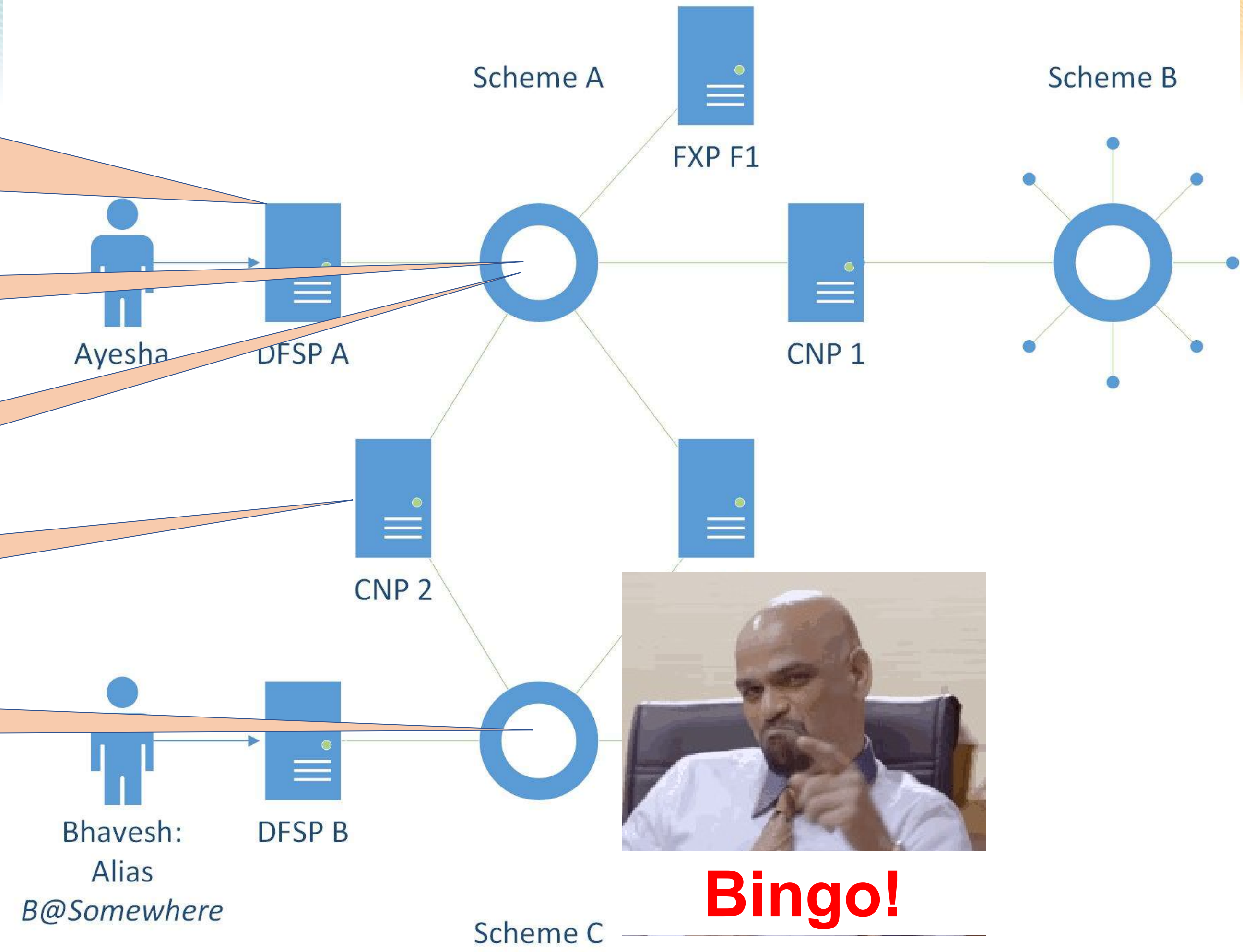
Ayesha starts with a request in the normal way. Her DFSP adds the currency

Scheme A's switch asks the ALS if it knows about Bhavesh's alias. It doesn't.

So the switch asks the ALS for the names of all its CNP providers. There are 3

The switch forwards the request to CNP 2

… and CNP 2 asks the switch in Scheme 3 if it recognises Bhavesh's alias.



Bingo!

Digression 2:
How does the DFSP respond?

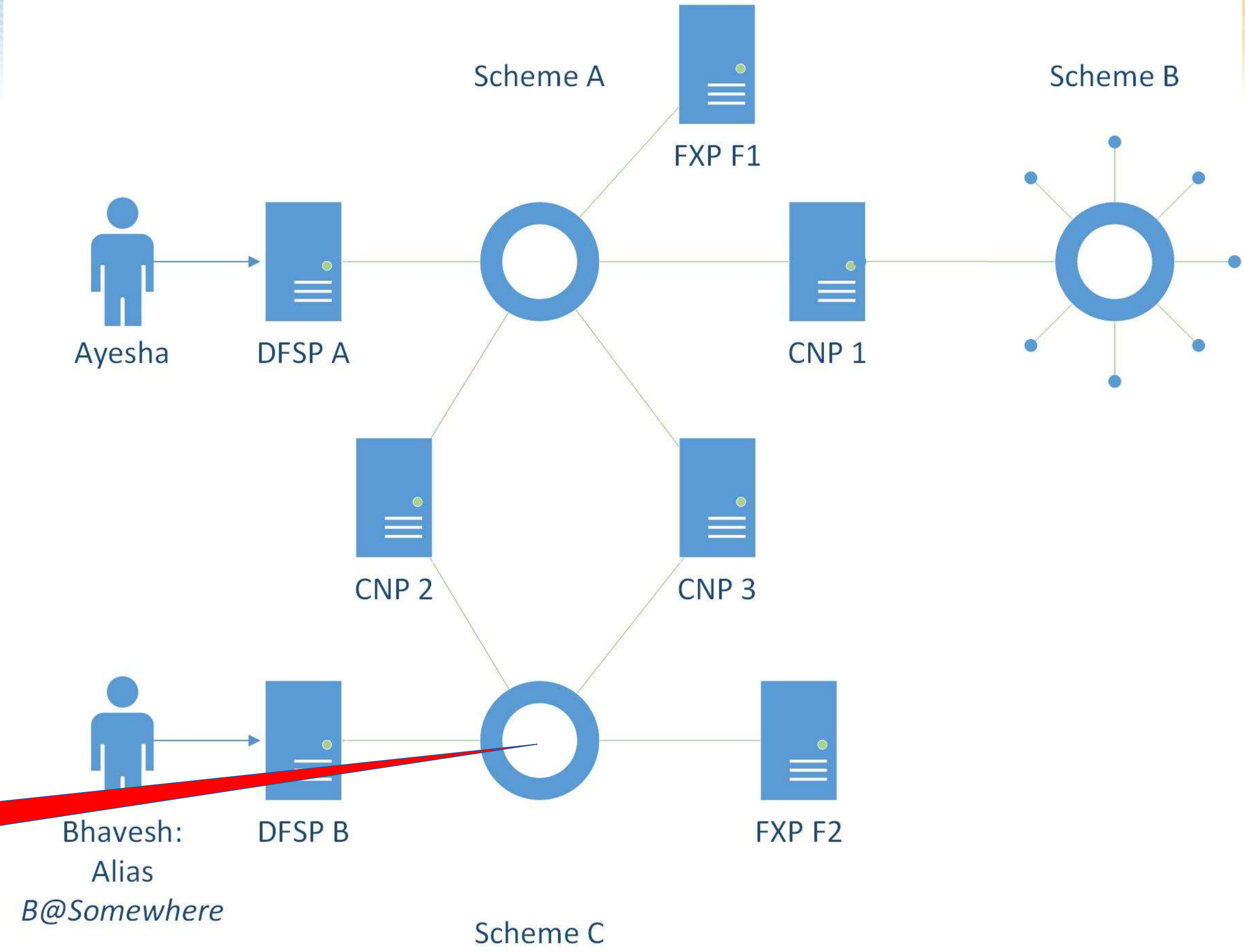
Caution:
Technical gobbledegook
ahead…

Digression 2:
How does the DFSP respond?

```
 "header":{
     "FSPIOP-Source": "DFSP B",
     "FSPIOP-Destination": "CNP 2"
     },
 "body":{
     "party": {
     "partyIdInfo": {
         "partyIdType": "ALIAS",
         "partyIdentifier": "B@Somewhere",
         "fspId": "DFSP B",
         "publicKey": "AAAAB3NzaC1yc2EAAAADAQABAAAAgQ
CqGKukO1De7zhZj6+H0qtjTkVxwTCpvKe4eCZ0FPqri0cb2JZfX
J/DgYSF6vUpwmJG8wVQZKjeGcjDOL5UlsuusFncCzWBQ7RK
NUSesmQRMSGkVb1/3j+skZ6UtW+5u09lHNsj6tQ51s1SPrCBke
dbNf0Tp0GbMJDyR4e9T04ZZw=="
     },
     "name": "Bhavesh",
```

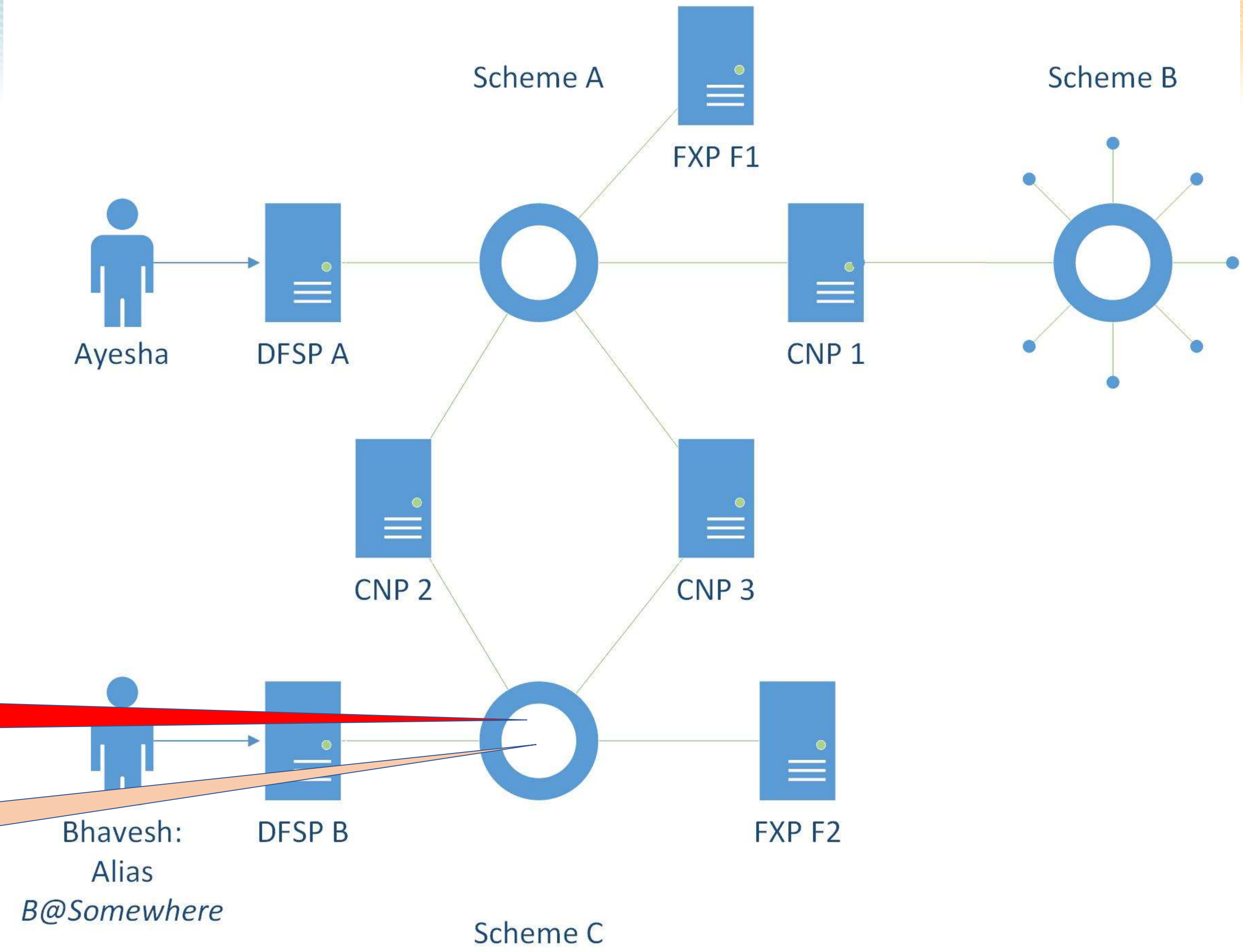
```
 "personalInfo": {
     "complexName": {
         "firstName": "Bhavesh",
         "middleName": "Other_name",
         "lastName": "Family_name"
     },
   "dateOfBirth": "1982-05-23"
 },
"accounts" [
     "account": {
         "currency": "XOF",
         "address": "SomeGuffOrOther"
     }
 ]
}
```

Charting the way home…



Bhavesh's DFSP sends the response back to the switch. The response is addressed to CNP 2

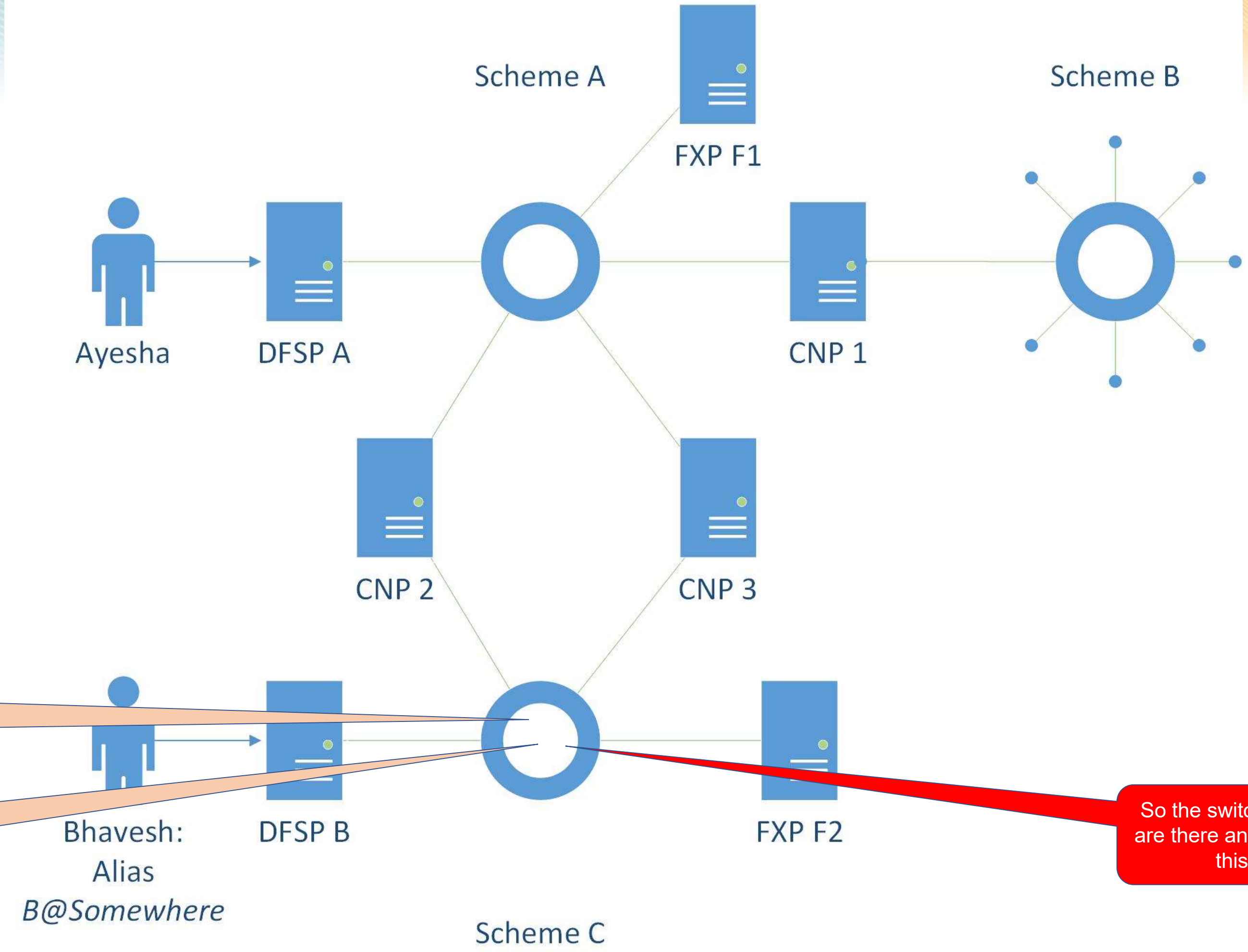
Charting the way home…



But wait! Now the switch can tell that currency conversion is required…

Bhavesh's DFSP sends the response back to the switch. The response is addressed to CNP 2

Charting the way home…

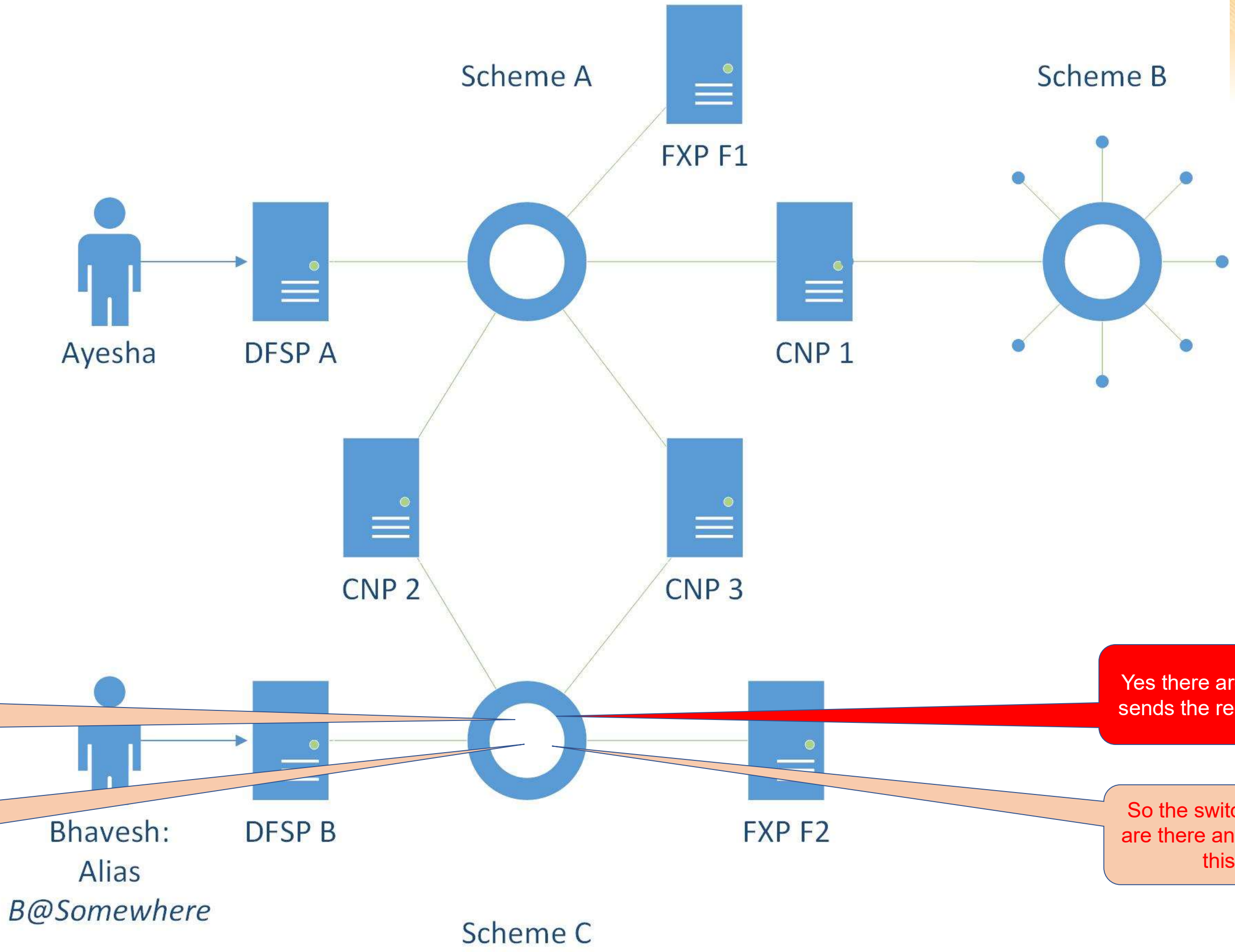


But wait! Now the switch can tell that currency conversion is required…

Bhavesh’s DFSP sends the response back to the switch. The response is addressed to CNP 2

So the switch asks the ALS: are there any FX providers in this switch?

Charting the way home…



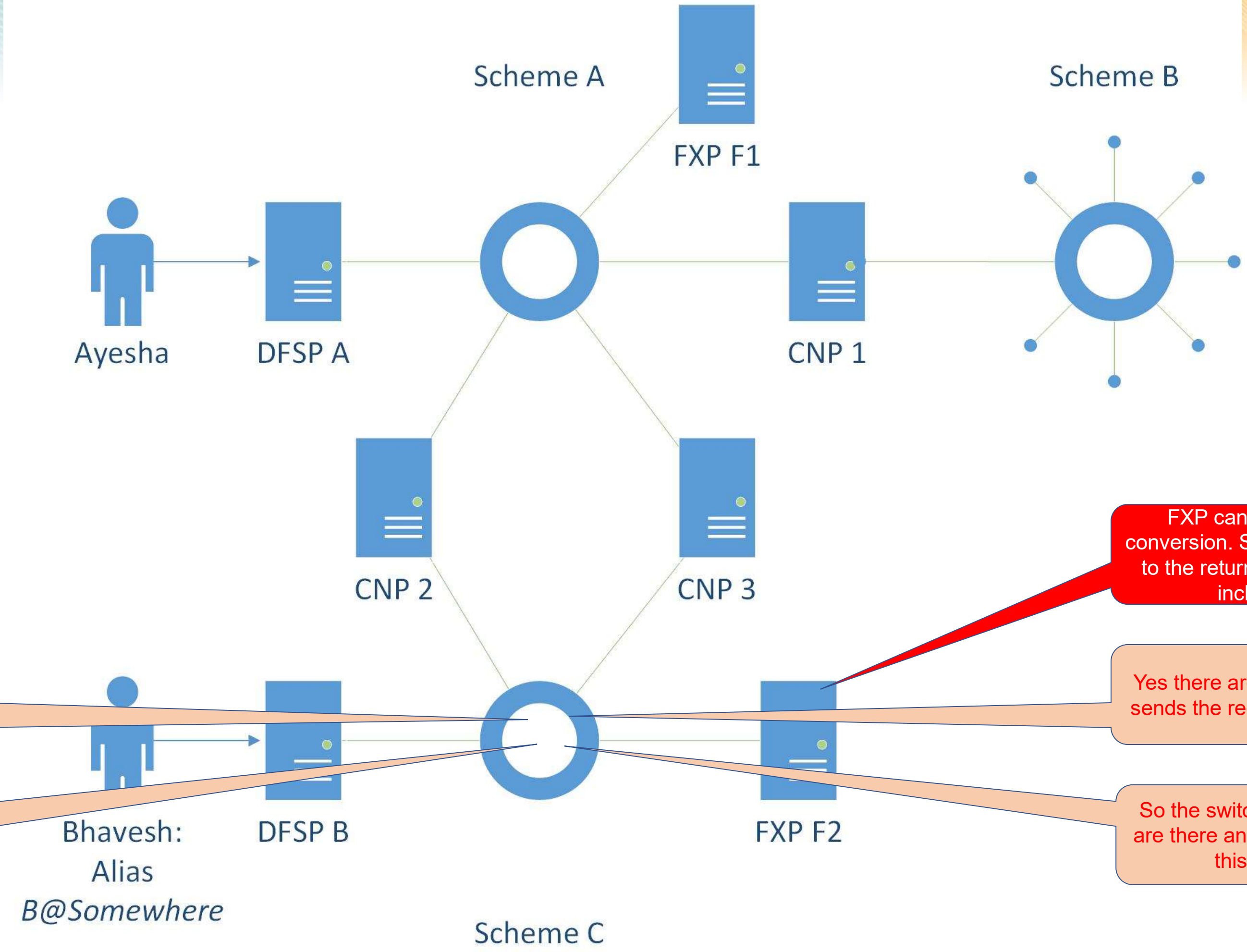
But wait! Now the switch can tell that currency conversion is required…

Yes there are. So the switch sends the request to FXP F2

Bhavesh's DFSP sends the response back to the switch. The response is addressed to CNP 2

So the switch asks the ALS: are there any FX providers in this switch?

Charting the way home…



Bhavesh's DFSP sends the response back to the switch. The response is addressed to CNP 2

But wait! Now the switch can tell that currency conversion is required…

So the switch asks the ALS: are there any FX providers in this switch?

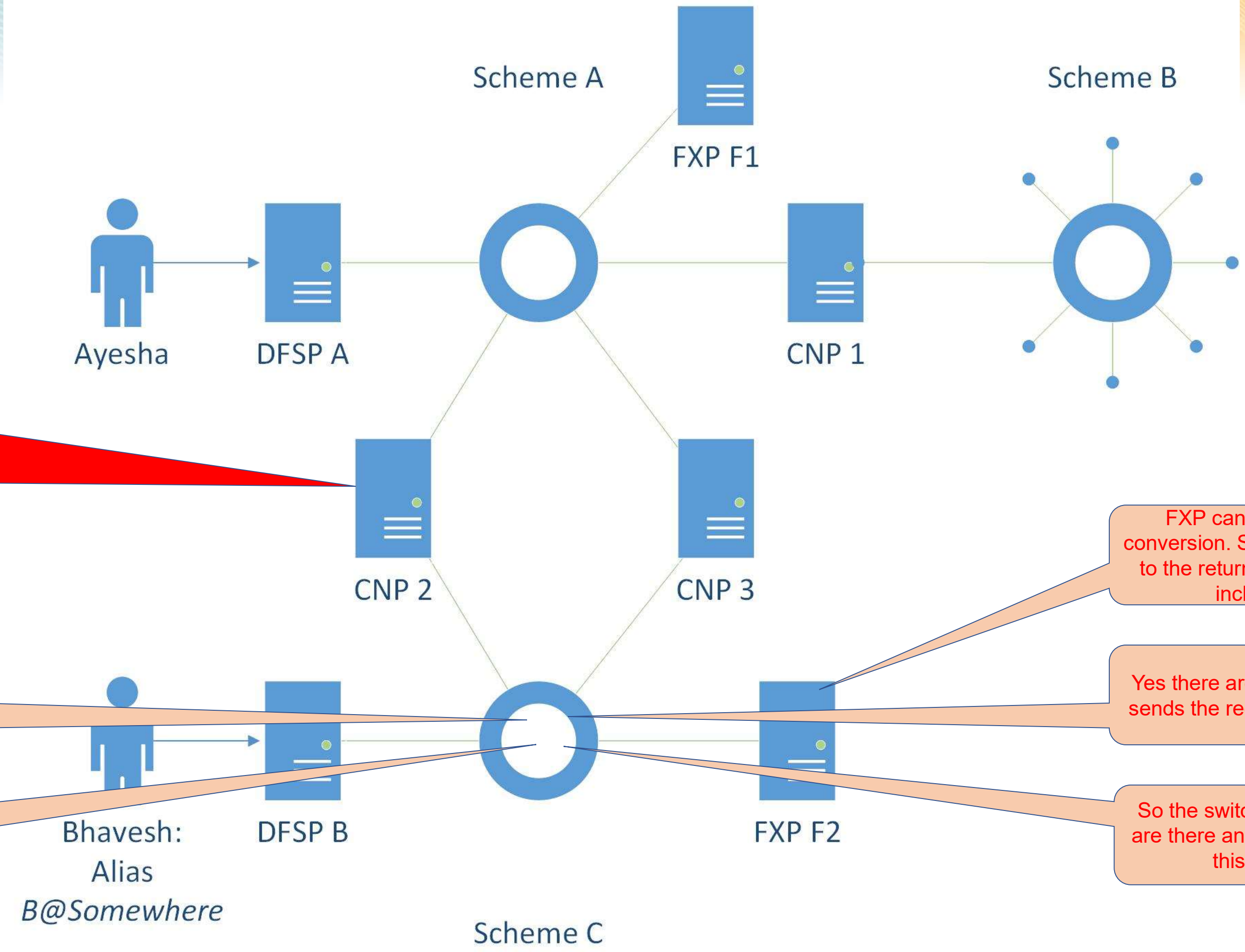
Yes there are. So the switch sends the request to FXP F2

FXP can perform the conversion. So it adds a route to the return message that includes it

Digression 3: routes

- We want to return a single response to the DFSP that originally requested the discovery
- That should include all the ways in which it’s possible to complete the transfer (as far as we know at this point…)
- So we define a **routes** collection in the response. This consists of a variable number of **route** objects.
- Participants who can provide FXP services to the transfer add themselves *in a new route* to the **routes** collection, since there will be only one FXP per route…
- Participants who can provide CNP services to the transfer add themselves to *all* of the current routes in the routes collection, since this is a collection specific to *this* CNP…

Charting the way home…



Now the response arrives at the original CNP. The CNP adds itself to all the current routes (i.e. the route that goes through FXP F2 and the (empty) route that doesn't…)

But wait! Now the switch can tell that currency conversion is required…

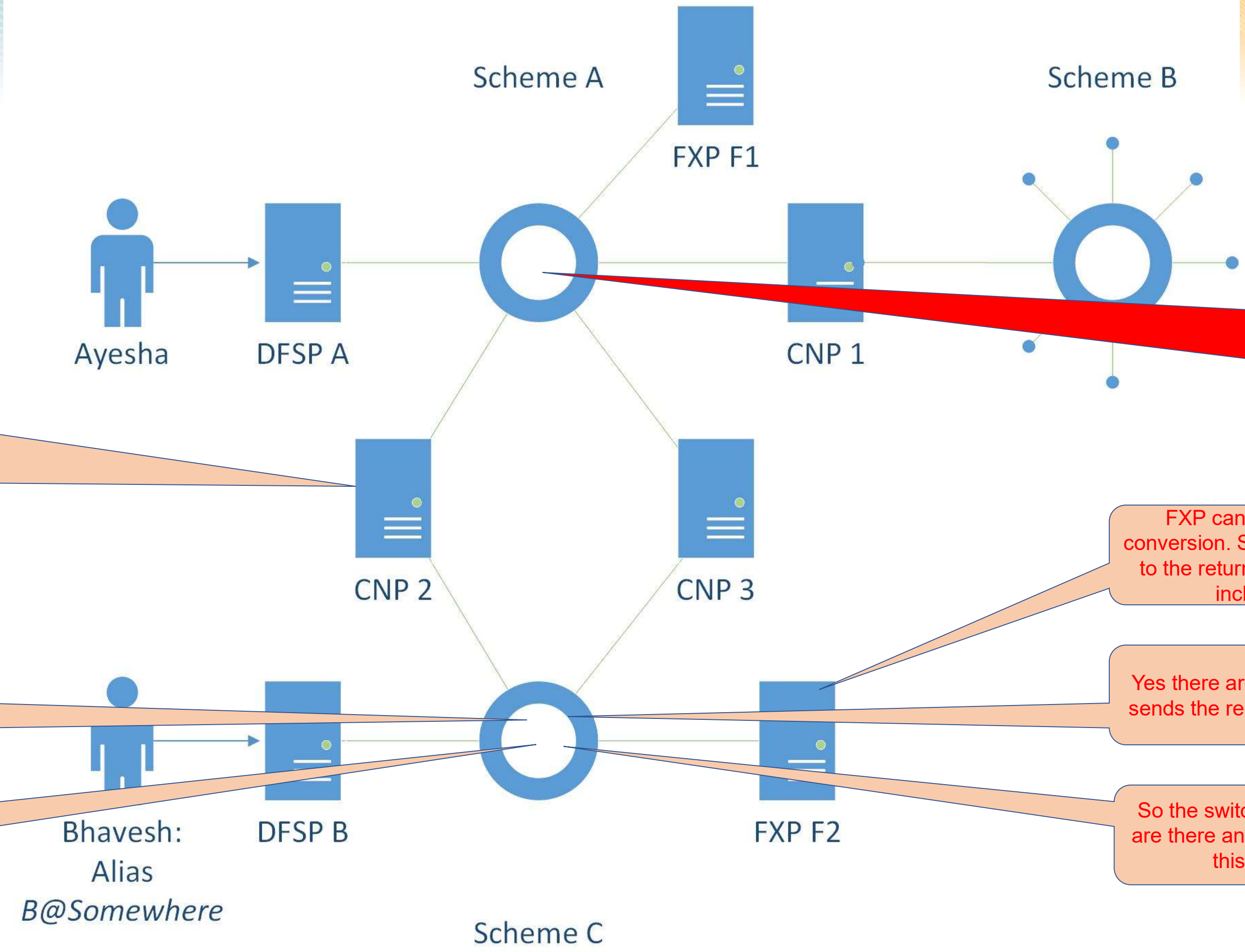
Bhavesh's DFSP sends the response back to the switch. The response is addressed to CNP 2

FXP can perform the conversion. So it adds a route to the return message that includes it

Yes there are. So the switch sends the request to FXP F2

So the switch asks the ALS: are there any FX providers in this switch?

Charting the way home…



Now the response arrives at the original switch. The switch can tell that FX is required, so it asks tha ALS if it knows of any FXP providers.

Now the response arrives at the original CNP. The CNP adds itself to all the current routes (i.e. the route that goes through FXP F2 and the (empty) route that doesn't…)

FXP can perform the conversion. So it adds a route to the return message that includes it

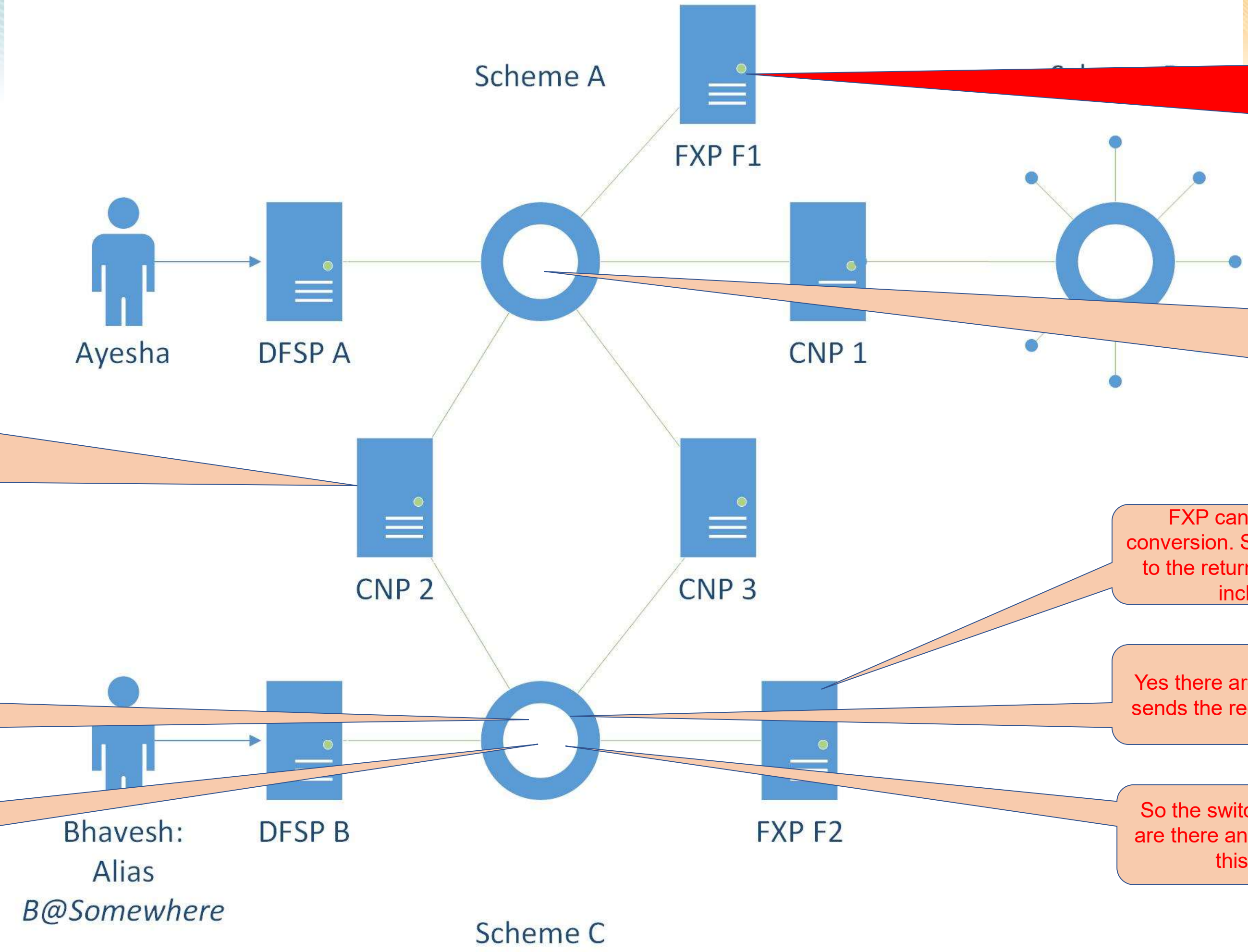
But wait! Now the switch can tell that currency conversion is required…

Yes there are. So the switch sends the request to FXP F2

Bhavesh's DFSP sends the response back to the switch. The response is addressed to CNP 2

So the switch asks the ALS: are there any FX providers in this switch?

Charting the way home…



Bhavesh's DFSP sends the response back to the switch. The response is addressed to CNP 2

But wait! Now the switch can tell that currency conversion is required…

So the switch asks the ALS: are there any FX providers in this switch?

Yes there are. So the switch sends the request to FXP F2

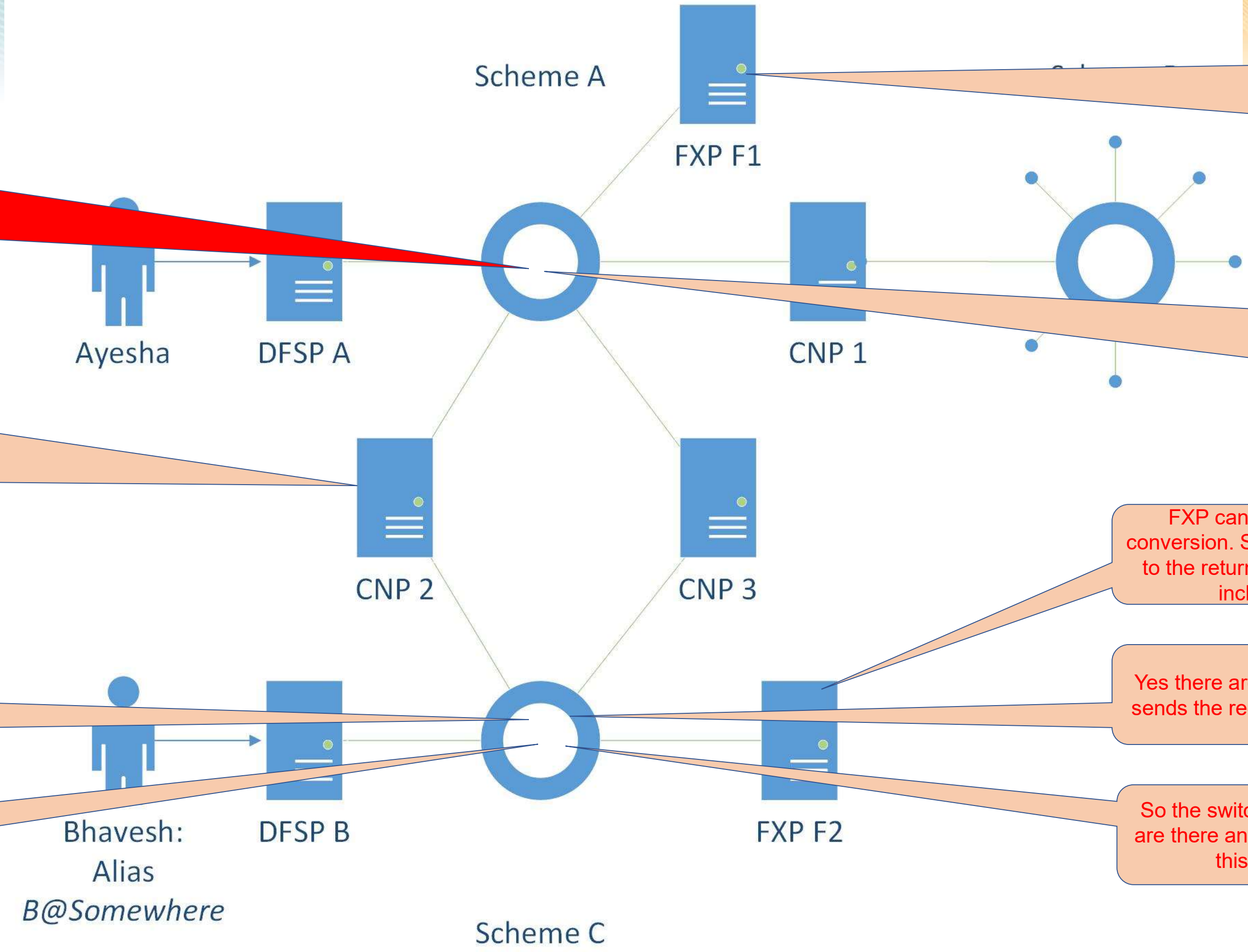
FXP can perform the conversion. So it adds a route to the return message that includes it

Now the response arrives at the original CNP. The CNP adds itself to all the current routes (i.e. the route that goes through FXP F2 and the (empty) route that doesn't…

Now the response arrives at the original switch. The switch can tell that FX is required, so it asks tha ALS if it knows of any FXP providers.

The route to which the FXP needs to add itself is the route with the smallest number of hops… The FXP also needs to confirm that the CNP it will be sending to can receive the converted currency…

Charting the way home…



So now we have two possible routes.
And two more CNPs to try…

Now the response arrives at the original CNP. The CNP adds itself to all the current routes (i.e. the route that goes through FXP F2 and the (empty) route that doesn't…)

But wait! Now the switch can tell that currency conversion is required…

Bhavesh's DFSP sends the response back to the switch. The response is addressed to CNP 2

The route to which the FXP needs to add itself is the route with the smallest number of hops… The FXP also needs to confirm that the CNP it will be sending to can receive the converted currency…

Now the response arrives at the original switch. The switch can tell that FX is required, so it asks tha ALS if it knows of any FXP providers.

FXP can perform the conversion. So it adds a route to the return message that includes it

Yes there are. So the switch sends the request to FXP F2

So the switch asks the ALS: are there any FX providers in this switch?

CNP 2 offers two routes to Bhavesh’s DFSP

- The first route goes via CNP 2 and FXP F2
- The second route goes via FXP F1 and CNP 2
- The switch in scheme A stores these two routes and tries out CNP 1 and CNP 3

CNP 1

- CNP 1 has no connection to Bhavesh’s DFSP, so nothing is returned…

CNP 3

- CNP 3 essentially offers the same facilities as CNP 2, except…

CNP 3

- CNP 3 essentially offers the same facilities as CNP 2, except…
- CNP 3 can’t transact in XOF, so there’s no point in FXP F1 doing the currency conversion
- So CNP 3 returns only one route, instead of two, via:
  - CNP 3 -> FXP F2

The result…

- The switch in Scheme A has collected three possible routes to Bhavesh’s DFSP:
  - **CNP 2 -> FXP F2 -> DFSP B**
  - **FXP F1 -> CNP 2 -> DFSP B**
  - **CNP 3 -> FXP F2 -> DFSP B**
- It returns those routes to DFSP A, and allows DFSP A to decide what to do next.

… and they look like this…

```
"routes" [
    "route": {
        "participantList": [
            "participant" {
                "fspId": "FXP F1",
                "transferCurrency": "XOF"
            },
            "participant" {
                "fspId": "CNP 2",
                "transferCurrency": "XOF"
            }
        ]
    },
    "route": {
        "participantList": [
            "participant" {
                "fspId": "CNP 2"
            },
```

```
            "participant" {
                "fspId": "FXP F2",
                "transferCurrency": "XOF"
            }
        ]
    },
    "route": {
        "participantList": [
            "participant" {
                "fspId": "CNP 3",
                "transferCurrency": "ZAR"
            },
            "participant" {
                "fspId": "FXP F2",
                "transferCurrency": "XOF"
            }
        ]
     }
 ]
```

Dost thou comprehend?