Mobile Money API Specification 1.2.0 Master

Document Summary

Official Document Number, Document Title and Version Number	Mobile Money API Specification 1.2.0 - Master
Official Document Type	Non-binding Permanent Reference Document
Change Request Security Classification	Non-confidential

Document History

Document Version	API Release Version	Date	Brief Description of Change	Editor / Company
0.1	1.2.0-beta	Aug 2020	• Initial draft of document	GSMA
0.2	1.2.0	Oct 2020	• Implemented RFC 2020-3, RFC 2020-5, RFC 2020-6, RFC 2020-7, RFC 2020-8, RFC 2020-9, RFC 202-10, RFC 2020-11, RFC 2020-13, RFC 2020-15, RFC 2020-16, RFC 2020-17, RFC 2020-18, RFC 2020-19	GSMA

Other Information

Type	Description
Document Owner	Mobile Money API Working Group
Editor / Company	GSMA

Table of Contents

1 Introduction

The purpose of this document is to specify the Mobile Money API endpoints, fields, objects, and enumerations.

For further reading, please refer to the following documents:

- **Mobile Money API Introduction**. Introduces the use and benefits of the Mobile Money API. Also provides a glossary of terms used by the Mobile Money API specifications.
- **Mobile Money API Fundamentals**. Specifies the design principles, behaviours, and error handling of the Mobile Money API.

All documentation can be found on the GSMA Mobile Money API Developer Portal.

This document contains the following sections:

2 API Endpoints

API endpoint fields are described in this specification as follows:

- The field **name**.
- The field **type**.
- **Description** of the field.
- **Optionality** of the field, i.e. whether the field must be supplied. Optionality is identified as per follows:

  → Request optionality

  ← Response optionality

  O Field is optional

  M Field is mandatory

  C Field is conditional

  NA Field does not need to be supplied. If supplied, it will be ignored.

- **Reference** where the field is an array and/or is defined by another object.
- **Validation** applied to the field, including enumeration, field length and use of regular expressions to validate format.

Please note that string fields have a default maximum length of 256 characters unless specified otherwise.

2.1 Transactions API

The Transactions APIs are used to support mobile money financial transaction use cases. Transactions are used for a wide range of use cases including merchant payments, international transfers, domestic transfers, and agent cash-in/cash-out.

The following paths are permitted:

Operation	Path	Description
POST	*POST /transactions/type/{transactiontype}*	To be used for transaction creation when the provider's API Gateway requires that the transaction *type* be identified in the URL.
View	*GET /transactions/{transactionReference}*	To view a transaction.
Update	*PATCH /transactions/{transactionReference}*	To update the *transactionStatus* of a transaction.

2.1.1 Transaction UML Class Diagram

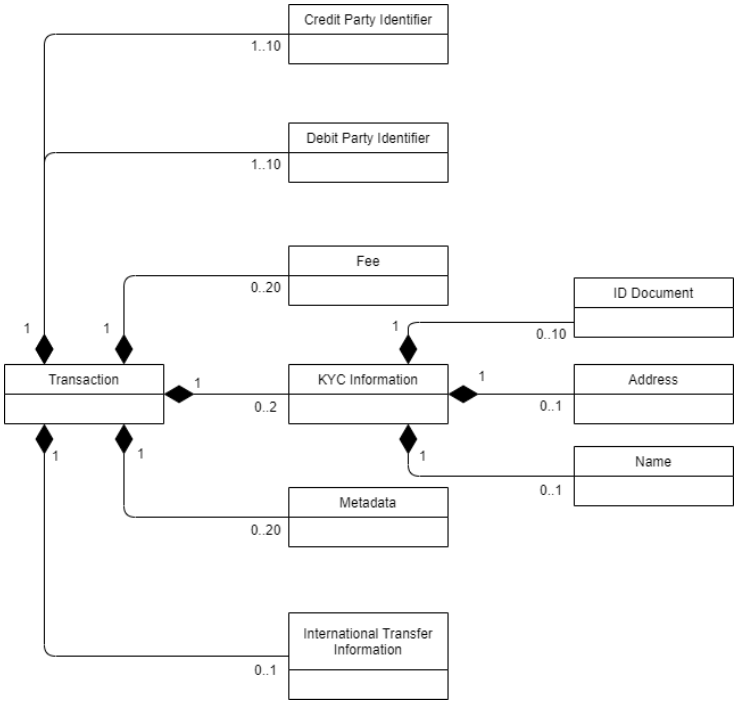

Figure 2-1 Transaction UML Class Diagram

2.1.2 Transaction Object Definition

Transaction Object

Name	Type	Description		Reference	Validation
transactionReference	string	Unique reference for the transaction. This is returned in the response by API provider.	→NA ←M		
requestingOrganisationTransactionReference	string	A reference provided by the requesting organisation that is to be associated with the transaction.	→O ←O		
originalTransactionReference	string	For reversals and refunds, this field indicates the transaction which is the subject of the reversal.	→O ←O		
creditParty	array	A series of key/value pairs that enable the credit party to be identified. Keys include MSISDN and Wallet Identifier.	→C ←C	Account Identifiers	creditParty must be supplied if debitParty is omitted. If debitParty is supplied, then creditParty is optional.
debitParty	array	A collection of key/value pairs that enable the debit party to be identified. Keys include MSISDN and Wallet Identifier.	→C ←C	Account Identifiers	debitParty must be supplied if creditParty is omitted. If creditParty is supplied, then debitParty is optional.
type	string	The harmonised Transaction Type (not required if passed in the URL).	→M ←M		Enumeration = Transaction Types
subType	string	A non-harmonised sub-classification of the type of transaction. Values are not fixed, and usage will vary according to Provider.	→O ←O		
transactionStatus	string	Indicates the status of the transaction as stored by the API provider.	→NA ←M		
amount	string	The transaction amount.	→M ←M		Please refer to API Fundamentals document for

					amount validation rules.
currency	string	Currency of the transaction amount.	→M ←M		Enumeration = ISO Currency Codes
descriptionText	string	Free format text description of the transaction provided by the client. This can be provided as a reference for the receiver on a notification SMS and on an account statement.	→O ←O		
fees	array	Allows the passing and/or returning of all fees pertaining to the transaction.	→O ←O	Fees Object	
geoCode	string	Indicates the geographic location from where the transaction was initiated.	→O ←O		
internationalTransferInformation	object	A collection of fields detailing information specifically used for international transfers.	→O ←O	International Transfer Information	
oneTimeCode	string	A one-time code that can be supplied in the request or can be generated in the response depending upon the use case. An authorisation code can be supplied in this field for requests that have been pre-authorised.	→O ←O		
recipientKyc	object	A collection of fields detailing the KYC of the transaction recipient.	→O ←O	KYC Information	
senderKyc	object	A collection of fields detailing the KYC of the transaction sender.	→O ←O	KYC Information	
requestingOrganisation	object	The originating organisation of the request.	→O ←O	Requesting Organisation	
servicingIdentity	string	The field is used to identify the servicing identity for	→O ←O		

		transactions, e.g. till, POS ID, assistant ID.			
transactionReceipt	string	Transaction receipt number as notified to the parties. This may differ from the Transaction Reference.	→NA ←O		
creationDate	date-time	Date and time when the transaction was created by the API Provider	→NA ←O		
modificationDate	date-time	Date and time when the transaction was modified by the API Provider	→NA ←O		
requestDate	date-time	The date and time of the transaction request as supplied by the client.	→O ←O		
customData	string	A collection of key/value pairs that can be used for provider specific fields.	→O ←O	Custom Data Object	
metadata	array	A collection of key/value pairs. These can be used to populate additional properties that describe administrative information regarding the transaction.	→O ←O	Metadata	

2.2 Reversals API

The Reversals API is used to reverse, adjust or refund a financial transaction. The originating transaction reference must be provided in the path in order to identify the transaction to be reversed. For a partial reversal, the amount needs to be supplied. It should be noted that some API providers do not support partial reversals and will return an error if a partial amount is supplied.

For viewing and updating reversals, the Transactions API should be used.

The supported path is *POST /transactions/{originalTransactionReference}/reversals*.

2.2.1 Reversal UML Class Diagram

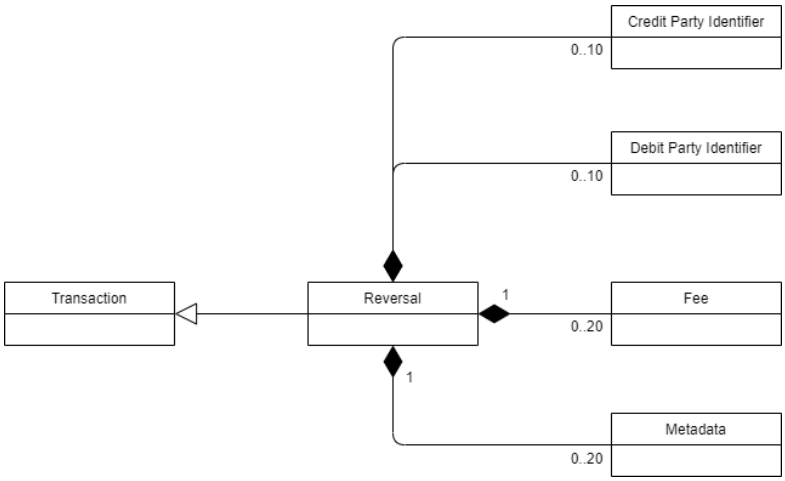

Figure 2-2 Reversal UML Class Diagram

2.2.2 Reversal Object Definition

Reversal Object					
Name	**Type**	**Description**		**Reference**	**Validation**
transactionReference	string	Unique reference for the transaction. This is returned in the response by API provider.	→NA ←M		
requestingOrganisationTransactionReference	string	A reference provided by the requesting organisation that is to be associated with the transaction.	→O ←O		
originalTransactionReference	string	For reversals and refunds, this field indicates the transaction which is the subject of the reversal.	→NA ←M		
creditParty	array	A series of key/value pairs that enable the credit party to be identified. Keys include MSISDN and Wallet Identifier.	→O ←O	Account Identifiers	
debitParty	array	A collection of key/value pairs that enable the debit party to be identified. Keys include MSISDN and Wallet Identifier.	→O ←O	Account Identifiers	
type	string	The harmonised Transaction Type.	→M ←M		Enumeration = Transaction Types Note that only Reversals and

					Refunds (adjustments) are supported.
subType	string	A non-harmonised sub-classification of the type of transaction. Values are not fixed, and usage will vary according to Provider.	→O ←O		
transactionStatus	string	Indicates the status of the transaction as stored by the API provider.	→NA ←M		
amount	string	The transaction amount.	→O ←O		Please refer to API Fundamentals document for amount validation rules.
currency	string	Currency of the transaction amount.	→O ←O		Enumeration = ISO Currency Codes.
descriptionText	string	Free format text description of the transaction provided by the client. This can be provided as a reference for the receiver on a notification SMS and on an account statement.	→O ←O		
fees	array	Allows the passing and/or returning of all fees pertaining to the transaction.	→O ←O	Fees Object	
geoCode	string	Indicates the geographic location from where the transaction was initiated.	→O ←O		
requestingOrganisation	object	The originating organisation of the request.	→O ←O	Requesting Organisation	
servicingIdentity	string	The field is used to identify the servicing identity for transactions, e.g. till, POS ID, assistant ID.	→O ←O		
transactionReceipt	string	Transaction receipt number as notified to the parties. This may differ from the Transaction Reference.	→NA ←O		
creationDate	date-time	Date and time when the transaction was created by the API Provider.	→NA ←O		

modificationDate	date-time	Date and time when the transaction was modified by the API Provider.	→NA ←O		
requestDate	date-time	The date and time of the transaction request as supplied by the client.	→O ←O		
customData	string	A collection of key/value pairs that can be used for provider specific fields.	→O ←O	Custom Data Object	
metadata	array	A collection of key/value pairs. These can be used to populate additional properties that describe administrative information regarding the transaction.	→O ←O	Metadata	

2.3 Batch Transactions

The Mobile Money API allows clients to submit, approve and view batches of transactions. The following steps describe an end to end flow for processing batch transactions. Two types of processing modes are supported:

- One shot processing without an approver.
- Maker/checker approval, i.e. transactions are not completed until approved via a second API request.

The individual APIs that are referenced in the steps below are fully documented in subsequent sub-sections.

2.3.1 Batch Transactions Workflow

2.3.1.1 One-Shot Batch Processing

Creating a Batch

1. Client submits the batch for processing via *POST /batchtransactions*.
2. The client will return the requestState object indicating whether a callback will be provided or polling is required.
3. The API provider will parse the batch in order to determine whether the transactions are capable of being processed.
4. Once parsing has completed, the API provider will set the batch status in the *batchTransactions* object to '**completed**'.

Verifying a Batch

5. The client will be able to retrieve the batch transaction object by invoking *GET /batchtransactions* using the object reference provided by the *requestState* object. Alternatively, if Callback is specified, the client will receive the representation of the batchTransactions object to their nominated URL set in the *X-Callback-URL* header.
6. If errors are indicated, i.e. some of the transactions failed parsing, the client is able to retrieve the errors via *GET /batchtransactions/rejections*. Successfully completed transactions can be viewed via *GET /batchtransactions/completions*.

2.3.1.2 Batch Processing with Maker/Checker

Creating a Batch

1. Client submits the batch for processing via *POST /batchtransactions*.
2. The client will return the *requestState* object indicating whether a callback will be provided or polling is required.
3. The API provider will parse the batch in order to determine whether the transactions are capable of being processed.
4. Once parsing has completed, the API provider will set the batch status in the *batchTransactions* object to '**created**'.

Verifying a Batch

5. The client will be able to retrieve the batch transaction object by invoking *GET /batchtransactions* using the object reference provided by the *requestState* object. Alternatively, if Callback is specified, the client will receive the representation of the *batchTransactions* object to their nominated URL set in the *X-Callback-URL* header.
6. If errors are indicated, i.e. some of the transactions failed parsing, the client is able to retrieve the errors via *GET /batchtransactions/rejections*.

Approving a Batch

7. A client can approve the batch for posting by issuing a *PATCH /batchtransactions* in order to update the status to '**approved**'.
8. As per step 2, a *requestState* object will be returned indicating whether a callback will be provided or polling is required.
9. The API provider will then post the transactions in the batch considering any scheduling considerations.
10. Once posting is completed, the API provider will set the batch status in the *batchTransactions* object to '**completed**'.
11. The client will be able to retrieve the *batchTransactions* object by invoking GET /batchtransactions using the object reference provided by the *requestState* object. Alternatively, if Callback is specified, the client will receive the representation of the *batchTransactions* object to their nominated URL set in the *X-Callback-URL* header.
12. the client will also be able to retrieve a list of successful transaction completions */batchtransactions/completions* and transaction failures */batchtransactions/rejections*.

2.4 Batch Transactions API

This API enables clients to submit and approve a batch of transactions. The API allows transactions of multiple types to be include in a single batch. The following paths are permitted:

- **Submit** a batch: *POST /batchtransactions*
- **Approve** a batch: *PATCH /bathtransactions/{batchID}*. The Batch Status needs to be set to 'approved'.
- **View** details regarding batch processing: *GET /batchtransactions/{batchID}*

Only asynchronous mode is supported for the POST and PATCH methods. For the GET method, only synchronous mode is supported.

There is a limit of 999,999 transaction records per batch.

2.4.1 Batch Transaction Object UML Diagram

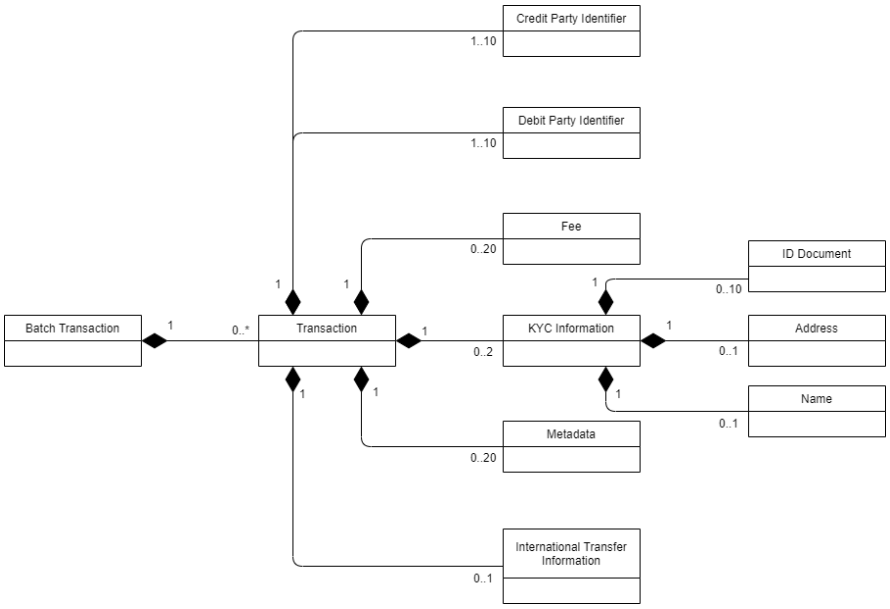

Figure 2-3 Batch Transaction UML Class Diagram

2.4.2 Batch Transaction Object Definition

Batch Transaction Object					
Name	**Type**	**Description**		**Reference**	**Validation**
batchID	string	Identifier for the Batch that is assigned by the API provider. This ID is used by the client on subsequent GET or PATCH methods.	→N/A ←M		
batchStatus	string	Indicates the status of the batch.	→O ←M		Enumeration = created, approved, completed
Transactions	array	Collection of Transactions that are to be processed. Note that the representation of each completed transaction can be retrieved via the '/completions API.	→M ←N/A	Transactions	
approvalDate	date-time	Indicates when the batch was approved as recorded by the API provider.	→NA ←M		
completionDate	date-time	Indicates when the batch was completed as recorded by the API provider.	→NA ←M		
batchTitle	string	Client-provided title for the batch.	→O ←O		

batchDescription	string	Client-provided description of the batch.	→O ←O		
processingFlag	boolean	Indicates whether the batch is currently undergoing processing by the API Provider.	→N/A ←O		
completedCount	integer	Indicates the number of records that have been successfully completed.	→NA ←O		
parsingSuccessCount	integer	Indicates the number of records that have been parsed successfully.	→NA ←O		
rejectionCount	integer	Indicates the number of records that have been rejected, either during parsing or during final processing.	→NA ←O		
requestingOrganisation	object	The originating organisation of the request.	→O ←O	Requesting Organisation	
scheduledStartDate	date-time	If the batch has been scheduled, the expected start time is provided here.	→O ←O		
creationDate	date-time	Indicates when the batch was created as recorded by the API provider.	→NA ←O		
modificationDate	date-time	Indicates when the batch was modified as recorded by the API provider.	→NA ←O		
requestDate	date-time	The date and time of the batch request as supplied by the client.	→O ←O		
customData	string	A collection of key/value pairs that can be used for provider specific fields.	→O ←O	Custom Data Object	

2.5 Batch Rejections API

This API enables clients to retrieve information on all transactions that have either failed parsing or have failed to complete. Only the GET method is supported. The path is *batchtransactions/{batchID}/rejections*.

To filter the number of records returned, the following query strings can be used:

Parameter	Type	Format	Description
limit	integer	N/A	Supports pagination. If this is not supplied, then the server will apply a limit of 50 records returned for each request.
offset	integer	N/A	Supports pagination. This value will indicate the cursor position from where to retrieve the set of records. For example, a limit of 50 and offset of 10 will return records 11 to 60.
fromDateTime	string	date-time	Indicates the minimum creationDate for which records should be returned.
toDateTime	string	date-time	Indicates the maximum creationDate for which records should be returned.

Note: HTTP response headers are returned with each response indicating the total number of records available (X-Records-Available-Count) and total number of records returned (X-Records-Returned-Count).

2.5.1 Batch Rejection UML Class Diagram

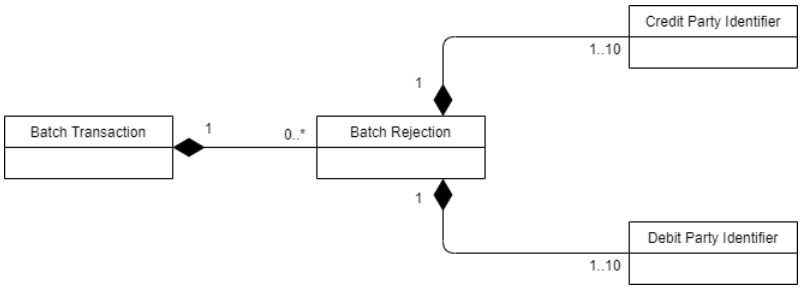

Figure 2-4 Batch Rejection UML Class Diagram

2.5.2 Batch Rejection Object Definition

Batch Rejection Object					
Name	**Type**	**Description**		**Reference**	**Validation**
transaction Reference	string	Transaction Reference as assigned by the API provider.	→N/A ←O		
requesting Organisation Transaction Reference	string	A reference provided by the requesting organisation that is to be associated with the transactions.	→N/A ←O		
creditParty	array	The credit party identifiers for the	→N/A ←M	Account Identifiers	

		transaction as specific in the batch request.			
debitParty	array	The debit party identifiers for the transaction as specific in the batch request.	→N/A ←M	Account Identifiers	
rejectionReason	string	The reason for the transaction request as indicated by the API provider.	→N/A ←M		
rejectionDate	date-time	Date and time of the rejection.	→N/A ←M		
customData	string	A collection of key/value pairs that can be used for provider specific fields.	→O ←O	Custom Data Object	

2.6 Batch Completions API

This API lists all transactions that have successfully completed for a given batch. Only the GET method is supported. The path format is *batchtransactions/{batchID}/completions*.

To filter the number of records returned, the following query strings can be used:

Parameter	Type	Format	Description
limit	integer	N/A	Supports pagination. If this is not supplied, then the server will apply a limit of 50 records returned for each request.
offset	integer	N/A	Supports pagination. This value will indicate the cursor position from where to retrieve the set of records. For example, a limit of 50 and offset of 10 will return records 11 to 60.
fromDateTime	string	date-time	Indicates the minimum creationDate for which records should be returned.
toDateTime	string	date-time	Indicates the maximum creationDate for which records should be returned.

Note: HTTP response headers are returned with each response indicating the total number of records available (X-Records-Available-Count) and total number of records returned (X-Records-Returned-Count).

2.6.1 Batch Completion UML Class Diagram

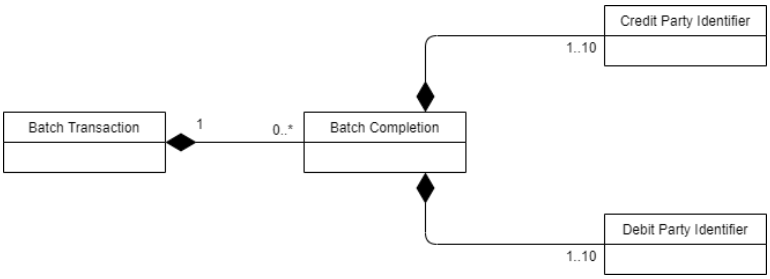

Figure 2-5 Batch Completion UML Class Diagram

2.6.2 Batch Completion Object Definition

Batch Completion Object					
Name	**Type**	**Description**		**Reference**	**Validation**
transactionReference	string	Transaction Reference as assigned by the API provider.	→N/A ←M		
requestingOrganisationTransactionReference	string	A reference provided by the requesting organisation that is to be associated with the transactions.	→N/A ←O		
creditParty	array	The credit party identifiers for the transaction as specified in the batch request.	→N/A ←M	Account Identifiers	
debitParty	array	The debit party identifiers for the transaction as specified in the batch request.	→N/A ←M	Account Identifiers	
completionDate	date-time	Date and time indicating when the transaction was completed.	→N/A ←M		
link	string	Provides a URL to the transaction resource.	→N/A ←M		
customData	string	A collection of key/value pairs that can be used for provider specific fields.	→O ←O	Custom Data Object	

2.7 Accounts APIs

The Accounts APIs are used to support a range of account-related operations. Types of accounts include mobile wallets, bank accounts, savings accounts, and loan accounts.

2.7.1 Identifying a Target Account

Two methods are provided for identifying an account, the single identifier method, and the multiple identifiers method.

2.7.1.1 Single Identifier Method

In the scenario where one identifier suffices to uniquely identify an account, the following path is to be used: */accounts/{identifierType}/{identifier}*.

2.7.1.2 Multiple Identifiers Method

Where a single identifier is not sufficient to identify an account, the following path is to be used:
/accounts/{accountIdentifier1}@{value1}${accountIdentifier2}@{value2}${accountIdentifier3}@{value3}.

The path uses a '$' delimiter to separate each identifier, up to a limit of three account identifiers. Each key/value is delimited by '@'.

The list of permitted account identifiers supported by the Mobile Money API can be found in the Account Identifiers section.

2.7.2 Account Creation

The Mobile Money API allows account creation for customers who are classified as individuals. The creation of a customer account can be triggered by various means including:

- Account creation via a mobile money agent.
- Automatic account creation upon SIM registration.
- App-based self-registration.

Regardless of the method of creation, new customers are generally provided with account and transaction limits based upon the level of KYC information they have provided and whether their KYC information has been physically verified.

To create an account, use *POST /accounts/{identityType}*, supplying '*individual*' as the *identityType*.

2.7.3 Account Retrieval

Account details, including associated account identities can be retrieved via the following paths:
GET /accounts/{accountIdentifierType}/{identifier} OR
GET /accounts/{Account Identifiers}

2.7.4 Account Update

2.7.4.1 Account-Level Updates

To update information held against an account, use one of the following paths:
PATCH /accounts/{accountIdentifierType}/{identifier} OR
PATCH /accounts/{Account Identifiers}

The following account fields can be updated:

Field	PATCH Body	Description
accountStatus	*"op": "replace", "path": "/accountStatus", "value": "string"*	Use to modify the status of an account.
accountSubStatus	*"op": "replace", "path": "/accountSubStatus", "value": "string"*	Use to modify the sub-status of an account.

For more information on the above fields please refer to the Account object.

2.7.4.2 Identity-Level Updates

To update information held against an identity associated with an account, use one of the following paths:
PATCH /accounts/{identifierType}/{identifier}/identities/{identityId} OR
PATCH /accounts/{Account Identifiers}/identities/{identityId}

The following identity fields can be updated:

Field	PATCH Body	Description
identity.identityStatus	*"op": "replace", "path": "/identityStatus", "value": "string"*	Use to modify the status of an identity associated with an account.
identity.kycVerificationStatus	*"op": "replace", "path": "/kycVerificationStatus", "value": "string"*	Use to change the KYC verification status of an identity associated with an account.
identity.kycVerificationEntity	*"op": "replace", "path": "/kycVerificationEntity", "value": "string"*	Use to indicate the entity (e.g. mobile money agent) that performed the verification.
identity.kycLevel	*"op": "replace", "path": "/kycLevel", "value": "integer"*	Use to modify the KYC level of an identity associated with an account.

For more information on the above fields please refer to the Identity object.

2.7.5 Account UML Diagram

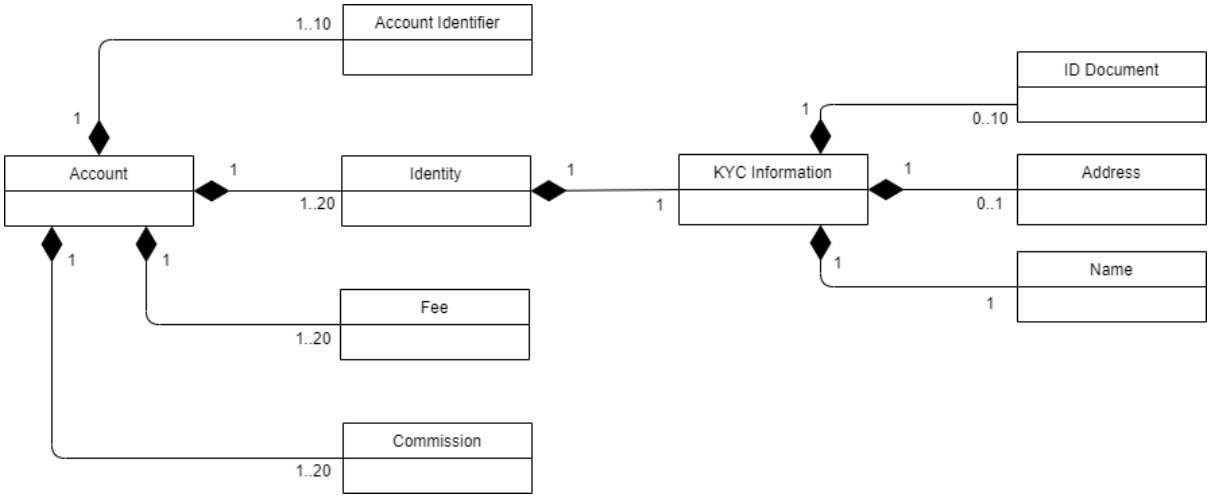

Figure 2-6 Account Creation UML Class Diagram

2.7.6 Account Object Definition

Account Object					
Name	**Type**	**Description**		**Reference**	**Validation**
accountIdentifiers	array	A series of key/value pairs that enable the account to be identified. Additional identifiers can also be assigned by the API Provider during creation.	→O ←M	Account Identifiers	
identity	array	An account must be associated with one or more identities.	→M ←M	Identity Object	
accountType	string	A non-harmonised field that indicates the type of the account.	→O ←O		
accountStatus	string	Indicates a simplified representation of the account status. This will be shown as 'available' or 'unavailable'. A state of 'unavailable' means that the account is in a state that does not	→NA ←M		

		allow posting of transactions.			
accountSubStatus	string	Field can be used to return a provider-specific status for the account.	→NA ←O		
currentBalance	string	The current outstanding balance on the account.	→NA ←O		Please refer to API Fundamentals document for amount validation rules.
availableBalance	string	Indicates the balance that is able to be debited for an account. This balance is only provided on some API provider systems.	→NA ←O		Please refer to API Fundamentals document for amount validation rules.
reservedBalance	string	Indicates the portion of the balance that is reserved, i.e. intended to be debited. This balance is only provided on some API provider systems.	→NA ←O		Please refer to API Fundamentals document for amount validation rules.
unClearedBalance	string	Indicates the sum of uncleared funds in an account, i.e. those that are awaiting a credit confirmation.	→NA ←O		Please refer to API Fundamentals document for amount validation rules.
currency	string	Currency of the account.	→NA ←O		Enumeration = ISO Currency Codes
customData	string	A collection of key/value pairs that can be used for provider specific fields.	→O ←O	Custom Data Object	
fees	string	Returns all fees related to the creation of the account.	→O ←O	Fees Object	
commissionEarned	string	Returns all commission earned by the registering entity for the creation of the account.	→NA ←O	Commission Object	
registeringEntity	string	The entity that registered the account, for example, a mobile money agent.	→O ←O		
creationDate	date-time	Indicates when the account was created as recorded by the API provider.	→NA ←O		

modificationDate	date-time	Indicates when the account was modified as recorded by the API provider.	→NA ←O		
requestDate	date-time	The date and time of the account request as supplied by the client.	→O ←O		

2.7.7 Retrieving Transactions for an Account

It is possible to return a range of transactions for an account as per the following paths:

GET /accounts/{identifierType}/{identifier}/transactions

or *GET /accounts/{Account Identifiers}/transactions*.

To filter the number of records returned, the following query strings can be used:

Parameter	Type	Format	Description
limit	integer	N/A	Supports pagination. If this is not supplied, then the server will apply a limit of 50 records returned for each request.
offset	integer	N/A	Supports pagination. This value will indicate the cursor position from where to retrieve the set of records. For example, a limit of 50 and offset of 10 will return records 11 to 60.
fromDateTime	string	date-time	Indicates the minimum creationDate for which records should be returned.
toDateTime	string	date-time	Indicates the maximum creationDate for which records should be returned.
transactionStatus	string	N/A	Indicates the status of the transactions to be returned.
transactionType	string	N/A	Indicates the type of the transactions to be returned.

Note 1: For a harmonised behavior, API Providers should make sure that the transactions are returned in descending date created order.

Note 2: HTTP response headers are returned with each response indicating the total number of records available (X-Records-Available-Count) and total number of records returned (X-Records-Returned-Count).

2.7.7.1 Account Transaction UML Class Diagram

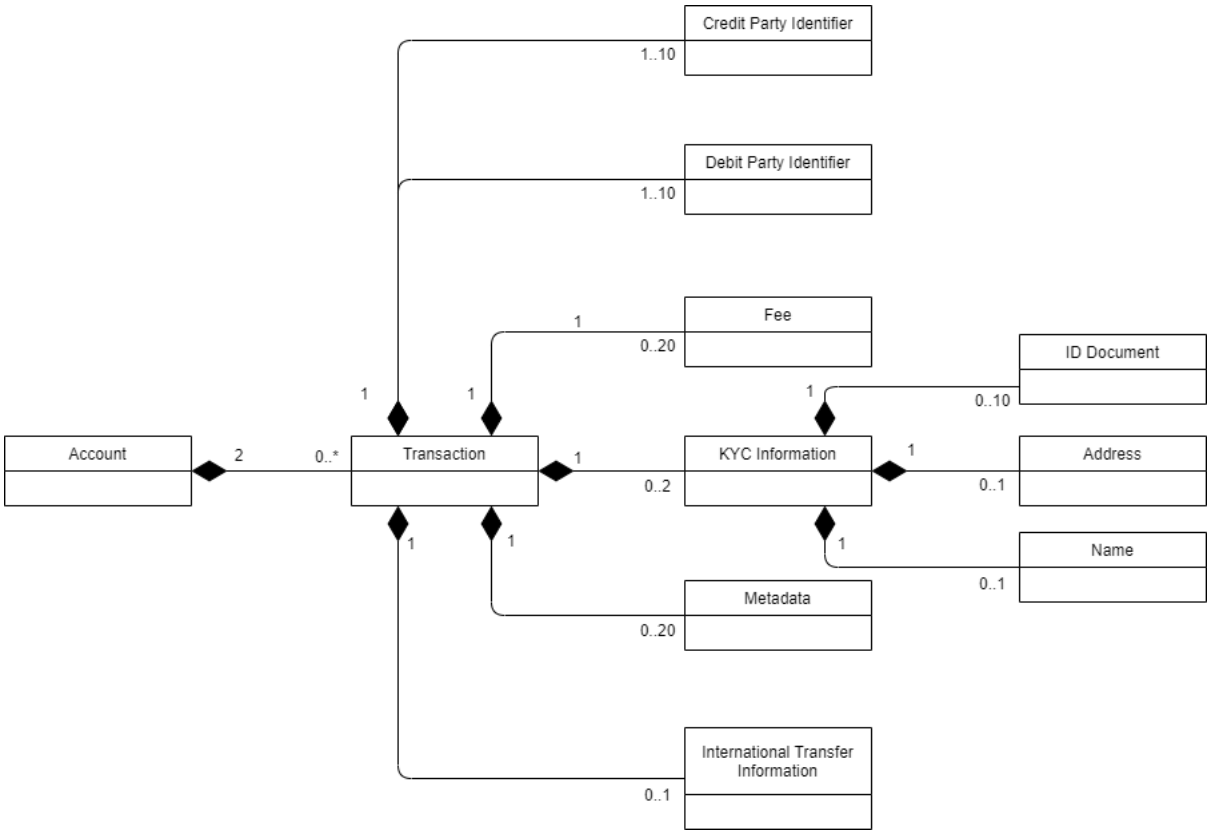

Figure 2-7 Account Transaction UML Class Diagram

2.7.8 Accounts Status API

The Accounts Status API returns a harmonised status of the account. The status enables the client to determine whether transactions can be subsequently posted against the account. Permitted paths are *GET /accounts/{identifierType}/{identifier}/status' or GET /accounts/{Account Identifiers}/status*.

2.7.8.1 Account Status UML Class Diagram

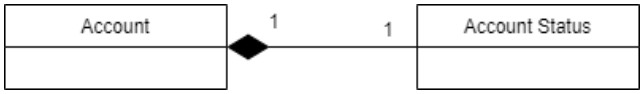

Figure 2-8 Account Status UML Class Diagram

2.7.8.2 Account Status Object Definition

Account Status Object					
Name	**Type**	**Description**		**Reference**	**Validation**
accountStatus	string	Indicates a simplified representation of the account status. This will be shown as 'available' or 'unavailable'. A state of 'unavailable' means that the account is in a state that does not allow posting of transactions. Unregistered indicates that although not available, a transaction posted with the account identifier(s) will result in an unregistered voucher creation.	→NA ←M		Enumeration = available, unavailable, unregistered
subStatus	string	Field can be used to return a provider-specific status for the account.	→NA ←O		
lei	string	Indicates the Legal Entity Identifier of the organisation holding the account.	→NA ←O		Refer to LEI format as defined here: https://www.leiroc.org/lei.htm

2.7.9 Account Balances API

This API defines specific fields for returning balances associated with an account. Permitted paths are *GET /accounts/{identifierType}/{identifier}/balance* or *GET /accounts/{Account Identifiers}/balance*.

A 'self' version is also available where the calling API client is the account holder. Path for the 'self' version is */accounts/balance*.

2.7.9.1 Account Balance UML Class Diagram

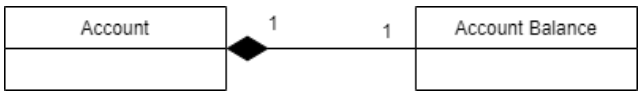

Figure 2-9 Account Balance UML Class Diagram

Balance Object					
Name	**Type**	**Description**		**Reference**	**Validation**
accountStatus	string	Indicates a harmonises representation of the account state. This will be shown as 'available' or 'unavailable'. A state of 'unavailable' means that the account is in a state that does not allow posting of transactions. Unregistered indicates that although not available, a transaction created with the account identifier(s) will result in an unregistered voucher creation.	→NA ←O		Enumeration = available, unavailable, unregistered
currentBalance	string	The current outstanding balance on the account.	→NA ←O		Please refer to API Fundamentals document for amount validation rules.
availableBalance	string	Indicates the balance that is able to be debited for an account. This balance is only provided on some API provider systems.	→NA ←O		Please refer to API Fundamentals document for amount validation rules.
reservedBalance	string	Indicates the portion of the balance that is reserved, i.e. intended to be debited. This balance is only provided on some API provider systems.	→NA ←O		Please refer to API Fundamentals document for amount validation rules.
unClearedBalance	string	Indicates the sum of uncleared funds in an account, i.e. those that are awaiting a credit confirmation.	→NA ←O		Please refer to API Fundamentals document for amount validation rules.
currency	string	Currency for all returned balances.	→NA ←O		Enumeration = ISO Currency Codes

2.7.10 Account Holder Name API

This API defines specific fields for returning account holder name information associated with an account. Permitted paths are *GET /accounts/{identifierType}/{identifier}/accountname* or *GET /accounts/{Account Identifiers}/accountname*.

2.7.10.1 Account Holder Name UML Class Diagram

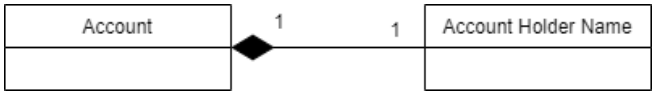

Figure 2-10 Account Holder Name UML Class Diagram

2.7.10.2 Account Holder Name Object Definition

Account Holder Name Object					
Name	**Type**	**Description**		**Reference**	**Validation**
name	reference	A collection of fields detailing the name of the Primary Account Holder.	→NA ←O	Name	
lei	string	Indicates the Legal Entity Identifier of the organisation holding the account.	→NA ←O		Refer to LEI format as defined here: https://www.leiroc.org/lei.htm

2.7.11 Account Statement Entries API

The Account Statement Entries API enables a generic representation of transactions to be returned so that they can be displayed in an account statement for the account holder. To return statement entries, an account or a transaction reference must be specified. The supported paths are as follows:

To return a specific statement entry:

- *GET /statemententries/{transactionReference}*

To return a range of statement entries:

- *GET /accounts/{ identifierType}/{identifier}statemententries* or */accounts/{Account Identifiers}/statemententries*.

To filter the number of records returned, the following query string parameters can be used:

Parameter	Type	Format	Description
limit	integer	N/A	Supports pagination. If this is not supplied, then the server will apply a limit of 50 records returned for each request.
offset	integer	N/A	Supports pagination. This value will indicate the cursor position from where to retrieve the set of records. For example, a limit of 50 and offset of 10 will return records 11 to 60.
fromDateTime	string	date-time	Indicates the minimum creationDate for which records should be returned.
toDateTime	string	date-time	Indicates the maximum creationDate for which records should be returned.
transactionStatus	string	N/A	Indicates the status of the transactions to be returned.
displayType	string	N/A	Indicates the Display Type of the transactions to be returned.

Note 1: For a harmonised behavior, API Providers should make sure that the statement entries are returned in descending date created order.

Note 2: HTTP response headers are returned with each response indicating the total number of records available (X-Records-Available-Count) and total number of records returned (X-Records-Returned-Count).

2.7.11.1 Account Statement Entry UML Class Diagram

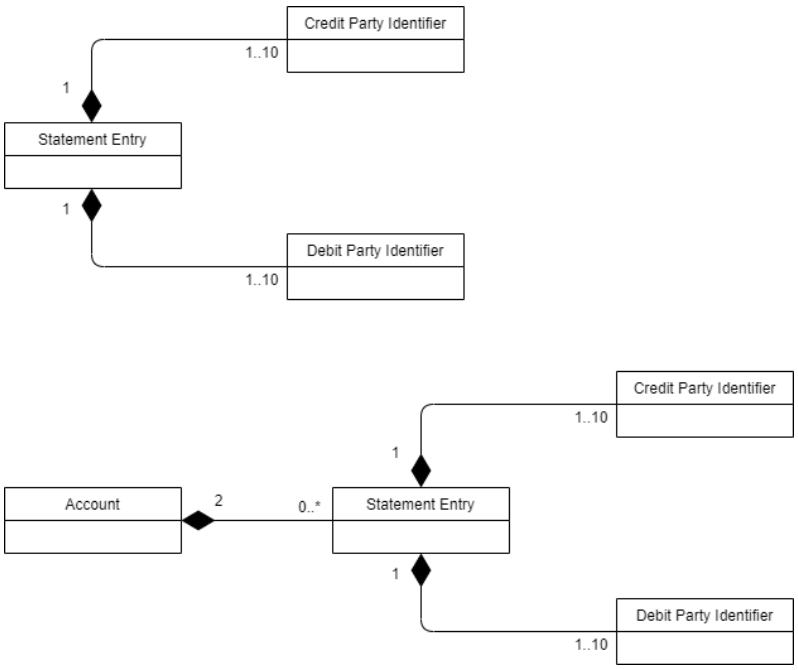

Figure 2-11 Statement Entry UML Class Diagram

2.7.11.2 Account Statement Entry Object Definition

Account Statement Entry Object					
Name	**Type**	**Description**		**Reference**	**Validation**
transaction Reference	string	Unique reference for the transaction. This is returned in the response by API provider.	→NA ←M		
creditParty	array	A series of key/value pairs that identify the credit party. Keys include MSISDN and Wallet Identifier.	→NA ←M	Account Identifiers	
debitParty	array	A collection of key/value pairs that identify the debit. Keys include MSISDN and Wallet Identifier.	→NA ←M	Account Identifiers	
transaction Status	string	Indicates the status of the transaction as represented by the API provider.	→NA ←M		
amount	string	Amount of the transaction.	→NA ←M		Please refer to API Fundamentals document for amount validation rules.
currency	string	Currency of the transaction.	→NA ←M		Enumeration = ISO Currency Codes
description Text	string	Free format text description of the transaction provided by the client. This can be provided as a reference for the receiver on a notification SMS and on an account statement.	→NA ←O		
displayType	string	The transaction type that is to be used for presentation to the account holder as determined by the API provider. This is not necessarily the actual transaction type.	→NA ←O		
transaction Receipt	string	Transaction receipt number as notified to the parties. This may differ from the Transaction Reference.	→NA ←O		
creationDate	date-time	Date and time when the transaction was created by the API Provider.	→NA ←O		

modificationDate	date-time	Date and time when the transaction modified by the API Provider.	→NA ←O		
requestDate	date-time	The date and time of the transaction request as supplied by the client.	→NA ←O		
customData	string	A collection of key/value pairs that can be used for provider specific fields.	→O ←O	Custom Data Object	

2.8 Bills API

The Bills APIs are used to return all outstanding bills associated with an account. The main purpose of the API is to support Bill Presentment, i.e. presenting all applicable bills for a payer to view and select for payment. To pay a bill, the Bill Payments API is used. Permitted paths are *GET /accounts/{identifierType}/{identifier}/bills* or *GET /accounts/{Account Identifiers}/bills*.

To filter the number of records returned, the following query string parameters can be used:

Parameter	Type	Format	Description
limit	integer	N/A	Supports pagination. If this is not supplied, then the server will apply a limit of 50 records returned for each request.
offset	integer	N/A	Supports pagination. This value will indicate the cursor position from where to retrieve the set of records. For example, a limit of 50 and offset of 10 will return records 11 to 60.
fromDateTime	string	date-time	Indicates the minimum creationDate for which records should be returned.
toDateTime	string	date-time	Indicates the maximum creationDate for which records should be returned.

Note 1: For a harmonised behavior, API Providers should make sure that the bills are returned in descending date created order.

Note 2: HTTP response headers are returned with each response indicating the total number of records available (X-Records-Available-Count) and total number of records returned (X-Records-Returned-Count).

2.8.1 Bill UML Class Diagram

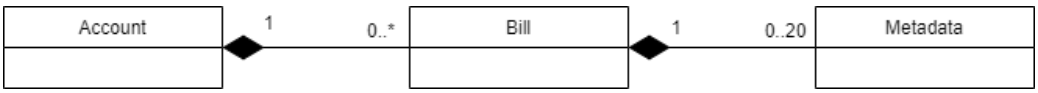

Figure 2-12 Bill UML Class Diagram

2.8.2 Bill Object Definition

Bill Object					
Name	**Type**	**Description**		**Reference**	**Validation**
billReference	string	Reference number for the Bill that the payer can use when making a payment.	→NA ←O		

billStatus	string	Identifies the status of the Bill.	→NA ←O		'paid', 'unpaid', 'partialpaid'
amountDue	string	Amount outstanding on the bill to be paid.	→NA ←O		Please refer to API Fundamentals document for amount validation rules.
billDescription	string	Description of the bill that is to be paid.	→NA ←O		
currency	string	Currency of the bill to be paid.	→NA ←O		Enumeration = ISO Currency Codes
dueDate	date	Date on which the Bill is due to be paid.	→NA ←O		
minimumAmountDue	string	The minimum amount that is outstanding on the bill to be paid.	→NA ←O		Please refer to API Fundamentals document for amount validation rules.
creationDate	date-time	Indicates when the bill was created by the API provider.	→NA ←O		
modificationDate	date-time	Indicates when the bill was modified by the API provider.	→NA ←O		
customData	string	A collection of key/value pairs that can be used for provider specific fields.	→O ←O	Custom Data Object	
metadata	array	A collection of key/value pairs. These can be used to populate additional properties that describe administrative information regarding the bill.	→NA ←O	Metadata	

2.9 Bill Payments API

The Bill Payments APIs are used to pay a specific bill associated with an account held with a service provider. Bill payments can also be retrieved. Permitted paths are:

Path	Usage
/accounts/{identifierType}/{identifier}/bills/{billReference}/payments	Use when a single identifier suffices to identify the bill account.

/accounts/*{Account Identifiers}*/bills/{billReference}/payments	Use when two or three identifiers are required to identify an account.
/bills/{billReference}/payments	Use when a bill payment is not associated with a service provider account.
/accounts/{identifierType}/{identifier}/bills/payments* OR */accounts/*{Account Identifiers}*/bills/payments	Use when a bill does not have a bill reference.

As per MM API standards, POST is used to create a bill payment whereas GET is used to retrieve all payments associated with a bill.

When retrieving bill payments, the following query string parameters can be used to filter the number of records returned:

Parameter	Type	Format	Description
limit	integer	N/A	Supports pagination. If this is not supplied, then the server will apply a limit of 50 records returned for each request.
offset	integer	N/A	Supports pagination. This value will indicate the cursor position from where to retrieve the set of records. For example, a limit of 50 and offset of 10 will return records 11 to 60.
fromDateTime	string	date-time	Indicates the minimum creationDate for which records should be returned.
toDateTime	string	date-time	Indicates the maximum creationDate for which records should be returned.

Note 1: For a harmonised behavior, API Providers should make sure that the bill payments are returned in descending date created order.

Note 2: HTTP response headers are returned with each response indicating the total number of records available (X-Records-Available-Count) and total number of records returned (X-Records-Returned-Count).

2.9.1 Bill Payment UML Class Diagram

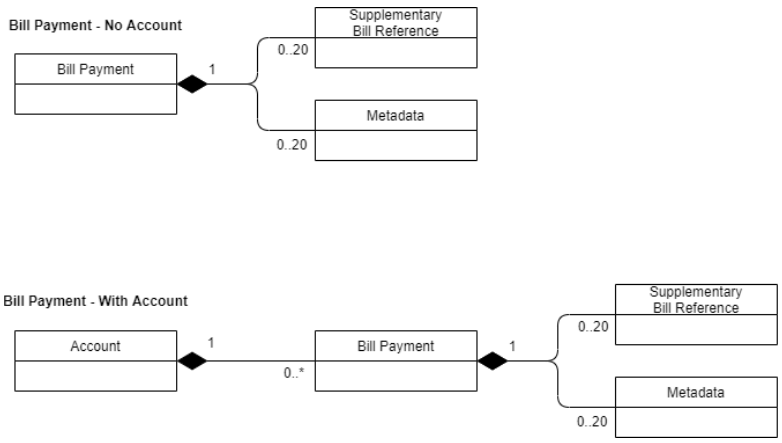

Figure 2-13 Bill Payment UML Class Diagram

2.9.2 Bill Payment Object Definition

Bill Payment Object					
Name	**Type**	**Description**		**Reference**	**Validation**
serviceProviderPaymentReference	string	Reference for the payment generated by the service provider.	→O ←O		
requestingOrganisationTransactionReference	string	The mobile money provider's (or Financial Institution's) transaction reference used to debit the customer and credit the service provider.	→O ←O		
paymentType	string	Describes the type of Bill Payment, i.e. whether a full or partial payment.	→O ←O		Enumeration = 'fullpayment', 'partialpayment'
billPaymentStatus	string	Indicates the status of the bill payment as stored by the API provider.	→N/A ←M		
amountPaid	string	Amount that is being paid.	→M ←M		Please refer to API Fundamentals document for amount validation rules.
currency	string	Currency of the amount that is being paid.	→M ←M		Enumeration = ISO Currency Codes
customerReference	string	Textual reference provided by the customer paying the bill.	→O ←O		
requestingOrganisation	string	The originating mobile money provider or financial institution that holds the wallet/account of the payer.	→O ←O		

supplementaryBillReferenceDetails	array	In some cases, a single reference is not sufficient to identify a bill. This key-value collection enables further reference information to be supplied.	→C ←C	Bill References	Not applicable if billReference is no passed in the path.
serviceProviderComment	string	Allows the Service Provider to include specific information regarding the bill payment.	→N/A ←O		
serviceProviderNotification	string	Allows the Service Provider to include specific information that will be included on the notification to the customer by the mobile money provider.	→N/A ←O		
creationDate	date-time	Indicates when the bill payment was created as recorded by the API provider.	→NA ←O		
modificationDate	date-time	Indicates when the bill payment was modified as recorded by the API provider.	→NA ←O		
requestDate	date-time	The date and time of the bill payment request as supplied by the client.	→O ←O		
customData	string	A collection of key/value pairs that can be used for provider specific fields.	→O ←O	Custom Data Object	
metadata	array	A collection of key/value pairs. These can be used to populate additional properties that describe administrative information regarding the bill payment.	→O ←O	Metadata	

2.10 Bill Companies API

The Bill Companies APIs are used to return a list of Service Providers that accept bill payments. Permitted paths are as per below:

- Bill Companies irrespective of account:
  - Use *GET /billcompanies* to return a list of all bill payment service providers.
  - Use *GET /billcompanies/{serviceProvider}* to return a specific bill payment service provider.
- Bill Companies for a given account:
  - *GET /accounts/{identifierType}/{identifier}/billcompanies* or *GET /accounts/{Account Identifiers}/billcompanies*.

To filter the number of records returned, the following query strings can be used:

Parameter	Type	Format	Description
limit	integer	N/A	Supports pagination. If this is not supplied, then the server may apply a limit of records returned for each request.
offset	integer	N/A	Supports pagination. This value will indicate the cursor position from where to retrieve the set of records. For example, a limit of 50 and offset of 10 will return records 11 to 60.

Note: HTTP response headers are returned with each response indicating the total number of records available (X-Records-Available-Count) and total number of records returned (X-Records-Returned-Count).

2.10.1 Bill Company UML Class Diagram

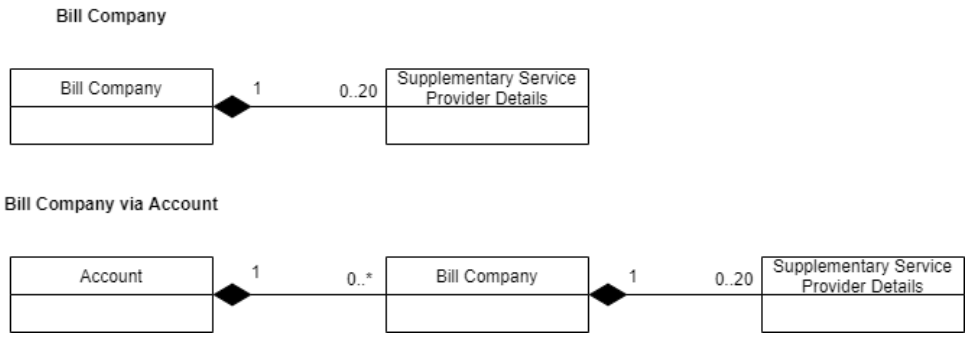

Figure 2-14 Bill Company UML Class Diagram

2.10.2 Bill Company Object Definition

Bill Companies Object					
Name	**Type**	**Description**		**Reference**	**Validation**
serviceProvider	string	Service Provider Reference Code.	→NA ←M		
service ProviderType	string	Type of Service Provider that accepts payments.	→NA ←O		
serviceProvider SubType	string	Sub-Type of Service Provider.	→NA ←O		
companyName	string	Display Name for the Service Provider.	→NA ←M		
supplementary ServiceProvider Details	array	In some cases, further information for a service provider can be returned. This key-value collection enables further information to be supplied.	→NA ←O	Supplementary Service Provider Details	

2.11 Debit Mandates API

The Debit Mandates APIs allow a mobile money customer to provide prior approval for payments to be taken from their account by a requesting payee. If the amount limit field is not supplied, the payee will be able to take any amount. Mandates can be open-ended or can be constrained by a quantified number of payments for a given frequency.

Mandates can be created, viewed, and modified. The request to create a debit mandate will be typically initiated by the service provider (payee) but can also be requested by the customer (payer).

The permitted paths are as follows. Note that the payer account is identified in the path whereas the payee account is identified in the request body.

- **Creation**: *POST /accounts/{identifierType}/{identifier}/debitmandates* or *POST /accounts/{Account Identifiers}*/debitmandates.
- **Update**: In order to update a debit mandate, a HTTP PATCH is used. Format is: *PATCH /accounts/{identifierType}/{identifier}/debitmandates/{mandateReference}* or *PATCH /accounts/{Account Identifiers}/debitmandates/{mandateReference}*
- **Read**. *GET /accounts/{identifierType}/{identifier}/debitmandates/{mandateReference}* or *GET /accounts/{Account Identifiers}/debitmandates/{mandateReference}.*

Synchronous and asynchronous modes are supported for the POST and PATCH methods whereas only synchronous mode is supported for the GET method.

The following fields are modifiable: *mandateStatus*, *startDate*, *endDate*, *frequencyType*, *numberOfPayments.*

2.11.1 Debit Mandate UML Class Diagram

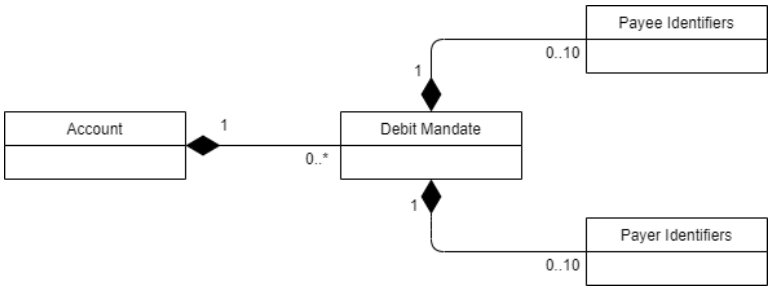

Figure 2-15 Debit Mandate UML Class Diagram

2.11.2 Debit Mandate Object Definition

Debit Mandate Object					
Name	**Type**	**Description**		**Reference**	**Validation**
mandateReference	string	Unique reference provided by the API Provider for the Debit Mandate.	→N/A ←M		
payee	array	A series of key/value pairs that enable the payee to	→O	Account Identifiers	

		be identified. Keys include MSISDN and Wallet Identifier.	←O		
mandateStatus	string	Indicates the status of the Debit Mandate as held in the API Provider system.	→O ←O		Enumeration = active, inactive
startDate	date	Date on which the mandate starts. If a frequencyType is specified, this will also be the date on which the first payment is to be taken.	→M ←M		
amountLimit	string	The maximum amount that can be taken by the Payee on a payment request.	→O ←O		Please refer to API Fundamentals document for amount validation rules.
currency	string	Currency of the amount limit.	→O ←O		Enumeration = ISO Currency Codes
endDate	date	Date on which the Debit Mandate ends.	→O ←O		
frequencyType	string	Indicates the frequency for which payments will be taken from the payers account.	→O ←O		Enumeration = Frequency
numberOfPayments	number	Indicates the number of consecutive payments that are to be taken.	→O ←O		
requestingOrganisation	object	The originating organisation of the request.	→O ←O	Requesting Organisation	
creationDate	date-time	Date and time when the Debit Mandate was created by the API Provider.	→NA ←O		
modificationDate	date-time	Date and time when the Debit Mandate was modified by the API Provider.	→NA ←O		
requestDate	date-time	The date and time of the debit mandate request as supplied by the client.	→O ←O		
customData	string	A collection of key/value pairs that can be used for provider specific fields.	→O ←O	Custom Data Object	

2.12 Links API

The Links APIs are used to establish a link between two separate accounts on the client and provider systems. The API can be used for example to link a mobile wallet account to a Microfinance Institution account or a bank account. The link object does not mandate the processes to verify and authenticate a link request - this depends upon the use case. A link needs to be associated with a mode of operation:

- **pull**. The link can be used by the client to debit the target account held by the provider.
- **push**. The link can be used by the client to credit the target account held by the provider.
- **both**. The link can be used for Push and Pull requests.

To identify the accounts that are to be linked, the target account is specified in the path whereas the source account is specified in the link object.

The permitted paths are as follows:

- **Creation**: POST */accounts/{identifierType}/{identifier}/links* or *POST /accounts/{ Account Identifiers}/links*.
- **Update** of status and/or mode fields: *PATCH /accounts/{identifierType}/{identifier}/links/{linkReference}* or *PATCH /accounts/{Account Identifiers}/links/{linkReference}*.
- **Read**. *GET /accounts/{identifierType}/{identifier}/links/{linkReference}* or *GET /accounts/{Account Identifiers}/links/{linkReference}*.

Synchronous and asynchronous modes are supported for POST and PATCH methods whereas only synchronous mode is supported for the GET method.

2.12.1 Link UML Class Diagram

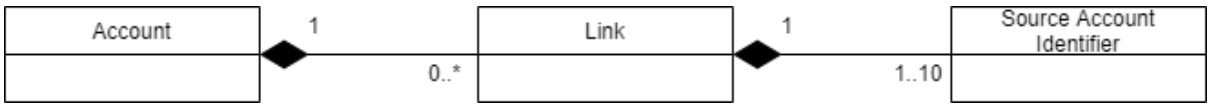

Figure 2-16 Link UML Class Diagram

2.12.2 Link Object Definition

Link Object					
Name	**Type**	**Description**		**Reference**	**Validation**
linkReference	string	Indicates the Link reference. This enables a linked account to be uniquely identified.	→NA ←M		
sourceAccountIdentifiers	array	A series of key/value pairs that identify the source	→M ←M	Account Identifiers	

		account. Keys include MSISDN and Wallet Identifier.			
mode	string	Indicates the mode of operation for the Link.	→M ←M		Enumeration = push, pull, both
status	string	Indicates the status of the Link.	→M ←M		Enumeration = active, inactive
requesting Organisation	object	The originating organisation of the request.	→O ←O	Requesting Organisation	
creationDate	date-time	Indicates when the link was created as recorded by the API provider.	→NA ←O		
modificationDate	date-time	Indicates when the link was modified as recorded by the API provider.	→NA ←O		
requestDate	date-time	The date and time of the link request as supplied by the client.	→O ←O		
customData	string	Contains provider specific fields.	→O ←O	Custom Data Object	

2.13 Authorisation Codes API

The Authorisation Codes APIs allow a payer to generate a payment code which when presented to the payee, can be redeemed for an amount associated with the code.

Authorisation codes can be set to expire. Note that expiry time can be specified via the API, however the mobile money provider may mandate a common expiry period for all codes.

Authorisation Codes are used widely in the industry across a range of use cases, including:

- ATM Codes for card-less withdrawals. A code is generated in advance by the customer and entered into the ATM to facilitate the withdrawal.
- Pre-authorised codes for agent withdrawals. A code is generated in advance by the customer and given to the agent to facilitate the withdrawal.
- Pre-authorised codes for merchant payments. The customer generates a code which can be redeemed at a merchant.

Once an authorisation code has been generated, it can be presented through multiple means, including encoding into a QR code. Typically, an authorisation code will expire.

The following paths are permitted:

- **Generate** an Authorisation Code. *POST /accounts/{identifierType}/{identifier}/authorisationcodes* or *POST /accounts/{Account Identifiers}/authorisationcodes*
- **Cancel** an Authorisation Code (*codeState* = 'cancelled')*. PATCH /accounts/{identifierType}/{identifier}/authorisationcodes/{authorisationCode}* or *PATCH /accounts/{Account Identifiers}/authorisationcodes/{authorisationCode}*.
- **View** An Authorisation Code for a given account. *GET /accounts/{identifierType}/{identifier}/authorisationcodes/{authorisationCode}* or *GET /accounts/{Requestor Account Identifiers}/authorisationcodes/{authorisationCode}*.
- **View** all Authorisation Codes for a given account. *GET /accounts/{identifierType}/{identifier}/authorisationcodes* or *GET /accounts/{Requestor Account Identifiers}/authorisationcodes*.

When retrieving authorisation codes, the following query string parameters can be used to filter the number of records returned:

Parameter	Type	Format	Description
limit	integer	N/A	Supports pagination. If this is not supplied, then the server will apply a limit of 50 records returned for each request.
offset	integer	N/A	Supports pagination. This value will indicate the cursor position from where to retrieve the set of records. For

			example, a limit of 50 and offset of 10 will return records 11 to 60.
fromDateTime	string	date-time	Indicates the minimum creationDate for which records should be returned.
toDateTime	string	date-time	Indicates the maximum creationDate for which records should be returned.
codeState	string	string	Allows filtering on the state of the authorisation code.

Note 1: For a harmonised behavior, API Providers should make sure that the authorisation codes are returned in descending date created order.

Note 2: HTTP response headers are returned with each response indicating the total number of records available (X-Records-Available-Count) and total number of records returned (X-Records-Returned-Count).

Synchronous and asynchronous modes are supported for the POST and PATCH methods whereas only synchronous mode is supported for the GET method.

2.13.1 Authorisation Code UML Class Diagram

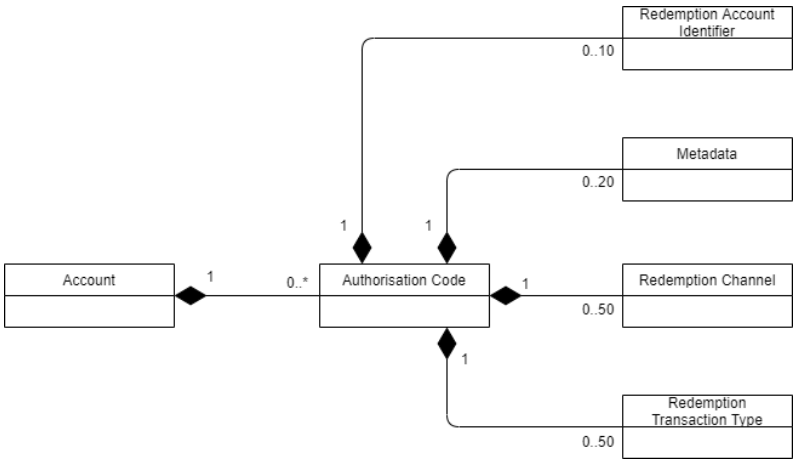

Figure 2-17 Authorisation Code UML Class Diagram

2.13.2 Authorisation Code Object Definition

Authorisation Codes Object					
Name	**Type**	**Description**		**Reference**	**Validation**
authorisation Code	string	The code that will be presented to the other party for redemption.	→NA ←M		
codeState	string	Indicates the state of the Authorisation Code.	→NA ←M		Enumeration = 'active', 'expired', 'cancelled'
amount	string	Indicates the amount associated with the	→O		Please refer to API

		authorisation code. Typically, this is set by the client.	←O		Fundamentals document for amount validation rules.
currency	string	Indicates the amount currency. Must be supplied when an amount is supplied.	→O ←O		Enumeration = ISO Currency Codes
amountType	string	The amount for the authorisation can be an exact amount or can be a maximum amount, i.e. redemption up to but not higher than the amount specified.	→O ←O		Enumeration = 'exact', 'maximum'
codeLifetime	integer	Indicates the expiry time in seconds of the code. Depending upon the use case, this can be set by the client or server.	→O ←O		If supplied, then must be 1 second or greater.
holdFundsIndicator	boolean	Indicates whether funds should be reserved against the payer's account where the payer is the requestor.	→O ←O		
redemptionAccountIdentifiers	array	A series of key/value pairs that identify the account where the code must be redeemed. Only needed if the redemption account needs to be explicitly stated.	→O ←O	Account Identifiers	
redemptionChannels	string	Indicates the channel(s) that the code can be redeemed against, e.g. ATM, Merchant, etc..	→O ←O	Channel Types Object	
redemptionTransactionTypes	string	Indicates the Transaction Types(s) that the code can be redeemed against.	→O ←O	Transaction Types Object	
requestingOrganisation	object	The originating organisation of the request.	→O ←O	Requesting Organisation	
creationDate	date-time	Indicates when the link was created as recorded by the API provider.	→NA ←O		
modificationDate	date-time	Indicates when the link was modified as	→NA ←O		

		recorded by the API provider.			
requestDate	date-time	The date and time of the request as provided by the client.	→O ←O		
customData	string	A collection of key/value pairs that can be used for provider specific fields.	→O ←O	Custom Data Object	
metadata	array	A collection of key/value pairs. These can be used to populate additional properties that describe administrative information regarding the authorisation code.	→O ←O	Metadata	

2.14 Quotations API

The Quotations APIs are used to obtain one or multiple quotes for a mobile money customer that wishes to transfer money. The creation of a quote involves returning any fees that will be levied on the sending customer and if the request is international, the forex rate. A request is made for a quotation by the requesting Service Provider in response to a customer request. The quotation is calculated and returned. If the customer is satisfied with the quotation, then they can confirm and proceed with a transaction request using the /transactions API.

The following paths are permitted:

- **Creation** of a quotation: *POST /quotations*
- **View** a quotation: *GET /quotations/{Quotation Reference}*

2.14.1 Quotation UML Class Diagram

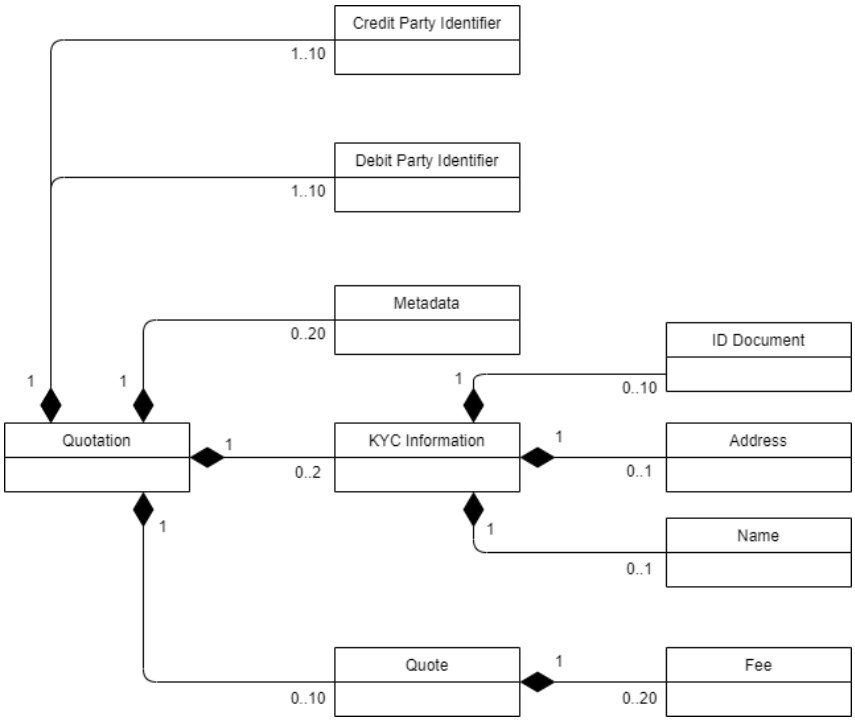

Figure 2-18 Quotation UML Class Diagram

2.14.2 Quotation Object Definition

Quotation Object Definition					
Name	**Type**	**Description**		**Reference**	**Validation**
quotationReference	string	Unique reference for the quotation as provided by the API Provider.	→NA ←M		
creditParty	array	A series of key/value pairs that enable the credit party to be	→M ←M	Account Identifier	

		identified. Keys include MSISDN and Wallet Identifier.			
debitParty	array	A collection of key/value pairs that enable the debit party to be identified. Keys include MSISDN and Wallet Identifier.	→M ←M	Account Identifier	
type	string	The transaction type that the quotation has been requested for.	→O ←O		Enumeration = Transaction Types
subtype	string	The transaction sub-type that the quotation has been requested for.	→O ←O		
quotationStatus	string	Indicates the creation state of the quotation.	→NA ←O		Enumeration = pending, rejected, completed
requestAmount	string	Requested Quotation amount.	→M ←M		Please refer to API Fundamentals document for amount validation rules.
requestCurrency	string	Currency of the requested quotation amount.	→M ←M		Enumeration = ISO Currency Codes
availableDelivery Method	string	Delivery Method that is possible for the intended recipient.	→NA ←O		Enumeration = Delivery Method
chosenDeliveryM ethod	string	The delivery method chosen by the sending end user as the specific delivery method to be used in the quotes received.	→O ←O		Enumeration = Delivery Method
originCountry	string	The originating country of the quotation request, i.e. the country where the request was initiated.	→O ←O		
receivingCountry	String	Destination country of the quotation request, i.e. the country that the sender wishes to send to.	→O ←O		
quotes	array	A collection of quotes. A quote can be received from a single receiving financial service provider or from multiple providers.	→NA ←O	Quotes	

recipientKyc	object	A collection of fields detailing the KYC of the transaction recipient, typically used for International Transfers.	→O ←O	KYC Information	
senderKyc	object	A collection of fields detailing the KYC of the transaction sender, typically used for International Transfers.	→O ←O	KYC Information	
recipientBlockingReason	string	The reason for blocking the quotation, based on AML checks on the recipient.	→NA ←O		
senderBlockingReason	string	The reason for blocking the quotation, based on AML checks on the sender.	→NA ←O		
requestingOrganisation	object	The originating organisation of the request.	→O ←O	Requesting Organisation	
sendingServiceProviderCountry	string	The country of the sending service provider that holds the account of the sender.	→O ←O		
creationDate	date-time	Date and time when the quotation was created by the API Provider.	→NA ←O		
modificationDate	date-time	Date and time when the quotation was modified by the API Provider.	→NA ←O		
requestDate	date-time	The date and time of the quotation request as supplied by the client.	→O ←O		
customData	string	A collection of key/value pairs that can be used for provider specific fields.	→O ←O	Custom Data Object	
metadata	array	A collection of key/value pairs. These can be used to populate additional properties that describe administrative information regarding the quotation.	→O ←O	Metadata	

3 Supporting Objects

3.1 International Transfer Information Object

The International Transfer Information object contains details that are specific to international money transfers.

International Transfer Information Object					
Name	**Type**	**Description**		**Reference**	**Validation**
quotationReference	string	Reference for the quotation that was provided to the sender. (refer to Quotations API for more information).	→O ←O		
quoteId	string	The specific quote associated with the quotation (refer to Quotes object for more information).	→O ←O		
originCountry	String	The originating country of the transaction, i.e. the country where the transaction was initiated.	→M ←M		Enumeration = ISO Country Codes
deliveryMethod	string	The recipient delivery method as chosen by the sender.	→O ←O		Enumeration = Delivery Method Types
receivingCountry	string	Destination country of the international transfer.	→O ←O		
relationshipSender	string	Indicates the relationship (if any) between the sender and the receiver.	→O ←O		
recipientBlockingReason	string	The reason for blocking the transaction, based on AML checks on the recipient.	→NA ←O		
senderBlockingReason	string	The reason for blocking the transaction, based on AML checks on the sender.	→NA ←O		
remittancePurpose	string	field providing a description of the reason for the international transfer.	→O ←O		
sendingServiceProviderCountry	string	The country of the sending service provider that holds the account of the sender.	→O ←O		

3.2 KYC Information Object

KYC refers to 'Know your Customer'. The KYC object contains a number of fields that enable the identity of subject to be verified. KYC can be provided with transfers for the sending identity and the receiving identity. There are no mandatory KYC object fields.

KYC Information Object					
Name	**Type**	**Description**		**Reference**	**Validation**
birthCountry	string	The country of birth of the KYC subject.	→O ←O		Enumeration = ISO Country Codes
dateOfBirth	date	Birth date of the KYC subject.	→O ←O		
contactPhone	string	Contact phone number (mobile or landline) of the KYC subject.	→O ←O		Must contain between 6 and 15 consecutive digits First character can contain a '+' or digit Can contain spaces.
emailAddress	string	Email address of the KYC subject.	→O ←O		
employerName	string	Employer name of the KYC subject.	→O ←O		
gender	string	Gender of the KYC Object.	→O ←O		Length=1, Enumeration = (m)ale, (f)emale, (u)nspecified
idDocument	array	An array of fields containing the forms of identification that are associated with the subject.	→O ←O	Id Document	
nationality	string	Nationality of the KYC subject.	→O ←O		Enumeration = ISO Country Codes
postalAddress	object	A collection of fields that details the postal address of the KYC subject.	→O ←O	Address	
occupation	string	Occupation of the KYC subject.	→O ←O		
subjectName	object	Refers to the name fields for the KYC subject.	→O ←O	Name	

3.3 Name Object

The name object identifies the name details for the subject identity.

Name Object					
Name	**Type**	**Description**		**Reference**	**Validation**
title	string	The given title of the KYC subject, e.g. Mr, Mrs, Dr.	→O ←O		
firstName	string	First name (also referred to as given name) of the KYC subject.	→O ←O		
middleName	string	Middle Name of the KYC subject.	→O ←O		
lastName	string	Surname (also referred to as last or family name) of the KYC subject.	→O ←O		
fullName	string	The full name of the KYC subject.	→O ←O		
nativeName	string	The full name expressed as in the native language.	→O ←O		

3.4 Id Document Object

As part of KYC information, identification documentation is normally required. The Id Document Object enables documents pertaining to a subject's identity to be described.

Id Document Object					
Name	**Type**	**Description**		**Reference**	**Validation**
idType	string	Indicates the type of identification for the KYC subject, e.g. passport, driving licence etc..	→M ←M		Enumeration = ID Types
idNumber	string	Reference pertaining to the type of identification for the KYC subject.	→O ←O		
issueDate	date	Date of issue for the identification document.	→O ←O		
expiryDate	date	Date of expiry for the identification document.	→O ←O		
issuer	string	Indicates the organisation/government entity that issued the ID document.	→O ←O		
issuerPlace	string	Place of issue for the identification type.	→O		

			←O		
issuerCountry	string	Country where the identification type was issued.	→O ←O		Enumeration = ISO Country Codes
otherIdDescription	string	Where an ID Type of 'otherid' is specified, a description of the type of identification can be provided in this field.	→O ←O		

3.5 Address Object

The address object holds the postal address of the subject. Due to variability of address information in a number of mobile money markets, only country is mandatory.

Address Object					
Name	**Type**	**Description**		**Reference**	**Validation**
addressLine1	string	First line of the address.	→O ←O		
addressLine2	string	Second line of the address.	→O ←O		
addressLine3	string	Third line of the address.	→O ←O		
city	string	City/Town.	→O ←O		
stateProvince	string	State or Province.	→O ←O		
postalCode	string	Postal Code.	→O ←O		
country	string	Country.	→M ←M		Enumeration = ISO Country Codes

3.6 Account Identifier Object

The Account Identifier object enables one or multiple identifiers to be provided to enable the recipient system to resolve the account/party.

Account Identifier Object					
Name	**Type**	**Description**		**Reference**	**Validation**
key	string	Provides the account identifier type.	→M ←M		Enumeration = Account Identifiers

value	string	Provides the account identifier type value.	→M ←M		

3.7 Identity Object

The Identity object defines the information for an identity associated with an account. Between one and twenty identities can be associated with an account.

Identity Object					
Name	**Type**	**Description**		**Reference**	**Validation**
identityId	string	A unique id for the identity as assigned by the API Provider.	→NA ←M		
identityType	string	Indicates the type of the identity. Currently, only 'individual' is supported.	→NA ←M		'individual'
identityStatus	string	A non-harmonised field describing the status of the identity.	→NA ←O		
identityKyc	object	A collection of fields detailing the KYC held for the identity.	→M ←M	KYC Information	
accountRelationship	string	Describes the relationship that the identity holds with the account.	→M ←M		'accountholder'
kycVerificationStatus	string	Indicates the status of the identity's KYC verification.	→O ←O		'verified', 'unverified', 'rejected'
kycVerificationEntity	string	Indicates the entity (e.g. mobile money agent) that has verified the KYC of the identity.	→O ←O		
kycLevel	integer	Indicates the KYC level that the identity is associated with.	→O ←O		
customData	array	A collection of key/value pairs that can be used for provider specific fields.	→O ←O		

3.8 Quote Object

Quotations can consist of multiple quotes. The fields for a quote are defined in the object.

Quote Object					
Name	**Type**	**Description**		**Reference**	**Validation**
quoteId	string	The unique ID for this quote.	→NA ←M		

receivingAmount	string	The total amount as it will be received by the receiving end user.	→NA ←M		Please refer to API Fundamentals document for amount validation rules.
receivingCurrency	string	The currency of the quote.	→NA ←M		Enumeration = ISO Currency Codes
sendingAmount	string	Requested quotation amount as supplied by the sender.	→NA ←M		Please refer to API Fundamentals document for amount validation rules.
sendingCurrency	string	Currency of the requested quotation amount.	→NA ←M		Enumeration = ISO Currency Codes
deliveryMethod	string	The delivery method that is applicable to the quotation.	→NA ←O		Enumeration = Delivery Method
fees	array	Returns all fees that are applicable to the quote.	→NA ←O	Fees Object	
fxRate	string	The conversion rate applicable between the sending and the receiving currency for the requested transaction.	→NA ←O		Please refer to API Fundamentals document for amount validation rules. Note 10 decimal places supported.
quoteExpiryTime	date-time	The timestamp when the quote will expire.	→NA ←O		
receivingServiceProvider	string	The name of the receiving service provider, i.e. the provider that the quote is associated with.	→NA ←O		

3.9 Metadata Object

The metadata object allows fields to be specified to convey administrative information regarding the associated resource in the form of key/value pairs. Additional fields should only be used where no suitable defined field match can be found. The number of key/value pairs is limited to 20.

Metadata Object					
Name	**Type**	**Description**		**Reference**	**Validation**
key	string	Identifies the type of additional fields.	→M ←M		
value	string	Identifies the value of the additional field.	→M ←M		

3.10 Custom Data Object

The custom data object allows additional fields to be specified for the associated resource in the form of key/value pairs. Additional fields should only be used where no suitable defined field match can be found. The number of key/value pairs is limited to 20.

Custom Data Object					
Name	**Type**	**Description**		**Reference**	**Validation**
key	string	Identifies the type of additional fields.	→M ←M		
value	string	Identifies the value of the additional field.	→M ←M		

3.11 Supplementary Bill Reference Object

This object enables additional payment references to be specified for a bill payment in the form of key/value pairs. Additional fields should only be used where no suitable defined field match can be found. The number of key/value pairs is limited to 20.

Supplementary Bill Reference Object					
Name	**Type**	**Description**		**Reference**	**Validation**
paymentReferenceType	string	Identifies the type of the additional payment reference.	→M ←M		
paymentReferenceValue	string	Identifies the value of the additional payment reference.	→M ←M		

3.12 Transaction Type Object

This object enables multiple transaction types to be specified along with paired sub-types. This object is used where multiple transaction types need to be passed in an API.

Transaction Type Object					
Name	**Type**	**Description**		**Reference**	**Validation**

transactionType	string	Identifies the Transaction Type.	→M ←M		Enumeration = Transaction Types
transactionSubType	string	Identifies the Transaction Sub-Type.	→O ←O		

3.13 Channel Type Object

This object enables multiple channel types to be specified. This object is used where multiple channel types need to be passed in an API.

Channel Type Object					
Name	**Type**	**Description**		**Reference**	**Validation**
channelType	string	Identifies the Channel Type.	→M ←M		

3.14 Fees Object

An object that enables fees differentiated by type to be provided and/or returned.

Fees Object					
Name	**Type**	**Description**		**Reference**	**Validation**
feeType	string	Defines the type of fee.	→M ←M		
feeAmount	string	Defines the amount of the fee.	→M ←M		Please refer to API Fundamentals document for amount validation rules.
feeCurrency	string	Defines the currency for the given fee.	→M ←M		Enumeration = ISO Currency Codes

3.15 Commission Object

An object that enables earned commission that is calculated by the API provider to be returned.

Commission Object					
Name	**Type**	**Description**		**Reference**	**Validation**
commissionType	string	Defines the type of commission.	→M ←M		

commission Amount	string	Defines the amount of the commission.	→M ←M		Please refer to API Fundamentals document for amount validation rules.
commission Currency	string	Defines the currency of the commission.	→M ←M		Enumeration = ISO Currency Codes

3.16 Requesting Organisation Object

An object that details the originating organisation of the request.

Requesting Organisation Object					
Name	**Type**	**Description**		**Reference**	**Validation**
requestingOrganisationIdentifierType	string	Identifies the identifier type of the requesting organisation.	→M ←M		'swiftbic', 'lei', 'organisationid'
requestingOrganisationIdentifier	string	Contains the requesting organisation identifier.	→M ←M		

4 Enumerations

4.1 ISO Currency Codes

The three-character alphabetic code for currency as defined by ISO 4217 is to be used for all currency fields. The full list of codes is maintained by Swiss Interbank Clearing on behalf of the International Organisation for Standardisation. This list can be obtained via the following website - http://www.currency-iso.org/en/home/tables/table-a1.html.

4.2 Transaction Types

A transaction type is used to classify the nature of a transaction.

Code	Description
billpay	Payment of bill from a business for goods and/or services.
deposit	Exchange of cash in return for e-Money at a physical agent or via ATM.
disbursement	Disbursement of funds (making payments from an organisation (business, NGO, government entity) to a mobile money recipient.
transfer	Transfer of funds between mobile money provider and another provider or financial institution in the same country.
merchantpay	Purchases of goods and/or services from shops (payer present) or online (payer not present).
inttransfer	Transfer of funds to a recipient in another country, either directly to/from a mobile wallet or via an international money transfer provider.
adjustment	General adjustments to an account via an adjustment transaction (e.g. refunds).
reversal	Reversal of a prior transaction to return funds to the payer.
withdrawal	Exchange of e-Money in return for cash at a physical agent or via ATM.

4.3 ID Types

The ID Types enumeration contains accepted identification types. Due to the wide international variation in accepted types of identification, a catch-all type of ‘otherid’ is included.

ID Type	Description
passport	Payment of bill from a business for goods and/or services.
nationalregistration	National Registration Number.
othered	Catch-all for IDs not on the list.
drivinglicence	Driving Licence Number.
socialsecurity	Social Security Number.
alienregistration	Alien Registration ID.
nationalidcard	National Identity Card.
employer	Employers Identification.

taxid	Tax Identification Number.
seniorcitizenscard	Senior Citizens ID Card.
marriagecertificate	Marriage Certificate.
birthcertificate	Birth Certificate.
healthcard	Health Card.
votersid	Voters Identification.
villageelderletter	Letter of confirmation from village elder.
pancard	Credit/debit card number (Primary Account Number).
officialletter	Official letter confirming identity.

4.4 Account Identifiers

The Account Identifier enumeration lists all possible means to identify a target account. Identifiers can be combined if necessary, to provide a unique identifier for the target account.

Code	Short Description	Type	Description
accountcategory	Account Category	string	Can be used to identify the sources of funds category where there are multiple accounts (wallets) held against an account holder.
bankaccountno	Bank Account Number	string	Financial institution account number that is typically known by the account holder.
accountrank	Account Rank	string	Is used to identify the rank of the source of funds where there are multiple accounts (wallets) held against an account holder.
identityalias	Identity Alias	string	An alias for the identity, e.g. short code for an agent till.
iban	IBAN	string	Internationally agreed system of identifying bank accounts across national borders to facilitate the communication and processing of cross border transactions. Can contain up to 34 alphanumeric characters.
accountid	Account Holder Identity	string	Identifier for the account holder.
msisdn	MSISDN	string	Must contain between 6 and 15 consecutive digits First character can contain a '+' or digit Can contain spaces.

swiftbic	SWIFTBIC	string	A bank identifier code (BIC) is a unique identifier for a specific financial institution. A BIC is composed of a 4-character bank code, a 2-character country code, a 2-character location code and an optional 3-character branch code. BICs are used by financial institutions for letters of credit, payments and securities transactions and other business messages between banks. Please refer to ISO 9362 for further information.
sortcode	Bank Sort Code	string	Sort code to identify the financial institution holding the account.
organisationid	Organisation Account Identifier	string	Used to identify the organisation for which a payment is to be made.
username	Username	string	Used to identify target account via an associated username.
walletid	Wallet Identifier	string	A means to identify a mobile money wallet, particularly where multiple wallets can be held against an MSISDN. typically used in conjunction with MSISDN or identity alias to identify a particular wallet.
linkref	Link Reference	string	A means to uniquely identify an account via an account to account link. E.g. wallet account link to bank account.
consumerno	Consumer Number	String	Identifies the consumer associated with the account.
serviceprovider	Service Provider	String	Provides a reference for a Service Provider.
storeid	Store ID	String	Identifies the transacting store / retail outlet.
bankname	Bank Name	String	Name of the bank.
bankaccounttitle	Bank Account Title	String	The title of the bank account.
emailaddress	Email Address	String	emailaddress of the party.
mandatereference	Debit Mandate Reference	String	A means to identify an account via a debit mandate reference.

4.5 ISO Country Codes

The two-character alphabetic code for country as defined by ISO 3166 is to be used for all fields specifying a country or nationality. The full list of codes is maintained by the International Organisation for Standardisation. The list can be obtained via the following website - http://www.iso.org/iso/country_codes.

4.6 Delivery Method Types

When a customer requests a quotation they are able to specify their preferred method of delivery of the transfer to the recipient. Permitted delivery methods are provided below.

Delivery Method	Description
directtoaccount	The transfer is to be delivered into the account (wallet) of the recipient.
agent	The recipient can visit an agent and get the transferred funds.
personaldelivery	a supplementary service where an authorised person can deliver the funds, in hand, to the receiving end user.

4.7 Frequency Type

When requesting a debit mandate, the API client is able to specify the frequency of which the payment should be taken. Valid values are defined in the table below.

Frequency Type	Description
weekly	Payment will be taken weekly.
fortnight	Payment will be taken every two weeks.
monthspecificdate	Payment to be taken on a specific date every month.
twomonths	Payment to be taken every two months.
threemonths	Payment to be taken every three months.
fourmonths	Payment to be taken every four months.
sixmonths	Payment to be taken every six months.
yearly	Payment to be taken yearly.
lastdaymonth	Payment to be taken on the last calendar day of the month.
lastdaymonthworking	Payment to be taken on the last working day of the month according to working days as per the resident country of the account.
lastmonday	Payment to be taken on the last Monday of the month.
lasttuesday	Payment to be taken on the last Tuesday of the month.
lastwednesday	Payment to be taken on the last Wednesday of the month.
lastthursday	Payment to be taken on the last Thursday of the month.
lastfriday	Payment to be taken on the last Friday of the month.
lastsaturday	Payment to be taken on the last Saturday of the month.
lastsunday	Payment to be taken on the last Sunday of the month.

specificdaymonthly	Payment to be taken on a specific day of the month.

This document is produced by the GSMA with input from the GSMA Mobile Money API Working Group. It is our intention to provide a quality product for your use. If you find any errors or omissions, please contact us with your comments. You may notify us at support.mmapi@gsma.com.