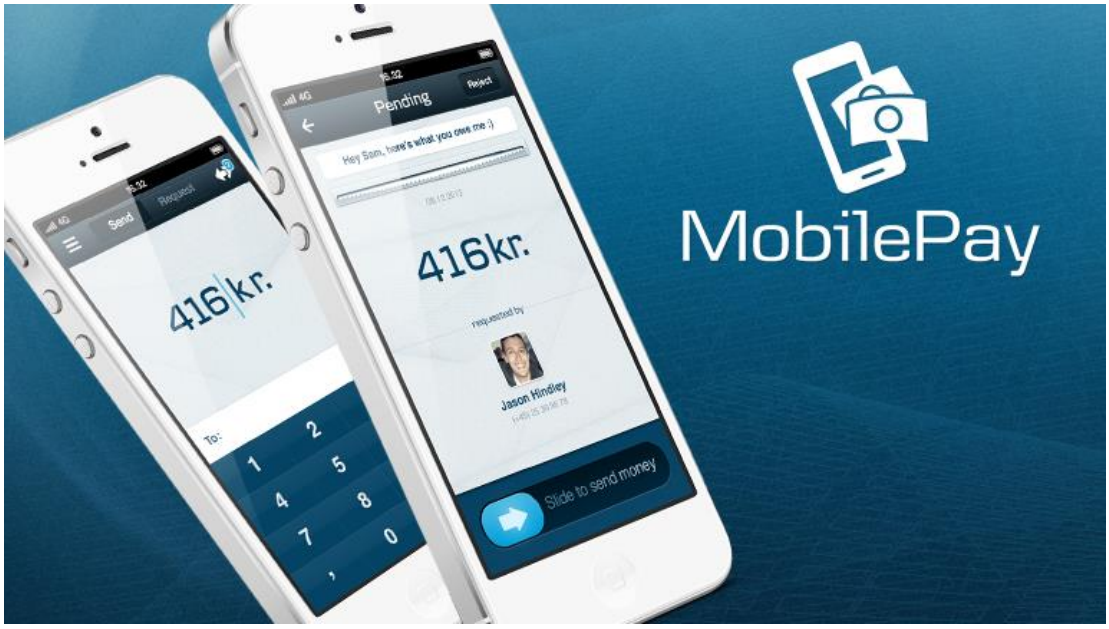


MobilePay AppSwitch SDK Implementation Guide

Version 1.3

November 2016

Contents

1 Document log

Version	Date	Amendment
1.0	15.12.2015	TKI: Document created
1.1	06.01.2016	TKI/DUY: Corrections to text and layout
1.2	29.06.2016	TKI: Callback URL in SDK 1.8.0 added to chap 4.2
1.3	02.11.2016	RIRAD: Corrections to naming.

2 Purpose of this guide

This implementation guide explains the implementation design of MobilePay AppSwitch. This includes descriptions of MobilePay AppSwitch communication, SDK error scenarios, different payment scenarios etc.

2.1 Target groups

The target group is project managers, system architects, and developers.

Implementation details such as coding details and platform specific details (iOS, Android and Windows Phone) are not part of this guide, but can be found on GitHub under MobilePay's AppSwitch SDK repository.

2.2 Technical support

For technical questions, please contact mp_appswitch@mobilepay.dk .

Please prepare the following scheme when requesting support.

Subject	**Description / comments**
Error	*[Headline/title]*
Description	*[Error message with a short description]*
Service	*[Which service is affected?]*
Screenshot	*[Screen shot if possible]*
Date/time	*[Timestamp]*
Mobile number	*[Registered mobile number of user]*
E-mail address	*[Registered email address of merchant]*
Platform	*e.g. iOS*
OS version	*e.g. iOS 8.1*
Merchant app version	*e.g. Version 3.60*
OrderID	*e.g. 2015-06-08 000001*
TransactionID	*e.g. 1089237509*

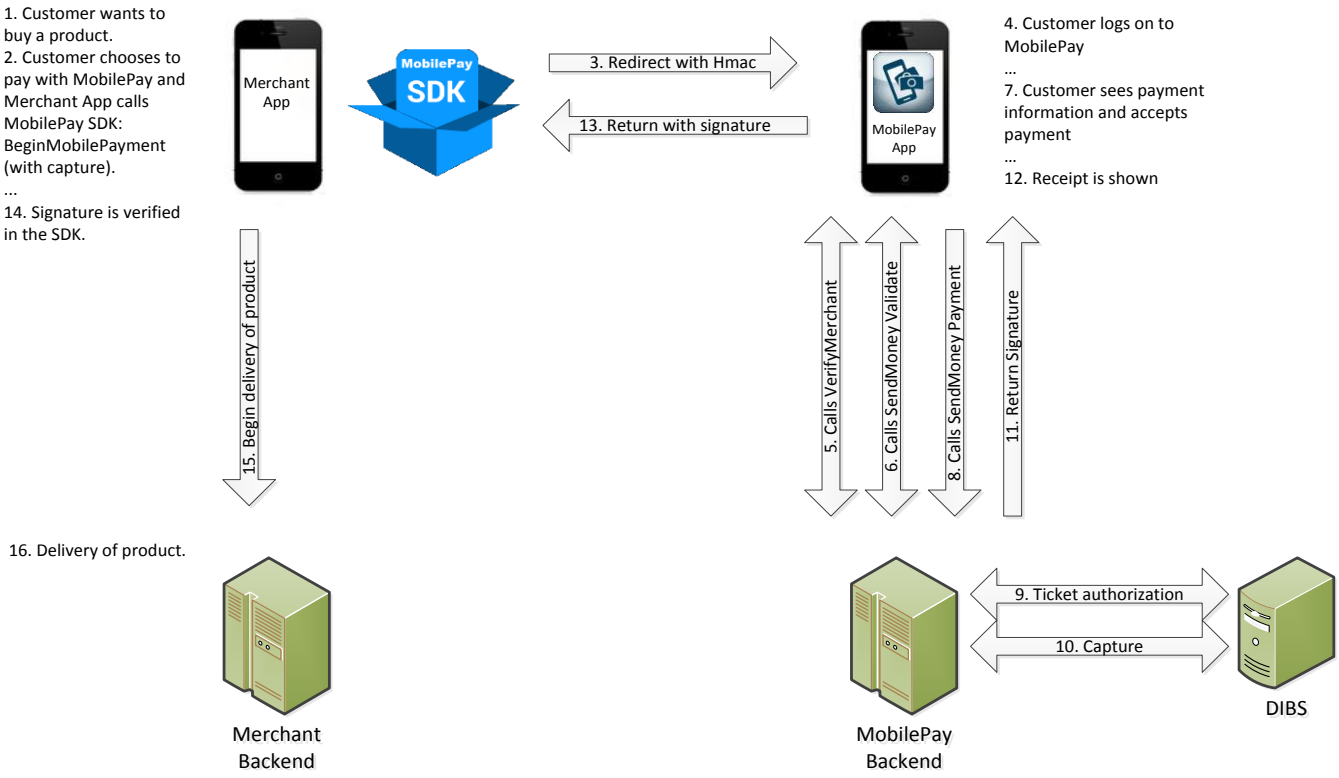
3 MobilePay AppSwitch SDK

The MobilePay AppSwitch SDK has been developed to make it easy for app developers to integrate MobilePay in their own apps. Embedding the MobilePay AppSwitch SDK will enable the merchant app to receive payments through the MobilePay app.

Three platforms are supported; iOS, Android and Windows Phone. Documentation is provided on GitHub and includes further implementation details, which is not part of this guide.

The figure below illustrates the app-only solution of MobilePay AppSwitch. The integrity and identity is guarded by an HMAC authentication and a signature validation, respectively. The HMAC is calculated in the SDK and sent to the MobilePay backend for verification (step 5 in below figure). This is done for all calls in MobilePay. The signature validation is required only when CaptureType = "Y".

Please note that this solution is best suited for smaller businesses as no lifecycle management of the transaction (via backend) is possible.



4 Communication

Communication steps

The merchant app communicates with MobilePay AppSwitch SDK and typically also with a merchant backend.

The MobilePay app communicates with MobilePay AppSwitch SDK and MobilePay Backend.

Selection of payment option MobilePay in the merchant app implies that the MobilePay AppSwitch SDK redirects to the MobilePay app. This redirection includes verification at the MobilePay backend of current merchantID (ID provided by MobilePay) and specific secure data (HMAC calculation based opon an agreed key) used for validation of current message content sent from merchant app.

The customer follows the well-known MobilePay steps as logon, payment approval (swipe) and payment completion.

When the requested payment is completed in MobilePay, a signature is returned to the merchant app. The signed information contains the original merchant orderID and the MobilePay transactionID. The MobilePay AppSwitch SDK validates the signature and ensures that the orderID given to the MobilePay AppSwitch SDK by the merchant app matches the orderID received from MobilePay (Note: signature validation is only active for instant captures).

Services

The MobilePay app uses different services to communicate with the MobilePay backend.

The services can be divided into two types of services namely, authenticated and not authenticated services. Authenticated services require PIN code, installed MobilePay app id and related MobilePay mobile phone number. Not authenticated services include services for registration of new users, getting app release information, Help texts, Contact information etc.

Examples of MobilePay app services are:

1. VerifyMerchant: Verifies that the HMAC and the merchant ID are correct.
2. Logon: Enable MobilePay access (authenticated)
3. SendMoneyAppSwitch: Transfers the money from the customer to the merchant account. This step involves creation of a transaction id (payment id) which is a signature on the payment. The SendMoneyAppSwitch service has two actions: VALIDATE and PAYMENT. The VALIDATE action verifies whether the customer can do the payment (ability in relation to the amount of payment and paycard status), but no payment is done. The PAYMENT action executes the actual payment. The VALIDATE action will also verify if current specified OrderID (provided from Merchant App) in current payment request has already been paid. If already paid, the payment details and signature will be returned (authenticated).
4. ShowList: Returns all the customers' transactions in MobilePay (authenticated).

5. ShowDetails: Returns the payment receipt and transaction id in MobilePay (authenticated).
6. AppVersion: Returns the minimum MobilePay app version required.

All security operations are executed in the MobilePay backend and not in the MobilePay app.

Furthermore, there is no need for any user administration or access control for the merchant to servers at MobilePay and there is no need for the merchant to host servers that MobilePay needs access to.

4.1 Communication between the merchant app and MobilePay app

The data exchange between the merchant app and MobilePay app and vice versa is defined in sections 6.1 and 6.2, respectively. The merchant app utilises the MobilePay AppSwitch SDK to start a payment that will redirect the payment inputs to the MobilePay app.

The MobilePay app validates the input and requests the user to login. After login the user will be presented with a payment request once a payment validation has taken place (user looked up in DIBS regarding ability to pay with current pay card in MobilePay).

When the user accepts (swipes), the payment is authorised at DIBS. The parameter "Capture" from the SDK is either 'Y' (yes) or 'N' (no). If it is "yes" the payment is captured instantly after the authorisation. If it is "no" the payment is not captured at this stage – the payment has been authorized and reserved.

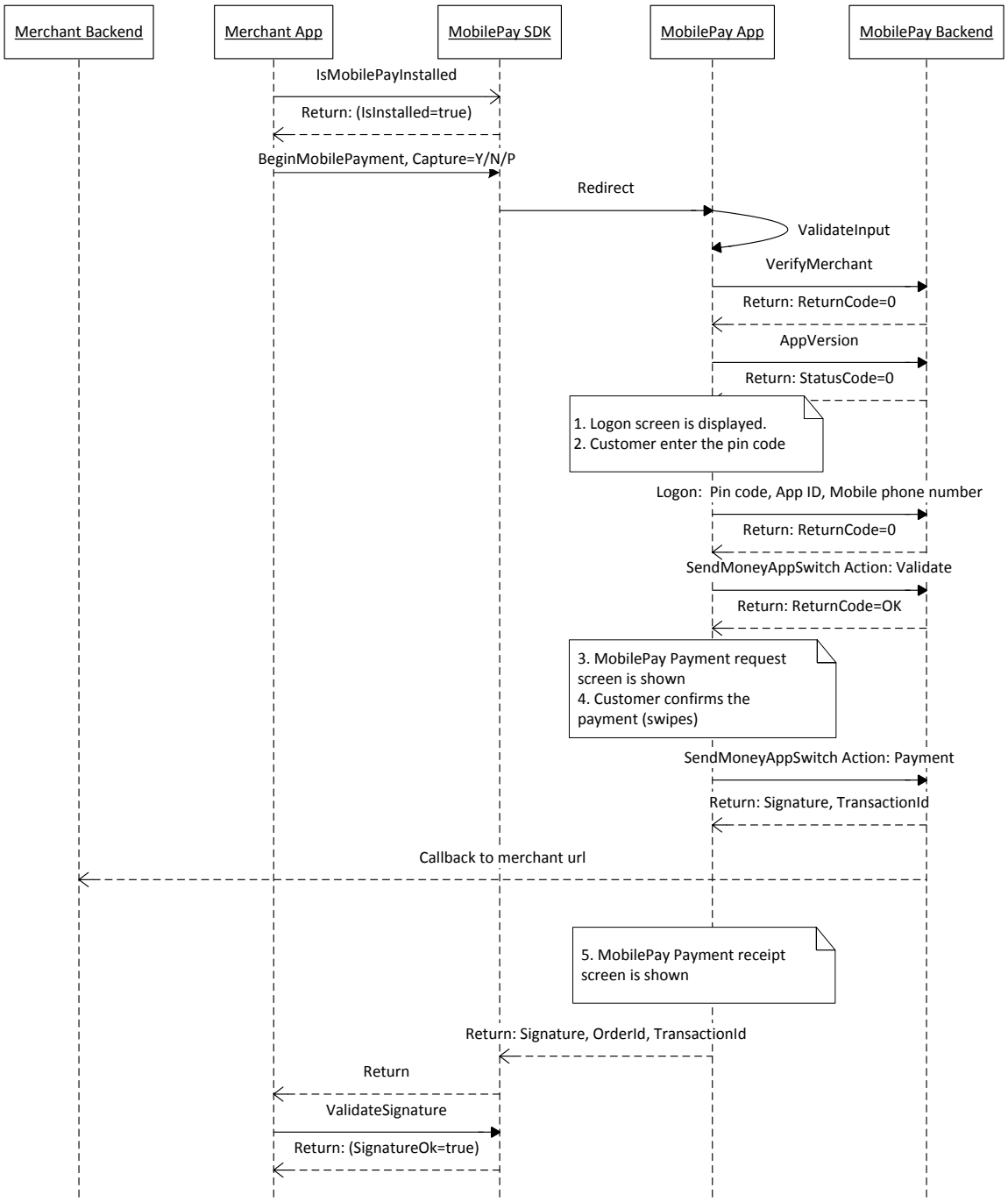
The request is confirmed in MobilePay backend – a response receipt is sent to the MobilePay app. The MobilePay app will then redirect back to the merchant app with a signature of the payment.

If the transaction is captured, not reserved, the signature will contain the status of the payment.

4.2 Communication between MobilePay backend and merchant backend

From SDK version 1.8.0 it is possible to set an url to which the MobilePay backend can send a string containing the initial status and details of the current transaction. An example of a reply returned to a merchant url:

https://example.com/appswitchstatus?TimeStamp=2016-06-29T08:43:42.939&OrderId=1122334455&MerchantId=APPDK1175851001&TransactionId=1234567890&Amount=00001500&Currency=DKK&Country=DK&PaymentStatus=RES&ReturnCode=00&ReasonCode=00



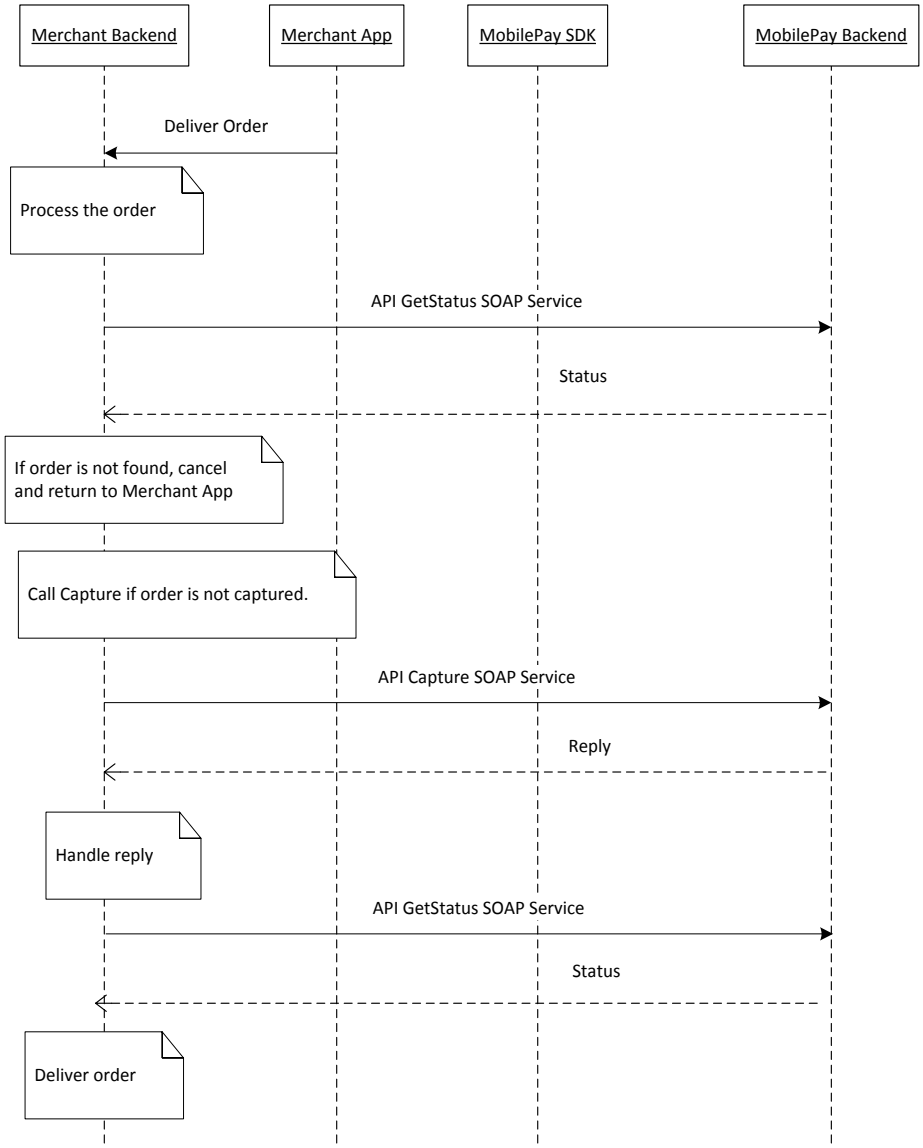
NOTE: Only for merchants using a backend setup

If "capture" is "N" or "P" the merchant backend can process the order and just before delivery of the order, the merchant backend may call the GetStatus service to verify the authenticity of the payment, and then the Capture service at MobilePay to finalise the payment.

If these steps are successful, the order can be delivered.

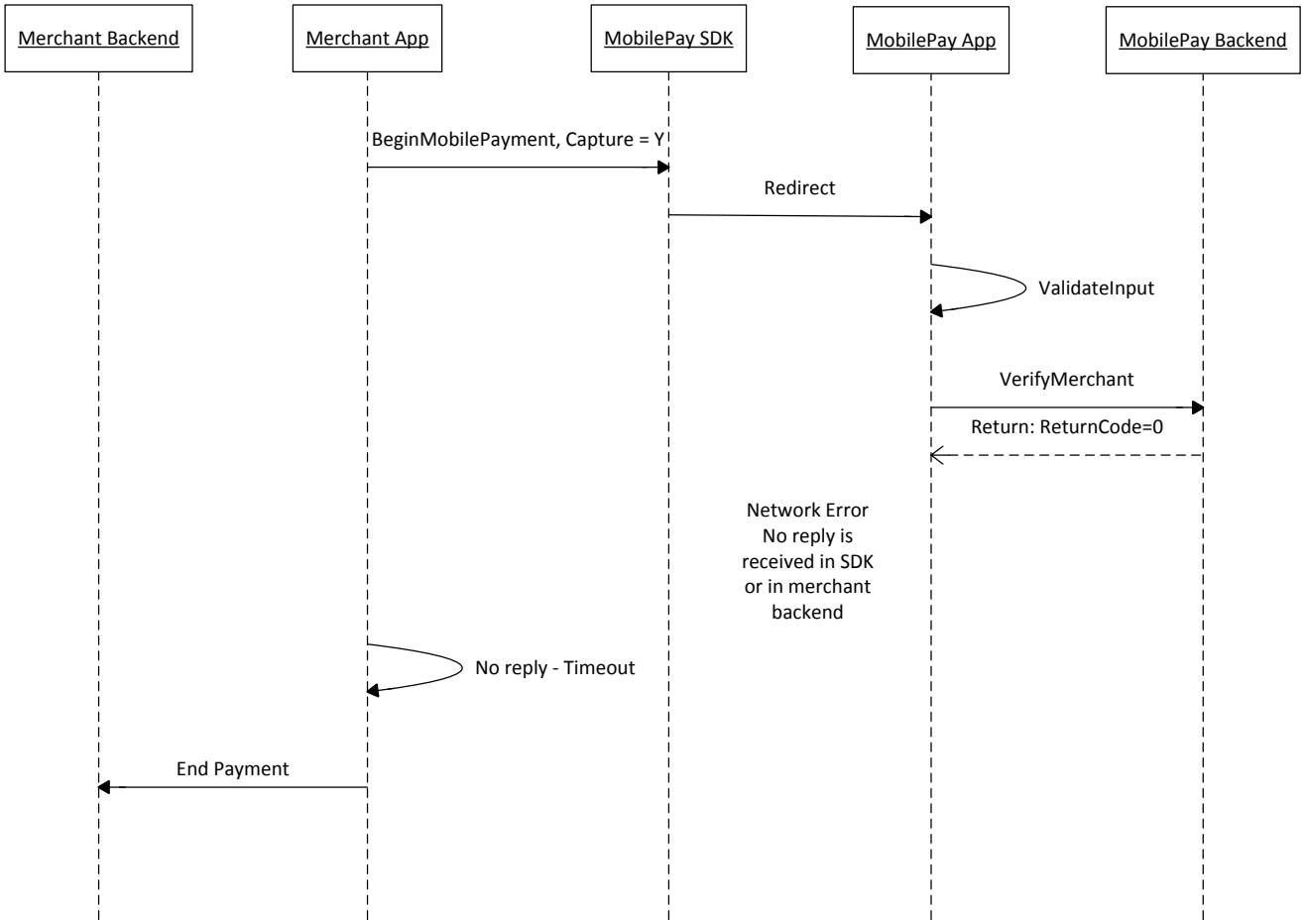
However, if the order cannot be delivered, but a reservation has been made, the merchant backend may call the Cancel service at MobilePay to cancel the authorisation, and hence the payment. The reservation made on the current card is then deleted. If the merchant does not call the Cancel service, the reservation will automatically be cancelled when it expires (default is seven days).

Once a capture has been performed, it is not possible to call the Cancel service. In this case the Refund service can be used.



4.3 Order has been paid in MobilePay but merchant app does not get a reply

We are currently working on a solution that allows merchants to ask for a status through the SDK. For a more robust solution, it is recommended to implement the full MobilePay AppSwitch suite thus incorporating a backend setup.



4.4 How to avoid double payments

The MobilePay app has a built in validation which ensures that the customer will not accidently pay for the same service/product twice.

1. If the merchant app sends a new order ID to the MobilePay app, the customer will be able to pay for the order, and the money will be transferred to the merchant's account.

2. If the merchant app sends an order ID to the MobilePay app (within 24 hours), which the customer has already paid for, the payment signature including the existing transaction ID (payment ID) will be returned without a new payment has taken place.

MobilePay will ensure a one to one relation between these artefacts:

- Order ID (from merchant app)
- Actual payment (money transfer)

Normally, there will also be a one to one relation between Order ID and the transaction ID:

- Transaction ID (payment ID from DIBS).

However, if the payment is refunded or created more than 24 hours after the initial payment, a new transaction ID will be created. In this case two transaction IDs will exist for the same order ID. In this case, it will always be possible to distinguish between the current transaction ID and the original transaction ID.

The customer is redirected from WIFI to phone network if any network issues occur. The network might resend the transaction, but double payments will not occur, because the second payment will be rejected due to the same order ID not being allowed for payment confirmation.

4.5 How to ensure authentication of the calling merchant app

The merchant is registered in the MobilePay backend with an AppSwitch ID (also called merchant ID). This ID is attached to a Business Online agreement. It will not be possible for merchants to get an AppSwitch ID without having a Business Online agreement with MobilePay.

The MobilePay backend will verify the merchant in the following steps:

- Is the merchant ID registered in the MobilePay backend database?
- Is it registered as being active?
- Is the HMAC correct (please refer to chapter 6 - "Security")?

4.6 How to ensure authentication of the payment

A MobilePay AppSwitch payment returns a digital signature to the calling merchant app. This signature ensures the authentication of the payment, because the MobilePay private key has been used for signing and therefore the MobilePay public key can be used for verification at the merchant side. The public key is contained in a certificate and exchanged in a secure manner separate from the payment transaction.

NOTE: Only for merchants using a backend setup

In case the merchant setup is depending on a backend solution (e.g. if merchant wants to do a reserve / capture setup), the authentication of the payment will be ensured by calling the GetStatus service. This service will return the status of the current transaction, and thus allow for a more specific handling of the transaction.

5 Error codes

The following sections describe the different error codes that can be returned from the MobilePay AppSwitch SDK. NOTE: These error codes must not be mistaken for the API error codes available in the 'Merchant API description' document.

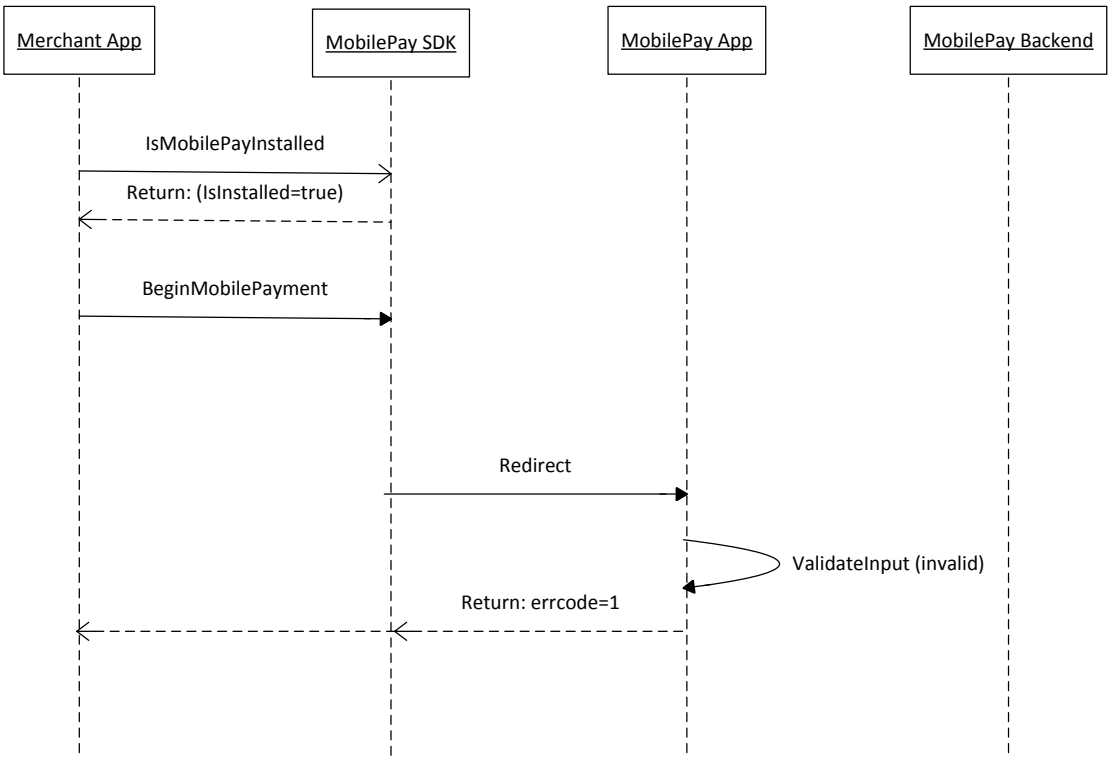
The error codes from the MobilePay AppSwitch SDK are listed in the table below:

No.	Type
1	Invalid parameters to MobilePay app
2	Validate merchant request fails
3	MobilePay app version is out of date
4	Merchant ID is not valid
5	HMAC parameter is not valid
6	MobilePay timeout
7	MobilePay amount exceeded
8	Timeout set in merchant app exceeded
9	Invalid signature. This means that the payment is invalid - MobilePay has not signed it.
10	MobilePay AppSwitch SDK version is outdated
11	The order ID sent to MobilePay has already been used for a confirmed payment by the same merchant (within 24 H).
12	MobilePay user is screened for fraud behavior.

5.1 Invalid parameters to MobilePay app

The MobilePay AppSwitch SDK will return error code no. 1 to the merchant app if the input sent to the MobilePay app is invalid. The input can be invalid if e.g. the price is lower than 0 or if a required input is missing.

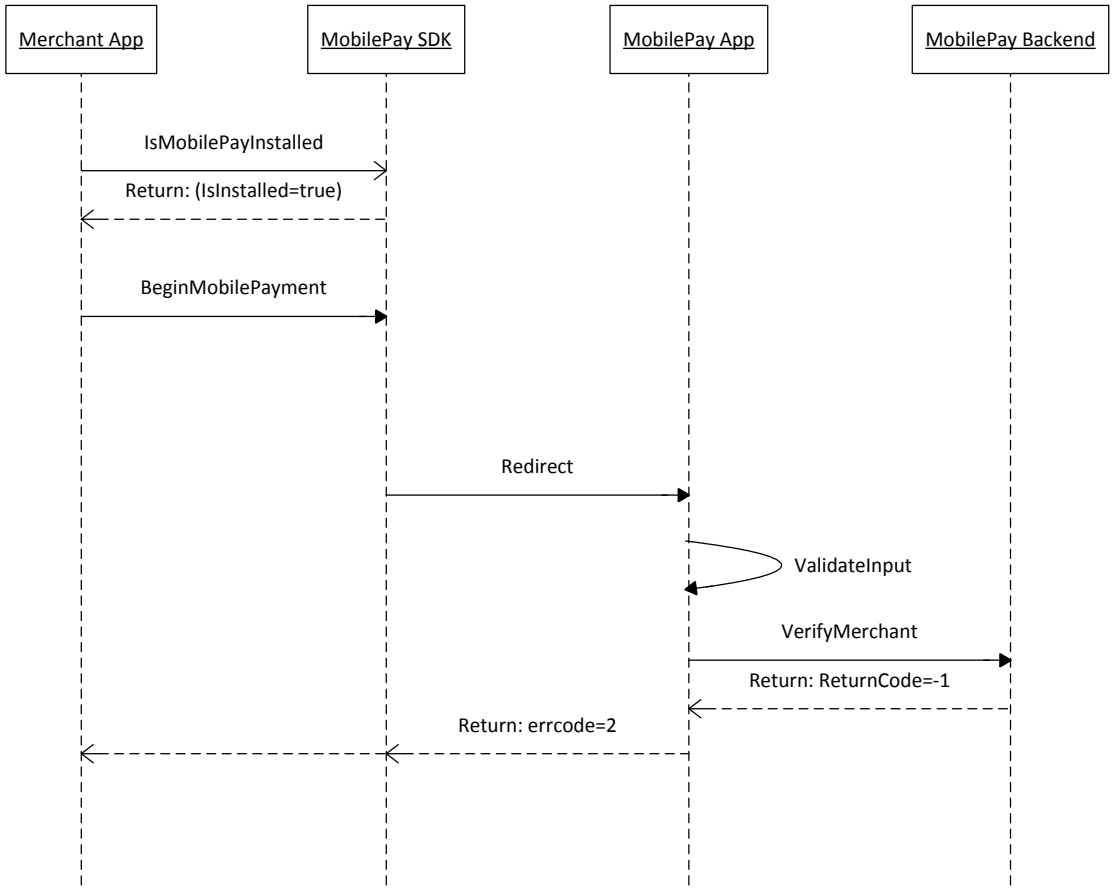
This error code should never be received in the merchant app in production. The error code should be handled in the merchant app by showing a message to the user and create a log in the merchant backend (if this is possible) that a problem with initiating a payment request occurred.



5.2 Validate merchant request fails

The MobilePay AppSwitch SDK will return error code no. 2 to the merchant app if the validation of the merchant failed due to network failure or timeout. The MobilePay app will validate the merchant ID sent to it by making a validation request to the MobilePay backend.

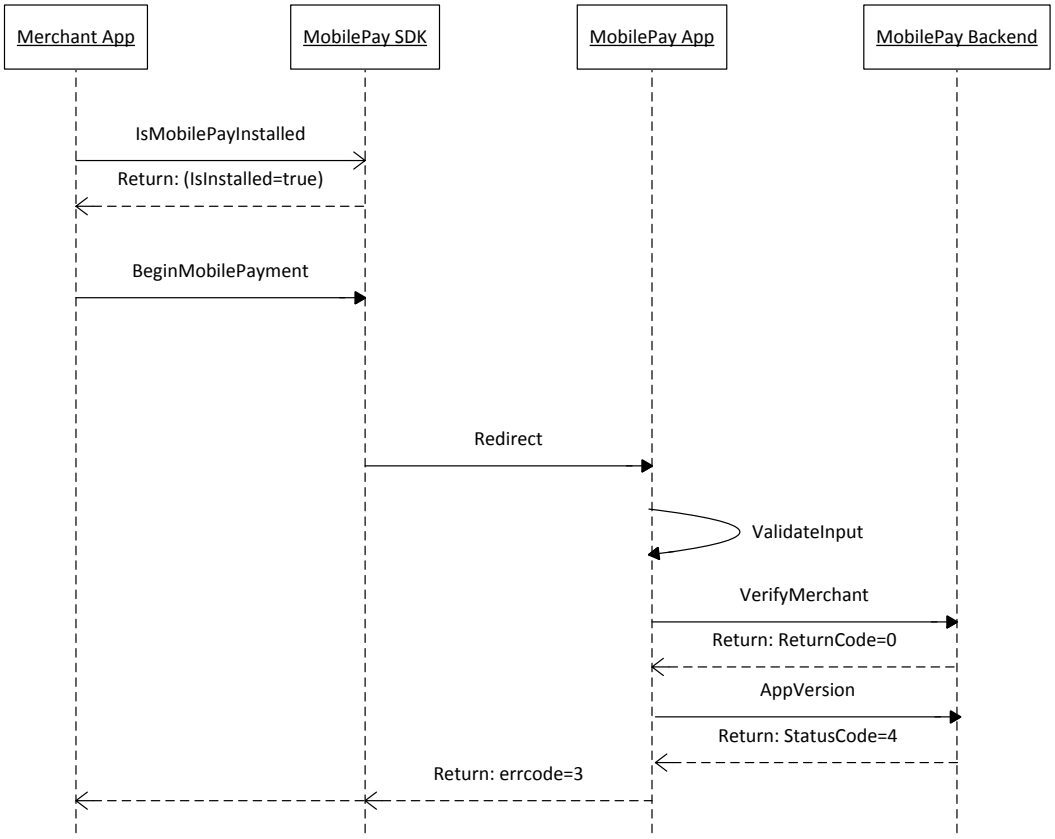
This error code should be handled in the merchant app by showing a message asking the user to check network connectivity and try again.



5.3 MobilePay app is out of date and must be updated

The MobilePay AppSwitch SDK will return error code no. 3 if the MobilePay app must be updated which can be due to security or legal requirements.

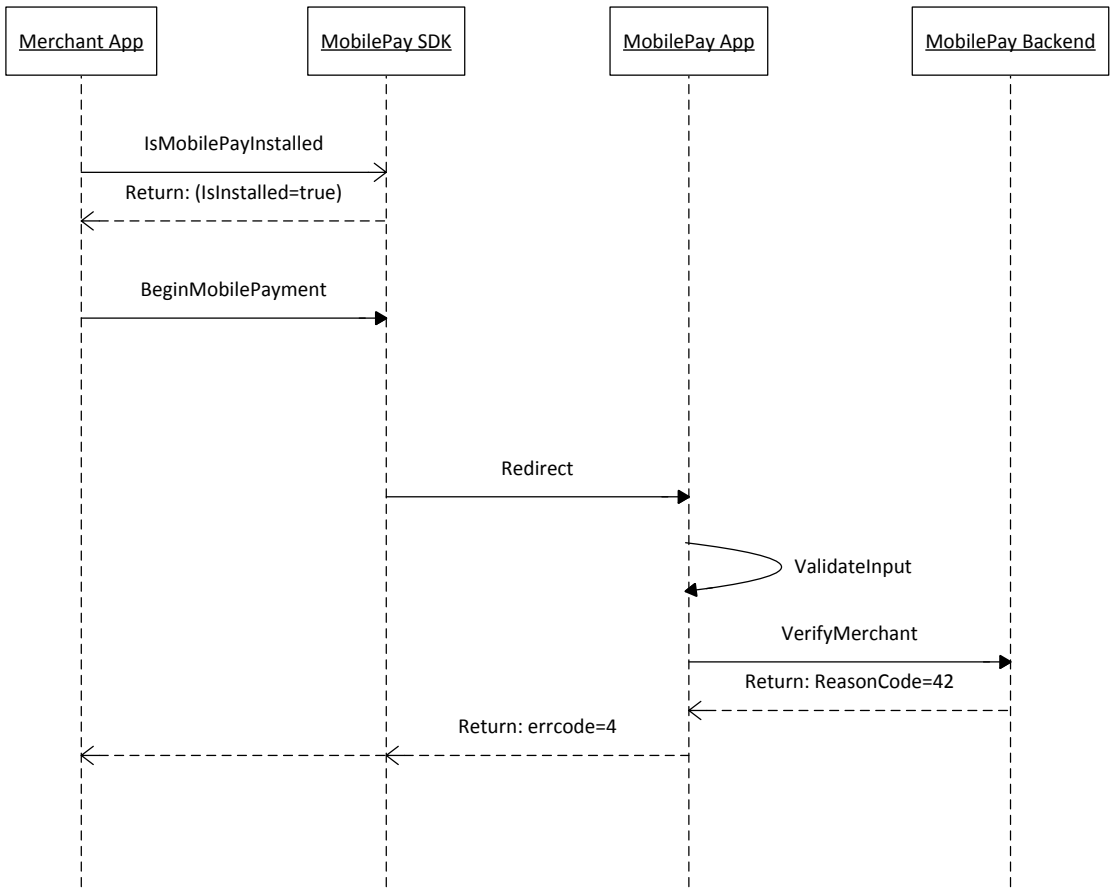
It is the responsibility of the merchant app to inform the user that the MobilePay app should be updated before trying again. The merchant app can show a message to the user and provide a link to an app store to download the latest version of MobilePay.



5.4 Merchant ID is not valid

The MobilePay AppSwitch SDK will return error code no. 4 if the merchant ID received by the MobilePay app is invalid. The MobilePay app validates the merchant ID by making a validation request to the MobilePay backend.

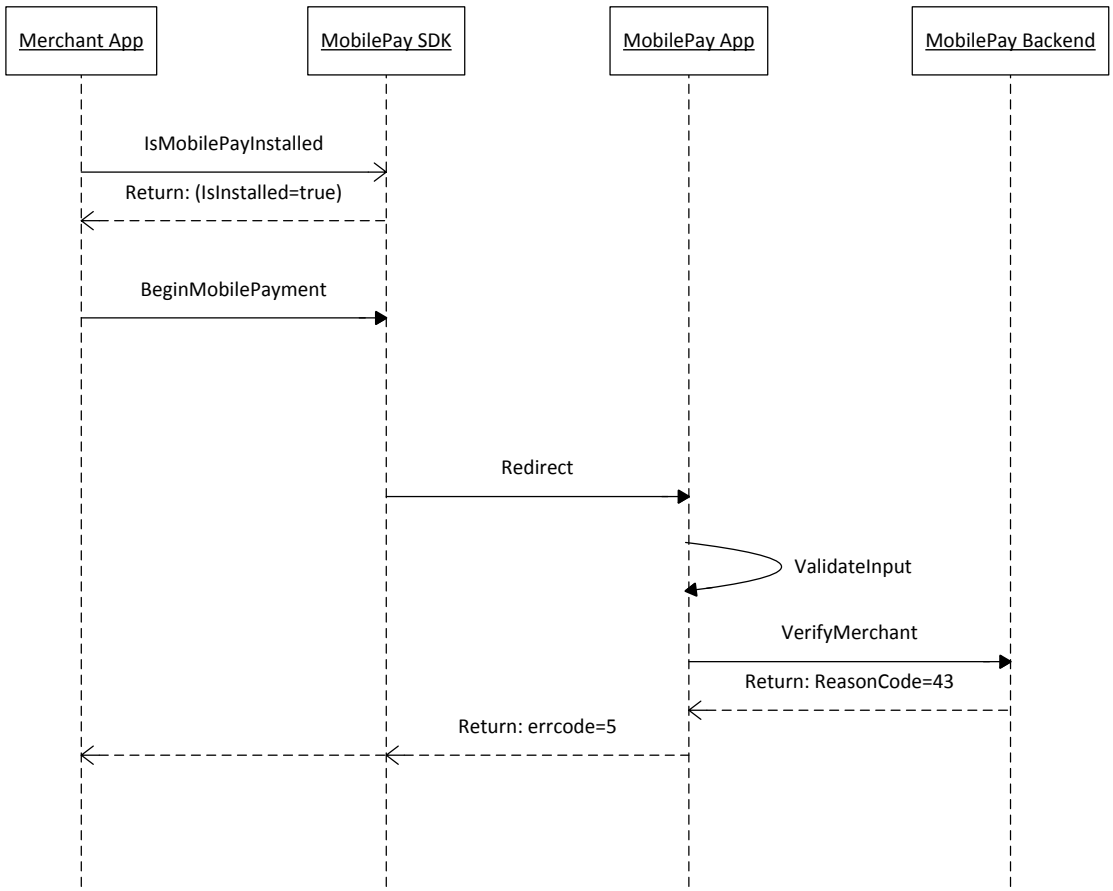
This error code should never be received in the merchant app in production. The error code should be handled in the merchant app, by showing a message to the user and log it to the merchant backend (if possible).



5.5 HMAC parameter is not valid

The MobilePay AppSwitch SDK will return error code no. 5 to the merchant app, if the HMAC parameter received by the MobilePay app is not valid due to value not matching – if not caused by corrupted data it might be caused by missing synchronisation in understanding of calculation – calculation method, content of message, and/or wrong used key.

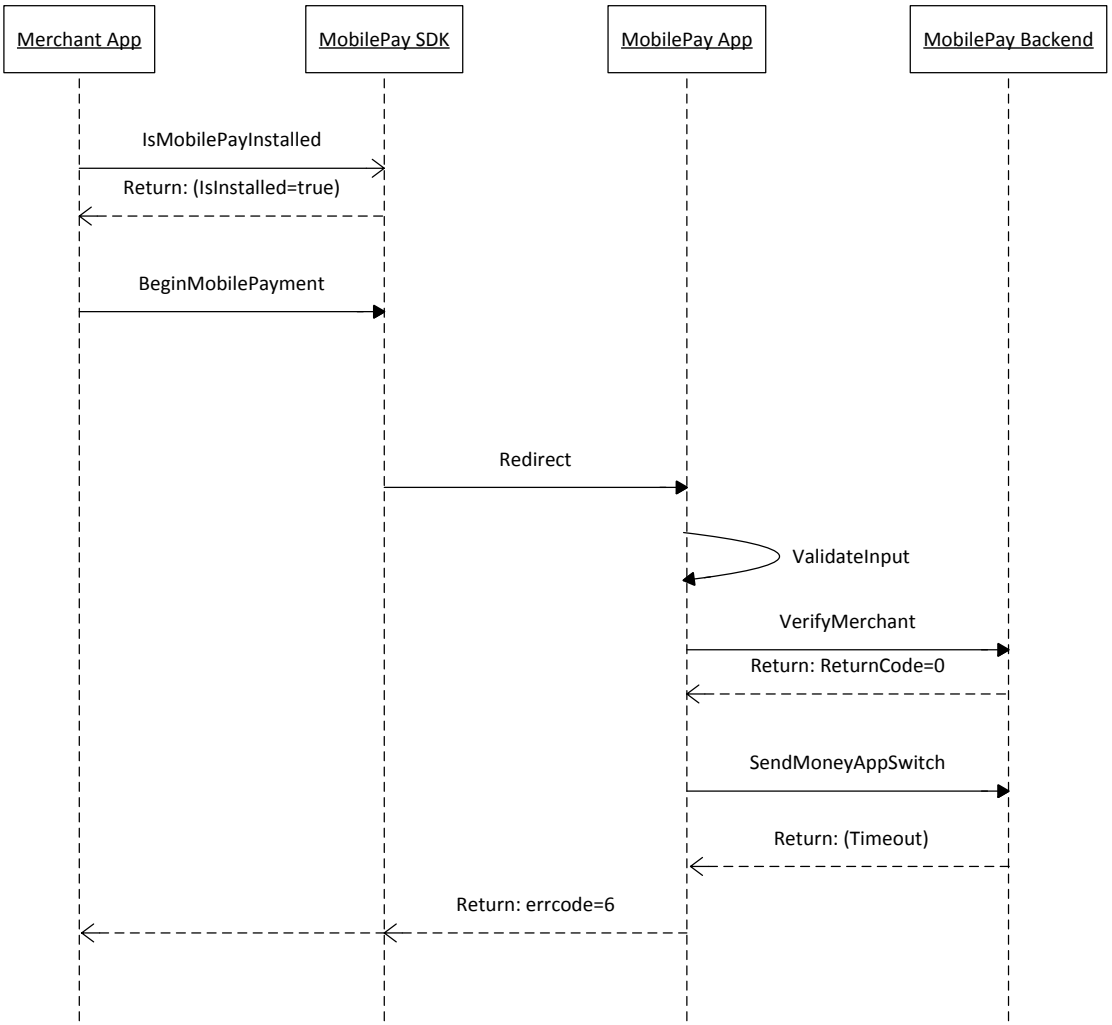
This error code should never be received in the merchant app in production. The error code should be handled in the merchant app by showing a message to the user and log it to the merchant backend (if possible).



5.6 MobilePay timeout

The MobilePay AppSwitch SDK will return error code no. 6 to the merchant app if the call to the MobilePay backend (SendMoneyAppSwitch handles the payment) times out which is by default set to 5 minutes.

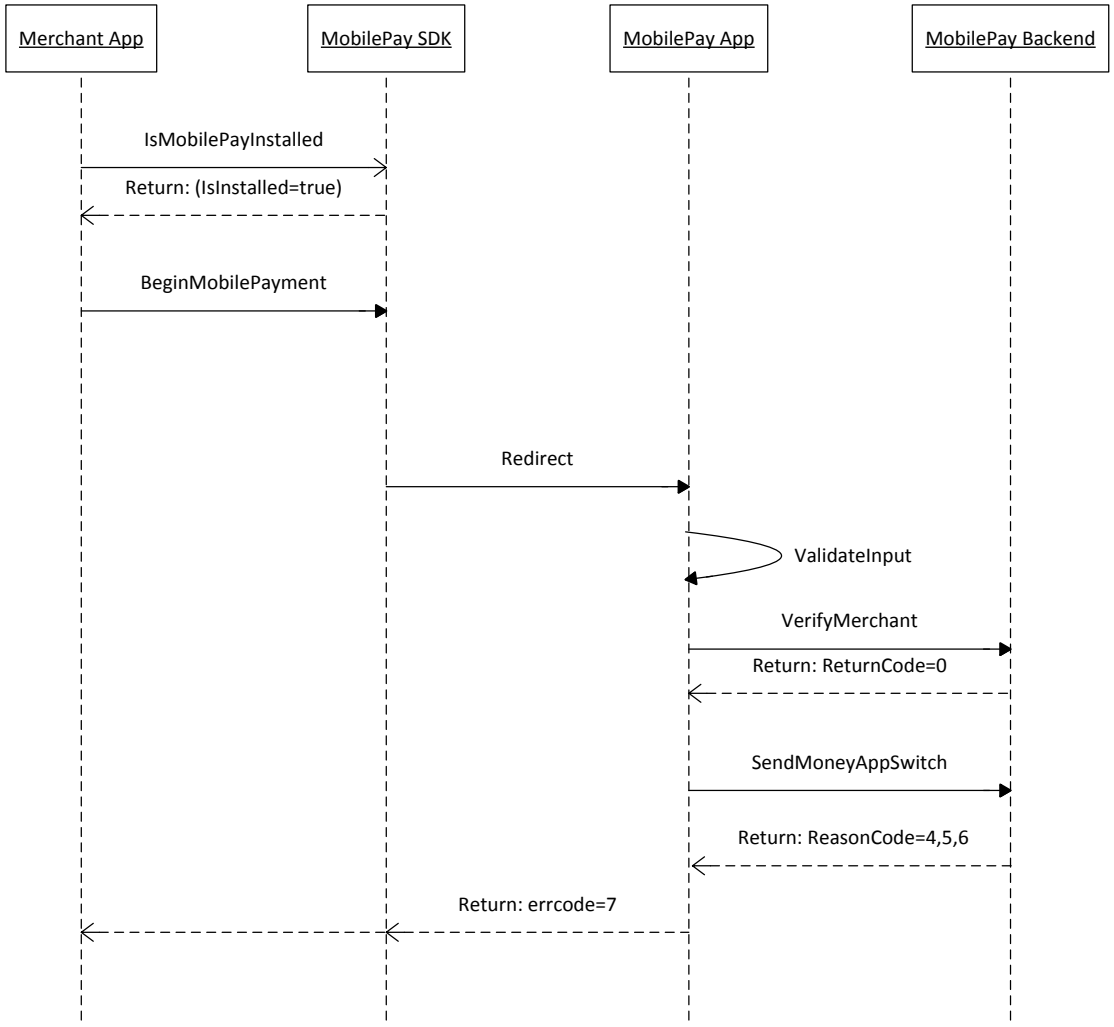
The merchant app should show a message to the user informing about the timeout and let the user try the same transaction again. The second attempt will then check if the previous request did succeed in the MobilePay backend and base its reply upon the result of this check.



5.7 MobilePay amount exceeded

The MobilePay AppSwitch SDK will return error code no. 7 to the merchant app if the user's own daily or yearly limits are exceeded by the requested amount in the order.

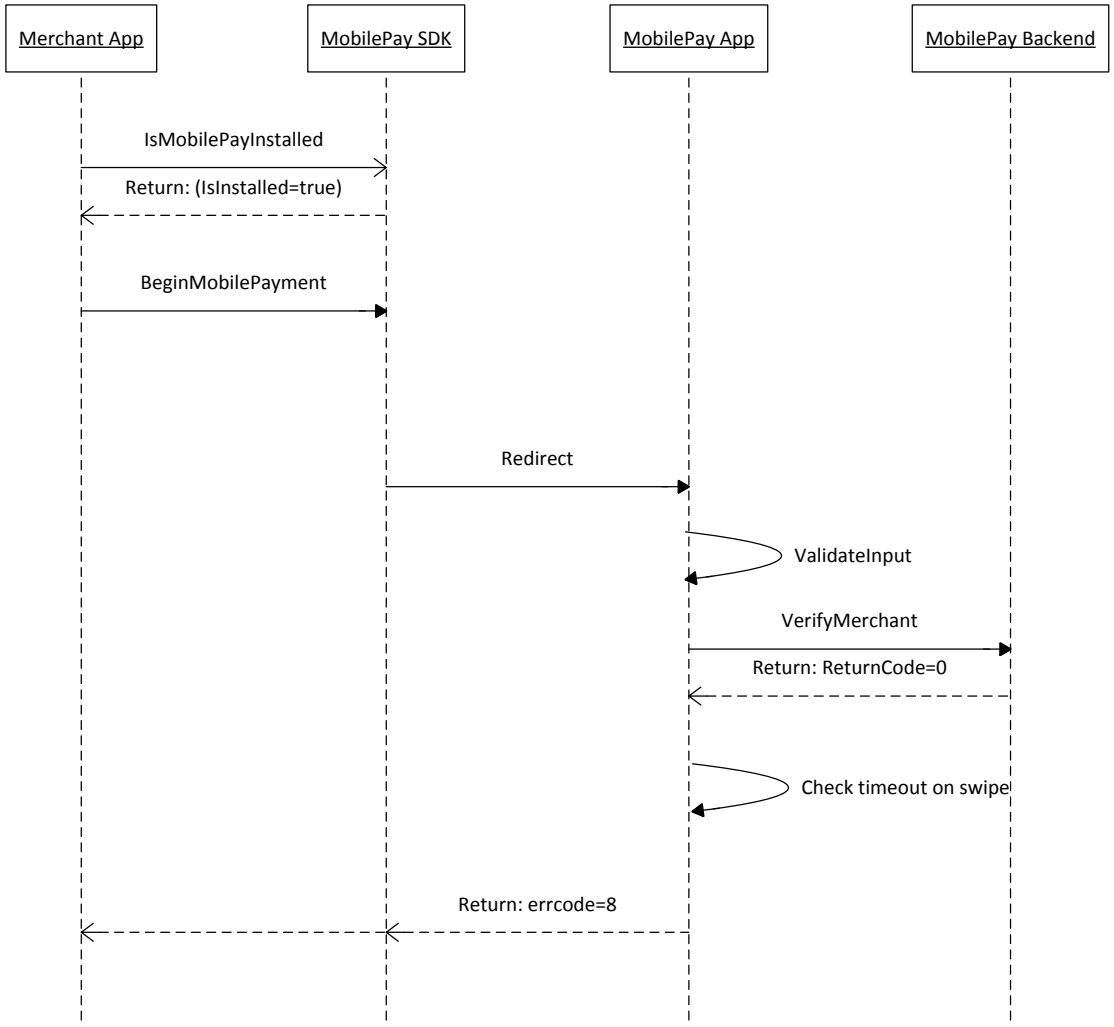
The merchant app should show a message informing the user about exceeding the limit and that the user can view the limits under "Beløbsgrænser" in the MobilePay app.



5.8 Timeout set in merchant app exceeded

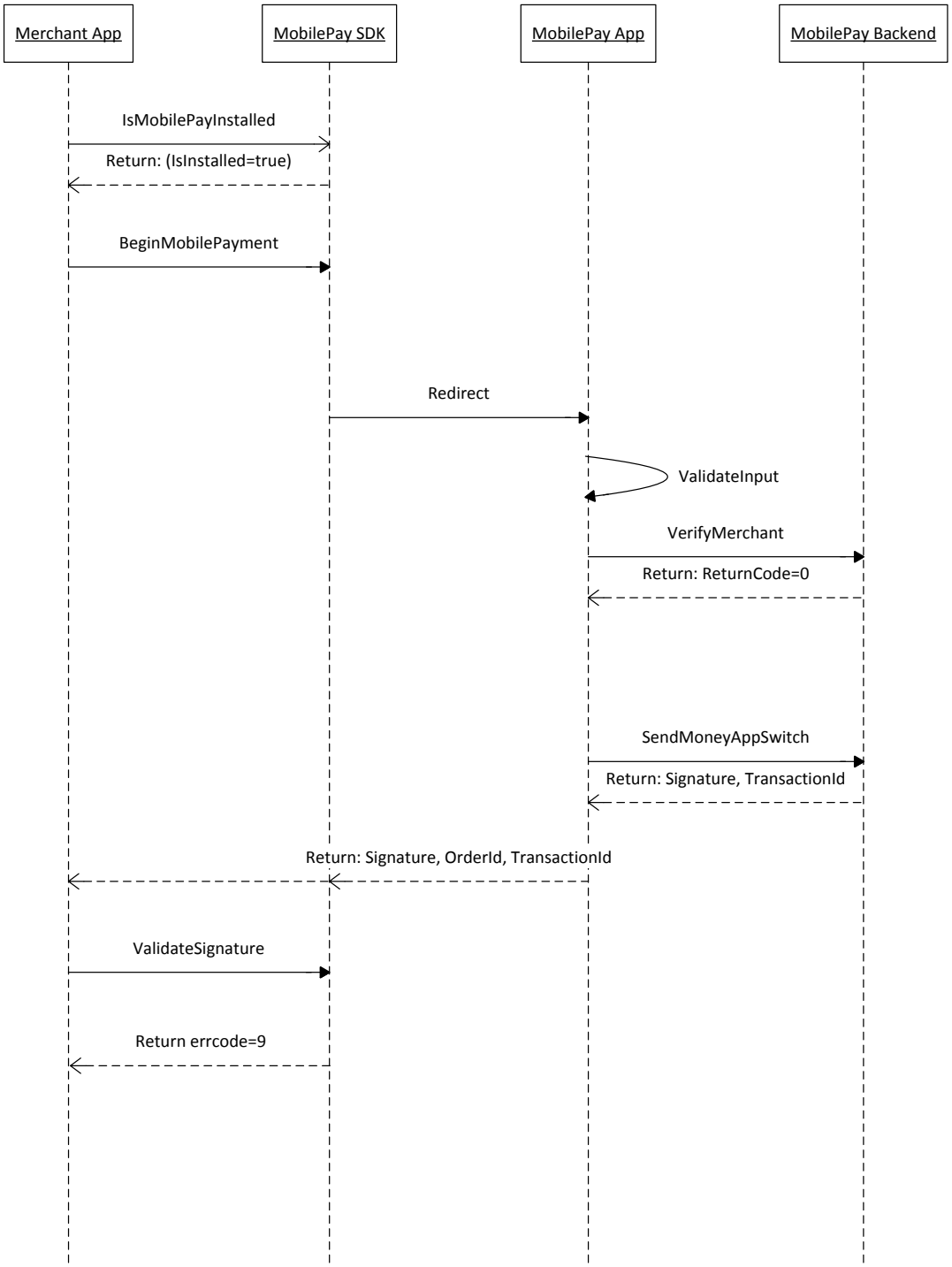
The MobilePay AppSwitch SDK will return error code no. 8 to the merchant app if the purchase takes longer than defined by the merchant app. The default timeout is to 5 minutes. The timeout will be checked in the MobilePay app when confirming the payment.

The merchant app should show a message informing that the user should try again before the timeout.



5.9 Invalid signature

The MobilePay AppSwitch SDK will return error code no. 9 to the merchant app if the SDK validation of the signature fails due to an invalid signature, or in the case that the same transaction id is sent to the merchant app twice after a single payment.

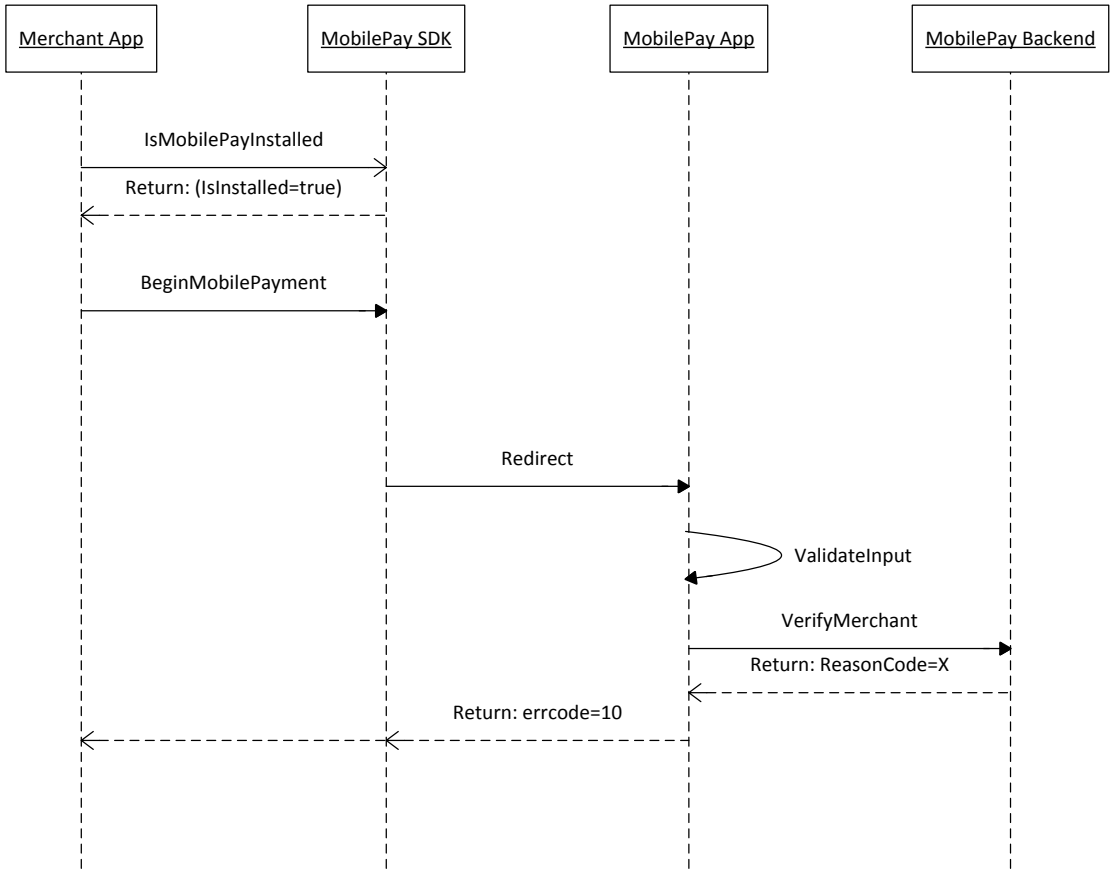


5.10 MobilePay AppSwitch SDK version is outdated

The MobilePay AppSwitch SDK will return error code no. 10 to the merchant app if the API version used by the SDK is declared obsolete by the MobilePay backend.

The merchant app receives this error code if the merchant app is not updated to the minimum required version of the MobilePay AppSwitch SDK.

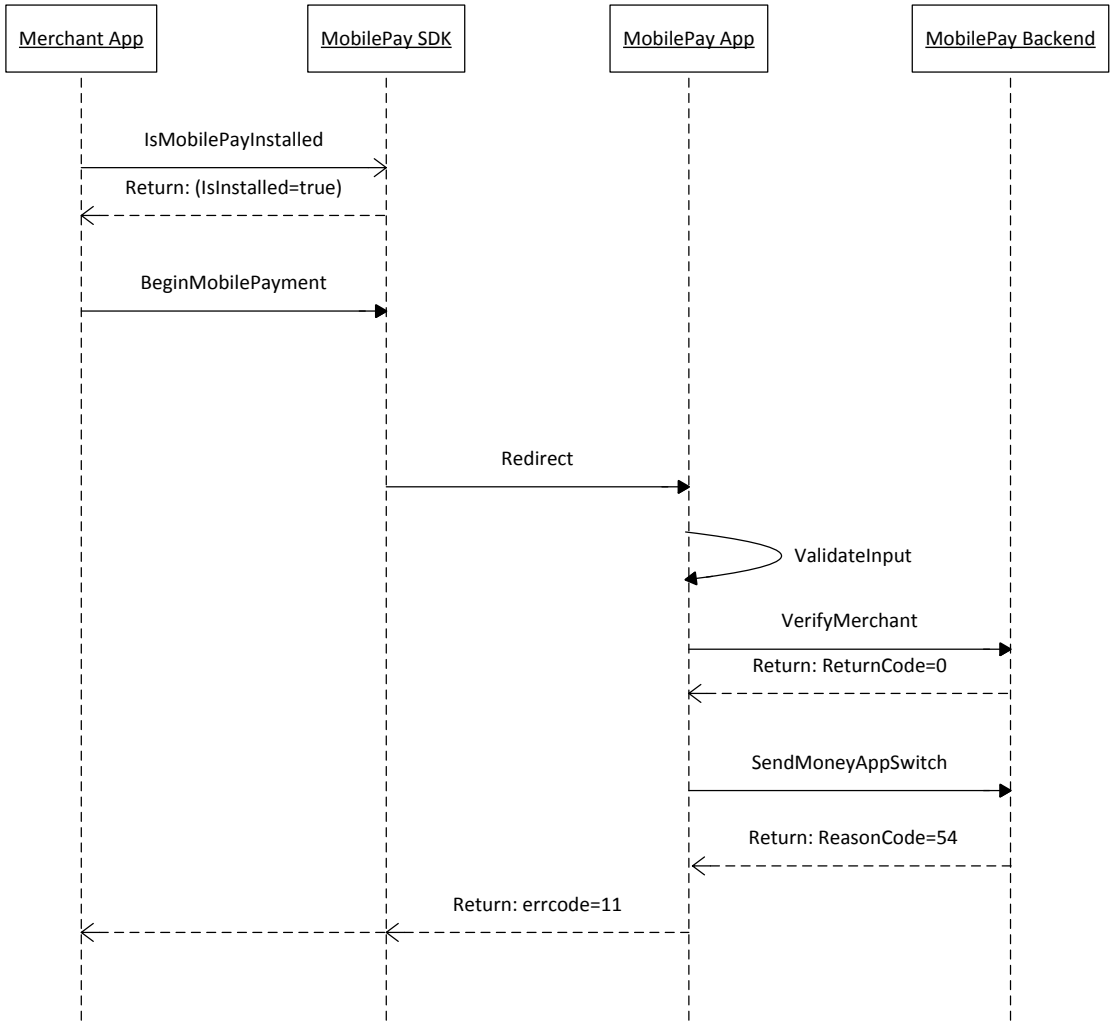
The merchant app should show a message asking the user to update the merchant app.



5.11 Order ID already used

Error code no. 11 will be returned to the merchant app if the order ID sent to MobilePay has already been used for a confirmed payment by the same merchant but not the current customer.

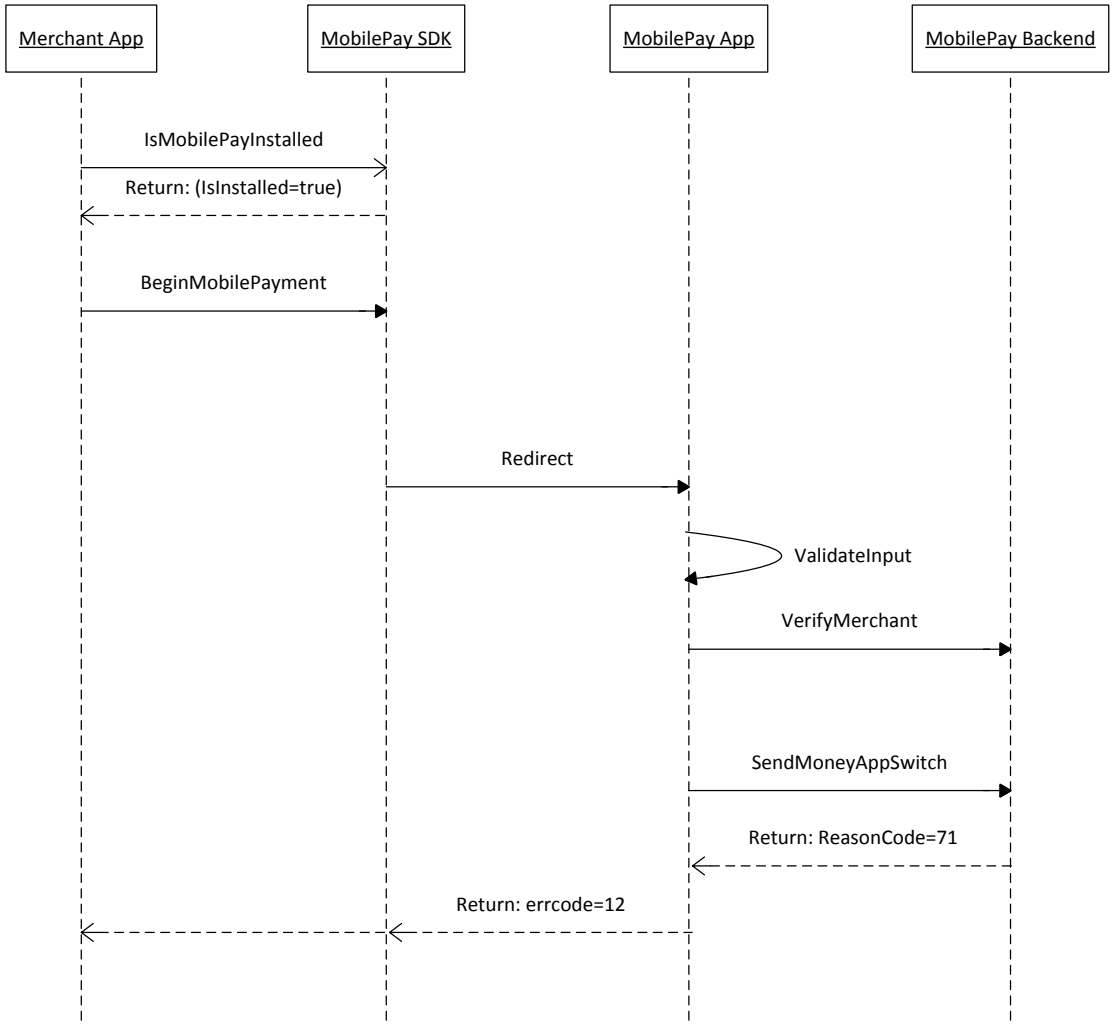
The merchant app should show a message to the user, and should create a new order ID attempting to make the payment again.



5.12 Fraud screening

Error code no. 12 will be returned to the merchant app if the user is caught in the fraud screening process due to suspicious behaviour.

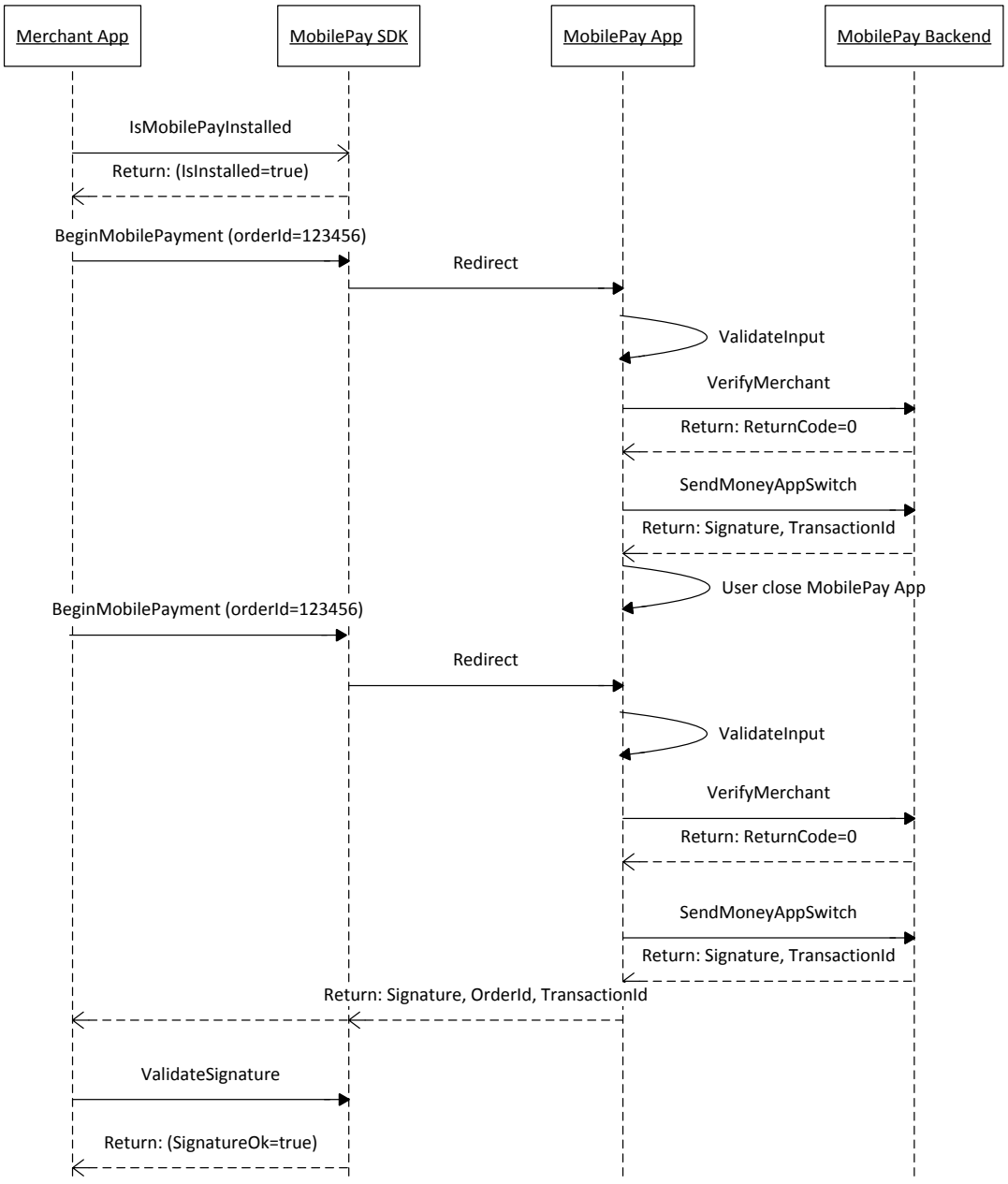
The merchant app should show a message informing the user of the rejection, and the user should contact MobilePay to resolve the situation.



5.13 Interrupted payment scenarios - MobilePay app is closed down while doing payment

In this case the merchant app will not receive a reply and this case can be handled as a normal timeout scenario.

The merchant app can choose to resend the same order ID to MobilePay, which will ensure that the customer will not pay twice for the same order and in all cases MobilePay will return the payment information, including the signature.

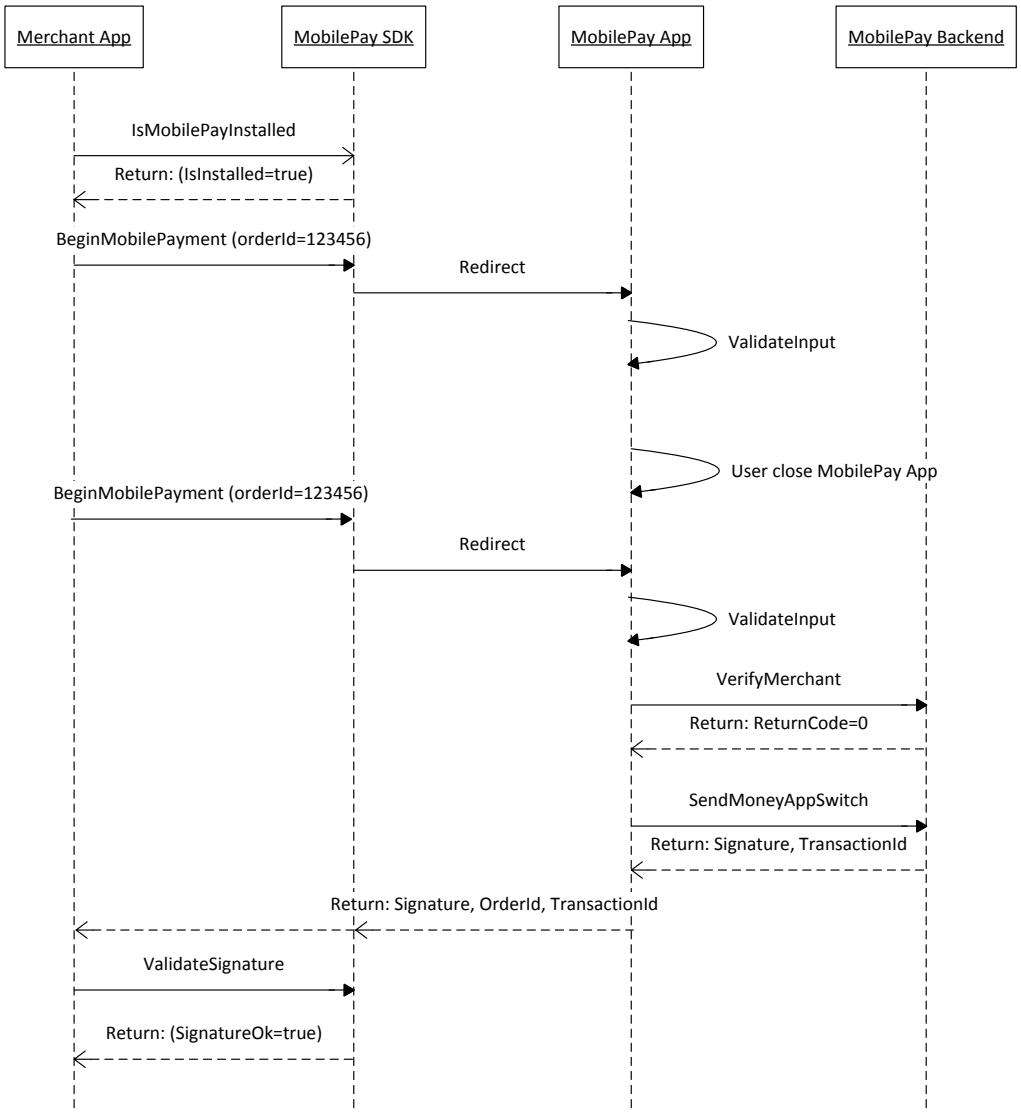


5.14 Interrupted payment scenarios - Customer navigates away from MobilePay

The MobilePay app can be in two different modes, namely AppSwitch mode and normal payment mode. MobilePay is in AppSwitch mode when an AppSwitch payment is initiated and in normal payment mode when e.g. used in relation to P2P payments.

If the customer navigates away from MobilePay during an AppSwitch payment (e.g. the phone is answered) the payment flow is cancelled and the MobilePay app mode changes from AppSwitch mode to normal payment mode and the user is logged out.

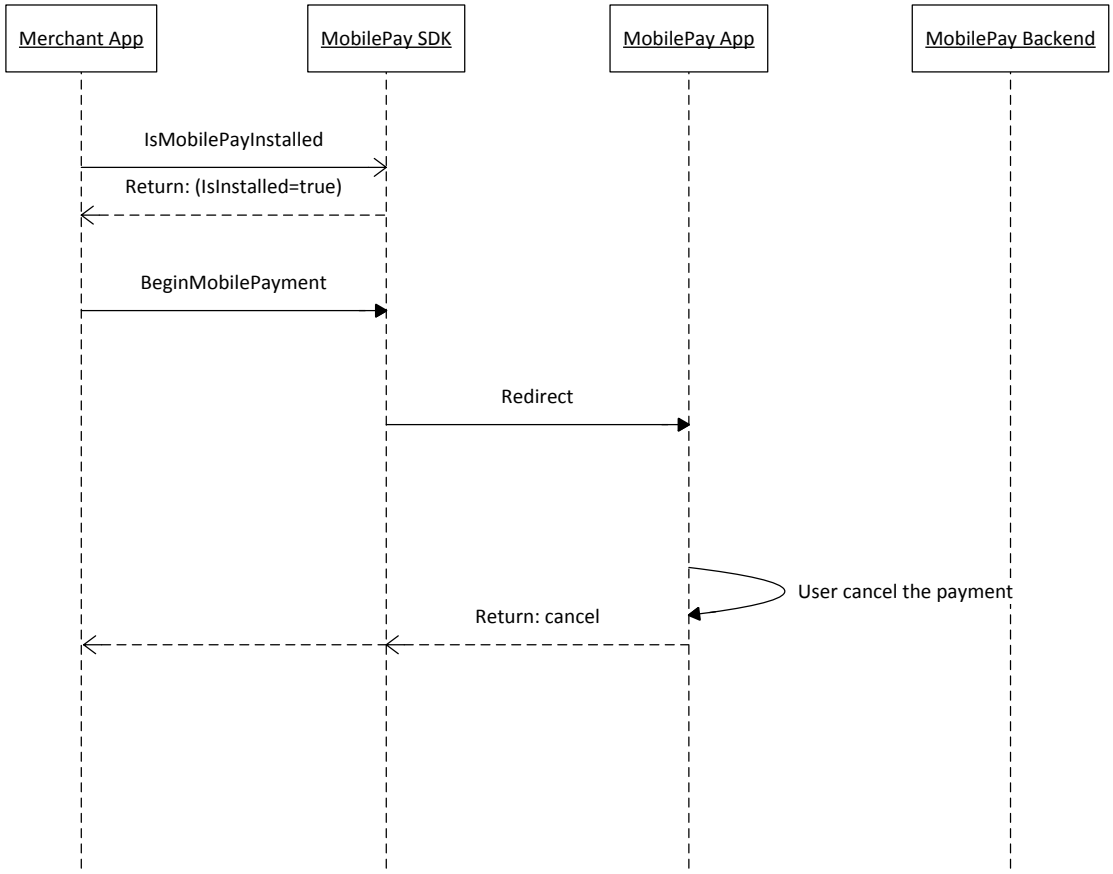
If the customer wants to pay after the phone call is completed the customer has to restart the payment from the merchant app.



5.15 Interrupted payment scenarios - Payment is cancelled in MobilePay

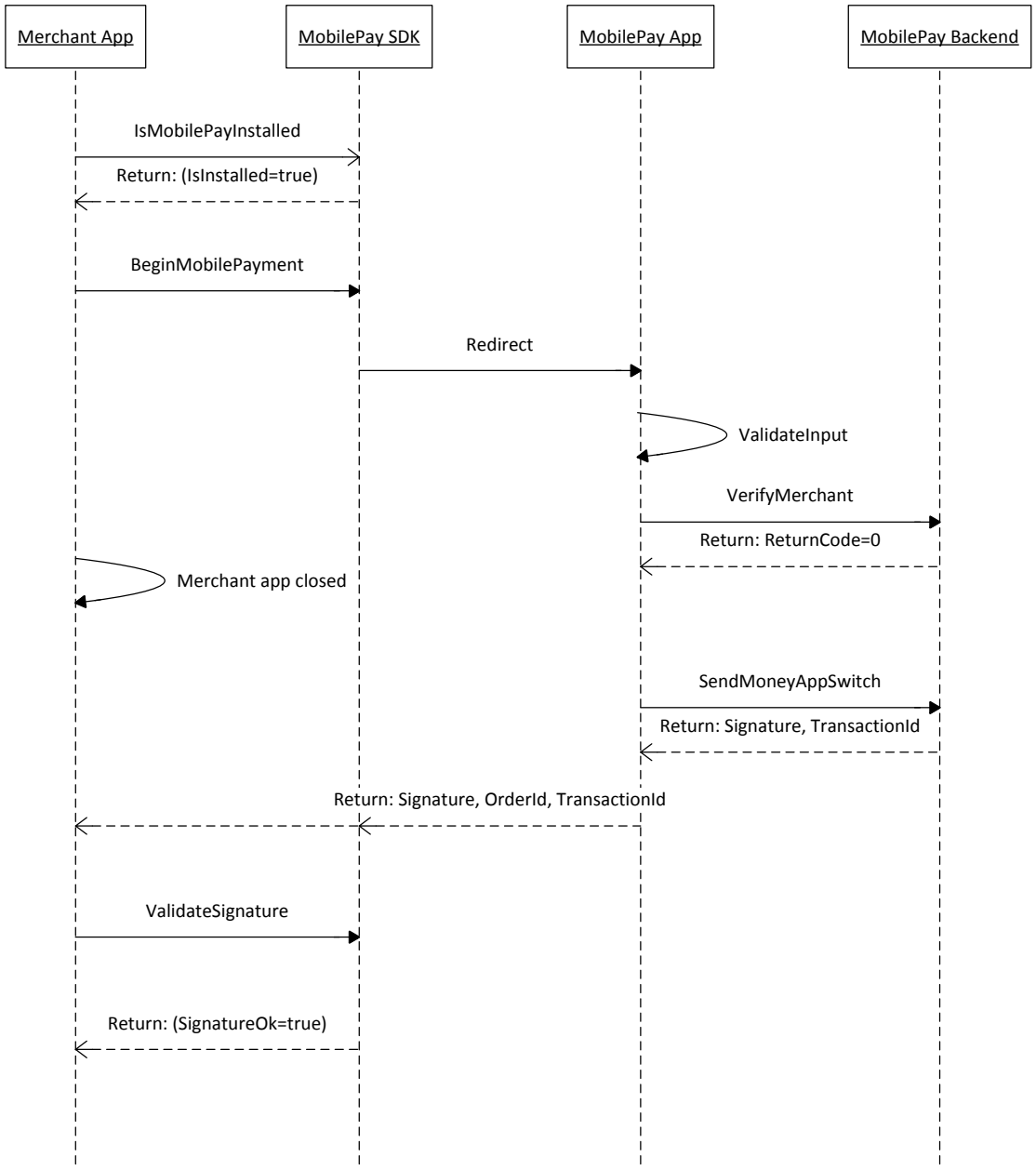
If the customer chooses to cancel the payment in the MobilePay app, the merchant app will be notified of the cancellation.

The merchant app can show a message to the user stating that the payment was cancelled and let the user continue shopping in the merchant app.



5.16 Interrupted payment scenarios - Customer closes merchant app

If the customer closes the merchant app (e.g. using the multitask feature of the OS) while doing a payment in the MobilePay app, the merchant app should also respond correctly when receiving the call-back from MobilePay, i.e. show the correct screen depending on the content of the callback.



5.17 Interrupted payment scenarios - Same order ID is sent to MobilePay twice

In case of network errors etc. it might happen that the merchant app resends the pending order (same order ID) to MobilePay, which the customer has already paid for.

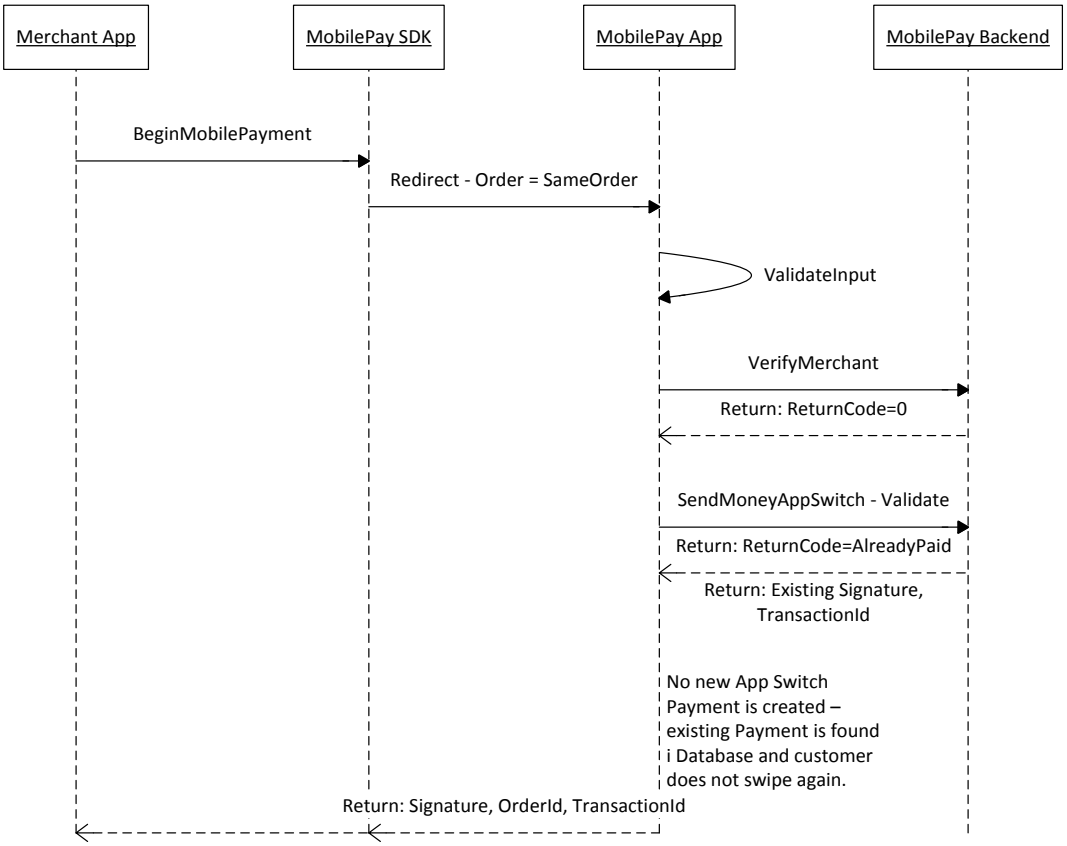
This error scenario is handled in MobilePay by returning the payment information to the merchant app letting the merchant app complete the order.

MobilePay uses a unique ID on the payment, which consists of the merchant ID and the order ID – called Global ID. This ID is used to identify the payment transaction to DIBS and DIBS will reject the ticket authorisation the second time, if the same ID is used.

This means the customer will not be able to swipe for payment, but the MobilePay app will immediately show the receipt.

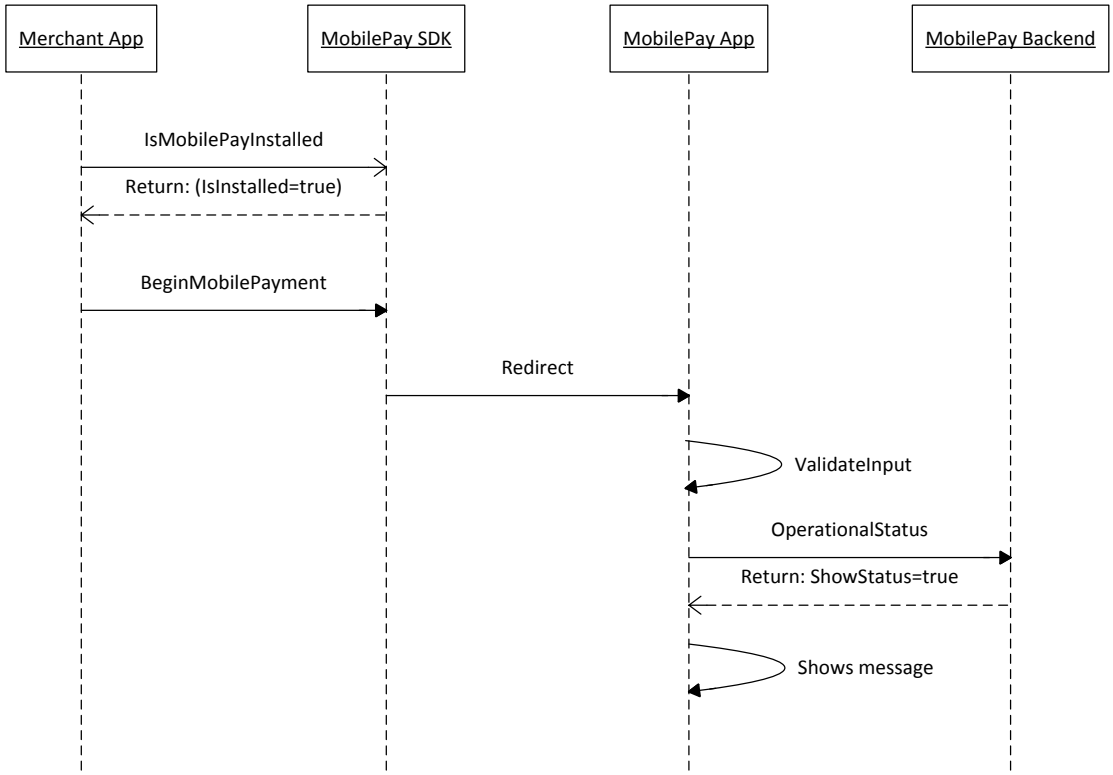
Please note that:

- Two redirects to MobilePay with same order ID returns signatures in both cases, but the customer has only paid once for this order. It must be ensured at merchant side, that it will not be possible to receive two services or products using the same order ID.



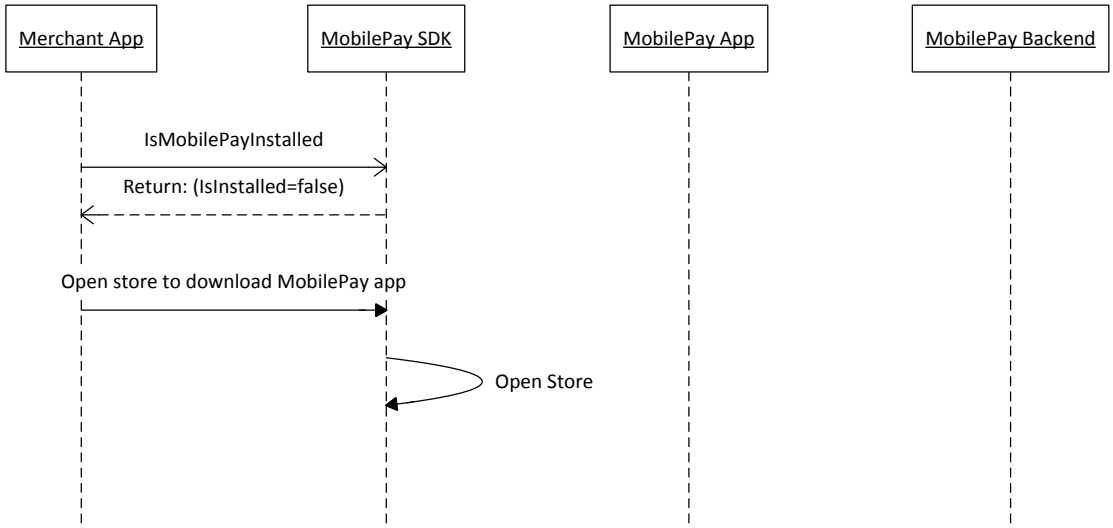
5.18 Interrupted payment scenarios - MobilePay is out of service

If MobilePay is out of service a notice will be displayed in the MobilePay app. In a case like this the customer can choose to cancel the transaction in MobilePay and the flow will then continue as explained in 0.



5.19 Installation issues - MobilePay is not downloaded

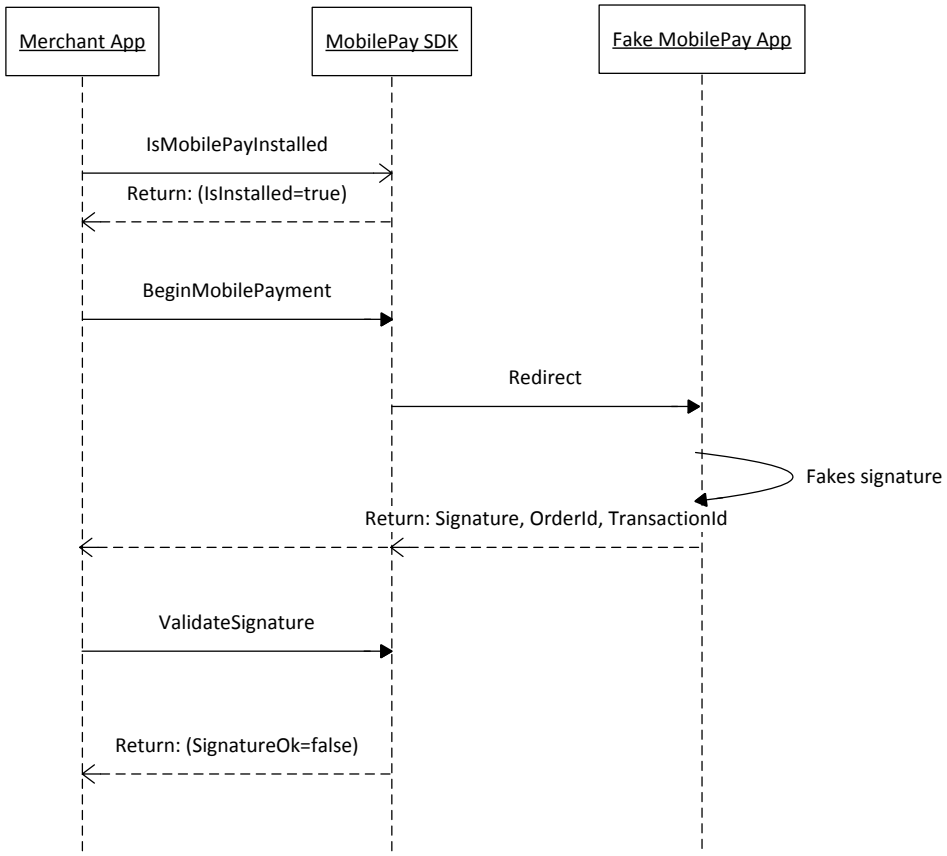
In cases where MobilePay is not installed on the customer's phone the merchant app can be set up to display a relevant message. The SDK on GitHub has a method for checking whether MobilePay has been installed.



5.20 Installation issues - Fake MobilePay app installed

A fake MobilePay app will not be able to generate a valid signature. In cases where “capture” is set to “N” or “P”, the fake app will fail due to backend calls not being able to complete transactions since no payments have been created at MobilePay in the first place.

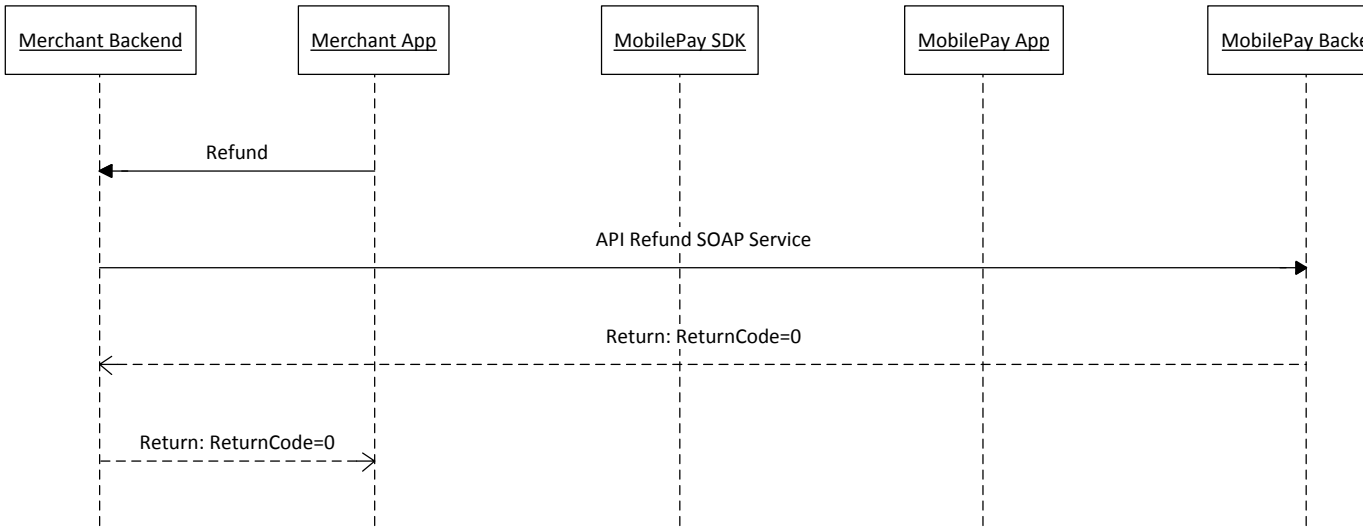
See also section 4.6 “How to ensure authentication of payment”.



5.21 Refund issues - The customer wants his/her money back

NOTE: Only for merchants using a backend setup

If the customer wants a refund the merchant can use the API Refund SOAP Service. The service is called from the merchant backend to the MobilePay backend (only available through the MobilePay AppSwitch Suite solution).



6 Security

This section describes how the communication back and forth between the merchant app and the MobilePay app is secured.

The communication and security are both ensured by the MobilePay AppSwitch SDK on GitHub.

6.1 From merchant app to MobilePay app

The merchant app will deliver the following data to the MobilePay app. Current version of the SDK (AppSwitch version 1.7):

Merchant ID	Char(60)
Order ID	Char(50)
Product Name	Char(40)
Product Price	Decimal
Receipt Message	Char(66)
BulkRef	Char(18)
SDK Version	Char(20)
Signature Version	Decimal
Success URL	Char(100)
Failure URL	Char(100)
Cancel URL	Char(100)
Capture (N/Y/P)	Char(1)
HMAC (data)	Char(64)

The BulkRef parameter allows the merchant to bulk product/service account references by assigning a name to a specific product/service, e.g. if the merchant is selling ice cream, and the sold product is a strawberry ice cream the BulkRef parameter can be set to "Strawberry" thus naming the account reference. All strawberry ice creams sold the current day will then be grouped under the reference "Strawberry".

HMAC (data) is a SHA-256 hash value of all the preceding data. The MobilePay app will ask the MobilePay backend to verify the HMAC on this data. The key for the HMAC calculation is stored in the merchant app (MobilePay AppSwitch SDK part) and in MobilePay Backend.

6.2 From MobilePay app to merchant app

If the parameter Capture specified in the input is 'Y', the MobilePay app will send the following data to the merchant app:

Order ID	Char(50)
Merchant ID	Char(60)
Transaction ID (Payment ID)	Char(20)
Signature	Char(2500)

And the signature will contain these data:

Signature version 3.0	
Order ID	Char(50)
Merchant ID	Char(60)
Transaction ID (Payment ID)	Char(20)

Amount	Dec(15,2)
Currency	Char(3)
Country	Char(2)

The MobilePay AppSwitch SDK will verify that the MSIGN certificate in the signature has been issued by the MobilePay root certificate. It is important that the root certificate is used for signature validation. Otherwise this solution will stop working when a new MSIGN certificate is issued. The private key of MSIGN is securely stored at MobilePay.

The signature is validated using the MobilePay root certificate in the following way:

A check is made to ensure that the signature is signed by a certificate that has a valid certificate chain to the MobilePay root certificate. It is also validated that the signature is of the expected order ID, merchant ID, transaction ID, and amount.

NOTE: Only for merchants using a backend setup

If the parameter Capture specified in the input is 'N' or 'P' (only reservations), there will be no signature verification although the signature is returned to the app. The signature is not needed in this case, because the merchant backend will verify the payment status by calling GetStatus.

6.3 Security from MobilePay app to MobilePay backend

The data sent from the MobilePay app to the MobilePay backend is sent via HTTP with SSL.

When a user logs on to the MobilePay app the user will have a session with the MobilePay backend. This session is created using the cryptographic encryption algorithms AES and RSA.

6.4 Data at Rest

The input that the MobilePay app receives from the merchant app is stored temporarily, i.e. no data is persists in the MobilePay app.

All payment and customer data is stored in the MobilePay backend (DB2 database).

7 MobilePay AppSwitch SDK updates

The SDK updates according to the following scheme: {MAJOR}.{MINOR}.{PATCH}. Within a given version number category, each number is assigned in an increasing order which means that version 1.4.3 is newer than 1.4.2.

The {MAJOR} number is increased when there are significant changes to functionality or changes to the framework which could cause incompatibility with interfacing systems.

The {MINOR} number is increased when minor features or significant fixes have been added.

The {PATCH} number is increased when minor bugs are fixed.

8 Test setup

It is not possible for merchants to communicate with the MobilePay test environment. Testing must therefore be done in the production environment.

Test merchant IDs are available for countries Denmark, Norway, and Finland. All test IDs are of the format "APP<country code>0000000000". When the test merchant ID is used it is possible to complete the payment flow without transferring any money. This means the merchant is able to test the security setup, SDK etc. without creating any payments.

When the merchant wants to test reconciliation files, the merchant must use the production merchant ID in order to create real payments. However, small amounts of e.g. 0,01 DKK/NOK/EUR can be used in this test setup and it will also be possible to set up a merchant ID for testing purposes.

9 Key terms and definitions

Terms	Definitions
AES	Advanced Encryption Standard (AES) is a specification for the encryption of electronic data established by the U.S
Alias	See merchant ID
API	Application Programming Interface
DIBS	Dansk Internet Betalings System
Data at Rest	Data at Rest is used as a complement to the terms Data in Use and Data in Motion, which together define the three states of digital Data.
GitHub	GitHub - Code sharing service. It is a Git repository web-based hosting service, which offers all of the distributed revision control and source code management (SCM) functionality of Git as well as adding its own features.
HMAC	Hash Message Authentication Code is a specific construction for calculating a message authentication code (MAC) involving a cryptographic hash function in combination with a secret cryptographic key.
HMAC Message	The message the HMAC calculation is based upon
HMAC Key	Agreed upon (between sender and receiver) key used for HMAC calculation
HTTP	Hypertext Transfer Protocol (HTTP) is an application protocol for distributed, collaborative, hypermedia information systems. HTTP is the foundation of data communication for the World Wide Web
Merchant App	Synonym for an app that involves payment transactions - typically related to selling of goods or services.
Merchant ID	Merchant identification number - A unique merchant ID provided by MobilePay.
MSIGN	Name of signing public key pair for MobilePay signatures. The MSIGN public key is included in a certificate issued by the MobilePay root CA.
Order ID	Order identification number - Here referring to the unique provided order ID (Shop's order ID) sent along with a payment request from a merchant (app) to MobilePay (app)
Payment ID	See Transaction ID
P2P	Peer-to-peer payment
PKI	Public Key Infrastructure
REST	Representational state transfer (REST) is a simple stateless architecture that generally runs over HTTP. REST is used between the MobilePay app and the MobilePay backend.
RSA	RSA is one of the first practicable public-key cryptosystems and is widely used for secure data transmission. RSA involves a public key and a private key. The public key can be known by everyone and is used for encrypting messages. Messages encrypted with the public key can only be decrypted in a reasonable amount of time using the private key.
SDK	Software Development Kit – MobilePay AppSwitch SDK is designed to be embedded in a merchant app.
SIM	Subscriber Identification Module
SSL	Secure Sockets Layer- a standard cryptographic protocol designed to provide communication security over the Internet.

SOAP	Simple Object Access Protocol. SOAP is used between the merchant backend and the MobilePay backend.
Transaction ID	DIBS provided payment transaction ID
UI	User Interface
WSDL	Web Services Description Language - is an XML-based interface definition language that is used for describing the functionality offered by a web service.
WSDL file	An acronym used for any specific WSDL description of a web service, which provides a machine-readable description of how the service can be called, what parameters it expects, and what data structures it returns.