htng

MOVING
HOSPITALITY
FORWARD

OPEN PAYMENTS ALLIANCE STANDARDS SPECIFICATION

25 January 2021

Version 2.0

About HTNG

Hospitality Technology Next Generation (HTNG) is a non-profit association with a mission to foster, through collaboration and partnership, the development of next-generation systems and solutions that will enable hoteliers and their technology vendors to do business globally in the 21st century. HTNG is recognized as the leading voice of the global hotel community, articulating the technology requirements of hotel companies of all sizes to the vendor community. HTNG facilitate the development of technology models for hospitality that will foster innovation, improve the guest experience, increase the effectiveness and efficiency of hotels, and create a healthy ecosystem of technology suppliers.

TABLE OF CONTENTS

1 This Specification at a Glance

While there have been many advances and innovation in the distribution channel, payments have not evolved at the same pace due to their intrinsic complexity. There are many forms of payment used in the indirect sales channel and each one of them contains different payment and processing instructions. This hinders hotel reservations and front desk operations causing inefficiencies and overhead costs which ultimately affects the traveler's experience at a property. These inefficiencies manifest themselves in multiple ways:

- Distribution technology does not always support the transmission of payment data and/or changes required in the data for compliance and regulatory reasons
- Post-booking changes to a reservation require modifications to the payment instructions or amounts, and these changes cannot be communicated through existing technology
- Limited or non-existing technology to transmit payment instructions with the reservation
- Loss of revenue or anticipated profit due to manual and legacy processes and technology

The Open Payments Alliance "OPA" is introducing the concept of a Payment Manager. The Payment Manager provides a travel industry specific payment solution. Travel companies will be able to send payment instructions to the Payment Manager specifying the terms for which payment for a given booking should be executed. The Payment Manager leverages a network of partners and guarantees that money flows into the intended recipient's bank account.

The Payment Manager is one part of a system to streamline B2B payments in the hotel industry. This document provides specifications for a set of messages that will flow between a Payment Manager and the hotel's PMS and/or CRS. This specification also includes a set of messages between travel booking partners and the Payment Manager. While this document includes a comprehensive list of possible messages, different participants in the B2B payment stream may only require a subset of the messaging.

This specification defines the role of a Payment Manager, a solution that transforms payable data into instructions that automatically initiates and settles payments between two parties. These solutions are expected to streamline and automate processing as well as reconciliate payments used in indirect sales channels.

The adoption of these specifications will standardize the way payments are processed today, improving operational efficiencies for hotels while providing a better check-in experience to travelers.

2 Document Information

2.1 DOCUMENT HISTORY

Version	Date	Author	Comments
1.0	21 MAY 2020	Sandy Angel	Includes messages to suport the transactions between the Payment Manager and the Payee
2.0	25 JAN 2021	Sandy Angel	Added messages to support the transactions between the Payment Manager and the Payer

2.2 Document Purpose

The goal of this specification is to provide the foundation for decoupling payments from bookings and create a frictionless industry specific B2B payment channel. The resulting payment channel should feature:

- **Support for any form of B2B payment -** Hotels should be able to leverage different forms of payment in order to create optimal payment strategies for each client based on factors such as risk, contractual relationships, processing fees, etc.
- **Automated payment processing -** Irrespective of the form of payment, payments should be processed in due course with no manual intervention.
- **Transparent flow of payment information –** This supports the transparent bidirectional flow of relevant payment information between the travel company booking engines and the hotel PMS. This includes the flow of payment instructions to the hotels and the satisfaction of bill-back requirements to travel companies.

2.3 Scope

This specification does not define the Payment Manager architecture, but creates the software standards, business rules and best practices so that companies wishing to implement a Payment Manager can develop a solution using industry-accepted open standards.

Diagram 1 below illustrates the decoupling of the payment instructions and details from the booking.

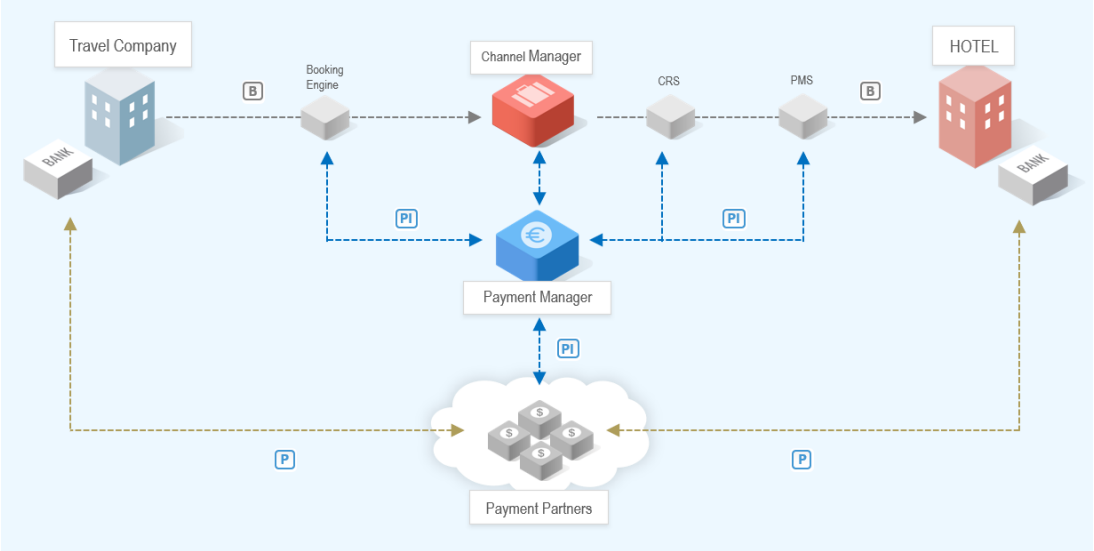


Diagram 1 - Decoupling of the Payment Instructions from the Booking

Key:
B – Booking
PI – Payment Instructions (Details)
P - Payment

Diagram 2 illustrates how the Payment Manager can be used to facilitate payments between two parties outside of the booking scenario. This may be useful for the payment of commissions from a hotel to an OTA or the payment of services from a hotel to a service provider, such as a laundry service.

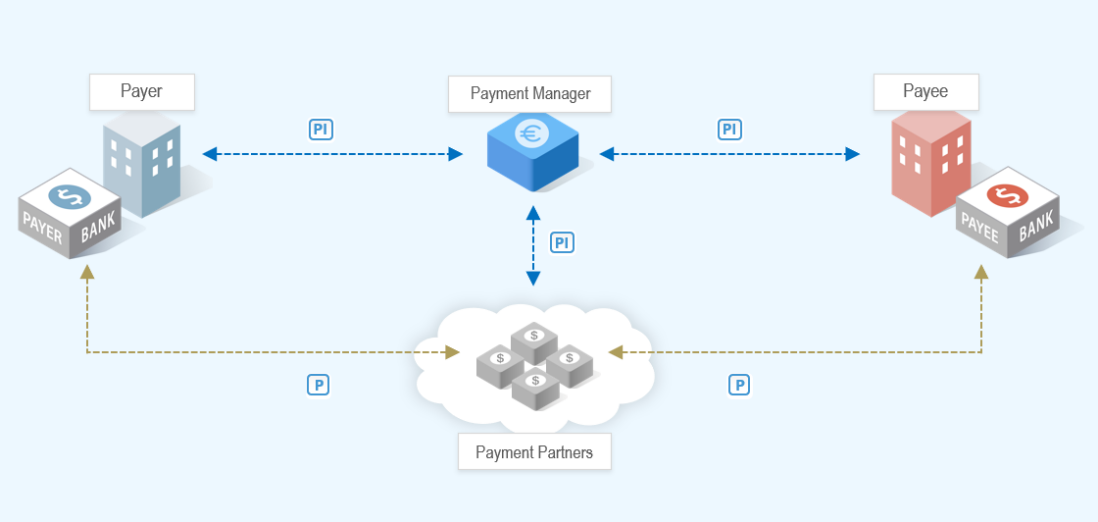


Diagram 2 - Payment Manager Architecture for Non-Booking Payments

Key:
PI – Payment Instructions (Details)
P - Payment

2.4 Relationship to Other Standards

This specification and its supporting schemas leverage the existing OpenTravel Alliance methodology for message construction and draws upon data definitions common to several HTNG specifications as of December 15, 2020.

Related specifications:

- HTNG Specifications
    - Most relevant to this specification:
        - HTNG Product Distribution Seamless Shop and Book Specification V4
        - HTNG Product Distribution Reservation Specification V4
- OpenTravel Alliance Specifications

2.5 Useful Resources

- Implementing Web Services Using HTNG Specifications – A Quick Start Guide for Software Developers
- HTNG Discussion Board – currently available at http://www2.htng.org/discussion
- Open Payment Alliance – www.openpaymentalliance,org

2.6 Audience

The Payment Manager standards have been designed for industry-wide adoption among hotels, their technology partners, payment companies, booking engines and third-party distribution and sales channels. The details herein will prove helpful to developers and integrators for hotel booking and payment systems, as well as property management systems and central reservation systems.

2.7 Further Considerations

This specification provides a framework for B2B payment processing. In this document, a number of use cases have been incorporated. These do not reflect the many other use cases that may build on the framework provided herein. Future considerations could include additional B2B use cases or potentially B2C transactions.

A number of functions required to complete the entire payment process will remain the responsibility of the Payment Manager or the payer/payee. The Payment Manager will develop connectivity to various payment companies. The process for refunds and reconciliation may fall to the Payment Manager, payment companies or the payer/payee.

3 Composite Scenarios

This section shows a list of potential composite scenarios and operations grouped by role: Payer and Payee. Each scenario will specify the proper API message that is needed to complete that operation.

The following notation is used:

Role Name	Definition
Payer	The party responsible for payment
Payee	The party receiving the payment
Payment Manager	The party that orchestrates payments between the Payer and the Payee
Payment Source	The party responsible for creating a payment instrument
Payment Processor	The party responsible for the transfer of the funds from the Payer to the Payee

Please note that the composite scenario diagrams only represent sequence, not time.

3.1 Payer Scenarios

This section is applicable to a Payer, or designated Requestor on behalf of the Payer (hereafter will be referred as the Payer), connecting to a Payment Manager.

3.1.1 Assumptions

- Payer is subscribed to the Payment Manager and has credentials
- Payer has a relationship with the Payee
- Payment Source is configured within the Payment Manager for the Payer

3.1.2 Composite Scenario 1 – Payment Creation – Booking

3.1.2.1 Overview

The Payer sends a request to the Payment Manager to create a payment.

3.1.2.1 Pre-conditions

None

3.1.2.1 Triggers

This triggers a need to send a payment to another party.

3.1.2.2 Basic course of events

1. The Payer creates a unique payment reference
2. The Payer sends the payment reference within the booking to the Payee
3. The Payer sends to the Payment Manager the Create Payment message including the payment reference (*HTNG_CreatePaymentRQ*)
4. If needed, the Payment Manager requests payment instrument from Payment Source
5. The Payment Manager creates a payment and assigns the payment reference provided by the Payer
6. The Payment Manager responds back to the Payer (*HTNG_CreatePaymentRS*)
7. Optionally, the Payment Manager sends to the Payee the Payment Detail Notification message to inform the Payee of the payment (*HTNG_PaymentDetailNotifRQ*)

3.1.2.3 Post conditions

- The payment exists in the Payment Manager
- Payee has payment reference

3.1.2.4 Exception path

- In case of an unsuccessful creation, the Payment Manager returns an error and the payment is not created
- The Payer follows their predetermined error policy

3.1.2.5 Message flow diagram

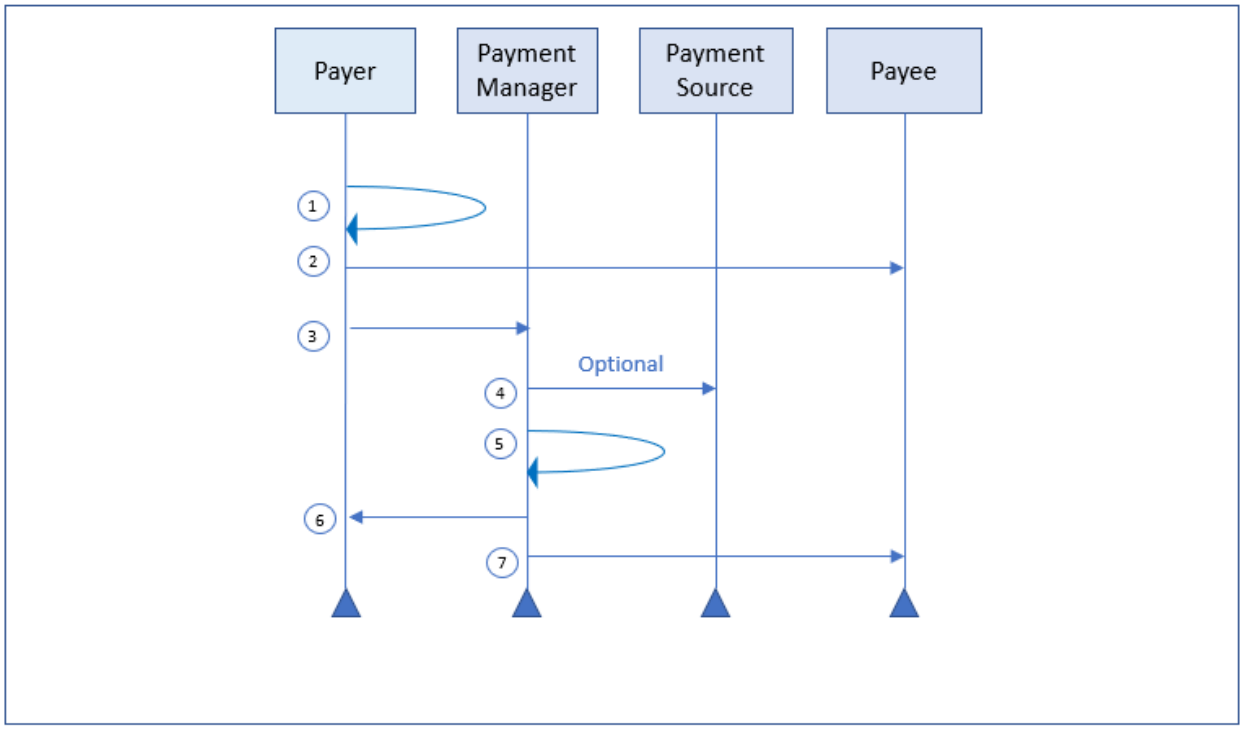


Diagram 3 - Create Payment - Booking

3.1.3 Composite Scenario 2 – Payment Creation – Non-Booking

3.1.3.1 Overview

The Payer sends a request to the Payment Manager to create a payment.

3.1.3.2 Pre-conditions

None

3.1.3.3 Triggers

This triggers a need to send a payment to another party.

3.1.3.4 Basic course of events

1. The Payer creates a unique payment reference
2. The Payer sends to the Payment Manager the Create Payment message including the payment reference (*HTNG_CreatePaymentRQ*)
3. If needed, the Payment Manager requests payment instrument from Payment Source
4. The Payment Manager creates a payment and assigns the payment reference provided by the Payer
5. The Payment Manager responds back to the Payer (*HTNG_CreatePaymentRS*)
6. The Payment Manager sends to the Payee the Payment Detail Notification message to inform the Payee of the payment including the payment reference (*HTNG_PaymentDetailNotifRQ*)

3.1.3.5 Post conditions

- The payment exists in the Payment Manager
- The Payee has the payment reference

3.1.3.6 Exception path

- In case of an unsuccessful creation, the Payment Manager returns an error and the payment is not created
- The Payer follows their predetermined error policy

3.1.3.7 Message flow diagram

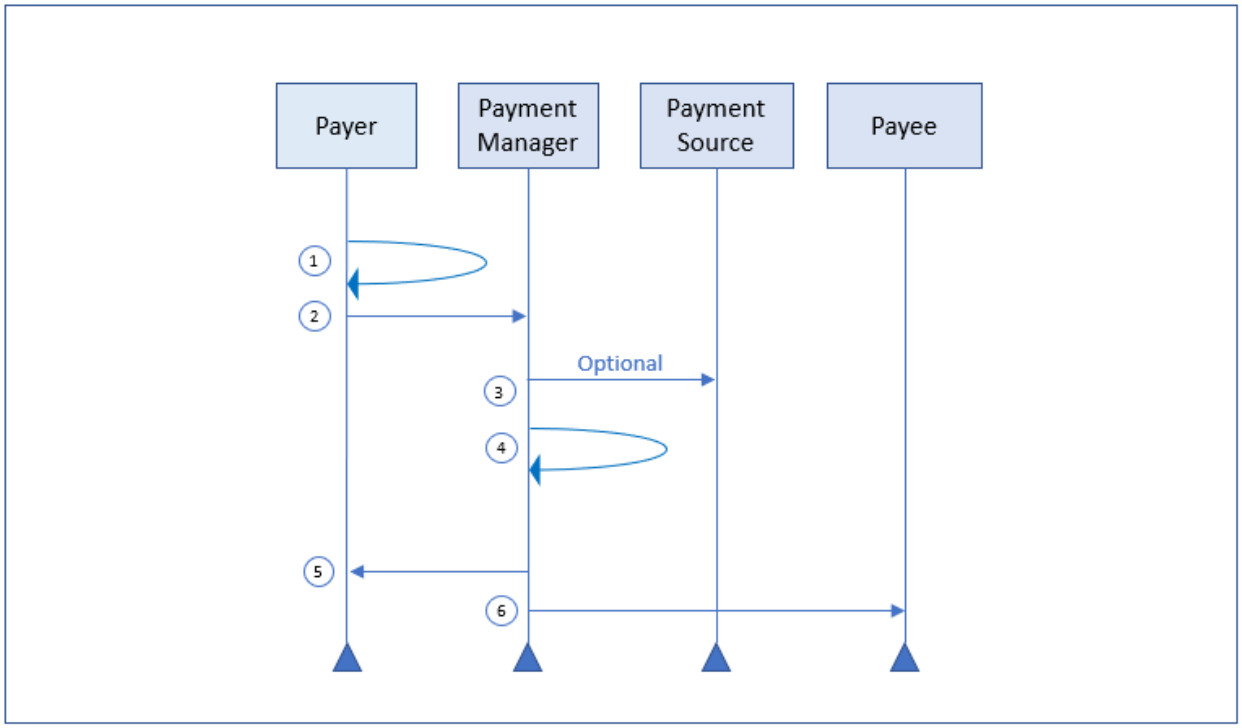


Diagram 4 - Create Payment - Non-Booking

3.1.4 Composite Scenario 3 – Payment Modification

3.1.4.1 Overview

The Payer needs to modify an existing payment in the Payment Manager.

Notes:

- *Use of this message is limited to modifications that do not require Payee approval. If approval is required, the payment should be canceled and a new payment should be created.*
- *NOT ALL information in the payment is eligible for modification (see Modifiable Payment Data section in the Appendix for more details).*

3.1.4.2 Pre-conditions

- The payment exists in the Payment Manager
- The payment is not already completed, expired, or cancelled

3.1.4.3 Triggers

The payment terms have changed and the payment in the Payment Manager needs to be updated accordingly.

3.1.4.4 Basic course of events

1. The Payer sends to the Payment Manager the Modify Payment message including the payment reference and the updated payment information (*HTNG_ModifyPaymentRQ*)
2. If needed, the Payment Manager connects to the Payment Source to update the payment instrument accordingly
3. The Payment Manager modifies the payment
4. The Payment Manager responds back to the Payer (*HTNG_ModifyPaymentRS*)
5. Optionally, the Payment Manager sends to the Payee the Payment Detail Notification message to inform the Payee of the changes (*HTNG_PaymentDetailNotifRQ*)

3.1.4.5 Post conditions

The payment is updated in the Payment Manager.

3.1.4.6 Message flow diagram

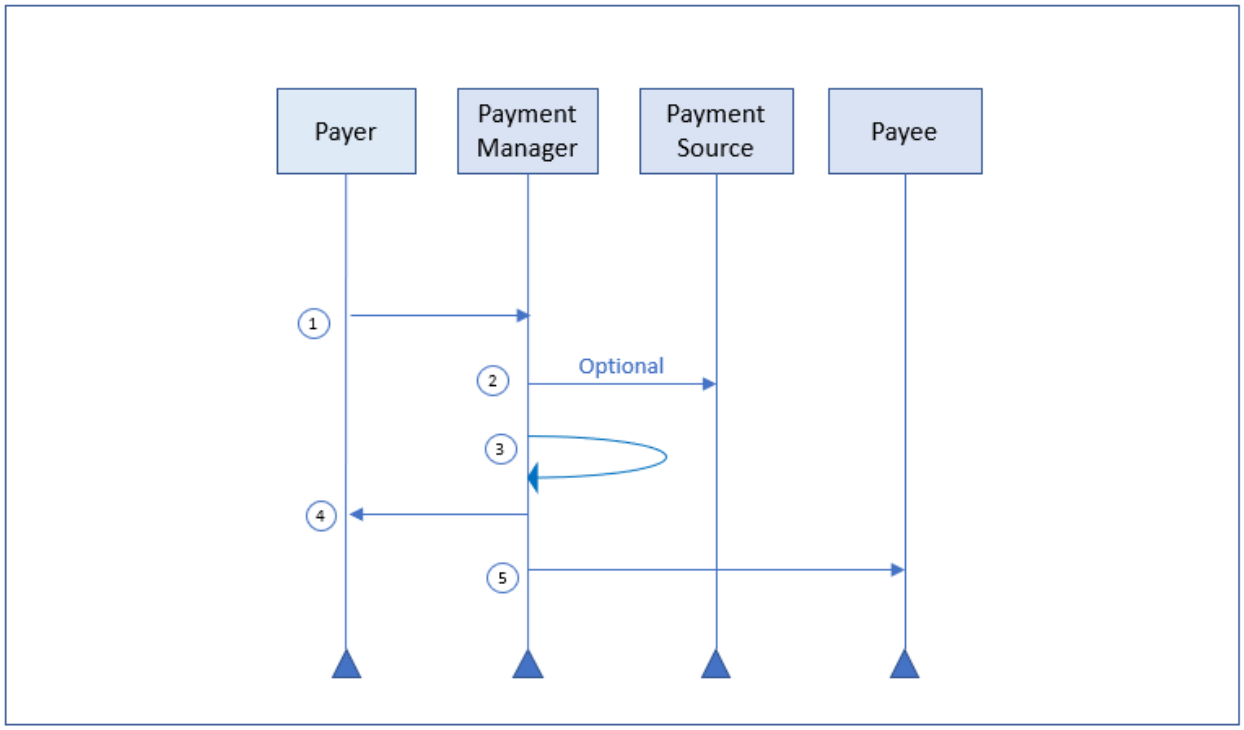


Diagram 5 - Payment Modification

3.1.5 Composite Scenario 4 – Payment Cancellation

3.1.5.1 Overview

A payment is no longer applicable and the Payer wants to remove it from the Payment Manager.

3.1.5.2 Pre-conditions

- The payment exists in the Payment Manager
- The payment is not already processed, expired or cancelled

3.1.5.3 Triggers

This triggers a need to cancel an existing payment to the Payee.

3.1.5.4 Basic course of events

1. The Payer sends to the Payment Manager the Cancel Payment message including the payment reference (*HTNG_CancelPaymentRQ*)
2. If needed, the Payment Manager connects to the Payment Source to cancel the payment instrument
3. The Payment Manager cancels the payment
4. The Payment Manager responds back to the Payer (*HTNG_CancelPaymentRS*)
5. Optionally, the Payment Manager sends to the Payee the Payment Status Notification message to inform the Payee of the status change (*HTNG_PaymentStatusNotifRQ*)

3.1.5.5 Post conditions

The payment is cancelled in the Payment Manager.

3.1.5.6 Message flow diagram

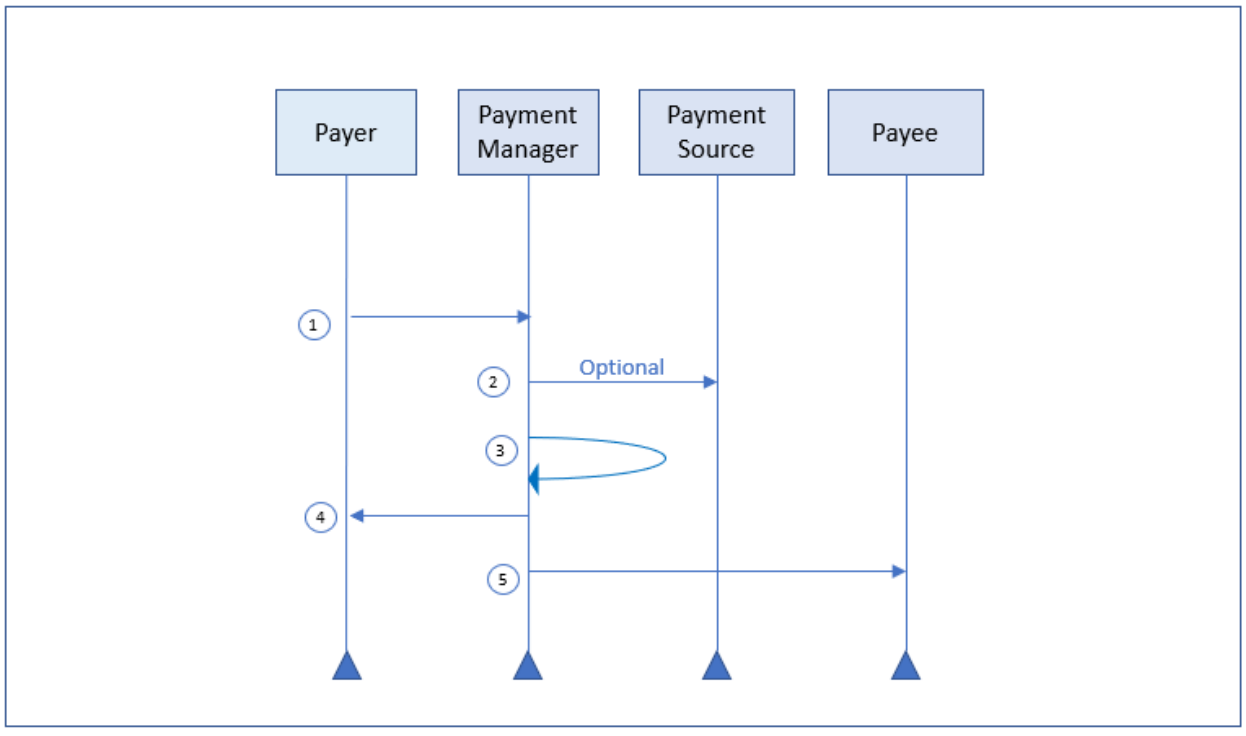


Diagram 6 - Payment Cancellation

3.1.6 Composite Scenario 5 – Payment Query

3.1.6.1 Overview

The Payer wants to understand the current payment information for a specific payment.

3.1.6.2 Pre-conditions

The payment exists in the Payment Manager.

3.1.6.3 Triggers

This triggers a need to know the details of an existing payment.

3.1.6.4 Basic course of events

1. The Payer sends to the Payment Manager the Get Payment Details message including the payment reference (*HTNG_GetPaymentDetailsRQ*)
2. The Payment Manager responds back to the Payer with the payment information (*HTNG_GetPaymentDetailsRS*)

3.1.6.5 Post conditions

The Payer has the latest information regarding the payment in the Payment Manager.

3.1.6.6 Message flow diagram

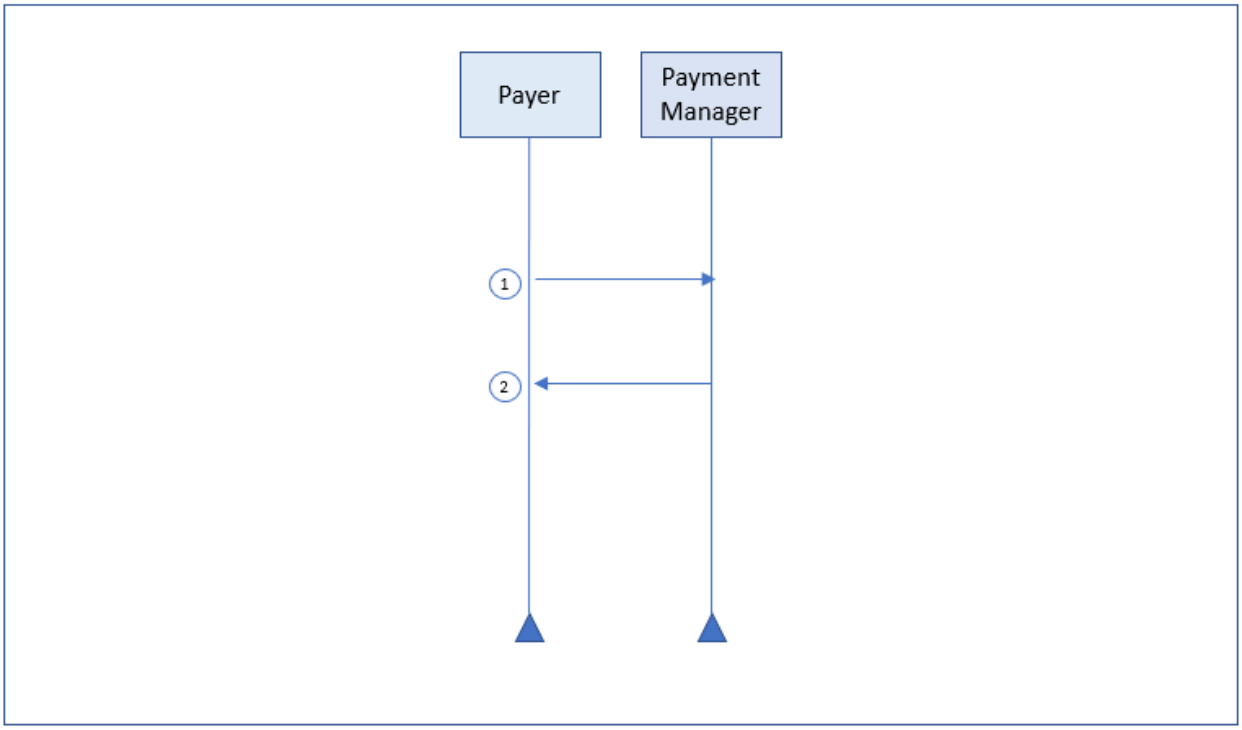


Diagram 7 - Payment Query

3.1.7 Composite Scenario 6 – Payment Status Query

3.1.7.1 Overview

The Payer wants to know the current status for a specific payment.

3.1.7.2 Pre-conditions

The payment exists in the Payment Manager.

3.1.7.3 Triggers

This triggers a need to know the status of an existing payment.

3.1.7.4 Basic course of events

1. The Payer sends to the Payment Manager the Get Payment Status message including the payment reference (*HTNG_GetPaymentStatusRQ*)
2. The Payment Manager responds back to the Payer with the payment status (*HTNG_GetPaymentStatusRS*)

3.1.7.5 Post conditions

The Payer has the latest status regarding the payment in the Payment Manager.

3.1.7.6 Message flow diagram

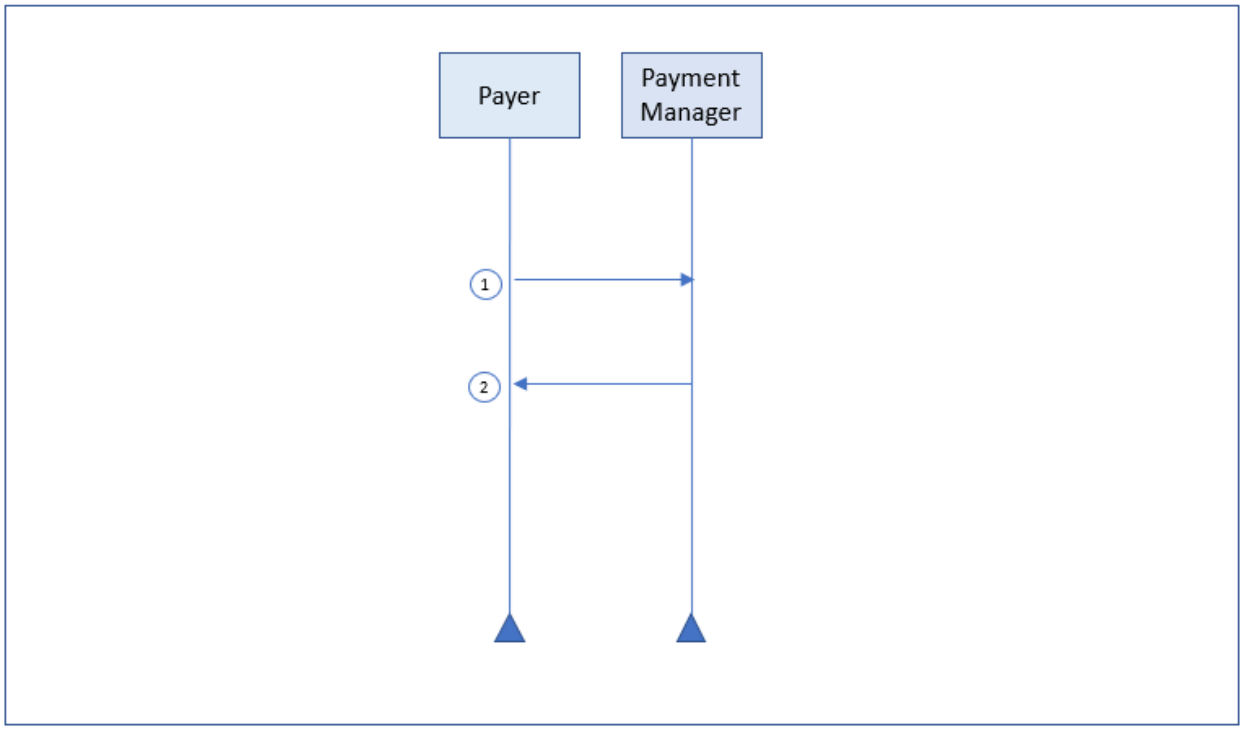


Diagram 8 - Payment Query Status

3.1.8 Composite Scenario 7 – Payment Initiation

3.1.8.1 Overview

The Payer wants to instruct the Payment Manager to start the payment process.

3.1.8.2 Pre-conditions

- The payment exists in the Payment Manager
- The payment status is pending or inactive

3.1.8.3 Triggers

This triggers the Payer to initiate a payment.

3.1.8.4 Basic course of events

1. The Payer sends to the Payment Manager the Initiate Payment message including the payment reference (*HTNG_InitiatePaymentRQ*)
2. The Payment Manager validates that the payment is eligible for initiation

3. If needed, the Payment Manager connects to the Payment Source to activate the payment instrument
4. The Payment Manager activates the payment
5. If needed, the Payment Manager initiates the payment by connecting to the Payment Processor
6. The Payment Manager responds back to the Payer (*HTNG_InitiatePaymentRS*)
7. Optionally, the Payment Manager sends to the Payee the Payment Status Notification message to inform them of the status change (*HTNG_PaymentStatusNotifRQ*)

3.1.8.5 Post conditions

The payment is initiated.

3.1.8.6 Message flow diagram

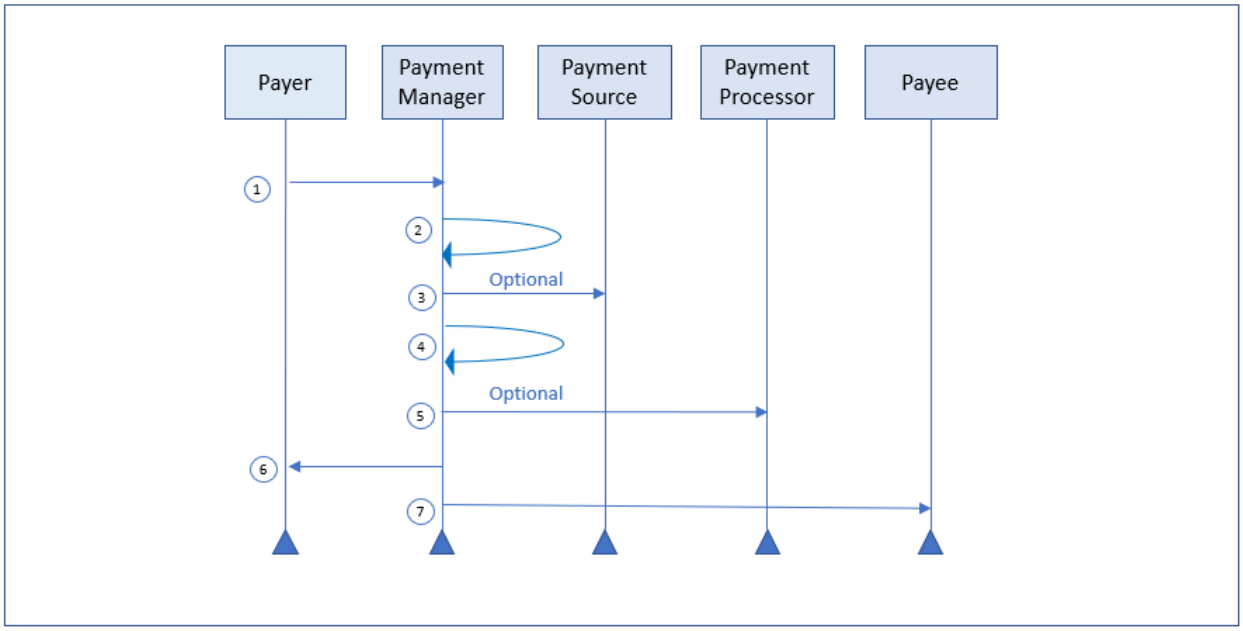


Diagram 9 - Payment Initiation

3.2 Payee Scenarios

This section is applicable to a Payee connecting to a Payment Manager.

3.2.1 Assumptions

- Payee is subscribed to the Payment Manager and has credentials
- Payee has a relationship with the Payer

3.2.2 Composite Scenario 1 – Payment Reference Query for a Booking

3.2.2.1 Overview

The Payee wants to verify, for a booking received, the existence of an associated payment in the Payment Manager.

3.2.2.2 Pre-conditions

A booking without a payment reference has been received from a Payer and the Payee expects that a payment was arranged through a designated Payment Manager.

3.2.2.3 Triggers

This triggers a need to ensure a payment has been created for a booking.

3.2.2.4 Basic course of events

1. The Payee checks for the existence of a payment for a booking by sending to the Payment Manager the Find Payment Ref message including the booking reference and the payer ID (*HTNG_FindPaymentRefRQ*)
2. The Payment Manager responds back to the Payee with the payment reference for the booking (*HTNG_FindPaymentRefRS*)

3.2.2.5 Post conditions

The Payee has the payment reference for the booking.

3.2.2.6 Exception path

1. The Payment Manager responds back to the Payee without the payment reference for the booking indicating that a payment reference was not found (*HTNG_FindPaymentRefRS*)
2. The Payee follows their predetermined error handling policy

3.2.2.7 Message flow diagram

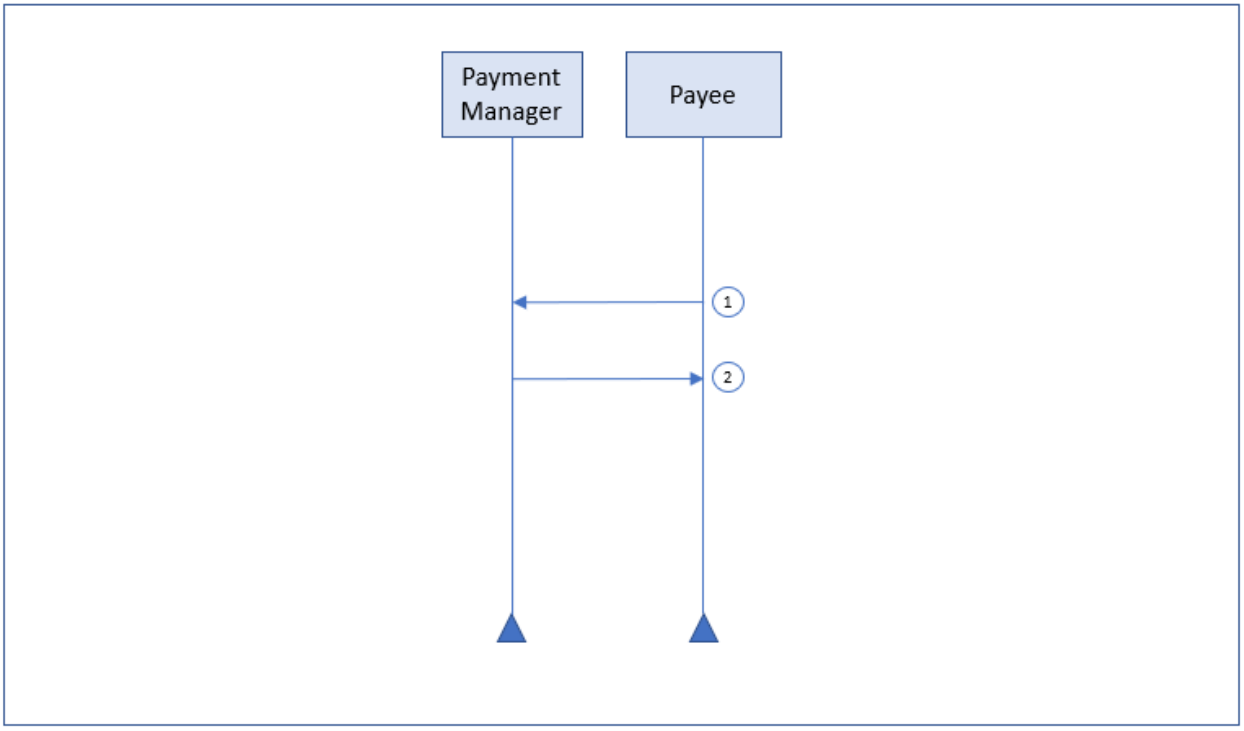


Diagram 10 - Payment Reference Query for a Booking

3.2.3 Composite Scenario 2 – Payment Query

3.2.3.1 Overview

The Payer wants to understand the current payment information for a specific payment.

3.2.3.2 Pre-conditions

The Payee has the payment reference of a payment that exists in the Payment Manager.

3.2.3.3 Triggers

This triggers a need to know the details of an existing payment.

3.2.3.4 Basic course of events

1. The Payee sends to the Payment Manager the Get Payment Details message including the payment reference (*HTNG_GetPaymentDetailsRQ*)
2. The Payment Manager responds back to the Payee with the payment information (*HTNG_GetPaymentDetailsRS*)

3.2.3.5 Post conditions

The Payee has the latest information regarding the payment in the Payment Manager.

3.2.3.6 Message flow diagram

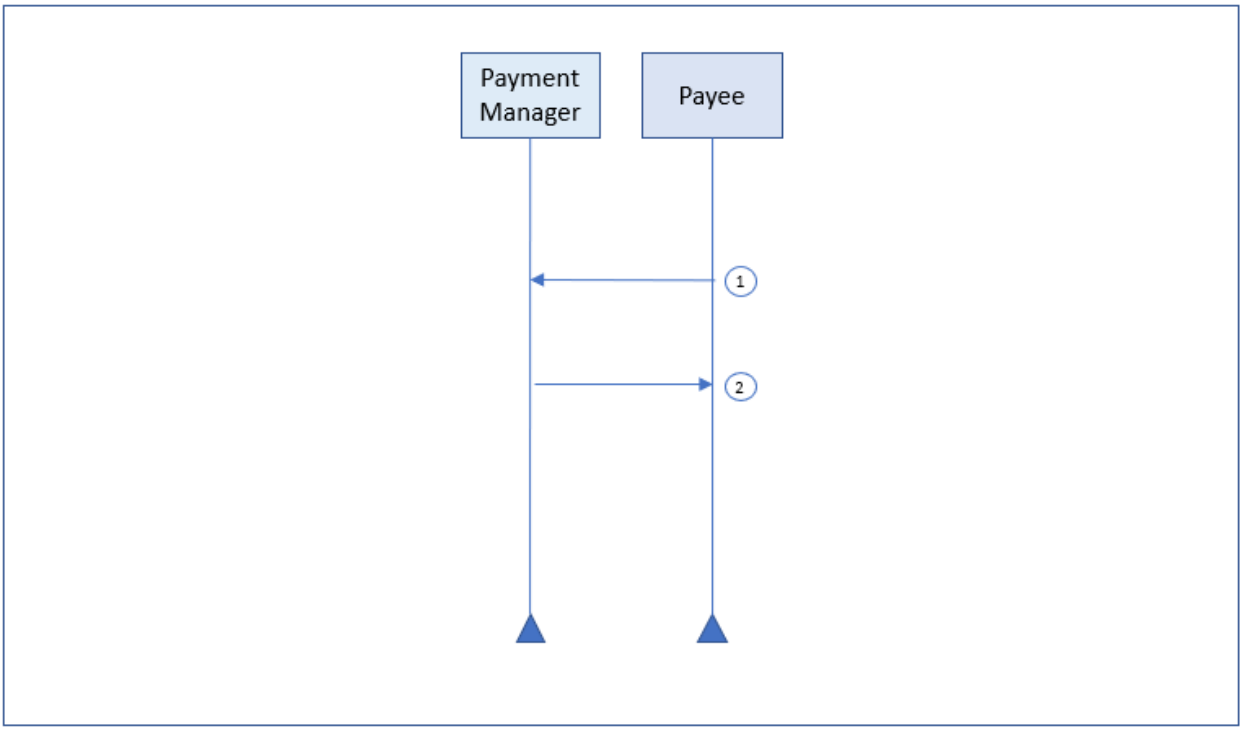


Diagram 11 - Payment Query

3.2.4 Composite Scenario 3 – Payment Status Query

3.2.4.1 Overview

The Payee wants to know the current status for a specific payment.

3.2.4.2 Pre-conditions

The payment exists in the Payment Manager.

3.2.4.3 Triggers

This triggers a need to know the status of an existing payment.

3.2.4.4 Basic course of events

1. The Payee sends to the Payment Manager the Get Payment Status message including the payment reference (*HTNG_GetPaymentStatusRQ*)
2. The Payment Manager responds back to the Payee with the payment status (*HTNG_GetPaymentStatusRS*)

3.2.4.5 Post conditions

The Payee has the latest status regarding the payment in the Payment Manager.

3.2.4.6 Message flow diagram

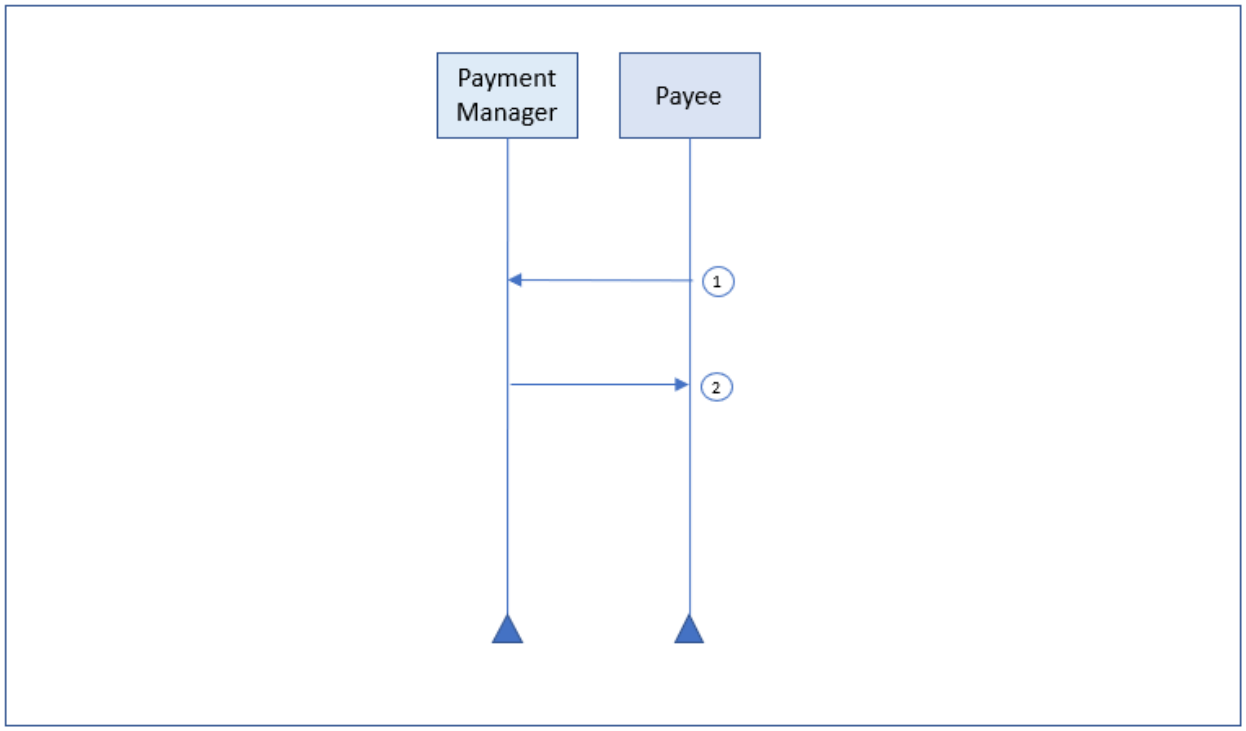


Diagram 12 - Payment Status Query

3.2.5 Composite Scenario 4 – Payment Initiation

3.2.5.1 Overview

The Payee wants to instruct the Payment Manager to start the payment process.

3.2.5.2 Pre-conditions

- The payment exists in the Payment Manager
- The Payment Processor is configured within Payment Manager for the Payee
- The payment is set to allow its initiation from the Payee

3.2.5.3 Triggers

This triggers the Payee to initiate a payment.

3.2.5.4 Basic course of events

1. The Payee sends to the Payment Manager the Initiate Payment message including the payment reference (*HTNG_InitiatePaymentRQ*)
2. The Payment Manager validates that the payment is eligible for initiation
3. The Payment Manager initiates the payment by connecting to the Payment Processor
4. The Payment Manager responds back to the Payee (*HTNG_InitiatePaymentRS*)
5. Optionally, the Payment Manager sends to the Payer the Payment Status Notification message to inform them of the status change

3.2.5.5 Post conditions

The payment is initiated.

3.2.5.6 Message flow diagram

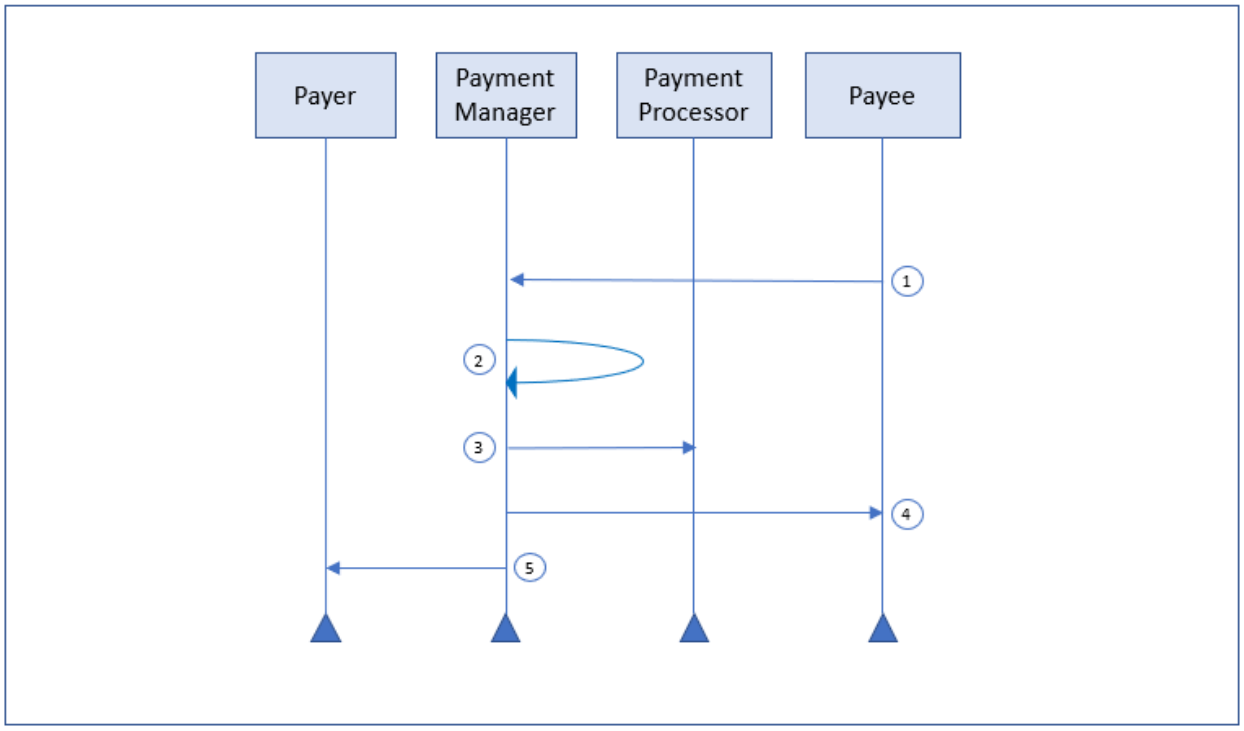


Diagram 13 - Payment Initiation

3.3 Payment Manager Scenarios

This section is applicable to a Payment Manager connecting to a Payer and Payee.

3.3.1 Assumptions

- Payee is subscribed to the Payment Manager's notification service
- Payer is subscribed to the Payment Manager's notification service

3.3.2 Composite Scenario 1 – Payment Status Change Initiated by the Payment Manager

3.3.2.1 Overview

The status of the payment has changed and the Payment Manager needs to inform those who have subscribed to the notification service.

3.3.2.2 Pre-conditions

The payment from the Payer to the Payee exists in the Payment Manager.

3.3.2.3 Triggers

This triggers a need to notify the Payer and/or the Payee of a payment status change that originated in the Payment Manager.

3.3.2.4 Basic course of events

1. If the Payer is subscribed, the Payment Manager sends to the Payer the Payment Status Notification message (*HTNG_PaymentStatusNotifRQ*)
2. If the Payee is subscribed, the Payment Manager sends to the Payee the Payment Status Notification message (*HTNG_PaymentStatusNotifRQ*)

3.3.2.5 Post conditions

The Payer and/or Payee have the latest status regarding the payment in the Payment Manager.

3.3.2.6 *Message flow diagram*

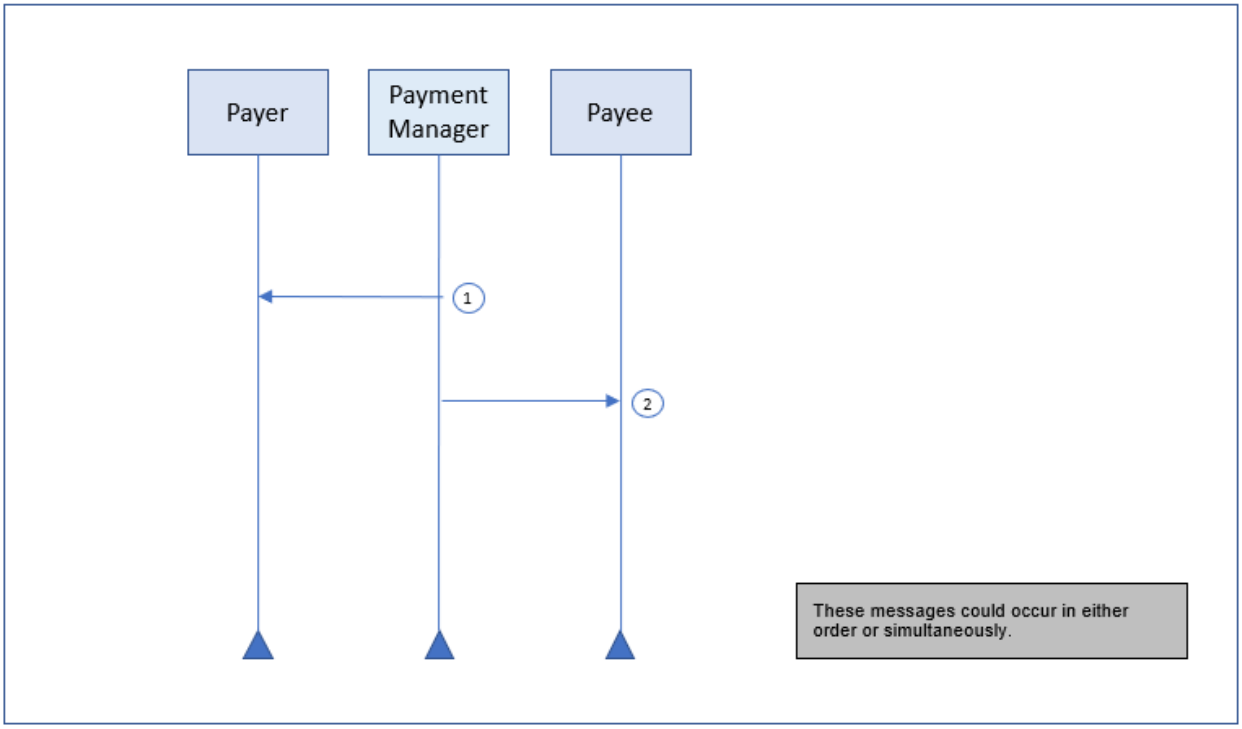


Diagram 14 - Payment Status Change Initiated by the Payment Manager

4 Use Cases

This section describes several end-to-end use case scenarios.

4.1 Use Case 1 – Hotel Booking & Subsequent Booking Modification

The consumer has booked a hotel reservation on an OTA website for a rate that requires prepayment to the OTA. The consumer provides their credit card which will be charged by the OTA at the time of booking. The OTA needs to communicate to the hotel how the hotel will be paid, which in this case is by using a virtual credit card (VCC) upon check-out.

Prior to the payment initiation, the consumer modifies their hotel booking on the OTA website to extend their check-out by one night. This results in an increase in the total payment due to the hotel. The Payment Manager needs to communicate any changes in how the payment will be processed to the hotel.

4.1.1 Assumptions

- OTA utilizes the Payment Manager to create the VCC
- Payment Manager connects to the Hotel's Payment Processor to initiate payment
- OTA and Hotel are both subscribed to receive updates from the Payment Manager regarding payment details and status

4.1.2 New Booking

1. OTA generates a payment reference
2. OTA sends a booking request to the hotel reservation system using *OTA_HotelResRQ* and includes the payment reference
3. OTA sends payment instructions using *HTNG_CreatePaymentRQ* to Payment Manager including payment reference, the OTA booking reference and requests a VCC to be issued by the Payment Manager; instructions indicate that payment is to be automatically initiated on payment date by Payment Manager
4. Payment Manager sends a request to the Payment Source to issue a VCC
5. Payment Manager sends payment details to Hotel using *HTNG_PaymentDetailsNotifRQ*

4.1.3 Booking Modified

1. OTA sends a request to the Hotel to modify the hotel reservation stay dates using *OTA_HotelResModifyRQ* and includes the payment reference; hotel confirms modification and the booking confirmation number remains the same
2. OTA sends a request to the Payment Manager to modify the payment instructions using *HTNG_ModifyPaymentRQ* and includes payment reference; instructions include a change to the payment amount and payment initiation date
3. Payment Manager sends a request to the Payment Source to change the VCC amount
4. Payment Manager sends payment details to Hotel using *HTNG_PaymentDetailsNotifRQ*

4.1.4 Payment Initiated

1. Conditions to initiate payment are met and the Payment Manager initiates the payment process
2. Payment Manager sends request to Hotel's Payment Processor to charge the VCC on behalf of Hotel

3. Payment Processor initiates settlement of payment with Hotel's financial institution
4. Payment Manager sends updated payment status to the OTA and Hotel using the *HTNG_PaymentStatusNotifRQ*

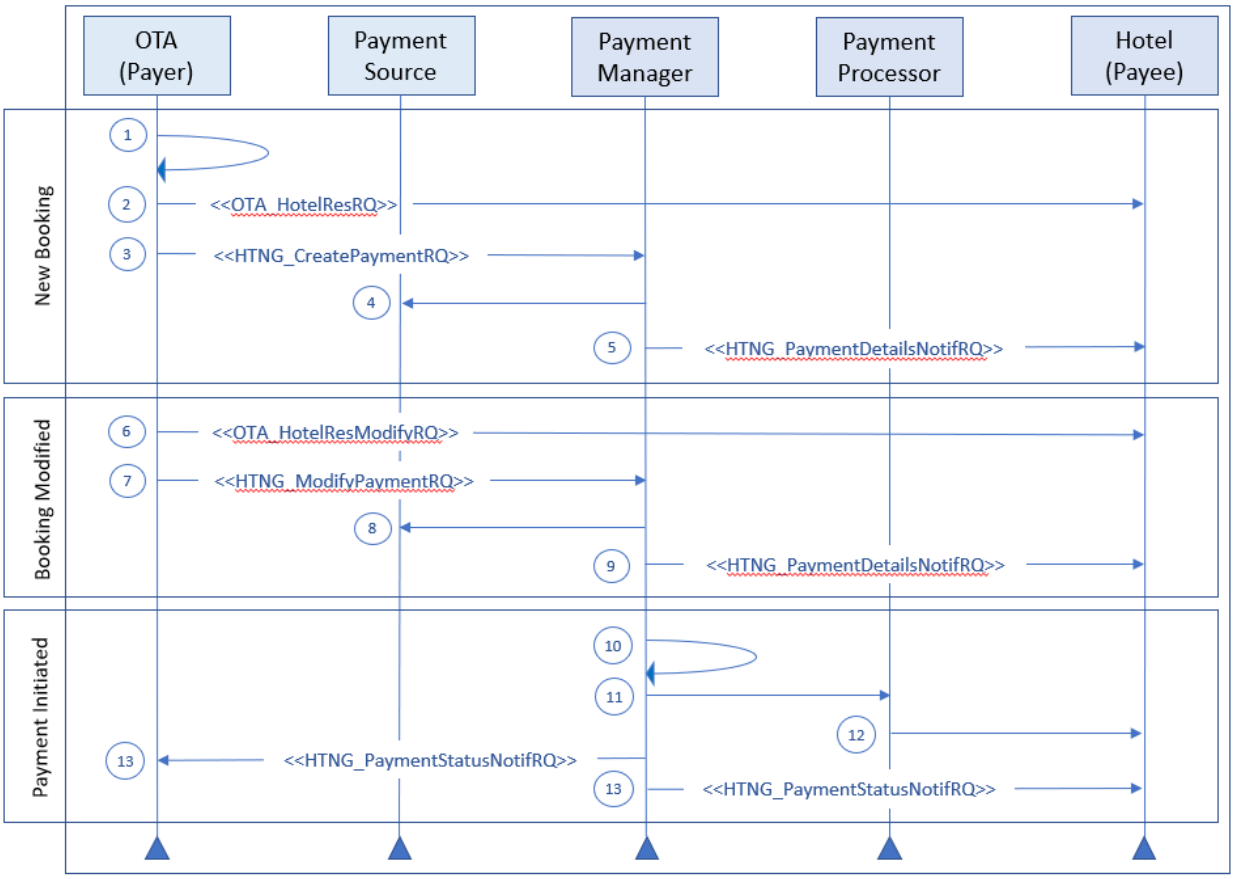


Diagram 15 - Hotel Booking & Subsequent Booking Modification Use Case

4.2 Use Case 2 – Hotel Booking Cancelled with Payment Due

In this use case, the consumer cancels their existing booking through the OTA website. The booking was for three nights and is subject to a one night cancellation fee so there is a reduced payment amount due to the hotel as a result of the booking.

4.2.1 Assumptions

- Original booking payment method was VCC which was obtained by the OTA from a third-party Payment Source
- Payment Manager connects to the Hotel's Payment Processor to initiate payment
- Payment conditions provided in the original payment allow for early payment for a cancelled booking

- OTA is subscribed to receive updates from the Payment Manager regarding payment details and status

4.2.2 Booking Canceled

1. OTA sends a request to Payment Source to modify the VCC appropriately
2. OTA sends a booking cancellation to the Hotel using *OTA_CancelRQ*
3. OTA sends a request to the Payment Manager to modify the payment instructions using *HTNG_ModifyPaymentRQ*; instructions include a change to the payment amount to cover the cancel fee only and payment date to pay now

4.2.3 Payment Initiated

1. Payment Manager confirms payment conditions are met and then automatically initiates payment
2. Payment Manager sends request to Hotel's Payment Processor to charge the VCC on behalf of Hotel
3. Payment Processor initiates settlement of payment with Hotel's financial institution
4. Payment Manager sends updated payment status to the OTA using the *HTNG_PaymentStatusNotifRQ*
5. Hotel requests payment status from Payment Manager using *HTNG_GetPaymentStatusRQ*

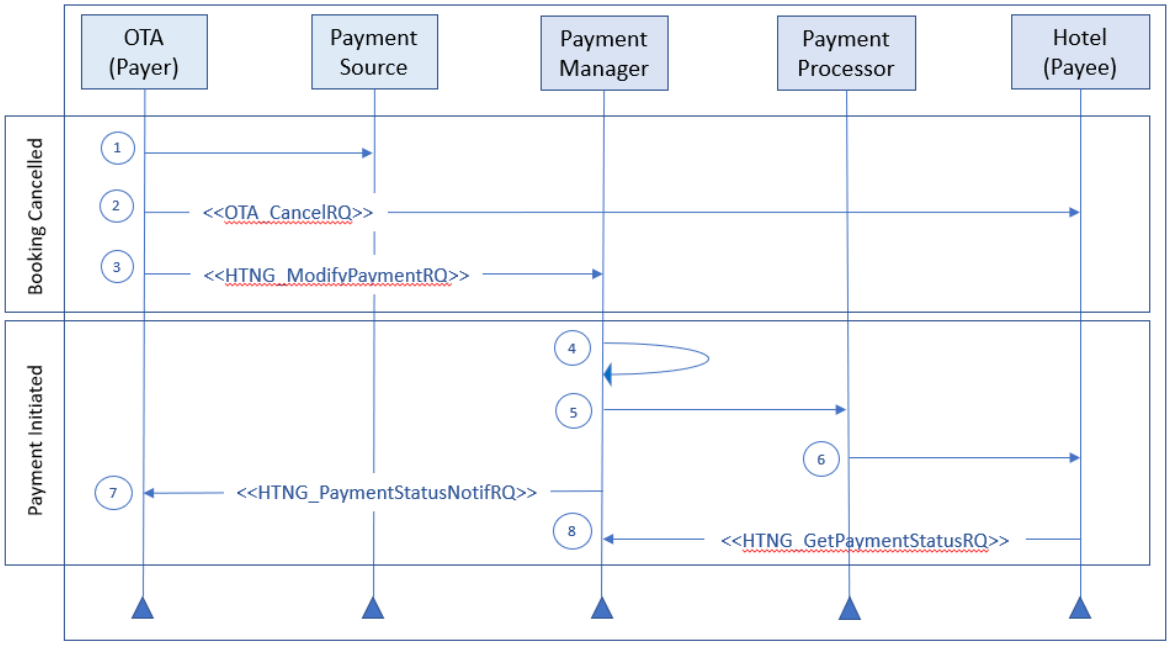


Diagram 16 - Hotel Booking Cancelled with Payment Due Use Case

4.3 Use Case 3 – Hotel Booking Cancelled without Payment Due

In this use case, the consumer cancels their existing hotel booking through the OTA website. The cancellation is not subject to any cancellation fee so there is no payment to be made to the Hotel.

4.3.1 Assumptions

- Original booking payment method was VCC which was obtained by the OTA from a third-party Payment Source
- OTA and Hotel are both subscribed to receive updates from the Payment Manager regarding payment details and status

4.3.2 Booking Cancelled

1. OTA sends *OTA_CancelRQ* to the Hotel to cancel the booking
2. OTA sends a request to the Payment Source to close the VCC
3. OTA sends payment cancellation request *HTNG_CancelPaymentRQ* to Payment Manager

4.3.3 Payment Cancelled

1. Payment Manager updates payment status to show as Cancelled
2. Payment Manager sends updated payment status to Hotel using *HTNG_PaymentStatusNotifRQ*

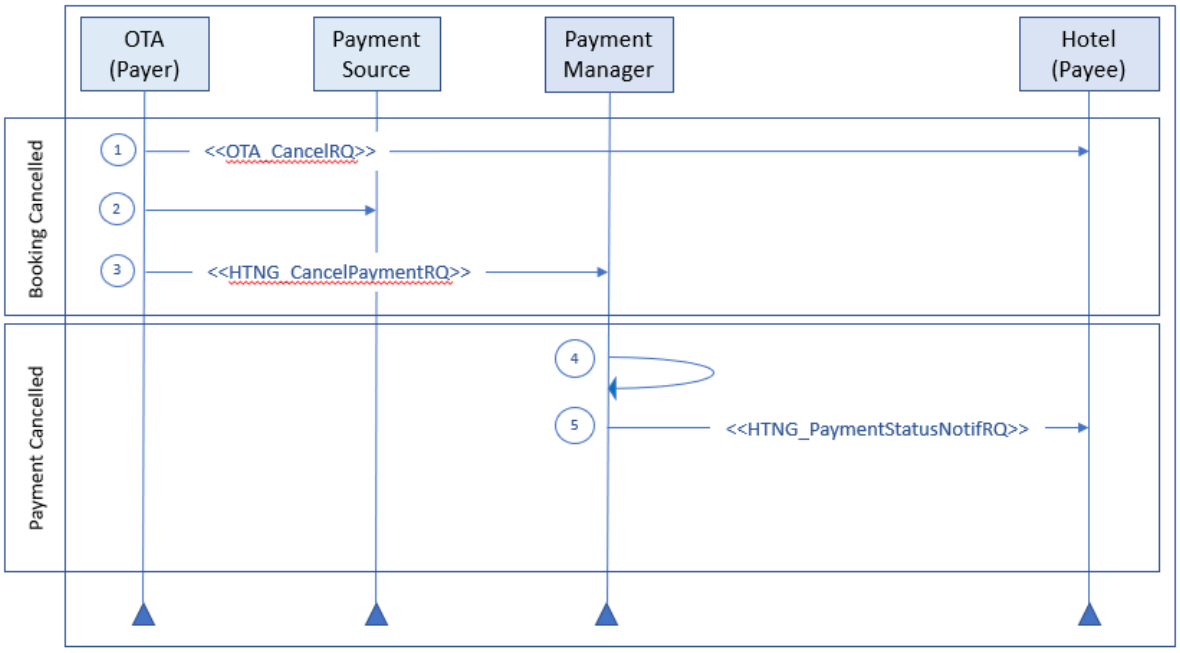


Diagram 17 - Hotel Booking Cancelled without Payment Due Use Case

4.4 Use Case 4 – Guest is a No-Show with Payment Due

In this use case, the guest does not check-in for their stay at the hotel and the booking is flagged as a no-show by the hotel. The original booking was for three nights and is subject to a no-show fee equal to a one night's stay so there is a reduced payment amount due to the hotel.

4.4.1 Assumptions

- Original booking payment method was VCC which was obtained by the OTA from a third-party Payment Source
- Payment Manager connects to the Hotel's Payment Processor to initiate payment
- OTA and Hotel are both subscribed to receive updates from the Payment Manager regarding payment details and status
- Payment instructions include early collection conditions for a no-show

4.4.2 No-Show Booking

The Hotel notifies the OTA of guest no-show.

4.4.3 Payment Initiated

1. Hotel initiates payment request to Payment Manager using *HTNG_InitiatePaymentRQ* and indicates reason as no-show
2. Payment Manager validates that payment conditions are met and initiates payment
3. Payment Manager sends request to Hotel's Payment Processor to charge the VCC on behalf of Hotel
4. Payment Processor initiates settlement of payment with Hotel's financial institution
5. Payment Manager sends updated payment status to the OTA using the *HTNG_PaymentStatusNotifRQ*

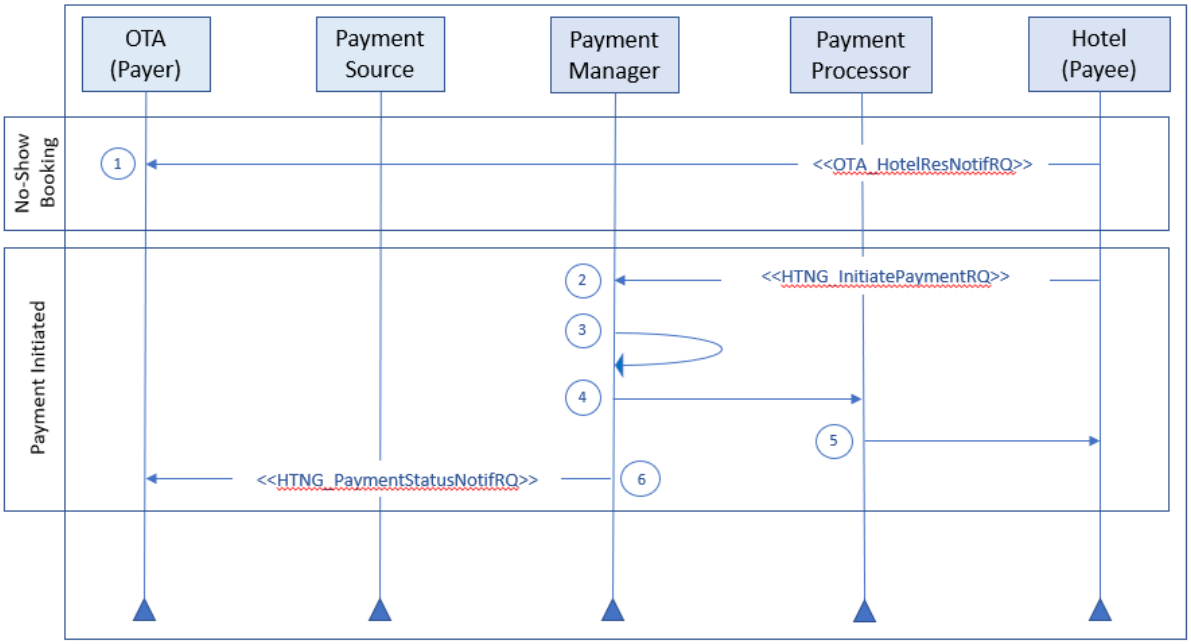


Diagram 18 - Guest is a No-Show with Payment Due Use Case

4.5 Use Case 5 – Payment of a Commission with Invoice

In this use case, a travel agency is owed commissions for multiple guests' stays at a hotel.

4.5.1 Assumptions

- Hotel requires an invoice for payment
- Payment method between the Hotel and the Agency is a bank transfer
- Payment Manager connects to the Hotel's Payment Processor to initiate payment
- Hotel and Agency are both subscribed to receive updates from the Payment Manager regarding payment details and status

4.5.2 Payment Created

1. Hotel sends payment instructions to Payment Manager using *HTNG_CreatePaymentRQ* including payment reference and multiple hotel confirmation numbers; instructions indicate that an invoice is required for payment
2. Payment Manager sends payment details to Agency using *HTNG_PaymentDetailsNotifRQ*; payment details indicate that an invoice is required for payment

4.5.3 Payment Initiated

1. Agency sends invoice to Hotel
2. Hotel initiates payment request to Payment Manager using *HTNG_InitiatePaymentRQ* and includes the invoice number and date
3. Payment Manager confirms that payment conditions are met and initiates payment
4. Payment Manager sends request to Hotel's Payment Processor to initiate transfer on behalf of Hotel
5. Hotel's Payment Processor initiates settlement of payment with Agency's financial institution
6. Payment Manager sends updated payment status to both the Agency and the Hotel using the *HTNG_PaymentStatusNotifRQ*

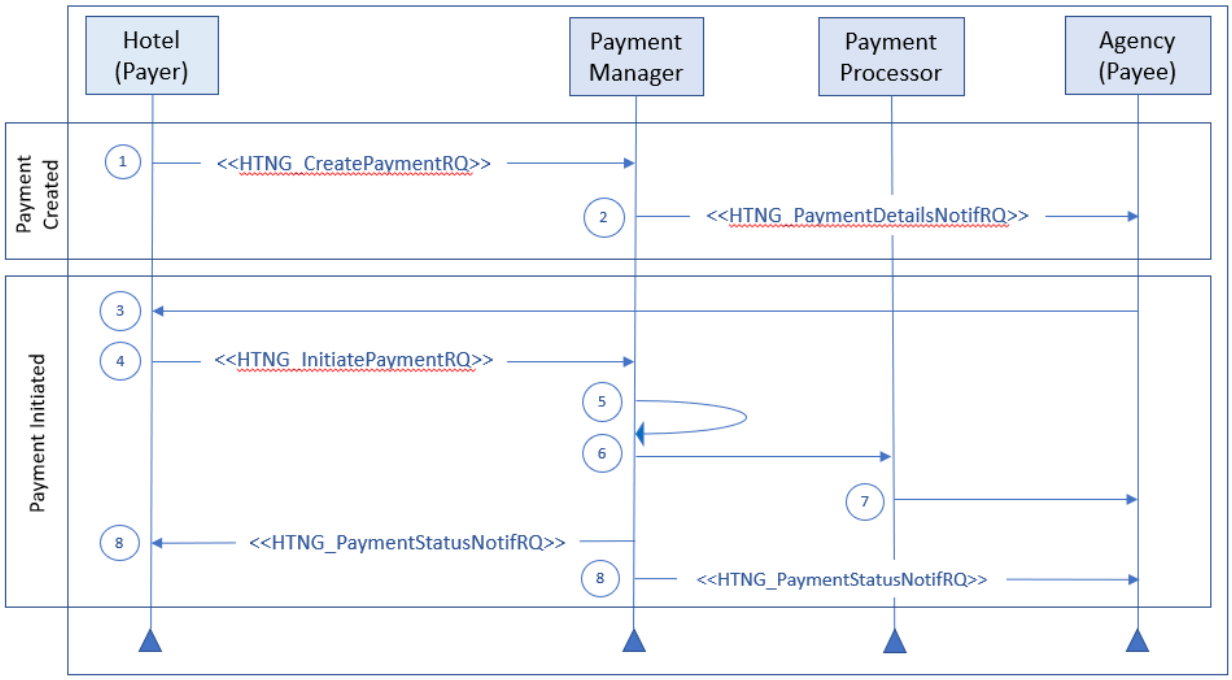


Diagram 19 - Payment of a Commission with Invoice Use Case

5 Messages

5.1 Create Payment

5.1.1 HTNG_CreatePaymentRQ

This message is used by the Payer to create a payment by sending payment instructions to the Payment Manager to create the payment order.

Element	Cardinality	Data Type	Description/Contents
HTNG_CreatePaymentRQ		Root Element	Payment instructions sent from the Payer to the Payment Manager to create a payment order
HTNG_CreatePaymentRQ/PaymentDetail	0..1	PaymentDetailType	Payment details for a specific PaymentRef requested

5.1.2 HTNG_CreatePaymentRS

This message is used in response to the create payment request to inform the Payer if the message was successfully processed. No payment details are provided in this message. If the requestor is interested in additional information this message should be followed by the *HTNG_GetPaymentDetailsRQ* or *HTNG_GetPaymentStatusRQ*.

Element	Cardinality	Data Type	Description/Contents
HTNG_CreatePaymentRS		Root Element	Returns the payment status for the payment being created
HTNG_CreatePaymentRS/PaymentRef	1	GUID	The unique identifier and index key and/or primary key for the payment

5.2 Cancel Payment

5.2.1 HTNG_CancelPaymentRQ

The Payer may use this message to cancel an existing payment order with the Payment Manager.

Element	Cardinality	Data Type	Description/Contents
HTNG_CancelPaymentRQ		Root Element	Instructions sent from the Payer to cancel a payment order
HTNG_CancelPaymentRQ/PaymentRef	1	GUID	The unique identifier and index key and/or primary key for the payment to be canceled

5.2.2 HTNG_CancelPaymentRS

The Payment Manager uses this message to respond to the cancellation request sent by the Payer. This message will inform the Payer if the cancellation was successfully processed. No payment details are provided in this message. If the requestor is interested in additional information this message should be followed by the *HTNG_GetPaymentDetailsRQ* or *HTNG_GetPaymentStatusRQ*.

Element	Cardinality	Data Type	Description/Contents
HTNG_CancelPaymentRS		Root Element	Response to a cancel payment request
HTNG_CancelPaymentRS/PaymentRef	1	GUID	The unique identifier and index key and/or primary key for the canceled payment

5.3 Modify Payment

5.3.1 HTNG_ModifyPaymentRQ

The Payer uses this message to request a modification to a payment order that exists in the Payment Manager. ***Please note that not all fields in this message are modifiable.***

Element	Cardinality	Data Type	Description/Contents
HTNG_ModifyPaymentRQ		Root Element	Payment instructions sent from the Payer to the Payment Manager to modify a payment order
HTNG_ModifyPaymentRQ/PaymentDetail	1	PaymentDetailType	Payment details for a specific PaymentRef to be modified

5.3.2 HTNG_ModifyPaymentRS

The Payment Manger uses this message to respond to the modify request sent by the Payer. This message will inform the Payer if the modification was successfully processed. No payment details are provided in this message. If the requestor is interested in additional information this message should be followed by the *HTNG_GetPaymentDetailsRQ* or *HTNG_GetPaymentStatusRQ*.

Element	Cardinality	Data Type	Description/Contents
HTNG_ModifyPaymentRS		Root Element	Returns the payment status for the payment being modified.
HTNG_ModifyPaymentRS/PaymentRef	1	GUID	The unique identifier and index key and/or primary key for the canceled payment.

5.4 Find Payment Reference

5.4.1 HTNG_FindPaymentRefRQ

The Payee may use this message to locate the payment reference for a booking using the booking reference and providing Payer information.

This message set is used when the payment reference is not received by the supplier. This may be the case if all parties in the booking channel have not implemented the GuaranteeType and GuaranteeID fields in the booking messages that are used to identify and send the payment reference. If multiple booking references are included in a single message and any of them fail, they will not be returned. It is the requestor's responsibility to identify which were not returned and send each one in a single message to receive an error response to obtain more information as to why the find was not successful.

Element	Cardi-nality	Data Type	Description/Contents
HTNG_FindPaymentRefRQ		Root Element	Used to find the payment reference for one or more specific bookings from the same payer The booking reference and booking date is submitted and the payment reference is returned
HTNG_FindPaymentRefRQ/PaymentInfo	1	PaymentInfoRequestType	Specifies the payer and booking information used to retrieve the payment reference

5.4.2 HTNG_FindPaymentRefRS

The Payment Manager uses this message to respond to the Payee to return a payment reference for a booking.

Element	Cardi-nality	Data Type	Description/Contents
HTNG_FindPaymentRefRS		Root Element	Used to return payment references for one or more bookings for a single payer
HTNG_FindPaymentRefRS/PaymentInfos	1	Element	A collection of Payment References for the booking references provided
./PaymentInfos/PaymentInfo	1..n	PaymentInfoType	A booking reference and the corresponding payment reference

5.5 Get Payment Details

5.5.1 HTNG_GetPaymentDetailsRQ

The Payer or Payee may use this message to retrieve payment details for a payment order from the Payment Manager. If multiple payment references are included in a single message and any of them fail, they will not be returned. It is the requestor's responsibility to identify which were not returned and send

each one in a single message to receive an error response to obtain more information as to why the retrieval was not successful.

Element	Cardinality	Data Type	Description/Contents
HTNG_GetPaymentDetailsRQ		Root Element	Used to request payment details from the Payment Manager
HTNG_GetPaymentDetailsRQ/PaymentRefs	1	Element	A collection of Payment References for which the payment details are being requested
./PaymentRefs/PaymentRef	1..n	GUID	The unique identifier and index key and/or primary key for the payment details

5.5.2 HTNG_GetPaymentDetailsRS

The Payment Manager uses this message to respond to the requestor providing the details of a payment order(s).

Element	Cardinality	Data Type	Description/Contents
HTNG_GetPaymentDetailsRS		Root Element	Returns the payment details for the payment refs requested
HTNG_GetPaymentDetailsRS/PaymentDetails	1	Element	A collection of payment details for the PaymentRefs requested
./PaymentDetails/PaymentDetail	1..n	PaymentDetailRecordType	Payment details for a specific PaymentRef requested

5.6 Get Payment Status

5.6.1 HTNG_GetPaymentStatusRQ

The Payer or Payee may use this message to retrieve the payment status for a payment order from the Payment Manager. If multiple payment references are included in a single message and any of them fail they will not be returned. It is the requestor's responsibility to identify which were not returned and send each one in a single message to receive an error response to obtain more information as to why the retrieval was not successful.

Element	Cardinality	Data Type	Description/Contents
HTNG_GetPaymentStatusRQ		Root Element	Used to request payment status from the Payment Manager
HTNG_GetPaymentStatusRQ/PaymentRefs	1	Element	A collection of Payment References for which the payment details are being requested

./PaymentRefs/PaymentRef	1..n	GUID	The unique identifier and index key and/or primary key for the payment details.

5.6.2 HTNG_GetPaymentStatusRS

The Payment Manger uses this message to respond to the requestor providing the payment status for the payment order(s).

Element	Cardinality	Data Type	Description/Contents
HTNG_GetPaymentDetailsRS		Root Element	Returns the payment details for the payment refs requested
HTNG_GetPaymentDetailsRS/Payments	1	Element	A collection of payments for which the status is being requested
./Payments/Payment	1..n	PaymentStatusReferenceType	A payment for which the status is being requested

5.7 Initiate Payment

5.7.1 HTNG_InitiatePaymentRQ

This message may be used by the Payer or Payee to initiate a payment when conditions within the payment order are met.

Element	Cardinality	Data Type	Description/Contents
HTNG_InitiatePaymentRQ		Root Element	Used to initiate the payment process
HTNG_InitiatePaymentRQ/Payment	1	PaymentInitiationType	A payment to be initiated

5.7.2 HTNG_InitiatePaymentRS

The Payment Manager uses this message to respond to the requestor to indicate if the payment initiation was successful.

Element	Cardinality	Data Type	Description/Contents
HTNG_InitiatePaymentRS		Root Element	Used to indicate the success or failure of payments that were requested to be initiated
HTNG_InitiatePaymentRS/PaymentRef	1	GUID	The unique identifier and index key and/or primary key for the payment

5.8 Payment Details Notification

5.8.1 HTNG_PaymentDetailsNotifRQ

This is a push message used by the Payment Manager to inform the Payee of current payment details for one or more payments.

Element	Cardin-ality	Data Type	Description/Contents
HTNG_PaymentDetailsNotifRQ		Root Element	Used to push payment details from one party to another
HTNG_PaymentDetailsNotifRQ/PaymentDetails	1	Element	A collection of payment details
./PaymentDetails/PaymentDetail	1..n	PaymentDetailRecordType	Payment details

5.9 Payment Status Notification

5.9.1 HTNG_PaymentStatusNotifRQ

This is a push message used by the Payment Manager to inform the Payer and/or Payee of the current payment status for one or more payments.

Element	Cardin-ality	Data Type	Description/Contents
HTNG_PaymentStatusNotifRQ		Root Element	Sends payment status for one or more payment references
HTNG_PaymentStatusNotifRQ/PaymentDetails	1	Element	A collection of payments for which the status is being provided
./PaymentDetails/PaymentDetail	1..n	PaymentStatusReferenceType	The payment status for a specific payment reference

6 Complex Types

These complex types are used within the messages and are defined just once here. You will see them referenced in the message data table in section 5. Some complex types are used within other complex types as well and again, you will find references within those types.

6.1 BookingInfoType

BookingInfoType identifies a booking by the booking reference and booking date.

Element	Cardinality	Data Type	Description/Contents
HTNG_OPA_CommonTypes			A file that contains the common types used across the Open Payment Alliance messages
BookingInfoType		Complex Type	Identifies a booking by the booking ref and booking date
BookingInfoType/BookingRefs	1..2	Element	Booking references for the reservation
BookingRefs/BookingRef	1	String	Booking reference for the reservation
BookingRefs/BookingType	1	String Enumeration	Specifies the entity that created the booking reference Possible options: • BookingSource • HotelConfirmationNumber
BookingInfoType/BookingDate	1	Date	The date the booking was made

6.2 BookingType

BookingType provides booking details related to the payment.

Element	Cardinality	Data Type	Description/Contents
HTNG_OPA_CommonTypes			A file that contains the common types used across the Open Payment Alliance messages
BookingType		Complex Type	Booking details related to the payment
BookingType/Booking	1..n	Element	A booking related to the payment
../Booking/BookingRefs	1..2	Element	Booking references for the reservation

../BookingRefs/BookingRef	1	String	Booking reference for the reservation
../BookingRefs/BookingType	1	String Enumeration	Specifies the entity that created the booking reference Possible options: • BookingSource • HotelConfirmationNumber
../Booking/BookingDate	0..1	Date	Date of generation of the reservation
../Booking/ServiceStartDate	0..1	Date	Start date of the service
../Booking/ServiceEndDate		Date	The amount related to this booking

6.3 CardPaymentType

CardPaymentType specifies the details related to a payment card.

Element	Cardin-ality	Data Type	Description/Contents
HTNG_OPA_CommonTypes			A file that contains the common types used across the Open Payment Alliance messages
CardPaymentType		Complex Type	Specifies the details related to a payment card
CardPaymentType/PAN	1	String	The Primary Account Number that identifies the issuer and the particular cardholder account If a TokenProviderID is present, then the card token is found in this field
CardPaymentType/Expiration Date	1	DateTime	The expiration date of the credit card
CardPaymentType/CVV	0..1	String	Card Verification Value of the credit card Refer to PCI DSS requirement 3.2
CardPaymentType/CardHolderName	0..1	String	Name of the card holder
CardPaymentType/TokenProviderID	0..1	String	Provider ID

6.4 CardRequestDetailsType

CardRequestDetailsType is the information to request a payment card.

Element	Cardi-nality	Data Type	Description/Contents

HTNG_OPA_CommonTypes			A file that contains the common types used across the Open Payment Alliance messages
CardRequestDetailsType		Complex Type	Information to request a payment card
CardRequestDetailsType/CardType	1	String Enumeration	The type of payment card (e.g. Visa, Mastercard, Bank of America, etc.) *See schema for possible list of values*
CardRequestDetailsType/Issuer	1	String Enumeration	The Primary Account Number that identifies the issuer and the particular cardholder account *See schema for possible list of values*
CardRequestDetailsType/CardAmount	1	MonetaryAmountType	Specifies the card amount
CardRequestDetailsType/SchemProgram	0..1	String	Identifies a particular program offered by the card issuer
CardRequestDetailsType/ValidFrom	1	DateTime	Specifies the earliest date that the requestor would like the credit card to be valid Expressed in ISO 8601 format
CardRequestDetailsType/ValidThrough	1	DateTime	Specifies the latest date that the requestor would like the credit card to be valid Expressed in ISO 8601 format

6.5 CardTokenType

CardTokenType specifies information related to a card token.

Element	Cardinality	Data Type	Description/Contents
HTNG_OPA_CommonTypes			A file that contains the common types used across the Open Payment Alliance messages
CardTokenType		Complex Type	Specifies information related to a card token
CardDetailsType/Mask	0..1	String (Restricted)	Masked card number
CardDetailsType/Token	1	String (Restricted)	Tokenized card number
CardDetailsType/TokenProviderID	1	String	Provider ID

6.6 CurrencyCodeType

CurrencyCodeType provides a currency code to reflect the currency in which an amount may be expressed.

Element	Cardin-ality	Data Type	Description/Contents
HTNG_OPA_CommonTypes			A file that contains the common types used across the Open Payment Alliance messages
CurrencyCodeType		Complex Type	Provides a currency code to reflect the currency in which an amount may be expressed
CurrencyCodeType/CurrencyCode	1	String (Restricted)	An ISO 4217 (3) alpha character code that specifies a monetary unit
CurrencyCodeType/DecimalPlaces	1	NonNegativeInteger	The ISO 4217 standard "minor unit" for the number of decimal places for the currency code

6.7 EarlyCollectionConditionType

EarlyCollectionConditionType specifies conditions for when an early collection of the payment is allowed.

Element	Cardin-ality	Data Type	Description/Contents
HTNG_OPA_CommonTypes			A file that contains the common types used across the Open Payment Alliance messages
EarlyCollectionConditionType		Complex Type	Specifies conditions for when an early collection of the payment is allowed
EarlyCollectionConditionType/EarlyCollectionCondition	1	Element	A term that applies to collect a payment prior to the payment date
./EarlyCollectionCondition/EarlyCollectionReason	1	String Enumeration	Specifies the condition for early collection Possible values: • NoShow • Cancel • EarlyCheckOut • EarlyPayAgreement
./EarlyCollectionCondition/EarlyCollectionDetail	1..n	Element	Specifies the date(s) and amount that apply for early collection

			Either the DateFrom or DateThrough field MUST be sent, however both may be sent together to indicate a date range
./EarlyCollectionDetail/DateFrom	0..1	DateTime	The date from which this term applies expressed in ISO 8601 format
./EarlyCollectionDetail/DateThrough	0..1	DateTime	The date through which this term applies expressed in ISO 8601 format
./EarlyCollectionDetail/Paym, entAmount	1	MonetaryAmountType	Specifies the amount of the payment and payment currency

6.8 EncryptedDataType

This data element is defined by the W3C and detail regarding its format and usage may be found at: https://www.w3.org/TR/xmlenc-core1/

6.9 FormOfPaymentDetailType

FormofPaymentDetailType provides detailed information regarding the type of payment.

Element	Cardin-ality	Data Type	Description/Contents
HTNG_OPA_CommonTypes			A file that contains the common types used across the Open Payment Alliance messages
FormOfPaymentDetailType		Complex Type	Provides detailed information regarding the type of payment
FormOfPaymentDetailType/PaymentCard	0..1	Element	Information regarding a payment card (e.g. credit or debit)
./PaymentCard/VirtualInd	0..1	Boolean	When true, the payment card is a virtual card
./PaymentCard/CardRequest Details	1	CardRequestDetailsType	Details pertaining to the payment when the Payment Manager is issuing the card. *Choice element, one of the three items is required: CardRequestDetails OR CardDetails OR CardToken*
./PaymentCard/CardPayment	1	CardPaymentType	Details regarding the card when issued prior to the Payment Manager
./PaymentCard/EncryptedData	1	EncryptedDataType	If the CardPayment element is encrypted, the EncryptedData element appears in place of the CardPayment element and can be found in the CipherData/CipherValue element The EncryptedData element is defined by the W3C XML Encrypted standard

FormOfPaymentType/BankTransfer	0..1	Element	Transfer of funds from one financial institution to another
./BankTransfer/PaymentServiceProvider	1	String	The payment service provider for the bank transfer

6.10 FormOfPaymentType

FormofPaymentType provides information regarding the type of payment.

Element	Cardinality	Data Type	Description/Contents
HTNG_OPA_CommonTypes			A file that contains the common types used across the Open Payment Alliance messages
FormOfPaymentType		Complex Type	Information regarding the type of payment
FormOfPaymentType/PaymentCard	0..1	Element	Information regarding a payment card (e.g. credit or debit)
./PaymentCard/VirtualInd	0..1	Boolean	When true, the payment card is a virtual card
./PaymentCard/CardType	0..1	String Enumeration	The type of payment card (e.g. Visa, Mastercard, Bank of America, etc.) *See schema for possible list of values*
./PaymentCard/Issuer	0..1	String Enumeration	The entity that issued the credit card *See schema for possible list of values*

6.11 InclusionsType

InclusionsType specifies items or services included in the payment.

Element	Cardinality	Data Type	Description/Contents
HTNG_OPA_CommonTypes			A file that contains the common types used across the Open Payment Alliance messages
InclusionsType		Complex Type	Specifies items or services included in the payment
InclusionsType/Inclusion	1..n	String Enumeration	An item included in the payment Possible values: • Room • Tax • Meals

			• Breakfast • Wifi • FullCredit
InclusionsType/Details	0..1	String	Textual information describing what is covered in the payment

6.12 InvoiceType

InvoiceType provides invoice details for a payment.

Element	Cardinality	Data Type	Description/Contents
HTNG_OPA_CommonTypes			A file that contains the common types used across the Open Payment Alliance messages
InvoiceType		Complex Type	Invoice details for a payment
InvoiceType/Invoice	1..n	Element	An invoice linked to the payment
./Invoice/InvoiceRef	1	String	Booking Ref assigned by the generator of the reservation
./Invoice/InvoiceDate	1	Date	Date of generation of the invoice
./Invoice/InvoiceAmount	0..1	MonetaryAmountType	The amount and currency for the invoice

6.13 MonetaryAmountType

MonetaryAmountType specifies an amount and the currency for the amount.

Element	Cardinality	Data Type	Description/Contents
HTNG_OPA_CommonTypes			A file that contains the common types used across the Open Payment Alliance messages
MonetaryAmountType		ComplexType	Specifies an amount and the currency for the amount
MonetaryAmountType/Amount	1	Money	The amount
MonetaryAmountType/Currency	1	CurrencyCodeType	The currency

6.14 PayeeType

PayeeType specifies the details of the entity receiving the payment.

Element	Cardin-ality	Data Type	Description/Contents
HTNG_OPA_CommonTypes			A file that contains the common types used across the Open Payment Alliance messages
PayeeType		ComplexType	Specifies the details of the entity receiving the payment
PayeeType/PayeeID	1	String	An identifier of the payment recipient entity
PayeeType/PayeeID_Context	1	String	Used to identify the source of the identifier
PayeeType/PayeeName	1	String	Name of the Payee
PayeeType/PayeeAddress	0..1	String	Address of the Payee
PayeeType/PayeeZipCode	0..1	String	Zip code of the Payee
PayeeType/PayeeRegionCode	0..1	String	Region code (state, province, etc.) of the Payee
PayeeType/PayeeCity	0..1		City of the Payee
PayeeType/PayeeCountry	0..1	ISO3166	Country of the Payee (2-character ISO 3166 Country Code)
PayeeType/Email	0..1	String	Email address of the Payee

6.15 PayerType

PayerType provides details related to the entity responsible for the payment.

Element	Cardin-ality	Data Type	Description/Contents
HTNG_OPA_CommonTypes			A file that contains the common types used across the Open Payment Alliance messages
PayerType		ComplexType	Details related to the entity responsible for the payment
PayerType/PayerID	1	String	An identifier of the paying entity
PayerType/PayerID_Context	1	String	Used to identify the source of the identifier
PayerType/PayerName	1	String	Company name of the Payer

6.16 PaymentConditionsType

PaymentConditionsType specifies the conditions for the payment such as if an invoice is required and the types of charges the payment will cover (i.e. Room, Tax, Meals).

Element	Cardinality	Data Type	Description/Contents
HTNG_OPA_CommonTypes			A file that contains the common types used across the Open Payment Alliance messages
PaymentCondtionsType		ComplexType	Specifies the conditions for the payment such as if an invoice is required and the types of charges the payment will cover (i.e. Room, Tax, Meals)
PaymentCondtionsType/AmountReconciliationBy	0..1	String Enumeration	Specifies whose responsibility it is to reconcile the payment amount Possible values: • Payer • PaymentManager
PaymentCondtionsType/AmountReconciliationRules	0..1	String Enumeration	Specifies the rules for payment amount to be accepted (e.g the amount must be less than or equal to the amount field in this message) Possible values: • LET_Amount • Amount
PaymentCondtionsType/InvoiceRequiredInd	0..1	Boolean	When true, the Payment Recipient is required to submit an invoice to receive payment
PaymentCondtionsType/PayeeInitiationRequiredInd	0..1	Boolean	When true, the Payee must send a request to the Payment Manager to initiate the payment
PaymentCondtionsType/PayerInitiationRequiredInd	0..1	Boolean	When true, the Payer must send a request to the Payment Manager to initiate the payment
PaymentCondtionsType/Comments	0..1	String	Comment regarding the payment conditions

6.17 PaymentDetailRecordType

PaymentDetailRecordType specifies the details and the current status related to the payment. This is used in messages to retrieve or push the details of a payment record.

Element	Cardinality	Data Type	Description/Contents

HTNG_OPA_CommonTypes			A file that contains the common types used across the Open Payment Alliance messages
PaymentDetailRecordType		ComplexType	Specifies the details and the current status related to the payment This is used in messages to retrieve or push the details of a payment record
PaymentDetailRecordType/PaymentRef	1	GUID	The unique identifier and index key and/or primary key for the payment details
PaymentDetailRecordType/Requestor	0..1	RequestorType	The party who requested the payment
PaymentDetailRecordType/Payer	1	PayerType	The party responsible for payment of the reservation
PaymentDetailRecordType/Payee	1	PayeeType	The party receiving the payment
PaymentDetailRecordType/Payment	1	PaymentType	Details of the payment
PaymentDetailRecordType/PaymentDocument	1	PaymentDocumentType	References linked to the payment
PaymentDetailRecordType/Inclusions	0..1	InclusionsType	Specifies what is included in the payment
PaymentDetailRecordType/	0..1	PaymentConditionsType	Conditions that must be met prior to payment initiation
PaymentDetailRecordType/	0..1	EarlyCollectionConditionType	A collection of terms that apply to collect a payment prior to the DefaultPaymentDate
PaymentDetailRecordType/	1	PaymentStatusType	Status of the payment

6.18 PaymentDetailType

PaymentDetailType specifies the details related to a payment.

Element	Cardinality	Data Type	Description/Contents
HTNG_OPA_CommonTypes			A file that contains the common types used across the Open Payment Alliance messages
PaymentDetailTypeType		ComplexType	Specifies all of the details related to a payment
PaymentDetailTypeType/PaymentRef	1	GUID	The unique identifier and index key and/or primary key for the payment details

PaymentDetailTypeType/Requestor	1	RequestorType	Information related to the requestor of the payment
PaymentDetailTypeType/Payer	1	PayerType	Information related to the Payer
PaymentDetailTypeType/Payee	1	PayeeType	Information related to the Payee
PaymentDetailTypeType/Payment	1	Element	Information pertaining to the payment details
./Payment/FormOfPayment	1	FormOfPaymentType	Specifies the form of payment
./Payment/PaymentAmount	1	MonetaryAmountType	Specifies the amount of the payment and payment currency
./Payment/Tolerance	0..1	Percentage	Specifies the percentage over the amount that the Payer is willing to pay without further approval
./Payment/DefaultPaymentDate	0..1	DateTime	The date the payment will automatically be initiated by the Payment Manager expressed in ISO 8601 format
./Payment/EarliestPaymentDate	0..1	DateTime	The earliest date that the Payee can initiate the payment expressed in ISO 8601 format.
./Payment/LatestPaymentDate	0..1	DateTime	The latest date that the payment can be initiated expressed in ISO 8601 format
PaymentDetailType/PaymentDocument	1	PaymentDocumentType	References linked to the payment
PaymentDetailType/Inclusions	0..1	InclusionsType	Specifies what is included in the payment
PaymentDetailType/PaymentConditions	0..1	PaymentConditionsType	Conditions that must be met prior to payment initiation
PaymentDetailType/EarlyCollectionConditions	0..1	EarlyCollectionConditionType	A collection of terms that apply to collect a payment prior to the DefaultPaymentDate

6.19 PaymentDocumentType

PaymentDocumentType specifies where the payment is to be applied (i.e. a booking or an invoice).

Element	Cardinality	Data Type	Description/Contents
HTNG_OPA_CommonTypes			A file that contains the common types used across the Open Payment Alliance messages

PaymentDocumentType		Complex Type	Specifies where the payment is to be applied (i.e. a booking or an invoice) *This contains a choice where you send either the Bookings **OR** Invoices structure*
PaymentDocumentType/Bookings	1	BookingType	Information related to the bookings linked to the payment
PaymentDocumentType/Invoices	1	InvoiceType	Information related to the invoices linked to the payment

6.20 PaymentInfoRequestType

PaymentInfoRequestType provides information used to retrieve a payment reference.

Element	Cardinality	Data Type	Description/Contents
HTNG_OPA_CommonTypes			A file that contains the common types used across the Open Payment Alliance messages
PaymentInfoRequestType		ComplexType	Provides information used to retrieve a payment reference
PaymentInfoRequestType/Payer	1	PayerType	The party responsible for payment of the reservation
PaymentInfoRequestType/BookingInfos	1	Element	A collection of booking information for which the payment reference is being requested
./BookingInfos/BookingInfo	0..1	BookingInfoType	Booking information for a single booking whose payment reference is being requested

6.21 PaymentInfoType

PaymentInfoType provides information to link a payment with a booking.

Element	Cardinality	Data Type	Description/Contents
HTNG_OPA_CommonTypes			A file that contains the common types used across the Open Payment Alliance messages
PaymentInfoType		ComplexType	Provides information to link a payment with a booking
PaymentInfoType/PaymentRef	1..n	GUID	The unique identifier and index key and/or primary key for the payment details
PaymentInfoType/BookingRef	1	String	Booking Ref assigned by the generator of the reservation

PaymentInfoType/BookingDate	1	Date	The date the booking was made

6.22 PaymentInitiationType

PaymentInitiationType specifies the information required to initiate a payment.

Element	Cardinality	Data Type	Description/Contents
HTNG_OPA_CommonTypes			A file that contains the common types used across the Open Payment Alliance messages
PaymentInitiationType		ComplexType	Specifies the information required to initiate a payment
PaymentInitiationType/PaymentRef	1	GUID	The unique identifier and index key and/or primary key for the payment to be initiated
PaymentInitiationType/PaymentAmount	1	MonetaryAmountType	The amount of the payment to be initiated
PaymentInitiationType/EarlyCollectionReason	0..1	String Enumeration	Specifies the reason the payment is being initiated (e.g. Guest was a no-show, guest checked out early, etc.) If the payment is being initiated by the Payee, the reason MUST be sent if the date of initiation is prior to the DefaultPaymentDate Possible values: • Cancel • EarlyCheckOut • NoShow • EarlyPayAgreement
PaymentInitiationType/Comment	0..1	String	Used to provide additional information on the reason the payment is being initiated
PaymentInitiationType/InvoiceRef	0..1	String	The invoice reference for the payment that is being initiated
PaymentInitiationType/InvoiceDte	0..1	Date	The invoice date for the payment that is being initiated

6.23 PaymentStatusReferenceType

PaymentStatusReferenceType provides the payment status for a specific payment reference.

Element	Cardinality	Data Type	Description/Contents

HTNG_OPA_CommonTypes			A file that contains the common types used across the Open Payment Alliance messages
PaymentStatusReferenceType		ComplexType	Provides the payment status for a specific payment reference
PaymentIStatusReferenceType/ PaymentRef	1	GUID	The unique identifier and index key and/or primary key for the payment
PaymentIStatusReferenceType/ PaymentStatus	1	String	Provides the status of the payment

6.24 PaymentStatusType

PaymentStatusType specifies the status of the payment.

Element	Cardin-ality	Data Type	Description/Contents
HTNG_OPA_CommonTypes			A file that contains the common types used across the Open Payment Alliance messages
PaymentStatusType		ComplexType	Specifies the status of the payment
PaymentStatusType/Status	1	String Enumeration	The current state of the payment Possible values: • Active • Canceled • Error • Expired • Inactive • Pending • Settled • Submitted
PaymentIStatusType/StatusDescription	0..1	String	Description related to the payment status
PaymentIStatusType/SubStatus	0..1	String Enumeration	A sub-status related to the status (e.g. A Pending Status may have a sub-status of Invoice indicating that the payment is contingent on an invoice being received) Possible values: • Invoice • PaymentInitiation
PaymentStatusType/SubStatusDescription	0..1	String	Description related to the sub-status

6.25 PaymentType

PaymentType specifies the details of the payment.

Element	Cardi-nality	Data Type	Description/Contents
HTNG_OPA_CommonTypes			A file that contains the common types used across the Open Payment Alliance messages
PaymentType		ComplexType	Specifies details of the payment
PaymentIType/FormOfPayment	1	GUID	The unique identifier and index key and/or primary key for the payment
PaymentIStatusReferenceType/ PaymentStatus	1	String	Provides the status of the payment

6.26 RequestorType

RequestorType specifies information related to the requestor of the payment.

Element	Cardi-nality	Data Type	Description/Contents
HTNG_OPA_CommonTypes			A file that contains the common types used across the Open Payment Alliance messages
RequestorType		ComplexType	Specifies information related to the requestor of the payment
RequestorType/RequestorID	1	String	Information related to the requestor of the payment
RequestorType/RequestorName	1	String	Company name of the entity that creates the payment instruction

7 Payment Status

When a payment is created in the Payment Manager, a status is assigned to it. This section describes each of the payment statuses and how a payment may transition from one status to another.

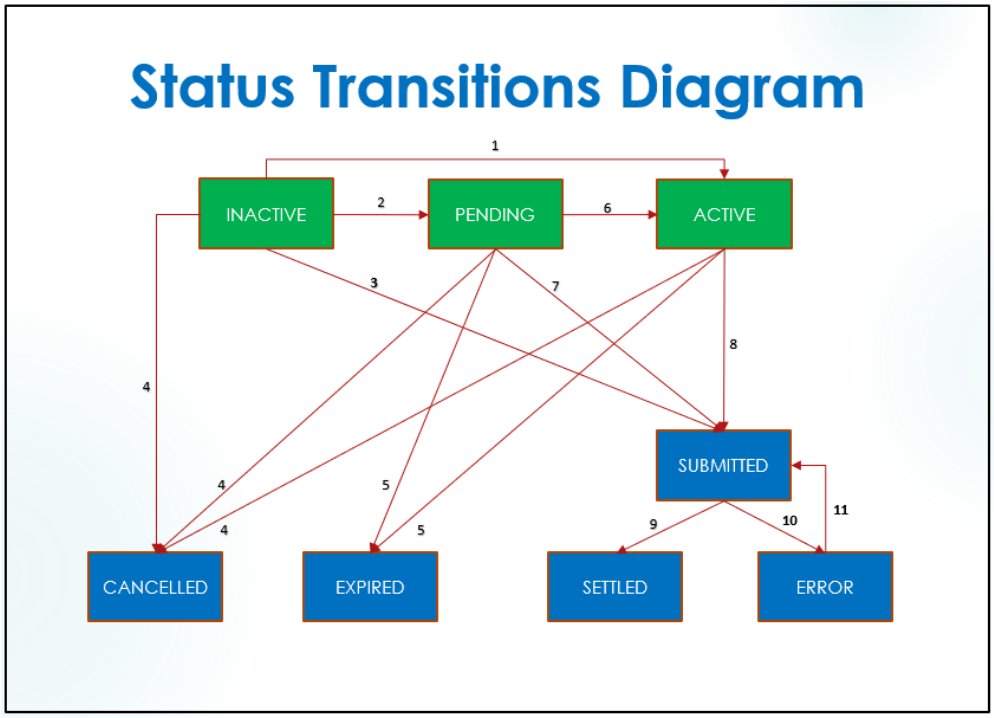
7.1 Payment Status Definitions

The following terms specify the possible payment statuses and their definitions.

Status	Definition
Inactive	The payment date has not been reached yet.
Active	The payment is ready to be initiated by the Payee.
Pending	A payment rule is not met, thus preventing initiation. A sub-status is used to specify what is blocking the initiation (e.g. pending invoice, awaiting initiation by the Payee).
Cancelled	The payment has been cancelled.
Expired	The payment has not yet been initiated and has reached its expiry date.
Submitted	The payment has been initiated through the Payment Manager.
Settled	The payment has been settled (i.e. the settlement process is complete). Note: This does not mean that the funds have already been credited into the Payee's account.
Error	The settlement process failed.

7.2 Payment Status Transitions and Triggers

The following diagram shows all of the payment statuses and their transitions from one to another.



When a payment is created, it can be assigned one of the following initial statuses:

- **Inactive**: The payment date is specified but not reached yet
- **Pending**: The payment date is specified and it is reached, but some payment rules are not met so the payment is not yet available for initiation.
- **Active**: The payment date is not specified

The following table specifies the requirements to transition from one payment status to another and who can trigger each transition.

Status Transition*	Initiator	Trigger
#1 Inactive to Active	Payment Manager	• Payment date reached AND • Payment NOT to be initiated by the PM, AND • Payment rules associated to the payment (e.g. invoice issuance), if any, met
#1 Inactive to Active	Payer	• Payment date NOT reached AND • Payment NOT to be initiated by the PM AND

		• Payment initiation instruction sent by the Payer
#1 Inactive to Active	Payee	• Payment date NOT reached AND • Payment NOT to be initiated by the PM AND • Payment initiation instruction sent by the Payee
#2 Inactive to Pending	Payment Manager	• Payment date reached, AND • Payment rules associated to the payment (e.g. invoice required) NOT met
#3 Inactive to Submitted	Payment Manager	• Payment date reached, AND • Payment to be initiated by the PM, AND • Payment rules associated to the payment (e.g. invoice required), if any, met
#3 Inactive to Submitted	Payer	• Payment date NOT reached, AND • Payment to be initiated by the PM, AND • Payment initiation instruction sent by the Payer
#3 Inactive to Submitted	Payee	• Payment date NOT reached, AND • Payment to be initiated by the PM, AND • Payment initiation instruction sent by the Payee
#3 Inactive to Submitted	Payee	• Payment date NOT reached, AND • Payment initiation instruction sent by the Payee
#4 Any Status Prior to Payment Initiation to Cancelled	Payer	• Payment cancellation instruction sent by the Payer
#5 Any Status Prior to Payment Initiation to Expired	Payment Manager	• Maximum date for payment initiation reached
#6 Pending to Active	Payment Manager	• Payment rules for payment initiation (e.g. invoice required) met, AND • Payment NOT to be initiated by the PM
#6		• Payment rules for payment initiation (e.g. invoice required) NOT met, AND

Pending to Active	Payer	• Payment NOT to be initiated by the PM, AND • Payment initiation instruction sent by the Payer
#6 Pending to Active	Payee	• Payment rules for payment initiation (e.g. invoice required) NOT met, AND • Payment NOT to be initiated by the PM, AND • Payment initiation instruction sent by the Payee
#7 Pending to Submitted	Payment Manager	• Payment rules for payment initiation (e.g. invoice required) met, AND • Payment to be initiated by the PM
#7 Pending to Submitted	Payer	• Payment rules for payment initiation (e.g. invoice required) NOT met, AND • Payment to be initiated by the PM, AND • Payment initiation instruction sent by the Payer
#7 Pending to Submitted	Payee	• Payment rules for payment initiation (e.g. invoice required) NOT met, AND • Payment to be initiated by the PM, AND • Payment initiation instruction sent by the Payee
#8 Active to Submitted	Payee	• Payment initiation instruction sent by the Payee
#9 Submitted to Settled	Payment Processor	• Settlement confirmation sent from the Payment Processor back to the PM
#10 Submitted to Error	Payment Processor	• Settlement error sent from the Payment Processor back to the PM
#11 Error to Submitted	Payee	• Payment initiation instruction sent by the Payee

Numbers correspond to the Status Transition Diagram

7.3 Payment Initiation Requirements

Payment may be initiated by the Payment Manager, the Payer or the Payee depending on how the payment instructions are set. The payment conditions that need to be met to initiate the payment are as follows:

Initiation by Payment Manager

- The payment date is specified
- The Payee's Payment Processor is integrated with the Payment Manager
- The Payee delegates the automated collection process to the Payment Manager

Initiation by the Payer

- The Payer implements the PM's payment initiation API message (*HTNG_InitiatePaymentRQ*)
- The payment status is Inactive or Pending

Initiation by the Payee

- The Payee's Payment Processor is integrated with the Payment Manager
- The Payee implements the Payment Manager's payment initiation API message (*HTNG_InitiatePaymentRQ*)
- The payment status is Inactive or Pending

8 Error Handling

For many years, software designers have recognized the value of separating error processing from the normal processing of a message. For example, the HTTP protocol uses a series of statuses to determine if a response is successful or erroneous. The SOAP protocols also use the SOAP Fault in response when a SOAP message can't be processed correctly. This specification follows the conventions that have been established for SOAP by the W3C and RESTful HTTP messages as established by the IETF. To support commonality, we define fault messages that are consistent between SOAP 1.2 and RESTful XML and JSON messages.

8.1 Recommended Approach

It is recommended to follow the best practice of using SOAP faults when using SOAP and HTTP status codes when using REST. For RESTful APIs along with the HTTP Status we adopt the convention of using a SOAP fault element in the body of the error response when using XML and the JavaScript/JSON equivalent when using JSON.

8.2 Errors and HTTP Statuses

REST services are typically carried over HTTP and use HTTP status codes for error handling. The five categories of HTTP status codes defined by the IETF are:

100 series: Continue

100 series codes give information to the client about how to continue sending the request.

200 series: Successful

200 series codes are used when the clients request is successful.

300 series: Redirection

300 series codes indicate that the client needs to take further action, typically sending the request to a new URL.

400 series: Client Error

There is a problem with the data sent by the client.

500 series: Internal Server Error

The server encountered an unexpected condition which prevented it from fulfilling the request.

The detailed information about individual statuses in each series is detailed in section 6 of the IETF RFC-7231 for HTTP 1.1.

A compact listing of the status codes can be found at:
https://en.wikipedia.org/wiki/List_of_HTTP_status_codes

HTTP statuses are extensible in that if there is a need to add a status, the server is free to return any error number in the appropriate series that is not already assigned. If a client does not recognize the error, then it is to be treated as the base, i.e. 400 or 500 status. Additional information about the specifics of the error may be sent in header fields or the request body.

Common restful practice is to use HTTP status codes to indicate errors and add details in the body with the media types: application/problem+json or application/problem+xml for JSON and XML, respectively. These media types are defined in RFC-7807.

8.3 SOAP Faults

SOAP, the Simple Object Access Protocol is a set of W3C (World Wide Web Consortium) standards for Web Services that have been in use for over 20 years. SOAP messages are not constrained to HTTP but are commonly used with the HTTP protocols. The fault definition changed between SOAP 1.1 and 1.2. The specification uses the newer SOAP 1.2 Fault definition. The SOAP 1.2 Fault specification can be found in section 5.4 of the SOAP 1.2 specification at: https://www.w3.org/TR/soap12-part1/#soapfault

SOAP 1.2 faults include the following elements:

Code

A required Node element contains a Value element which holds one of the enumerated values shown below:

- VersionMismatch – invalid element inside the SOAP envelope
- MustUnderstand – an immediate child element of the header was not understood
- DataEncodingUnknown – an element uses an unsupported data encoding
- Sender – an error occurred on the client side (typically a problem with the message)
- Receiver – an error on the server side with the processing of the message

Optional Subcode elements each with their own Value element are used to further distinguish the fault.

Reason

The required Reason element contains a Text element that describes the human readable reason for the fault.

Node

The optional Node element contains a URL (AnyURL schema type) for the node that generated the fault.

Role

The optional Role element contains the URI describing the Role the Node operating in.

Detail

The optional Detail element is used to carry application specific information and contains elements providing information regarding the context of the fault.

Other Fault sub-elements MAY be present, provided they are namespace qualified. Below is an example SOAP 1.2 Fault response:

SOAP 1.2 Fault Example

HTTP/1.1 404 Not Found
Date: Wed, 18 Nov 2020 22:30:37 GMT
Server: Apache/2.4.25 (Debian)
Content-Length: 528
Content-Type: application/xml+soap; charset=utf-8

```
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
  <env:Header/>
  <env:Body>
    <env:Fault>
      <env:Code>
        <env:Value>env:Sender</env:Value>
      </env:Code>
      <env:Reason>
        <env:Text xml:lang="en-US">PaymentRef not found</env:Text>
      </env:Reason>
      <env:Detail>
        <opa:PaymentRef>982c916c-45f3-41b9-b90e-69c19e4d6ecc</opa:PaymentRef>
      </env:Detail>
    </env:Fault>
  </env:Body>
</env:Envelope>
```

8.4 RESTful XML

RESTful XML uses XML for the representation of data exchanged using a RESTful interface, typically built using the HTTP protocol. While a body is not required by the response, this specification recommends the use of elements that match those of the Fault element in the SOAP 1.2 response as shown in the example below:

RESTful (Simple) XML Example

```
HTTP/1.1 404 Not Found
Date: Wed, 18 Nov 2020 22:30:37 GMT
Server: Apache/2.4.25 (Debian)
Content-Length: 224
Content-Type: application/; charset=utf-8

<Fault>
   <Code>
      <Value>Sender</Value>
   </Code>
   <Reason>
      <Text>PaymentRef not found</Text>
   </Reason>
   <Detail>
      <Text>982c916c-45f3-41b9-b90e-69c19e4d6ecc</Text>
   </Detail>
</Fault>
```

Note: The content types are specified in RFC-7807

8.5 RESTful JSON

RESTful JSON uses JSON for the representation of data exchanged using a RESTful interface, typically built using the HTTP protocol. While a body is not required by the response, this specification recommends the use of elements that match those of the Fault element in the SOAP 1.2 response as shown in the example below:

RESTful JSON Fault Example:

HTTP/1.1 404 Not Found
Date: Wed, 18 Nov 2020 22:30:37 GMT
Server: Apache/2.4.25 (Debian)
Content-Length: 178
Content-Type: application/problem+json; charset=utf-8


```
{
  "fault": {
    "code": {
      "value": "Sender"
    }
  },
  "reason": "PaymentRef not found",
  "detail": {
    "paymentRef": "982c916c-45f3-41b9-b90e-69c19e4d6ecc"
  }
}
```


8.6 Recommended Error Codes

The following subsections provide some recommendations for common errors that may be encountered.

8.6.1 Client Errors – 400 Series Errors

Spec	Number	Name	Reason	Description
HTTP	400	Bad Request	Malformed Message: The server cannot or will not process the request due to an apparent client error.	This code is used for errors such as syntax errors or formatting errors in the message or header, or if the message size is too large.
HTTP	404	Not Found	The requested resource could not be found but may be available in the future.	This error may be returned if the data to be update or retrieved is not found. Subsequent requests by the client are permissible.
HTTP	405	Method Not Allowed	The request method is not supported for the requested resource.	This error may be returned if a method such as a “Get” is not a supported method for the resource.

HTTP	406	Not Acceptable	The requested resource is only capable of generating content that is not acceptable according to the Accept headers sent in the request.	This error may be used if the request would return data that does not meet the requirements in the header "Accept" field.
HTTP	415	Unsupported Media Type	The expecting message was in one language but the received message is in another.	This is used in the instance where the wrong media type is received; you may accept Application/JSON but are sent Application/XML.
HTTP	422	Unprocessable Entity	The request was well formed but was unable to be processed due to semantic errors.	Data within the message does not validate against the schema.

8.6.2 Server Errors – 500 Series Errors

Spec	Number	Name	Reason	Description
HTTP	500	Internal Error	A generic error message given when an unexpected condition was encountered, and no more specific message is suitable.	This error is used to return a server error if no additional detail is being returned.
HTTP	501	Not Implemented	The server either does not recognize the request method, or it lacks the ability to fulfill the request.	This error is used when the request method cannot be fulfilled, but usually implies future availability.
HTTP	503	Service Unavailable	The server cannot handle the request.	This error is used for instances such as when the server is overloaded or is down for maintenance.
HTTP	506	Variant Also Negotiates	Transparent content negotiation for the request results in a circular reference	This error is used if there is a circular reference encountered.
HTTP	507	Insufficient Storage	The server is unable to store the representation needed to complete the request.	This error is used if there is insufficient storage to process the request.

9 Example Messages

These example XML and JSON messages do not include all the available data fields in the messages but are intended to represent how each message may be used in a real-life scenario.

9.1 Create Payment

This example illustrates a $600 payment with a VCC to be created by the Payment Manager, for a three-day stay booking that includes accommodation and meals. The payment also defines the amount and date of collection in case of a no-show.

9.1.1 HTNG_CreatePaymentRQ XML Example

```
<?xml version="1.0" encoding="UTF-8"?>
<HTNG_CreatePaymentRQ xsi:schemaLocation="http://htng.org/2019A HTNG_CreatePaymentRQ.xsd"
xmlns="http://htng.org/2019A" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <PaymentDetail>
                <PaymentRef>00946db5-cff5-428f-88df-97c947ef568d</PaymentRef>
                <Requestor>
                        <RequestorID>12689</RequestorID>
                        <RequestorName>Travel Company 1</RequestorName>
                </Requestor>
                <Payer>
                        <PayerID>12689</PayerID>
                        <PayerID_Context>PAYER</PayerID_Context>
                        <PayerName>Travel Company 1</PayerName>
                </Payer>
                <Payee>
                        <PayeeID>SUPPLIERID_1</PayeeID>
                        <PayeeID_Context>PAYER</PayeeID_Context>
                        <PayeeName>Hotel 1</PayeeName>
                        <PayeeAddress>100 Main Street</PayeeAddress>
                        <PayeeZipCode>90002</PayeeZipCode>
                        <PayeeRegionCode>California</PayeeRegionCode>
                        <PayeeCity>Los Angeles</PayeeCity>
                        <PayeeCountry>US</PayeeCountry>
                        <PayeeEmail>payments@hotel1.com</PayeeEmail>
                </Payee>
                <Payment>
                        <FormOfPayment>
                                <PaymentCard>
                                        <VirtualInd>true</VirtualInd>
                                        <CardRequestDetails>
                                                <CardType>AmericanExpress</CardType>
                                                <Issuer>AmericanExpress</Issuer>
                                                <CardAmount>
                                                        <Amount>60000</Amount>
```

```
                                        <Currency>
                                            <CurrencyCode>USD</CurrencyCode>
                                            <DecimalPlaces>2</DecimalPlaces>
                                        </Currency>
                                    </CardAmount>
                                    <SchemeProgram/>
                                    <ValidFrom>2020-12-30T09:30:47Z</ValidFrom>
                                    <ValidThrough>2021-01-07T09:30:47Z</ValidThrough>
                                </CardRequestDetails>
                            </PaymentCard>
                        </FormOfPayment>
                        <PaymentAmount>
                            <Amount>60000</Amount>
                            <Currency>
                                <CurrencyCode>USD</CurrencyCode>
                                <DecimalPlaces>2</DecimalPlaces>
                            </Currency>
                        </PaymentAmount>
                        <DefaultPaymentDate>2021-01-03T09:30:47Z</DefaultPaymentDate>
                        <LatestPaymentDate>2021-01-07T09:30:47Z</LatestPaymentDate>
                    </Payment>
                    <PaymentDocument>
                        <Bookings>
                            <Booking>
                                <BookingRefs>
                                    <BookingRef>BK-324798</BookingRef>
                                    <BookingRefType>BookingSource</BookingRefType>
                                </BookingRefs>
                                <BookingDate>2020-10-01</BookingDate>
                                <ServiceStartDate>2020-12-31</ServiceStartDate>
                                <ServiceEndDate>2021-01-03</ServiceEndDate>
                                <BookingAmount>
                                    <Amount>60000</Amount>
                                    <Currency>
                                        <CurrencyCode>USD</CurrencyCode>
                                        <DecimalPlaces>2</DecimalPlaces>
                                    </Currency>
                                </BookingAmount>
                            </Booking>
                        </Bookings>
                    </PaymentDocument>
                    <Inclusions>
                        <Inclusion>Room</Inclusion>
                        <Inclusion>Breakfast</Inclusion>
```

```
            </Inclusions>
            <EarlyCollectionConditions>
                <EarlyCollectionCondition>
                    <EarlyCollectionReason>NoShow</EarlyCollectionReason>
                    <EarlyCollectionDetail>
                        <DateFrom>2020-12-31T09:30:47Z</DateFrom>
                        <DateThrough>2021-01-07T09:30:47Z</DateThrough>
                        <PaymentAmount>
                            <Amount>20000</Amount>
                            <Currency>
                                <CurrencyCode>USD</CurrencyCode>
                                <DecimalPlaces>2</DecimalPlaces>
                            </Currency>
                        </PaymentAmount>
                    </EarlyCollectionDetail>
                </EarlyCollectionCondition>
            </EarlyCollectionConditions>
        </PaymentDetail>
</HTNG_CreatePaymentRQ>
```

9.1.2 HTNG_CreatePaymentRQ JSON Example

```
{
  "PaymentDetail": {
    "PaymentRef": "00946db5-cff5-428f-88df-97c947ef568d",
    "Requestor": {
      "RequestorID": "12689",
      "RequestorName": "Travel Company 1"
    },
    "Payer": {
      "PayerID": "12689",
      "PayerID_Context": "PAYER",
      "PayerName": "Travel Company 1"
    },
    "Payee": {
      "PayeeID": "SUPPLIERID_1",
      "PayeeID_Context": "PAYER",
      "PayeeName": "Hotel 1",
      "PayeeAddress": "100 Main Street",
      "PayeeZipCode": "90002",
      "PayeeRegionCode": "California",
      "PayeeCity": "Los Angeles",
      "PayeeCountry": "US",
      "PayeeEmail": "payments@hotel1.com"
    },
```

```
"Payment": {
  "FormOfPayment": {
    "PaymentCard": {
      "VirtualInd": true,
      "CardRequestDetails": {
        "CardType": "AmericanExpress",
        "Issuer": "AmericanExpress",
        "CardAmount": {
          "Amount": 60000,
          "Currency": {
            "CurrencyCode": "USD",
            "DecimalPlaces": "2"
          }
        },
        "SchemeProgram": "",
        "ValidFrom": "2020-12-30T09:30:47Z",
        "ValidThrough": "2021-01-07T09:30:47Z"
      }
    }
  },
  "PaymentAmount": {
    "Amount": 60000,
    "Currency": {
      "CurrencyCode": "USD",
      "DecimalPlaces": "2"
    }
  },
  "DefaultPaymentDate": "2021-01-03T09:30:47Z",
  "LatestPaymentDate": "2021-01-07T09:30:47Z"
},
"PaymentDocument": {
  "Bookings": [
    {
      "Booking": [
        {
          "BookingRefs": [
            {
              "BookingRef": "BK-324798",
              "BookingType": "BookingSource"
            }
          ],
          "BookingDate": "2020-10-01",
          "ServiceStartDate": "2020-12-31",
          "ServiceEndDate": "2021-01-03",
```

```
          "BookingAmount": {
            "Amount": 60000,
            "Currency": {
              "CurrencyCode": "USD",
              "DecimalPlaces": "2"
            }
          }
        }
      ]
    }
  ]
 },
 "Inclusions": {
  "Inclusion": [
   "Room",
   "Breakfast"
  ]
 },
 "EarlyCollectionConditions": [
  {
   "EarlyCollectionReason": "NoShow",
   "EarlyCollectionDetail": [
    {
     "DateFrom": "2020-12-31T09:30:47Z",
     "DateThrough": "2021-01-07T09:30:47Z",
     "PaymentAmount": {
      "Amount": 20000,
      "Currency": {
       "CurrencyCode": "USD",
       "DecimalPlaces": "2"
      }
     }
    }
   ]
  }
 ]
}
```


9.1.3 HTNG_CreatePaymentRS XML Example


```
<?xml version="1.0" encoding="UTF-8"?>
<HTNG_CreatePaymentRS xsi:schemaLocation="http://htng.org/2019A HTNG_CreatePaymentRS.xsd"
xmlns="http://htng.org/2019A" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <PaymentRef>00946db5-cff5-428f-88df-97c947ef568d</PaymentRef>
```

```
</HTNG_CreatePaymentRS>
```

9.1.4 HTNG_CreatePaymentRS JSON Example

```
{
  "PaymentRef": "00946db5-cff5-428f-88df-97c947ef568d"
}
```

9.2 Modify Payment

This example illustrates a payment modification resulting from an extension of the stay from three days to four days. As a consequence the payment information, such as the payment amount or the VCC activation window, needs to be updated accordingly.

9.2.1 HTNG_ModifyPaymentRQ XML Example

```
<?xml version="1.0" encoding="UTF-8"?>
<HTNG_ModifyPaymentRQ xsi:schemaLocation="http://htng.org/2019A HTNG_ModifyPaymentRQ.xsd"
xmlns="http://htng.org/2019A" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <PaymentDetail>
                <PaymentRef>00946db5-cff5-428f-88df-97c947ef568d</PaymentRef>
                <Requestor>
                        <RequestorID>12689</RequestorID>
                        <RequestorName>Travel Company 1</RequestorName>
                </Requestor>
                <Payer>
                        <PayerID>12689</PayerID>
                        <PayerID_Context>PAYER</PayerID_Context>
                        <PayerName>Travel Company 1</PayerName>
                </Payer>
                <Payee>
                        <PayeeID>SUPPLIERID_1</PayeeID>
                        <PayeeID_Context>PAYER</PayeeID_Context>
                        <PayeeName>Hotel 1</PayeeName>
                        <PayeeAddress>100 Main Street</PayeeAddress>
                        <PayeeZipCode>90002</PayeeZipCode>
                        <PayeeRegionCode>California</PayeeRegionCode>
                        <PayeeCity>Los Angeles</PayeeCity>
                        <PayeeCountry>US</PayeeCountry>
                        <PayeeEmail>payments@hotel1.com</PayeeEmail>
                </Payee>
                <Payment>
                        <FormOfPayment>
                                <PaymentCard>
                                        <VirtualInd>true</VirtualInd>
                                        <CardRequestDetails>
                                                <CardType>AmericanExpress</CardType>
```

```
                                    <Issuer>AmericanExpress</Issuer>
                                    <CardAmount>
                                        <Amount>80000</Amount>
                                        <Currency>
                                            <CurrencyCode>USD</CurrencyCode>
                                            <DecimalPlaces>2</DecimalPlaces>
                                        </Currency>
                                    </CardAmount>
                                    <SchemeProgram/>
                                    <ValidFrom>2020-12-30T09:30:47Z</ValidFrom>
                                    <ValidThrough>2021-01-08T09:30:47Z</ValidThrough>
                                </CardRequestDetails>
                            </PaymentCard>
                        </FormOfPayment>
                        <PaymentAmount>
                            <Amount>80000</Amount>
                            <Currency>
                                <CurrencyCode>USD</CurrencyCode>
                                <DecimalPlaces>2</DecimalPlaces>
                            </Currency>
                        </PaymentAmount>
                        <DefaultPaymentDate>2021-01-04T09:30:47Z</DefaultPaymentDate>
                        <LatestPaymentDate>2021-01-08T09:30:47Z</LatestPaymentDate>
                    </Payment>
                    <PaymentDocument>
                        <Bookings>
                            <Booking>
                                <BookingRefs>
                                    <BookingRef>BK-324798</BookingRef>
                                    <BookingRefType>BookingSource</BookingRefType>
                                </BookingRefs>
                                <BookingDate>2020-10-01</BookingDate>
                                <ServiceStartDate>2020-12-31</ServiceStartDate>
                                <ServiceEndDate>2021-01-04</ServiceEndDate>
                                <BookingAmount>
                                    <Amount>80000</Amount>
                                    <Currency>
                                        <CurrencyCode>USD</CurrencyCode>
                                        <DecimalPlaces>2</DecimalPlaces>
                                    </Currency>
                                </BookingAmount>
                            </Booking>
                        </Bookings>
                    </PaymentDocument>
```

```
            <Inclusions>
                <Inclusion>Room</Inclusion>
                <Inclusion>Breakfast</Inclusion>
            </Inclusions>
            <EarlyCollectionConditions>
                <EarlyCollectionCondition>
                    <EarlyCollectionReason>NoShow</EarlyCollectionReason>
                    <EarlyCollectionDetail>
                        <DateFrom>2020-12-31T09:30:47Z</DateFrom>
                        <DateThrough>2021-01-07T09:30:47Z</DateThrough>
                        <PaymentAmount>
                            <Amount>20000</Amount>
                            <Currency>
                                <CurrencyCode>USD</CurrencyCode>
                                <DecimalPlaces>2</DecimalPlaces>
                            </Currency>
                        </PaymentAmount>
                    </EarlyCollectionDetail>
                </EarlyCollectionCondition>
            </EarlyCollectionConditions>
        </PaymentDetail>
</HTNG_ModifyPaymentRQ>
```

9.2.2 HTNG_ModifyPaymentRQ JSON Example

```
{
  "PaymentDetail": {
    "PaymentRef": "00946db5-cff5-428f-88df-97c947ef568d",
    "Requestor": {
      "RequestorID": "12689",
      "RequestorName": "Travel Company 1"
    },
    "Payer": {
      "PayerID": "12689",
      "PayerID_Context": "PAYER",
      "PayerName": "Travel Company 1"
    },
    "Payee": {
      "PayeeID": "SUPPLIERID_1",
      "PayeeID_Context": "PAYER",
      "PayeeName": "Hotel 1",
      "PayeeAddress": "100 Main Street",
      "PayeeZipCode": "90002",
      "PayeeRegionCode": "California",
      "PayeeCity": "Los Angeles",
```

```
    "PayeeCountry": "US",
    "PayeeEmail": "payments@hotel1.com"
  },
  "Payment": {
    "FormOfPayment": {
      "PaymentCard": {
        "VirtualInd": true,
        "CardRequestDetails": {
          "CardType": "AmericanExpress",
          "Issuer": "AmericanExpress",
          "CardAmount": {
            "Amount": 80000,
            "Currency": {
              "CurrencyCode": "USD",
              "DecimalPlaces": "2"
            }
          },
          "SchemeProgram": "",
          "ValidFrom": "2020-12-30T09:30:47Z",
          "ValidThrough": "2021-01-08T09:30:47Z"
        }
      }
    },
    "PaymentAmount": {
      "Amount": 80000,
      "Currency": {
        "CurrencyCode": "USD",
        "DecimalPlaces": "2"
      }
    },
    "DefaultPaymentDate": "2021-01-04T09:30:47Z",
    "LatestPaymentDate": "2021-01-08T09:30:47Z"
  },
  "PaymentDocument": {
    "Bookings": [
      {
        "Booking": [
          {
            "BookingRefs": [
              {
                "BookingRef": "BK-324798",
                "BookingType": "BookingSource"
              }
            ],
```

```
          "BookingDate": "2020-10-01",
          "ServiceStartDate": "2020-12-31",
          "ServiceEndDate": "2021-01-04",
          "BookingAmount": {
            "Amount": 80000,
            "Currency": {
              "CurrencyCode": "USD",
              "DecimalPlaces": "2"
            }
          }
        }
      ]
    }
  ]
 },
 "Inclusions": {
  "Inclusion": [
    "Room",
    "Breakfast"
  ]
 },
 "EarlyCollectionConditions": [
  {
    "EarlyCollectionReason": "NoShow",
    "EarlyCollectionDetail": [
      {
        "DateFrom": "2020-12-31T09:30:47Z",
        "DateThrough": "2021-01-07T09:30:47Z",
        "PaymentAmount": {
          "Amount": 20000,
          "Currency": {
            "CurrencyCode": "USD",
            "DecimalPlaces": "2"
          }
        }
      }
    ]
  }
 ]
 }
}
```


9.2.3 HTNG_ModifyPaymentRS XML Example


```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<HTNG_ModifyPaymentRS xsi:schemaLocation="http://htng.org/2019A HTNG_ModifyPaymentRS.xsd"
xmlns="http://htng.org/2019A" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <PaymentRef>00946db5-cff5-428f-88df-97c947ef568d</PaymentRef>
</HTNG_ModifyPaymentRS>
```

9.2.4 HTNG_ModifyPaymentRS JSON Example

```
{
  "PaymentRef": "00946db5-cff5-428f-88df-97c947ef568d"
}
```

9.3 Cancel Payment

This example illustrates the cancellation of a payment, usually as the result of a cancellation of the related booking.

9.3.1 HTNG_CancelPaymentRQ XML Example

```
<?xml version="1.0" encoding="UTF-8"?>
<HTNG_CancelPaymentRQ xsi:schemaLocation="http://htng.org/2019A HTNG_CancelPaymentRQ.xsd"
xmlns="http://htng.org/2019A" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <PaymentRef>00946db5-cff5-428f-88df-97c947ef568d</PaymentRef>
</HTNG_CancelPaymentRQ>
```

9.3.2 HTNG_CancelPaymentRQ JSON Example

```
{
  "PaymentRef": "00946db5-cff5-428f-88df-97c947ef568d"
}
```

9.3.3 HTNG_CancelPaymentRS XML Example

```
<?xml version="1.0" encoding="UTF-8"?>
<HTNG_CancelPaymentRS xsi:schemaLocation="http://htng.org/2019A HTNG_CancelPaymentRS.xsd"
xmlns="http://htng.org/2019A" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <PaymentRef>00946db5-cff5-428f-88df-97c947ef568d</PaymentRef>
</HTNG_CancelPaymentRS>
```

9.3.4 HTNG_CancelPaymentRS JSON Example

```
{
  "PaymentRef": "00946db5-cff5-428f-88df-97c947ef568d"
}
```

9.4 Get Payment Details

This example illustrates the query made by the Payee to the Payment Manager to retrieve detailed information of a given payment.

9.4.1 HTNG_GetPaymentDetailsRQ XML Example

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<HTNG_GetPaymentDetailsRQ xsi:schemaLocation="http://htng.org/2019A
HTNG_GetPaymentDetailsRQ.xsd" xmlns="http://htng.org/2019A"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <PaymentRefs>
                <PaymentRef>00946db5-cff5-428f-88df-97c947ef568d</PaymentRef>
        </PaymentRefs>
</HTNG_GetPaymentDetailsRQ>
```

9.4.2 HTNG_GetPaymentDetailsRQ JSON Example

```
{
  "PaymentRefs": [
    "00946db5-cff5-428f-88df-97c947ef568d"
  ]
}
```

9.4.3 HTNG_GetPaymentDetailsRS XML Example

```
<?xml version="1.0" encoding="UTF-8"?>
<HTNG_GetPaymentDetailsRS xsi:schemaLocation="http://htng.org/2019A
HTNG_GetPaymentDetailsRS.xsd" xmlns="http://htng.org/2019A"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <PaymentDetails>
                <PaymentDetail>
                        <PaymentRef>00946db5-cff5-428f-88df-97c947ef568d</PaymentRef>
                        <Payer>
                                <PayerID>CLIENTID_1</PayerID>
                                <PayerID_Context>PAYEE</PayerID_Context>
                                <PayerName>Travel Company 1</PayerName>
                        </Payer>
                        <Payee>
                                <PayeeID>H001</PayeeID>
                                <PayeeID_Context>PAYEE</PayeeID_Context>
                                <PayeeName>Hotel 1</PayeeName>
                                <PayeeAddress>100 Main Street</PayeeAddress>
                                <PayeeZipCode>90002</PayeeZipCode>
                                <PayeeRegionCode>California</PayeeRegionCode>
                                <PayeeCity>Los Angeles</PayeeCity>
                                <PayeeCountry>US</PayeeCountry>
                                <PayeeEmail>payments@hotel1.com</PayeeEmail>
                        </Payee>
                        <Payment>
                                <FormOfPayment>
                                        <PaymentCard>
                                                <VirtualInd>true</VirtualInd>
                                                <CardType>AmericanExpress</CardType>
                                                <Issuer>AmericanExpress</Issuer>
```

```
                    </PaymentCard>
                </FormOfPayment>
                <PaymentAmount>
                    <Amount>80000</Amount>
                    <Currency>
                        <CurrencyCode>USD</CurrencyCode>
                        <DecimalPlaces>2</DecimalPlaces>
                    </Currency>
                </PaymentAmount>
                <DefaultPaymentDate>2021-01-03T09:30:47Z</DefaultPaymentDate>
                <LatestPaymentDate>2021-01-07T09:30:47Z</LatestPaymentDate>
            </Payment>
            <PaymentDocument>
                <Bookings>
                    <Booking>
                        <BookingRefs>
                            <BookingRef>BK-324798</BookingRef>

<BookingRefType>BookingSource</BookingRefType>
                        </BookingRefs>
                        <BookingDate>2020-10-01</BookingDate>
                    <ServiceStartDate>2020-12-31</ServiceStartDate>
                    <ServiceEndDate>2021-01-04</ServiceEndDate>
                        <BookingAmount>
                            <Amount>80000</Amount>
                            <Currency>
                                <CurrencyCode>USD</CurrencyCode>
                                <DecimalPlaces>2</DecimalPlaces>
                            </Currency>
                        </BookingAmount>
                    </Booking>
                </Bookings>
            </PaymentDocument>
            <Inclusions>
                <Inclusion>Room</Inclusion>
                <Inclusion>Breakfast</Inclusion>
            </Inclusions>
            <EarlyCollectionConditions>
                <EarlyCollectionCondition>
                    <EarlyCollectionReason>NoShow</EarlyCollectionReason>
                    <EarlyCollectionDetail>
                        <DateFrom>2020-12-31T09:30:47Z</DateFrom>
                        <DateThrough>2021-01-07T09:30:47Z</DateThrough>
                        <PaymentAmount>
```

```
                                        <Amount>20000</Amount>
                                        <Currency>
                                                <CurrencyCode>USD</CurrencyCode>
                                                <DecimalPlaces>2</DecimalPlaces>
                                        </Currency>
                                </PaymentAmount>
                        </EarlyCollectionDetail>
                </EarlyCollectionCondition>
        </EarlyCollectionConditions>
        <PaymentStatus>
                <Status>Inactive</Status>
                <StatusDescription/>
        </PaymentStatus>
</PaymentDetail>
</PaymentDetails>
</HTNG_GetPaymentDetailsRS>
```

9.4.4 HTNG_GetPaymentDetailsRS JSON Example

```
{
  "PaymentDetails": [
    {
      "PaymentRef": "00946db5-cff5-428f-88df-97c947ef568d",
      "Payer": {
        "PayerID": "CLIENTID_1",
        "PayerID_Context": "PAYEE",
        "PayerName": "Travel Company 1"
      },
      "Payee": {
        "PayeeID": "H001",
        "PayeeID_Context": "PAYEE",
        "PayeeName": "Hotel 1",
        "PayeeAddress": "100 Main Street",
        "PayeeZipCode": "90002",
        "PayeeRegionCode": "California",
        "PayeeCity": "Los Angeles",
        "PayeeCountry": "US",
        "PayeeEmail": "payments@hotel1.com"
      },
      "Payment": {
        "FormOfPayment": {
          "PaymentCard": {
            "VirtualInd": true,
            "CardType": "AmericanExpress",
            "Issuer": "AmericanExpress"
```

```
      }
    },
    "PaymentAmount": {
      "Amount": 80000,
      "Currency": {
        "CurrencyCode": "USD",
        "DecimalPlaces": "2"
      }
    },
    "DefaultPaymentDate": "2021-01-03T09:30:47Z",
    "LatestPaymentDate": "2021-01-07T09:30:47Z"
  },
  "PaymentDocument": {
    "Bookings": [
      {
        "Booking": [
          {
            "BookingRefs": [
              {
                "BookingRef": "BK-324798",
                "BookingType": "BookingSource"
              }
            ],
            "BookingDate": "2020-10-01",
            "ServiceStartDate": "2020-12-31",
            "ServiceEndDate": "2021-01-04",
            "BookingAmount": {
              "Amount": 80000,
              "Currency": {
                "CurrencyCode": "USD",
                "DecimalPlaces": "2"
              }
            }
          }
        ]
      }
    ]
  },
  "Inclusions": {
                "Inclusion": [
                  "Room",
                  "Breakfast"
                ]
        },
```

```
    "EarlyCollectionConditions": [
     {
      "EarlyCollectionReason": "NoShow",
      "EarlyCollectionDetail": [
       {
        "DateFrom": "2020-12-31T09:30:47Z",
        "DateThrough": "2021-01-07T09:30:47Z",
        "PaymentAmount": {
         "Amount": 20000,
         "Currency": {
          "CurrencyCode": "USD",
          "DecimalPlaces": "2"
         }
        }
       }
      ]
     }
    ],
    "PaymentStatus": {
     "Status": "Inactive",
     "StatusDescription": ""
    }
   }
  ]
}
```


9.5 Payment Details Notification

This example illustrates the notification message sent by the Payment Manager to the Payee with detailed information of a given payment.

9.5.1 HTNG_PaymentDetailsNotifRQ XML Example


```
<?xml version="1.0" encoding="UTF-8"?>
<HTNG_PaymentDetailsNotifRQ xsi:schemaLocation="http://htng.org/2019A
HTNG_PaymentDetailsNotifRQ.xsd" xmlns="http://htng.org/2019A"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <PaymentDetails>
                <PaymentDetail>
                        <PaymentRef>00946db5-cff5-428f-88df-97c947ef568d</PaymentRef>
                        <Payer>
                                <PayerID>CLIENTID_1</PayerID>
                                <PayerID_Context>PAYEE</PayerID_Context>
                                <PayerName>Travel Company 1</PayerName>
                        </Payer>
                        <Payee>
```

```
                    <PayeeID>H001</PayeeID>
                    <PayeeID_Context>PAYEE</PayeeID_Context>
                    <PayeeName>Hotel 1</PayeeName>
                    <PayeeAddress>100 Main Street</PayeeAddress>
                    <PayeeZipCode>90002</PayeeZipCode>
                    <PayeeRegionCode>California</PayeeRegionCode>
                    <PayeeCity>Los Angeles</PayeeCity>
                    <PayeeCountry>US</PayeeCountry>
                    <PayeeEmail>payments@hotel1.com</PayeeEmail>
                </Payee>
                <Payment>
                    <FormOfPayment>
                        <PaymentCard>
                            <VirtualInd>true</VirtualInd>
                            <CardType>AmericanExpress</CardType>
                            <Issuer>AmericanExpress</Issuer>
                        </PaymentCard>
                    </FormOfPayment>
                    <PaymentAmount>
                        <Amount>80000</Amount>
                        <Currency>
                            <CurrencyCode>USD</CurrencyCode>
                            <DecimalPlaces>2</DecimalPlaces>
                        </Currency>
                    </PaymentAmount>
                    <DefaultPaymentDate>2021-01-03T09:30:47Z</DefaultPaymentDate>
                    <LatestPaymentDate>2021-01-07T09:30:47Z</LatestPaymentDate>
                </Payment>
                <PaymentDocument>
                    <Bookings>
                        <Booking>
                            <BookingRefs>
                                <BookingRef>BK-324798</BookingRef>

<BookingRefType>BookingSource</BookingRefType>
                            </BookingRefs>
                            <BookingDate>2020-10-01</BookingDate>
                        <ServiceStartDate>2020-12-31</ServiceStartDate>
                        <ServiceEndDate>2021-01-04</ServiceEndDate>
                            <BookingAmount>
                                <Amount>80000</Amount>
                                <Currency>
                                    <CurrencyCode>USD</CurrencyCode>
                                    <DecimalPlaces>2</DecimalPlaces>
```

```
                                        </Currency>
                                    </BookingAmount>
                                </Booking>
                            </Bookings>
                        </PaymentDocument>
                        <Inclusions>
                            <Inclusion>Room</Inclusion>
                            <Inclusion>Breakfast</Inclusion>
                        </Inclusions>
                        <EarlyCollectionConditions>
                            <EarlyCollectionCondition>
                                <EarlyCollectionReason>NoShow</EarlyCollectionReason>
                                <EarlyCollectionDetail>
                                    <DateFrom>2020-12-31T09:30:47Z</DateFrom>
                                    <DateThrough>2021-01-07T09:30:47Z</DateThrough>
                                    <PaymentAmount>
                                        <Amount>20000</Amount>
                                        <Currency>
                                            <CurrencyCode>USD</CurrencyCode>
                                            <DecimalPlaces>2</DecimalPlaces>
                                        </Currency>
                                    </PaymentAmount>
                                </EarlyCollectionDetail>
                            </EarlyCollectionCondition>
                        </EarlyCollectionConditions>
                        <PaymentStatus>
                            <Status>Inactive</Status>
                            <StatusDescription/>
                        </PaymentStatus>
                    </PaymentDetail>
                </PaymentDetails>
</HTNG_PaymentDetailsNotifRQ>
```

9.5.2 HTNG_PaymentDetailsNotifRQ JSON Example

```
{
  "PaymentDetails": [
    {
      "PaymentRef": "00946db5-cff5-428f-88df-97c947ef568d",
      "Payer": {
        "PayerID": "CLIENTID_1",
        "PayerID_Context": "PAYEE",
        "PayerName": "Travel Company 1"
      },
      "Payee": {
```

```
    "PayeeID": "H001",
    "PayeeID_Context": "PAYEE",
    "PayeeName": "Hotel 1",
    "PayeeAddress": "100 Main Street",
    "PayeeZipCode": "90002",
    "PayeeRegionCode": "California",
    "PayeeCity": "Los Angeles",
    "PayeeCountry": "US",
    "PayeeEmail": "payments@hotel1.com"
  },
  "Payment": {
    "FormOfPayment": {
      "PaymentCard": {
        "VirtualInd": true,
        "CardType": "AmericanExpress",
        "Issuer": "AmericanExpress"
      }
    },
    "PaymentAmount": {
      "Amount": 80000,
      "Currency": {
        "CurrencyCode": "USD",
        "DecimalPlaces": "2"
      }
    },
    "DefaultPaymentDate": "2021-01-03T09:30:47Z",
    "LatestPaymentDate": "2021-01-07T09:30:47Z"
  },
  "PaymentDocument": {
    "Bookings": [
      {
        "Booking": [
          {
            "BookingRefs": [
              {
                "BookingRef": "BK-324798",
                "BookingType": "BookingSource"
              }
            ],
            "BookingDate": "2020-10-01",
            "ServiceStartDate": "2020-12-31",
            "ServiceEndDate": "2021-01-04",
            "BookingAmount": {
              "Amount": 80000,
```

```
            "Currency": {
              "CurrencyCode": "USD",
              "DecimalPlaces": "2"
            }
          }
        }
      ]
    }
  ]
},
"Inclusions": {
              "Inclusion": [
                "Room",
                "Breakfast"
              ]
      },
"EarlyCollectionConditions": [
 {
  "EarlyCollectionReason": "NoShow",
  "EarlyCollectionDetail": [
   {
    "DateFrom": "2020-12-31T09:30:47Z",
    "DateThrough": "2021-01-07T09:30:47Z",
    "PaymentAmount": {
     "Amount": 20000,
     "Currency": {
      "CurrencyCode": "USD",
      "DecimalPlaces": "2"
     }
    }
   }
  ]
 }
],
"PaymentStatus": {
  "Status": "Inactive",
  "StatusDescription": ""
}
  }
 ]
}
```

9.6 Get Payment Status

This example illustrates the query made by the Payee to the Payment Manager to retrieve the status of a given payment.

9.6.1 HTNG_GetPaymentStatusRQ XML Example

```
<?xml version="1.0" encoding="UTF-8"?>
<HTNG_GetPaymentStatusRQ xsi:schemaLocation="http://htng.org/2019A
HTNG_GetPaymentStatusRQ.xsd" xmlns="http://htng.org/2019A"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <PaymentRefs>
                <PaymentRef>00946db5-cff5-428f-88df-97c947ef568d</PaymentRef>
        </PaymentRefs>
</HTNG_GetPaymentStatusRQ>
```

9.6.2 HTNG_GetPaymentStatusRQ JSON Example

```
{
  "PaymentRefs": [
    "00946db5-cff5-428f-88df-97c947ef568d"
  ]
}
```

9.6.3 HTNG_GetPaymentStatusRS XML Example

```
<?xml version="1.0" encoding="UTF-8"?>
<HTNG_GetPaymentStatusRS xsi:schemaLocation="http://htng.org/2019A
HTNG_GetPaymentStatusRS.xsd" xmlns="http://htng.org/2019A"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <Payments>
                <Payment>
                        <PaymentRef>00946db5-cff5-428f-88df-97c947ef568d</PaymentRef>
                        <PaymentStatus>
                                <Status>Submitted</Status>
                                <StatusDescription/>
                        </PaymentStatus>
                </Payment>
        </Payments>
</HTNG_GetPaymentStatusRS>
```

9.6.4 HTNG_GetPaymentStatusRS JSON Example

```
{
  "Payments": [
    {
      "PaymentRef": "00946db5-cff5-428f-88df-97c947ef568d",
      "PaymentStatus": {
        "Status": "Submitted",
        "StatusDescription": ""
```

```
    }
   }
  ]
}
```

9.7 Payment Status Notification

This example illustrates the notification message sent by the Payment Manager to the Payer or Payee with the status of a given payment.

9.7.1 HTNG_PaymentStatusNotifRQ XML Example

```
<?xml version="1.0" encoding="UTF-8"?>
<HTNG_PaymentStatusNotifRQ xsi:schemaLocation="http://htng.org/2019A
HTNG_PaymentStatusNotifRQ.xsd" xmlns="http://htng.org/2019A"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <Payments>
                <Payment>
                        <PaymentRef>00946db5-cff5-428f-88df-97c947ef568d</PaymentRef>
                        <PaymentStatus>
                                <Status>Submitted</Status>
                                <StatusDescription/>
                        </PaymentStatus>
                </Payment>
        </Payments>
</HTNG_PaymentStatusNotifRQ>
```

9.7.2 HTNG_PaymentStatusNotifRQ JSON Example

```
{
  "Payments": [
    {
      "PaymentRef": "00946db5-cff5-428f-88df-97c947ef568d",
      "PaymentStatus": {
        "Status": "Submitted",
        "StatusDescription": ""
      }
    }
  ]
}
```

9.8 Find Payment Reference

This example illustrates the query made by the Payee to the Payment Manager to retrieve the payment reference associated to a given booking.

9.8.1 HTNG_FindPaymentRefRQ XML Example

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<HTNG_FindPaymentRefRQ xsi:schemaLocation="http://htng.org/2019A
HTNG_FindPaymentRefRQ.xsd" xmlns="http://htng.org/2019A"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <PaymentInfo>
                <Payer>
                        <PayerID>CLIENTID_1</PayerID>
                        <PayerID_Context>YYY</PayerID_Context>
                        <PayerName>Travel Company 1</PayerName>
                </Payer>
                <BookingInfos>
                        <BookingInfo>
                                <BookingRefs>
                                        <BookingRef>BK-324798</BookingRef>
                                        <BookingRefType>BookingSource</BookingRefType>
                                </BookingRefs>
                                <BookingDate>2020-10-01</BookingDate>
                        </BookingInfo>
                </BookingInfos>
        </PaymentInfo>
</HTNG_FindPaymentRefRQ>
```

9.8.2 HTNG_FindPaymentRefRQ JSON Example

```
{
  "PaymentInfo": {
    "Payer": {
      "PayerID": "CLIENTID_1",
      "PayerID_Context": "YYY",
      "PayerName": "Travel Company 1"
    },
    "BookingInfos": [
      {
        "BookingRefs": [
          {
            "BookingRef": "BK-324798",
            "BookingType": "BookingSource"
          }
        ],
        "BookingDate": "2020-10-01"
      }
    ]
  }
}
```

9.8.3 HTNG_FindPaymentRefRS XML Example

```
<?xml version="1.0" encoding="UTF-8"?>
<HTNG_FindPaymentRefRS xsi:schemaLocation="http://htng.org/2019A
HTNG_FindPaymentRefRS.xsd" xmlns="http://htng.org/2019A"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <PaymentInfos>
                <PaymentInfo>
                        <PaymentRef>00946db5-cff5-428f-88df-97c947ef568d</PaymentRef>
                        <BookingInfo>
                                <BookingRefs>
                                        <BookingRef>BK-324798</BookingRef>
                                        <BookingRefType>BookingSource</BookingRefType>
                                </BookingRefs>
                                <BookingDate>2020-10-01</BookingDate>
                        </BookingInfo>
                </PaymentInfo>
        </PaymentInfos>
</HTNG_FindPaymentRefRS>
```

9.8.4 HTNG_FindPaymentRefRS JSON Example

```
{
  "PaymentInfos": [
    {
      "PaymentRef": [
        "00946db5-cff5-428f-88df-97c947ef568d"
      ],
      "BookingInfo": {
        "BookingRefs": [
          {
            "BookingRef": "BK-324798",
            "BookingType": "BookingSource"
          }
        ],
        "BookingDate": "2020-10-01"
      }
    }
  ]
}
```

9.9 Initiate Payment

This example illustrates the instruction sent by the Payer or Payee to the Payment Manager to initiate a given payment.

9.9.1 HTNG_InitiatePaymentRQ XML Example

```
<?xml version="1.0" encoding="UTF-8"?>
<HTNG_InitiatePaymentRQ xsi:schemaLocation="http://htng.org/2019A HTNG_InitiatePaymentRQ.xsd"
xmlns="http://htng.org/2019A" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <Payment>
                <PaymentRef>00946db5-cff5-428f-88df-97c947ef568d</PaymentRef>
                <PaymentAmount>
                        <Amount>20000</Amount>
                        <Currency>
                                <CurrencyCode>USD</CurrencyCode>
                                <DecimalPlaces>2</DecimalPlaces>
                        </Currency>
                </PaymentAmount>
                <EarlyCollectionReason>NoShow</EarlyCollectionReason>
                <Comment/>
        </Payment>
</HTNG_InitiatePaymentRQ>
```

9.9.2 HTNG_InitiatePaymentRQ JSON Example

```
{
  "Payment": {
    "PaymentRef": "00946db5-cff5-428f-88df-97c947ef568d",
    "PaymentAmount": {
      "Amount": 20000,
      "Currency": {
        "CurrencyCode": "USD",
        "DecimalPlaces": "2"
      }
    },
    "EarlyCollectionReason": "NoShow",
    "Comment": ""
  }
}
```

9.9.3 HTNG_InitiatePaymentRS XML Example

```
<?xml version="1.0" encoding="UTF-8"?>
<HTNG_InitiatePaymentRS xsi:schemaLocation="http://htng.org/2019A HTNG_InitiatePaymentRS.xsd"
xmlns="http://htng.org/2019A" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <PaymentRef>00946db5-cff5-428f-88df-97c947ef568d</PaymentRef>
</HTNG_InitiatePaymentRS>
```

9.9.4 HTNG_InitiatePaymentRS JSON Example

```
{
  "PaymentRef": "00946db5-cff5-428f-88df-97c947ef568d"
}
```

10 Appendices

10.1 Glossary of Terms

For the purpose of this document, the following terms have been defined as follows:

Term	Definition
Booking Engine	Tool used by travel companies to send their reservations to hotels, either via direct connect or connectivity hubs such as GDSs or channel managers.
Business to Business (B2B)	A transaction between two businesses.
Business to Consumer (B2C)	A transaction between a business or online retailer and their consumers.
Channel Manager	Hub which provides connectivity between booking engines and CRSs.
Central Reservation System (CRS)	Centralizes the collection of all bookings received by a hotel from different sources and sends them to the hotel's PMS.
Global Distribution System (GDS)	Interconnects booking engines and CRSs. The most popular with a market share higher that 90% are Amadeus, Sabre and Travelport.
Open Payments Alliance (OPA)	An alliance of HEDNA and HTNG members designed to create an efficient ecosystem for B2B payments.
Online Travel Agency (OTA)	A web-based marketplace that allows consumers to research and book travel products and services.
Payment Manager (PM)	The party that orchestrates payments between the Payer and the Payee.
Property Management System (PMS)	A software application for the day-to-day operations of hospitality accommodations.
Payment Processor	The party responsible for the transfer of the funds from the Payer to the Payee.
Payment Source	The party responsible for creating a payment instrument.
Virtual Credit Card (VCC)	A credit card number, designed for single and/or temporary uses, without a physical card produced.

10.2 Implementation Notes

The following information is intended to aid in the implementation process.

10.2.1 Payment Reference Generation

A UUID will be used for the Payment Reference following the established IETF standard rfc4122 https://tools.ietf.org/html/rfc4122. The Wikipedia page for UUID is provided for ease of understanding: https://en.wikipedia.org/wiki/Universally_unique_identifier (The assumption is that the Payment Reference is always sent using this UUID format).

10.2.2 Modifiable Payment Data

A limited set of data in a payment may be modified only when the payment is in a status which allows modifications; inactive, active and pending. This section specifies which data within a payment may be modified.

Description	Data Type	Element
Virtual card indicator	Boolean	HTNG_ModifyPaymentRQ/PaymentDetail/Payment/FormOfPayment/PaymentCard/VirtualInd
Virtual card amount	Money	HTNG_ModifyPaymentRQ/PaymentDetail/Payment/FormOfPayment/PaymentCard/CardRequestDetails/CardAmount/Amount
Virtual card valid-from date	DateTime	HTNG_ModifyPaymentRQ/PaymentDetail/Payment/FormOfPayment/PaymentCard/CardRequestDetails/ValidFrom
Virtual card valid-through date	DateTime	HTNG_ModifyPaymentRQ/PaymentDetail/Payment/FormOfPayment/PaymentCard/CardRequestDetails/ValidThrough
Card number	String	HTNG_ModifyPaymentRQ/PaymentDetail/Payment/FormOfPayment/PaymentCard/CardPayment/PAN
Card expiration date	DateTime	HTNG_ModifyPaymentRQ/PaymentDetail/Payment/FormOfPayment/PaymentCard/CardPayment/ExpirationDate
Card CVV	String	HTNG_ModifyPaymentRQ/PaymentDetail/Payment/FormOfPayment/PaymentCard/CardPayment/CVV
Card holder name	String	HTNG_ModifyPaymentRQ/PaymentDetail/Payment/FormOfPayment/PaymentCard/CardPayment/CardHolderName
Card token provider ID	String	HTNG_ModifyPaymentRQ/PaymentDetail/Payment/FormOfPayment/PaymentCard/CardPayment/TokenProviderID
Payment amount	Money	HTNG_ModifyPaymentRQ/PaymentDetail/Payment/PaymentAmount/Amount
Tolerance	Percentage	HTNG_ModifyPaymentRQ/PaymentDetail/Payment/Tolerance
Default payment date	DateTime	HTNG_ModifyPaymentRQ/PaymentDetail/Payment/DefaultPaymentDate
Earliest payment date	DateTime	HTNG_ModifyPaymentRQ/PaymentDetail/Payment/EarliestPaymentDate
Latest payment date	DateTime	HTNG_ModifyPaymentRQ/PaymentDetail/Payment/LatestPaymentDate

Secondary booking ref *	String	HTNG_ModifyPaymentRQ/PaymentDetail/PaymentDocument/Bookings/Booking/BookingRefs/BookingRef
Secondary booking type*	String Enumeration	HTNG_ModifyPaymentRQ/PaymentDetail/PaymentDocument/Bookings/Booking/BookingRefs/BookingType
Inclusions **	InclusionsType	HTNG_ModifyPaymentRQ/PaymentDetail/Inclusions
Payment conditions **	PaymentConditionsType	HTNG_ModifyPaymentRQ/PaymentDetail/PaymentConditions
Early collection conditions **	EarlyCollectionConditionType	HTNG_ModifyPaymentRQ/PaymentDetail/EarlyCollectionConditions

* Please note that the booking reference is not modifiable however a booking reference for a hotel's confirmation number may be sent after the booking is made, so a secondary (hotel confirmation number) booking reference may be sent in the modify message.

** Please note that if the Inclusions, Payment Conditions or Early Collection Conditions change, all of the items in that type must be sent in a modification message as this is a full overlay.

10.2.3 Sending the Payment Reference with the Booking

Existing fields in the OpenTravel messages are utilized to send the Payment Reference with the booking. The fields can be found within the OTA_Reservation.xsd here:

When sending as a guarantee:

- HotelReservationType/RoomStays/RoomStay/Guarantee/GuaranteesAccepted/GuaranteeAccepted/@GuaranteeTypeCode
- HotelReservationType/RoomStays/RoomStay/Guarantee/GuaranteesAccepted/GuaranteeAccepted/@GuaranteeID

When sending as a deposit or payment:

- HotelReservationType/RoomStays/RoomStay/DepositPayments/GuaranteePayment/AcceptedPayments/AcceptedPayment/@GuaranteeTypeCode
- HotelReservationType/RoomStays/RoomStay/DepositPayments/GuaranteePayment/AcceptedPayments/AcceptedPayment/@GuaranteeID

The GuaranteeTypeCode field is defined as type OTA_CodeType and uses the Payment Type (PMT) Codelist. A new code has been added for this purpose; 49 – Payment Reference.

The GuaranteeID field is defined as StringLength1to64 allowing for a payment reference of up to 64 characters in length.

To include the payment reference in the booking, send code #49 in the GuaranteeTypeCode field and send the Payment Reference in the GuaranteeID field.

Please note that if the receiving system is unable to accept these two fields, the Payment Reference may be retrieved from the Payment Manager using the *HTNG_FindPaymentRefRQ/RS* message set by providing the payer ID, booking reference and booking date.

10.2.4 HTNG API Registry

HTNG hosts an API Registry where implementers can post information about their products and APIs; others are then able to connect with those listed to learn about their products. There has been a new category filter added specifically for Payment Managers. More information on the API Registry can be found here: https://www.htng.org/page/APIRegistry

10.2.5 Links

To submit comments to HTNG Staff regarding this document, including (but not limited to) change requests or clarifications, please visit this form and submit your comments. HTNG Staff may contact you for follow up.