xavi

[Capte la atención del lector con un resumen atractivo. Este resumen es una breve descripción del documento. Cuando esté listo para agregar contenido, haga clic aquí y empiece a escribir.]

PORTADA

[Subtítulo del documento]

Contenido

[Lista de tablas 0](#_Toc516317257)

[Lista de figuras 1](#_Toc516317258)

[Lista de acrónimos 2](#_Toc516317259)

[Introducción 3](#_Toc516317260)

[1.1 Contextualización 3](#_Toc516317261)

[1.1.1 Funcionamiento de una integración 3](#_Toc516317262)

[1.1.2 Principales transacciones en una integración de vuelos 4](#_Toc516317263)

[1.1.3 Relación entre proveedores, TravelgateX y Logitravel 7](#_Toc516317264)

[1.1.4 GDS: Global Distribution System 8](#_Toc516317265)

[1.1.5 Conceptos y terminología del GDS Amadeus 9](#_Toc516317266)

[1.1.6 Web Services y WSDL 12](#_Toc516317267)

[1.1.7 Estado actual de los webservices de Amadeus 13](#_Toc516317268)

[1.1.8 Estado previo de la API de transportes de TravelgateX 14](#_Toc516317269)

[1.1.9 Certificación de Amadeus (proceso) 18](#_Toc516317270)

[1.2 Definición del problema 19](#_Toc516317271)

[1.2.1 Unidades que forman una reserva 19](#_Toc516317272)

[1.2.2 Por qué surge la necesidad de modificar una reserva 20](#_Toc516317273)

[1.2.3 En qué consiste la modificación de una reserva 21](#_Toc516317274)

[1.3. Respuesta al problema 22](#_Toc516317275)

[1.3.1. Idea básica del flujo implementado para modificar reservas 22](#_Toc516317276)

[1.3.2. Qué tiene que desarrollarse para que una agencia pueda modificar reservas de Amadeus a través de TravelgateX 23](#_Toc516317277)

[1.3.2.1. Nuevas transacciones a desarrollar para la API de transportes 23](#_Toc516317278)

[1.3.2.2 Flujo de transacciones con Amadeus 25](#_Toc516317279)

[1.3.2.3 Integración a desarrollar 27](#_Toc516317280)

[1.3.2.4 Formulario de pruebas 27](#_Toc516317281)

[1.3.3 Certificación de Amadeus (avance de los resultados) 28](#_Toc516317282)

[Plan del proyecto 0](#_Toc516317283)

[2.1 Alcance del proyecto 0](#_Toc516317284)

[2.1.1 Entregables 1](#_Toc516317285)

[2.2 Requisitos del proyecto 2](#_Toc516317286)

[2.2.1 Requisitos de proyecto 2](#_Toc516317287)

[2.2.2 Requisitos no funcionales 2](#_Toc516317288)

[2.2.3 Requisitos funcionales 3](#_Toc516317289)

[2.3 Criterios de aceptación 4](#_Toc516317290)

[2.3.1 Certificación de Amadeus (criterios de aceptación) 4](#_Toc516317291)

[2.3.2 Criterios de aceptación de la agencia 5](#_Toc516317292)

[2.4 Restricciones del proyecto 6](#_Toc516317293)

[2.4.1 Restricción de tiempo 6](#_Toc516317294)

[2.4.2 Factores ambientales: cultura del Dpto. de integraciones 6](#_Toc516317295)

[2.5 Organización 7](#_Toc516317296)

[2.5.1 Cultura y estilo de TravelgateX 7](#_Toc516317297)

[2.5.2 Participantes internos del proyecto 7](#_Toc516317298)

[2.6 Partes interesadas 7](#_Toc516317299)

[2.6.1 Interesados internos y externos 7](#_Toc516317300)

[2.6.2 Relación entre interesados 8](#_Toc516317301)

[2.7 Riesgos 8](#_Toc516317302)

[2.7.1 Complejidad del proyecto 9](#_Toc516317303)

[2.7.2 Planificación temporal 9](#_Toc516317304)

[2.8 Gestión del tiempo 9](#_Toc516317305)

[2.8.1 Estructura de descomposición del trabajo (EDT) 9](#_Toc516317306)

[2.8.2 Planificación de la duración de los paquetes de trabajo 10](#_Toc516317307)

[2.8.3 Diagrama de Gantt 13](#_Toc516317308)

[2.9 Plan de gestión de comunicaciones 1](#_Toc516317309)

[2.9.1 Comunicación interna 1](#_Toc516317310)

[2.9.2 Comunicación con Logitravel 1](#_Toc516317311)

[2.9.3 Comunicación con Amadeus 1](#_Toc516317312)

[Análisis 0](#_Toc516317313)

[3.1 Requisitos no funcionales 0](#_Toc516317314)

[3.1.1 Requisitos no funcionales de las transacciones de la API de transportes 0](#_Toc516317315)

[3.1.2 Requisitos no funcionales de la integración 1](#_Toc516317316)

[3.1.3 Requisitos no funcionales del formulario de pruebas 1](#_Toc516317317)

[3.2 Requisitos funcionales 2](#_Toc516317318)

[Diseño 0](#_Toc516317319)

[4.1 Transacciones API Transportes 0](#_Toc516317320)

[4.2 Diagramas de flujo de transacciones con el GDS 1](#_Toc516317321)

[4.3 Mockups del formulario de pruebas 1](#_Toc516317322)

[0](#_Toc516317323)

[Programación 0](#_Toc516317324)

[5.1 Transacciones de la API de Transportes 0](#_Toc516317325)

[5.1.1 Lenguaje/tecnología 0](#_Toc516317326)

[5.1.2 Descripción de las transacciones programadas 1](#_Toc516317327)

[5.2 Integración 7](#_Toc516317328)

[5.2.1 Lenguaje/tecnología 7](#_Toc516317329)

[5.2.2 Funcionalidades destacadas 7](#_Toc516317330)

[5.3 Formulario de pruebas 9](#_Toc516317331)

[5.3.1 Lenguaje/tecnología 9](#_Toc516317332)

[5.3.2 Ejemplo de uso (vídeo) 9](#_Toc516317333)

[5.4 Programación de tests unitarios 9](#_Toc516317334)

[5.4.1 Importancia 10](#_Toc516317335)

[5.4.2 Lenguaje/tecnología 10](#_Toc516317336)

[Pruebas 0](#_Toc516317337)

[6.1 Certificación Amadeus 0](#_Toc516317338)

[6.1.1 Escenarios a probar y presentar 0](#_Toc516317339)

[6.1.2 Primer intento 1](#_Toc516317340)

[6.1.2 Segundo intento 2](#_Toc516317341)

[6.2 Fase interna de pruebas 1](#_Toc516317342)

[Despliegue 0](#_Toc516317343)

[Referencias bibliográficas 0](#_Toc516317344)

## Lista de tablas

[Tabla 1: Petición de transacción Disponibilidad: DisponibilidadRQ](#Tabla1)

[Tabla 2: Respuesta de transacción Disponibilidad: DisponibilidadRS](#Tabla2)

[Tabla 3: Petición de transacción Valoracion: ValoracionRQ](#Tabla3)

[Tabla 4: Respuesta de transacción Valoracion: ValoracionRS](#Tabla4)

[Tabla 5: Petición de transacción Reserva: ReservaRQ](#Tabla5)

[Tabla 6: Respuesta de transacción Reserva: ReservaRS](#Tabla6)

[Tabla 7: Petición de transacción Emitir: EmitirRQ](#Tabla7)

[Tabla 8: Respuesta de transacción Emitir: EmitirRS](#Tabla8)

[Tabla 9: Petición de transacción CancelaReserva: CancelaReservaRQ](#Tabla9)

[Tabla 10: Respuesta de transacción CancelaReserva: CancelaReservaRS](#Tabla10)

[Tabla 11: Petición de transacción RecuperaReserva: RecuperaReservaRQ](#Tabla11)

[Tabla 12: Respuesta de transacción RecuperaReserva: RecuperaReservaRS](#Tabla12)

[Tabla 13: Elemento tramoDisponibilidad de la petición DMR\_RQ](#Tabla13)

[Tabla 14: Listado de entregables definidos en el alcance del plan de proyecto](#Tabla14)

[Tabla 15: Criterios de aceptación generales](#Tabla15)

[Tabla 16: Criterios de aceptación de gestión de sesión](#Tabla16)

[Tabla 17: Criterios de aceptación ATC](#Tabla17)

[Tabla 18: Paquetes de trabajo](#Tabla18)

[Tabla 19: Requisitos no funcionales de las transacciones de la API de transportes](#Tabla19)

[Tabla 20: Requisitos no funcionales de la integración](#Tabla20)

[Tabla 21: Requisitos no funcionales del formulario de pruebas](#Tabla21)

[Tabla 22: Requisitos funcionales](#Tabla22)

[Tabla 23: Elementos y atributos más importantes de DMR](#Tabla23)

[Tabla 24: Elementos y atributos más importantes de RMR](#Tabla24)

[Tabla 25: Elementos y atributos más importantes de EMR](#Tabla25)

## Lista de figuras

[Figura 1: Representación visual simple de una integración](#Figura1)

[Figura 2: Representación visual de una integración con usuario y producto](#Figura2)

[Figura 3: Formulario web de Disponibilidad](#Figura3)

[Figura 4: Formulario web de Valoración](#Figura4)

[Figura 5: Formulario web de Valoración una vez seleccionadas las opciones](#Figura5)

[Figura 6: Representación visual de una integración con TravelgateX y cliente/proveedor](#Figura6)

[Figura 7: One Way Trip](#Figura7)

[Figura 8: Round Trip](#Figura8)

[Figura 9: Circle Trip](#Figura9)

[Figura 10: Open Jaw simple en el destino](#Figura10)

[Figura 11: Open Jaw simple en el origen](#Figura11)

[Figura 12: Open Jaw doble](#Figura12)

[Figura 13: Ciclo de vida del proyecto de Amadeus](#Figura13)

[Figura 14: Esquema de la aplicación Modificar Reservas](#Figura14)

[Figura 15: Relación entre los interesados del proyecto](#Figura15)

[Figura 16: Estructura de descomposición del trabajo (EDT)](#Figura16)

[Figura 17: Diagrama de Gantt](#Figura17)

[Figura 18: Diseño de las clases de la transacción DMR](#Figura18)

[Figura 19: Diseño de clases de la transacción RMR](#Figura19)

[Figura 20: Diseño de clases de la transacción EMR](#Figura20)

[Figura 21: Diagrama de flujo de ejecución de la transacción DMR](#Figura21)

[Figura 22: Diagrama de flujo de ejecución de la transacción RMR](#Figura22)

[Figura 23: Diagrama de flujo de ejecución de la transacción EMR](#Figura23)

[Figura 24: Mockup de la pestaña Disponibilidad-Valoración](#Figura24)

[Figura 25: Mockup de la pestaña Reserva](#Figura25)

[Figura 26: Mockup de la pestaña Recuperar Reserva](#Figura26)

[Figura 27: Mockup de la pestaña Modificar Reserva](#Figura27)

[Figura 28: Diagrama de Gantt (desviaciones temporales)](#Figura28)

## Lista de acrónimos

RQ: Petición o *request* de una transacción

RS: Respuesta o *response* de una transacción

GDS: Global Distribution System

URL: Uniform Resource Locator

API: Application Programming Interface

IBM: International Business Machines

CRS: Customer Reservations System

SABRE: Semi Automatic Business Environment Research

OW: One Way Trip

RT: Round Trip

CT: Circle Trip

OJ: Open Jaw

PNR: Passenger Name Record

LTD: Last Ticketing Date

ADT: Adult passenger

CHD: Child passenger

INF: Infant passenger

WSDL: Web Services Description Language

XML: Extensible Markup Language

W3C: World Wide Web Consortium

DMR: DisponibilidadModificarReserva

DMR\_RQ: Petición de la transacción DisponibilidadModificarReserva

DMR\_RS: Respuesta de la transacción DisponibilidadModificarReserva

RMR: ReservaModificarReserva

RMR\_RQ: Petición de la transacción ReservaModificarReserva

RMR\_RS: Respuesta de la transacción ReservaModificarReserva

EMR: EmitirModificarReserva

EMR\_RQ: Petición de la transacción EmitirModificarReserva

EMR\_RS: Respuesta de la transacción EmitirModificarReserva

ATC: Amadeus Ticket Changer

OTA: Online Travel Agency

SOAP: Simple Access Object Protocol

FAQ: Frequently Asked Questions

EDT: Estructura de Descomposición del Trabajo

XSD: XML Schema Definition

CVV: Card Verification Value

TSM: Transitional Stored Miscellaneous

TST: Transitional Stored Ticket

EMD: Electronic Miscellaneous Document

POO: Programación Orientada a Objetos

IATA: International Air Transport Association

# Introducción

## 1.1 Contextualización

### 1.1.1 Funcionamiento de una integración

En este apartado se define qué es una integración y qué proceso siguen los sistemas que interactúan con la integración.

En su más pura definición, una integración se encarga de conectar dos sistemas a través de transacciones, petición [RQ](#AcronimoRQ) y respuesta [RS](#AcronimoRS). En el contexto de este trabajo, los sistemas que van a conectarse son dos servicios web: el [GDS](#AcronimoGDS) Amadeus, que actuaría como proveedor; y la agencia de viajes Logitravel, que actuaría como cliente. Los dos actores son independientes el uno del otro, y cada uno interactúa con la integración mediante una tecnología y formatos diferentes.

Se puede desglosar un proceso completo de una integración de la forma siguiente:

* El cliente realiza una petición que llega a la integración en un formato y tecnología **A**. La petición del cliente va en busca de un servicio o producto que sabe que el proveedor posee.
* La integración transforma la petición al formato y tecnología **B** que usa el proveedor.
* A continuación la integración realiza la petición ya transformada al proveedor.
* El proveedor procesa la petición que le llega y envía de vuelta a la integración como respuesta, el producto que el cliente ha solicitado.
* De nuevo, la integración tiene que transformar los formatos y tecnologías que use el proveedor. Esta vez transforma del formato **B** al formato **A** la respuesta que le llega desde el proveedor.
* Finalmente la integración envía la respuesta al cliente ya transformada al formato y tecnología **A**.

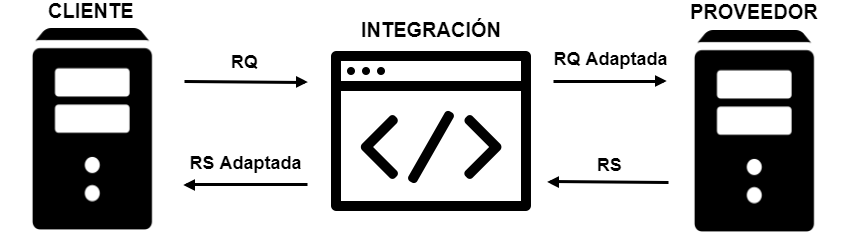
Sin entrar aun en las tecnologías y formatos de cliente y proveedor, a continuación, en la [Figura 1](#Figura1)*,* se sitúan conceptualmente a participantes, transacciones ([RQ](#AcronimoRQ) y [RS](#AcronimoRS)) e integración, en un diagrama.

Figura 1: Representación visual simple de una integración

El proceso representado en la figura anterior se denomina habitualmente transacción. Una transacción entre dos sistemas conectados por una integración, contiene entonces dos peticiones y dos respuestas.

Para hacer posible la conexión y completar una transacción, los 3 actores: cliente, integración y proveedor tienen que conocer previamente los puntos de acceso a los que enviar las peticiones y las respuestas. A estos puntos de acceso se les denomina comúnmente *endpoints*.

El acceso a un *endpoint* y consiguiente comunicación entre integración y cliente/proveedor y viceversa se consigue por norma general mediante:

* Dirección [URL](#AcronimoURL) del servicio web.
* Usuario y contraseña, o información que permita un agente identificarse.
* Protocolo criptográfico de mutuo acuerdo para la comunicación segura a través de internet.

Una vez establecida la definición de integración y transacción, pasan a detallarse cuáles son las transacciones más habituales de una integración de vuelos en el siguiente apartado [1.1.2](#_1.1.2_Principales_transacciones).

### 1.1.2 Principales transacciones en una integración de vuelos

En este apartado se introducen cuáles son las principales transacciones existentes en una integración de vuelos y cómo explicita éstas el cliente que se presenta en este proyecto en su sitio web.

Las transacciones que forman una integración de vuelos tienen como principal objetivo vender un producto o ayudar al cliente a que su compra sea más fácil, intuitiva o versátil. El producto, proporcionado por el proveedor, estará habitualmente almacenado en bases de datos.

Como intermediario entre la base de datos y los agentes que quieran acceder a los productos almacenados en ella, suele colocarse un *Web Service* o servicio web, proporcionado y construido por el mismo proveedor o por un tercero. Por lo tanto, el acceso a las bases de datos no va a realizarse directamente desde la integración a la base de datos mediante *querys*.

En la [Figura 2](#Figura2) presentada a continuación, se añaden los tres nuevos participantes mencionados hasta ahora en este capítulo: usuario, sitio web y base de datos.

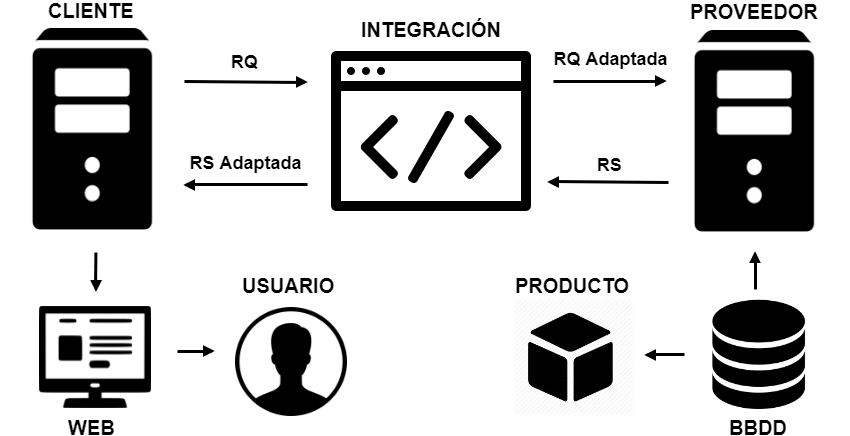


Figura 2: Representación visual de una integración con usuario y producto

El producto situado como meta en una integración de vuelos es una reserva aérea. Para poder ofrecer su venta, lo más habitual en una integración es desglosar ésta en tres transacciones, ejecutadas secuencialmente y en el orden que sigue:

#### Disponibilidad/Availability

Se solicita al proveedor qué productos están disponibles. La consulta a realizar dependerá de unos criterios predeterminados de búsqueda que el cliente escoja. Estos criterios son habitualmente tres:

* Fechas
* Localizaciones: aeropuertos o ciudades
* Pasajeros

Puede acotarse la búsqueda de disponibilidad de vuelos con otros criterios:

* Códigos de descuento
* Descuento de residente o familia numerosa

Como ya se ha introducido en el apartado [1.1.1](#_1.1.1_Funcionamiento_de), el proveedor procesa la petición y responde enviando, si están disponibles, los productos solicitados. La información de dichos productos se transforma y se envía al formato que el cliente entiende.

Ahora el cliente ya puede seleccionar una de las opciones que el proveedor tiene disponibles. Teniendo en cuenta los criterios de búsqueda que explicitados en la petición, el cliente selecciona una de las opciones con uno o más vuelos en unas fechas, desde y hasta unas localizaciones y para un número y tipo de pasajeros.

La información seleccionada, se emplea para construir la siguiente petición: valoración.

#### Valoración/Cotización/Valuation/Quote

Se encarga de comprobar si el producto seleccionado en la transacción de disponibilidad sigue disponible, actualiza el precio del mismo, ya que podría haber cambiado, devolviendo en este caso un precio y condiciones diferentes y ya definitivas.

La base de una valoración es por lo tanto verificar el precio y las condiciones del producto. La información del producto obtenido en la disponibilidad suele ampliarse en la valoración dado que la rapidez que se necesita para mostrar ésta al cliente no es tan crítica en esta transacción.

La información obtenida en una valoración se usa para construir la petición de reserva.

#### Reserva/Confirmación/Book

Se realiza la compra del elemento elegido. El pago del mismo al proveedor suele realizarse sin embargo, a continuación en otra transacción denominada Emisión o Ticketing.

Ayuda a comprender estas transacciones, imaginarlas en una página web de reservas de vuelos. Las tres transacciones definidas en este capítulo, se muestran a continuación a través de los habituales pasos que sigue un cliente en un sitio web de reservas:

#### Disponibilidad

Se elige un origen, un destino, una fecha y un número de pasajeros que viajarán en el avión. Cuando se está satisfecho con los datos introducidos y se pulsa el botón de búsqueda, implicando así que se realice una transacción de disponibilidad.

Figura 3: Formulario web de Disponibilidad

#### Valoración

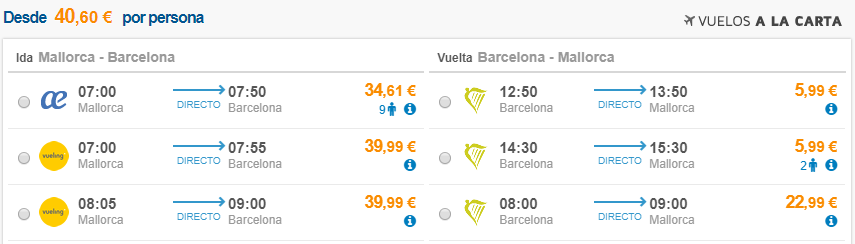
Se selecciona un vuelo de la lista de vuelos arrojada en la disponibilidad.

Figura 4: Formulario web de Valoración

Al seleccionar la opción se realiza automáticamente la transacción de valoración y a continuación aparecen los detalles del vuelo como se expone en la [Figura 5](#Figura5).

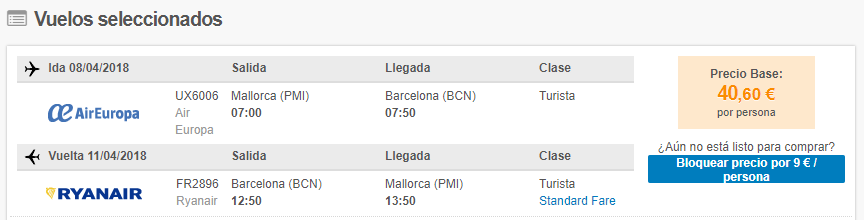


Figura 5: Formulario web de Valoración una vez seleccionadas las opciones

#### Reserva

Se completa la compra pulsando un botón “reservar” o “confirmar”. Esta acción conlleva posteriormente la introducción por parte del cliente de datos personales y de pago, entre otros. Al introducir los datos y aceptar, es cuando se realiza realmente la transacción de reserva.

Ya se han dado a conocer las tres transacciones principales que forman una integración de vuelos. El siguiente paso, expuesto en el siguiente capítulo [1.1.3](#_1.1.3_Relación_entre), es conocer y comprender la relación entre los tres participantes que interactuarán en las transacciones presentadas. Se expandirán los conceptos que envuelven a estos participantes, a lo largo del capítulo [1.1](#_1.1_Contextualización) restante.

### 1.1.3 Relación entre proveedores, TravelgateX y Logitravel

En este apartado se dará respuesta a las siguientes preguntas: ¿Qué ventajas tiene una agencia al conectarse a los proveedores a través de las integraciones de TravelgateX? ¿Por qué no conectarse directamente con los proveedores de vuelos?

TravelgateX, empresa en la que se ha realizado el desarrollo de este trabajo, dispone en producción, más de 20 integraciones de vuelos, entre ellas integraciones con webservices de compañías aéreas como Iberia, Vueling, Amercian Airlines o integraciones con sistemas [GDS](#AcronimoGDS) como Galileo, Sabre o Amadeus.

Al conectarse con TravelgateX, la agencia de viajes no tiene que entender ni traducir los formatos y tecnologías de los webservices que usen las 20 compañías aéreas, sino que para acceder al producto, únicamente tiene que entender el formato o [API](#AcronimoAPI) que utilice TravelgateX.

TravelgateX cuenta con una [API](#AcronimoAPI) de transportes, la cual da soporte a integraciones de vuelos, ferris y trenes, de tal manera que todos los productos de los proveedores pueden venderse a través de esta [API](#AcronimoAPI). Una agencia por lo tanto, tiene que conectarse y entender un único formato para poder acceder a los productos de todos los proveedores con los que cuenta TravelgateX.

La existencia de un intermediario que se integre con las compañías aéreas, supone una enorme diferencia de coste (tiempo y dinero) para una agencia cuando ésta decide empezar a vender los productos de un nuevo proveedor.

La agencia en este caso no tiene aprender qué peticiones y qué respuestas ha de enviar y recibir para poder comprar los vuelos a los proveedores y venderlos a sus clientes. Su única tarea es especificar a TravelgateX a qué proveedor quiere realizar las peticiones.

Los proveedores: compañías aéreas o [GDS](#AcronimoGDS)s se resumen en la [Figura 6](#Figura6) a través de uno único, Amadeus, ya que éste va a ser el proveedor en la integración. La relación entre Amadeus, TravelgateX y Logitravel se esquematiza en la [Figura 6](#Figura6).

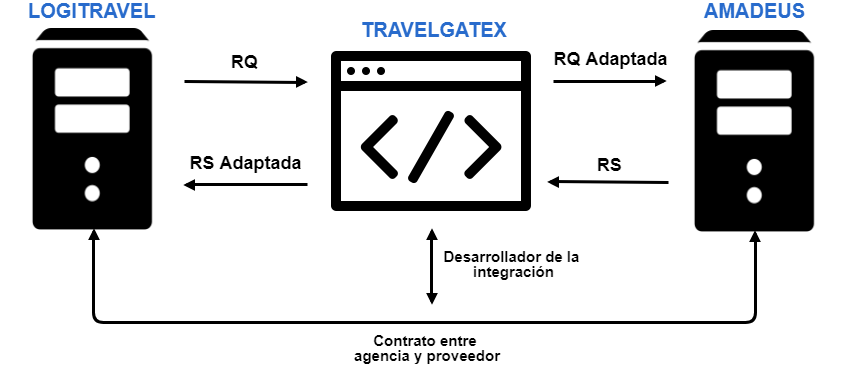


Figura 6: Representación visual de una integración con TravelgateX y cliente/proveedor

### 1.1.4 GDS: Global Distribution System

Desde un punto de vista de negocio, un [GDS](#AcronimoGDS) es un servicio centralizado y computarizado que permite la relación comercial entre los proveedores de servicios turísticos y las agencias o motores de búsqueda que demanden estos productos para su venta. Un [GDS](#AcronimoGDS) proporciona en directo, información sobre el producto, como su disponibilidad y precio.

David Chestler, vicepresidente ejecutivo de SiteMinder en América, compara un [GDS](#AcronimoGDS) con un supermercado diciendo: “Si quieres ser visible y recibir reservas, tu hotel necesita una presencia en las estanterías de este supermercado cuándo y dónde tus huéspedes salgan a comprar una reserva hotelera” [[1]](#Ref1).

Para entender mejor qué es un [GDS](#AcronimoGDS), tanto desde el punto de vista técnico como de negocio, hay que remontarse hasta su nacimiento y observar su evolución histórica a lo largo de los años hasta la actualidad.

Para elaborar el contenido que se cita a continuación, en el que se expone la evolución cronológica de los [GDS](#AcronimoGDS)s, se han utilizado las siguientes fuentes de información[[2]](#Ref2) [[3]](#Ref3) [[4]](#Ref4) [[5]](#Ref5).

* 1957 es el año en el que [IBM](#AcronimoIBM) y American Airlines empiezan a trabajar juntos en la creación de lo que se conoce como [CRS](#AcronimoCRS), un sistema semi-automatizado que llamaron [SABRE](#AcronimoSABRE).
* La versión final del proyecto [SABRE](#AcronimoSABRE) en 1962 se basaba en un sistema operativo que se ejecutaba en un servidor central y que se accedía a través de terminales de usuario. Estas terminales contaban con un *bios*, un monitor monocromático, un teclado y dos puertos serie para comunicarse con una impresora y un modem.
* [SABRE](#AcronimoSABRE) fue diseñado para que American Airlines fuese la única compañía capaz de usarla. Sin embargo con el tiempo el sistema fue vendido a otras compañías.
* Dada la ausencia de internet, cada día todas las aerolíneas tenían que pagar a los [CRS](#AcronimoCRS)s para ser incluidos en sus bases de datos.
* En 1971 United Airlines crearon un nuevo [CRS](#AcronimoCRS) llamado Apollo.
* Con el tiempo, algunas agencias de viajes recibían comisiones por parte de los [CRS](#AcronimoCRS)s dependiendo de cuantas reservas vendían. Esto complicó la relación comercial entre agencias y aerolíneas, provocando que estas últimas crearan sus propios sistemas: Worldspan y Galileo.
* En 1987, Air France, Lufthansa, Iberia y Scandinavian Airlines crean juntas un nuevo sistema, Amadeus que, además de la gestión del inventario de vuelos que hace un [CRS](#AcronimoCRS), se encargaba también de la distribución.
* [SABRE](#AcronimoSABRE) en los 90’ y más tarde el [GDS](#AcronimoGDS) Travelport, formado por los antiguos CRSs Apollo, Galileo y Worldspan, se transformaron con la llegada de internet. Los agentes de viaje a través de los terminales, se cambiaron por webservices.
* Las instrucciones para realizar reservas que se han venido utilizando y siguen utilizándose a día de hoy, han sido traducidas a funcionalidades implementadas en los webservices de cada uno de los [GDS](#AcronimoGDS)s.

### 1.1.5 Conceptos y terminología del GDS Amadeus

En este apartado se introducen algunos de los conceptos que han formado el [GDS](#AcronimoGDS) Amadeus desde su nacimiento y que, a pesar de la evolución tecnológica, siguen siendo parte funcional de estos sistemas.

Ya que cada [GDS](#AcronimoGDS) usa terminología propia, para acotar los conceptos definidos a continuación, se toman solamente las denominaciones bautizadas y utilizadas por Amadeus. Cómo ya se ha anticipado en capítulos anteriores, Amadeus el [GDS](#AcronimoGDS) con el que se trabajado para desarrollar la modificación de reservas que da título a este trabajo.

Algunos conceptos, aunque fueron acuñados por los [GDS](#AcronimoGDS)s, hoy en día se usan también para definir términos más generales, relacionados con los vuelos, pasajeros o reservas aéreas.

#### Segmento

Recorrido que realiza un avión de pasajeros, operado por una única compañía aérea, de un punto *A* a un punto *B*. Salvo que el avión, durante este recorrido, tenga que realizar una parada técnica prevista o una parada de emergencia, se considera que el segmento tiene un único número de vuelo y unas fechas de salida y llegada.

Ejemplo de segmento:

Palma – Madrid con Iberia, número de vuelo 7610 con fecha de salida 2018-10-17T12:15:00 y fecha prevista de llegada 2018-10-17T13:20:00.

#### Escala

Una escala es un punto de conexión entre dos segmentos.

Ejemplo de escala:

Palma – Barcelona con Iberia, número de vuelo 7611 con fecha de salida 2018-10-17T10:30:00 y fecha prevista de llegada 2018-10-17T11:15:00.

Barcelona – Madrid con Iberia, número de vuelo 7612 con fecha de salida 2018-10-17T11:40:00 y fecha prevista de llegada 2018-10-17T12:25:00.

Barcelona en este ejemplo es una escala.

#### Conexión

Segmento que conecta un punto *B* con un punto *C* cuando la ruta/itinerario vendido está establecido en la reserva como un recorrido de *A* a *C* de forma continuada. El punto *B* se considera una escala.

En el ejemplo anterior, el segmento Barcelona – Madrid es una conexión.

#### Viaje sólo de ida (One way o [OW](#AcronimoOW))

Los itinerarios de ida sola contienen viajes entre dos puntos vía una ruta específica, pero el viaje no regresa al punto de origen: [Figura 7](#Figura7).

Cada una de las líneas de la [Figura 7](#Figura7) representa gráficamente un segmento.

#### Viaje de ida y vuelta (Round Trip o [RT](#AcronimoRT))

Los itinerarios de los viajes de ida y vuelta contienen viajes que retornan al punto de origen: [Figura 8](#Figura8).

#### Viaje circular (Circle Trip o [CT](#AcronimoCT))

Un viaje circular retorna al punto de origen pasando antes n puntos de conexión: [Figura 9](#Figura9). Un viaje circular no usa la misma ruta en ambas direcciones.

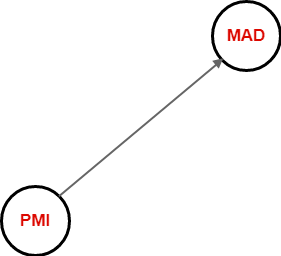
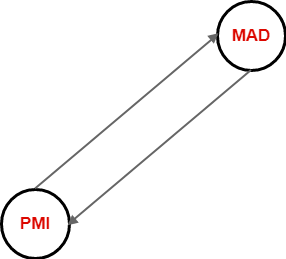
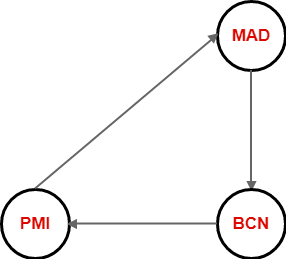
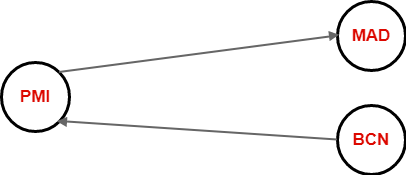
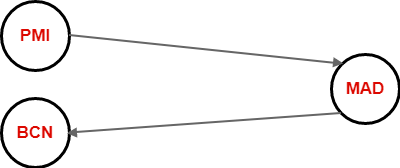
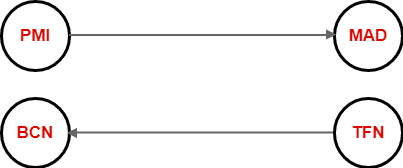


Figura 12: Open Jaw doble

Figura 11: Open Jaw simple en el origen

Figura 10: Open Jaw simple en el destino

Figura 9: Circle Trip

Figura 8: Round Trip

Figura 7: One Way Trip

#### Open Jaw simple ([OJ](#AcronimoOJ) simple) - en el destino

El punto de partida del retorno no coincide con el punto de llegada de la ida (open jaw en el destino): [Figura 10](#Figura10).

#### Open Jaw simple ([OJ](#AcronimoOJ) simple) - en el origen

El punto de salida de la partida no coincide con el punto de llegada del retorno (open jaw en el origen): [Figura 11](#Figura11).

#### Open Jaw doble ([OJ](#AcronimoOJ) doble)

El punto de salida de la partida no coincide con el punto de llegada del retorno ni el punto de salida del arribo no coincide con el punto de llegada de la partida: [Figura 12](#Figura12).

#### Passenger Name Record ([PNR](#AcronimoPNR))

El [PNR](#AcronimoPNR) es el registro de toda la información que hace referencia a una reserva. En este registro se introducen pasajeros, segmentos, billetes, información de contacto y forma de pago del cliente, así como maletas o asientos pre-reservados extra. Amadeus introduce automáticamente, información adicional para constatar que el [PNR](#AcronimoPNR) se ha creado correctamente.

Hay 7 elementos obligatorios que deben añadirse al [PNR](#AcronimoPNR):

* Vuelos seleccionados (número de vuelo, origen y destino y fechas).
* Nombres de los pasajeros.
* Forma de pago.
* Información de contacto del cliente.
* Fecha hasta la cual será válida una reserva (last ticketing date o [LTD](#AcronimoLTD)). Si no se ha abonado el importe de la reserva, ésta se cancela.
* Compañía aérea validadora.
* Comisión de la compañía aérea. Normalmente es un elemento que Amadeus introduce de forma automática.

Un [PNR](#AcronimoPNR) se mantiene activo hasta 4 días después de la fecha del último segmento aéreo y una vez superada ésta, se mantiene archivado en Amadeus para su posible reclamación durante los siguientes 3 años.

#### Localizador

Una vez introducido los campos obligatorios en el [PNR](#AcronimoPNR) y guardarse este en el sistema de distribución de Amadeus, se crea un localizador único para recuperar y poder así acceder de nuevo a la información previamente introducida en el [PNR](#AcronimoPNR). De esta forma puede seguirse introduciendo información o modificarse información ya existente.

#### Cabina

Comúnmente llamada también clase cabina. Las cabinas se separan para diferenciar las ventajas en los servicios que vende la compañía para un segmento. Entre los servicios más habituales se incluyen: asientos especiales, comidas, zonas de silencio, etc. Las cabinas más habituales vendidas por las compañías son:

**Y** – Económica o turista

**C** – Club o Business

**F** – Primera

#### Clase

Término que habitualmente se confunde con la cabina. Un segmento con dos clases distintas, no se distingue uno del otro por la calidad del servicio vendido por parte de la compañía, como si ocurre con la cabina.

El número máximo de asientos del avión se separa en clases (normalmente de 9 en 9 pasajeros). Algunas compañías separan estas clases en precios, que van subiendo o bajando a medida que los asientos del avión se van vendiendo.

#### Tarifa

Una tarifa está formada por los siguientes elementos:

* Uno o más segmentos
* Cabina (una por segmento)
* Clase (una por segmento)

Dependiendo del número y tipo de elementos que una tarifa incluya, ésta constará de un precio u otro.

#### Tipo de pasajero

Amadeus separa los pasajeros en tres tipos dependiendo de unos rangos de edad:

* Adultos ([ADT](#AcronimoADT)): Más de 12 años. Por norma general tiene que introducirse al menos un adulto en un PNR para que este sea válido.
* Niños ([CHD](#AcronimoCHD)): Entre 12 y 1 año.
* Bebés ([INF](#AcronimoINF)): Entre 0 y 1 año. Un bebé tiene que ir acompañado por un adulto.

#### Billete

Identificador asociado a un pasajero o elemento extra, como maleta o asiento, y que controla su emisión una única compañía: la compañía validadora o validating carrier. Cada compañía tiene unos dígitos de control para que el agente o cliente que realice la reserva pueda identificar de qué compañía aérea es el billete. Un billete puede ser de un máximo de 4 segmentos. Si la reserva tiene más de 4 segmentos, Amadeus crea un billete más para cada pasajero.

#### Emisión o Ticketing

Es el proceso a través del cual Amadeus abona el importe de la reserva a la compañía y ésta genera los billetes.

### 1.1.6 Web Services y WSDL

En este apartado se explica de forma breve que es un webservice y para qué sirve un [WSDL](#AcronimoWSDL).

Este apartado guarda una estrecha relación con el apartado [1.1.1](#_1.1.1_Funcionamiento_de) y permitirá al lector extender el conocimiento aportado en el primer capítulo de la contextualización.

Un servicio web o webservice describe una forma estandarizada de integrar varias aplicaciones o programas mediante el uso de estándares y protocolos web. Los webservices permiten a las organizaciones intercambiar datos sin necesidad de conocer los detalles de sus respectivos sistemas de información.

Los estándares y protocolos establecidos en un webservice, permiten a las distintas aplicaciones de software desarrolladas en lenguajes de programación diferentes, y ejecutadas sobre cualquier plataforma, intercambiar información.

[WSDL](#AcronimoWSDL) es un protocolo basado en [XML](#AcronimoXML) que describe los accesos al webservice. De una forma práctica es el manual de operación del webservice, ya que describe cuáles son las interfaces que provee éste y que tipos de datos necesitan las aplicaciones que inter-operarán para utilizarlo.

[WSDL](#AcronimoWSDL) es el lenguaje propuesto por el [W3C](#AcronimoW3C) para la descripción de servicios web y permite describir la interfaz de un servicio web en un formato [XML](#AcronimoXML). El [WSDL](#AcronimoWSDL) describe los servicios web a través de los mensajes que se intercambian entre el proveedor del servicio y el cliente.

### 1.1.7 Estado actual de los webservices de Amadeus

En este apartado se introducen cuáles son los principales webservices que ofrece Amadeus en la actualidad y qué utilidades y funcionalidades nos ofrecen cada uno de ellos.

Los webservices que se explican a continuación y algunas de sus variantes ya han sido utilizados en la integración de Amadeus de TravelgateX antes de realizar este trabajo. Algunos de ellos vuelven a utilizarse durante y para el desarrollo de este trabajo, por lo que se ha considerado conveniente explicar al lector qué función desempeña cada uno de ellos.

#### MasterPricer Search

Es el webservice que se utiliza para obtener la disponibilidad de los vuelos. La información base que ha de solicitarse obligatoriamente para obtener las tarifas es:

* Orígenes y destinos
* Fechas
* Pasajeros
* Número máximo de tarifas a devolver

El resultado obtenido es un conjunto de tarifas, apodadas *recommendations* y un conjunto de segmentos. En cada tarifa se indica qué segmentos de los incluidos en la respuesta pueden elegirse.

#### Informative Pricing (también con opción Best)

Permite la cotización de las tarifas devueltas en [MasterPricer Search](#MASTERPRICER). Existen tres versiones del webservice, cada una de ellas sirve con un propósito:

* Cotizar con [PNR](#AcronimoPNR): Cotizar un vuelo ya reservado, utilizado para re-cotizar el vuelo y obtener así un precio más económico con el que decidir luego si se desea modificar el PNR con el nuevo vuelo.
* Cotizar sin [PNR](#AcronimoPNR): Es la opción más utilizada puesto que es necesaria para obtener la cotización que luego ha de reservarse y crear así el [PNR](#AcronimoPNR). Ya que no existe aún el [PNR](#AcronimoPNR), la cantidad de pasajeros, forma de pago e itinerario, tiene que proporcionarse en la petición.
* Cotizar refrescando la disponibilidad: Antes de cotizar, se vuelve a lanzar internamente una disponibilidad para obtener así vuelos más baratos en una clase distinta.

#### Air Sell

Web service diseñado para comprar las tarifas obtenidas en [MasterPricer](#MASTERPRICER). Es una llamada necesaria y obligatoria que sirve como paso previo a la inclusión de los vuelos al [PNR](#AcronimoPNR). Una vez ejecutada la llamada, Amadeus pre-reserva la tarifa y elimina la plaza libre de la clase.

#### PNR Add Multi Elements

Web service esencial para añadir cualquier información de la reserva en el [PNR](#AcronimoPNR). Además de la introducción de datos, este webservice permite realizar el *commit* de los cambios realizados en el [PNR](#AcronimoPNR).

#### PNR Retrieve

Permite recuperar el [PNR](#AcronimoPNR) a través, por ejemplo, del localizador que Amadeus proporciona al crear la reserva.

#### PNR Cancel

Permite eliminar elementos guardados en el [PNR](#AcronimoPNR).

#### Ticket Process

Nomenclatura que engloba los webservices para realizar la emisión de billetes de pasajeros y elementos extra como maletas o asientos. En la respuesta se proporcionan los números de billete emitidos.

### 1.1.8 Estado previo de la API de transportes de TravelgateX

A continuación se destacan los elementos y atributos más importantes de algunas de las transacciones que incluye la [API](#AcronimoAPI) de transportes de TravelgateX. Los siguientes elementos y atributos (representados con una @) representan el trabajo realizado con anterioridad en la empresa.

Dado que se pretende presentar las transacciones de forma introductoria, la mayoría de elementos descritos en las siguientes tablas no se corresponden con los que realmente existen en la [API](#AcronimoAPI), puesto que su descripción sería demasiado extensa para el propósito que se busca en este apartado.

Las filas sombreadas de la tabla representan los elementos o atributos incluidos en la raíz de las peticiones/respuestas.

Tabla 1: Petición de transacción Disponibilidad: DisponibilidadRQ

|  |  |  |
| --- | --- | --- |
| **Elemento** | **Cardinalidad** | **Significado** |
| tramosDisponibilidad | 1..1 | Contiene una lista de tramos |
| tramoDisponibilidad | 1..n | Contiene la información necesaria del itinerario para realizar una búsqueda |
| locOrigen | 1..1 | Localización del origen |
| locDestino | 1..1 | Localización del destino |
| @fechaSalida | 1..1 | Fecha salida del origen |
| @fechaLlegada | 1..1 | Fecha llegada al destino |
| pasajerosDisponibilidad | 1..1 | Contiene una lista de pasajeros |
| pasajeroDisponibilidad | 1..n | Contiene la edad del pasajero |
| @edad | 1..1 | Edad del pasajero |
| preferencias | 1..1 | Contiene un conjunto de atributos que acota la predilección del cliente en una búsqueda |
| @claseCabina | 0..1 | Clase cabina preferida |
| @incluirLowcost | 0..1 | Incluir compañías lowcost en la búsqueda |
| @tipoViaje | 1..1 | Tipo de viaje de la búsqueda: [OW](#AcronimoOW), [RT](#AcronimoRT), [OJ](#AcronimoOJ) o [CT](#AcronimoCT) |

Tabla 2: Respuesta de transacción Disponibilidad: DisponibilidadRS

|  |  |  |
| --- | --- | --- |
| **Elemento** | **Cardinalidad** | **Significado** |
| segmentos | 1..1 | Lista de segmentos |
| segmento | 1..n | Contiene información de un vuelo |
| @numTransporte | 1..1 | Identificador del vuelo. Ejemplo: IB6622 |
| @operatingCia | 1..1 | Compañía aérea que opera el vuelo |
| @marketingCia | 1..1 | Compañía aérea que vende el vuelo |
| locOrigen | 1..1 | Localizacion del origen |
| locDestino | 1..1 | Localización del destino |
| @fechaSalida | 1..1 | Fecha y hora de la salida del origen |
| @fechaLlegada | 1..1 | Fecha y hora de la llegada al destino |
| @terminalSalida | 0..1 | Terminal de salida del origen |
| @terminalLlegada | 0..1 | Terminal de llegada al destino |
| tarifas | 1..1 | Contiene una lista de tarifas |
| tarifa | 1..n | Contiene información de una tarifa |
| condiciones | 1..n | Contiene una lista de condiciones de una tarifa: penalizaciones, reembolsable si/no… |
| desgloseImporte | 1..1 | Contiene los importes desglosados por tipo de pasajero: [ADT](#AcronimoADT), [CHD](#AcronimoCHD) y [INF](#AcronimoINF) |
| opciones | 1..1 | Contiene una lista de opciones |
| opcion | 1..n | Opción a cotizar que podrá elegir el cliente dentro de una tarifa. Cada opción corresponderá a uno de los tramosDisponibilidad de la petición |
| @refTramoDisponibilidad | 1..1 | Referencia al tramoDisponibilidad de la petición |
| @numEscalas | 1..1 | Cantidad de escalas |
| referenciasSegmentos | 1..n | Contiene referencias (id’s) de los segmentos incluidos en la lista de segmentos de DisponibilidadRS |
| equipajesIncluidos | 1..1 | Tipo y cantidad de equipajes que incluye la tarifa gratuitamente |
| configuracionesPasajeros | 1..n | Cantidad de pasajeros desglosados por tipo de pasajero: [ADT](#AcronimoADT), [CHD](#AcronimoCHD) y [INF](#AcronimoINF) |
| @tieneCargosPorTarjeta | 1..1 | Indica si la tarifa implica el abono de cargos adicionales si se paga con tarjeta |

Tabla 3: Petición de transacción Valoracion: ValoracionRQ

|  |  |  |
| --- | --- | --- |
| **Elemento** | **Cardinalidad** | **Significado** |
| preferencias | 1..n | Conjunto de preferencias para solicitar la disponibilidad de equipajes, asientos o extras |
| desgloses | 1..1 | Contiene una lista de desgloses |
| desglose | 1..n | Contiene la información que el cliente ha seleccionado en la transacción de Disponibilidad |
| @placa | 1..1 | Compañía aérea validadora del vuelo |
| tramos | 1..1 | Contiene una lista de tramos |
| tramo |  | Vuelo/s correspondiente al tramoDisponibilidad solicitado en la transacción de Disponiblidad |
| segmentosTramo | 1..1 | Contiene una lista de segmentosTramo |
| segmentoTramo | 1..n | Contiene información del vuelo y de la tarifa seleccionada en la transacción de Disponibilidad |
| segmento | 1..1 | Información del vuelo: fechas, duración, origen y destino… |
| clasesSegmento | 1..1 | Información de tarifa: claseCabina, asientos disponibles por clase… |
| desgloseImporte | 1..1 | Contiene los importes desglosados por tipo de pasajero: [ADT](#AcronimoADT), [CHD](#AcronimoCHD) y [INF](#AcronimoINF) |
| configuracionesPasajeros | 1..n | Cantidad de pasajeros desglosados por tipo de pasajero: [ADT](#AcronimoADT), [CHD](#AcronimoCHD) y [INF](#AcronimoINF) |

Tabla 4: Respuesta de transacción Valoracion: ValoracionRS

|  |  |  |
| --- | --- | --- |
| **Elemento** | **Cardinalidad** | **Significado** |
| desgloses | 1..1 | Información actualizada de cada uno de los desgloses solicitados en la petición |
| suplementos | 0..1 | Contiene los suplementos disponibles solicitados por el cliente a través de las preferencias de la petición ValoracionRQ |
| equipajes | 1..n | Lista de equipajes: cada equipaje tendrá unas características, precio y podrá reservarse para unos segmentos y pasajeros determinados por el proveedor |
| suplementosEspeciales | 1..n | Lista de suplementos especiales: cada suplemento tendrá unas características, precio y podrá reservarse para unos segmentos y pasajeros determinados por el proveedor |
| asientos | 1..n | Lista de asientos: cada asiento tendrá unas características, precio y podrá reservarse para unos segmentos y pasajeros determinados por el proveedor |

Tabla 5: Petición de transacción Reserva: ReservaRQ

|  |  |  |
| --- | --- | --- |
| **Elemento** | **Cardinalidad** | **Significado** |
| pasajeros | 1..1 | Contiene una lista de pasajeos |
| pasajero | 1..n | Contiene información del pasajero facilitada por el cliente una vez que ha decidido confirmar la reserva |
| @nombre | 1..1 | Nombre del pasajero |
| @apellidos | 1..1 | Apellidos del pasajero |
| @fechaNacimiento | 1..1 | Fecha de nacimiento del pasajero |
| @tipoDocumento | 1..1 | Documento de identidad del pasajero |
| @idDocumento | 1..1 | Tipo de documento de identidad del pasajero: Pasaporte, dni, dni extranjero… |
| @caducidadDocumento | 1..1 | Fecha de caducidad del documento de identidad del pasajero |
| @nacionalidad | 1..1 | Nacionalidad del pasajero |
| peticionesEspeciales | 0..n | Conjunto de peticiones (equipajes, asientos o suplementos especiales) que solicita el cliente del conjunto de disponibles que ha arrojado la transacción de Valoracion |
| cliente | 1..1 | Información de contacto del cliente que confirma la reserva |
| @nombre | 1..1 | Nombre del cliente |
| @apellidos | 1..1 | Apellidos del cliente |
| @email | 1..1 | Email del cliente |
| @telefono | 1..1 | Teléfono de contacto del cliente |
| @direccion | 1..1 | Dirección del cliente |
| infoPago | 1..1 | Información del pago que realizará el cliente: cash o tarjeta. Si se indica el cliente, se ha de proporcionar el número y tipo de tarjeta |
| @deltaPrice | 0..1 | Subida del precio que está dispuesto a asumir el cliente (la agencia) con respecto al precio que arroje la transacción de Valoracion |
| desgloses | 1..1 | Desgloses obtenidos en la transacción de Valoracion |

Tabla 6: Respuesta de transacción Reserva: ReservaRS

|  |  |  |
| --- | --- | --- |
| **Elemento** | **Cardinalidad** | **Significado** |
| localizadores | 1..1 | Contiene una lista de localizadores |
| localizador | 1..n | Localizador de la reserva con el que posteriormente podemos recuperarla para cancelarla, consultarla o emitir los billetes |
| pasajeros | 1..1 | Contiene la lista de pasajeros (la misma información proporcionada en la petición) |
| factura | 1..1 | Contiene información del desglose de importes que ha de abonarse con el método de pago especificado en la petición |
| @importeTotal | 1..1 | Cantidad total a abonar para confirmar la reserva |
| desglosesPasajeros | 1..n | Lista de importes desglosados por tipo de pasajero: [ADT](#AcronimoADT), [CHD](#AcronimoCHD) y [INF](#AcronimoINF) |
| cargos | 1..n | Lista de importes de cargo por equipajes, suplementos especiales y asientos extra |
| @importeTasas | 1..1 | Cantidad del importe total destinado a tasas |

Tabla 7: Petición de transacción Emitir: EmitirRQ

|  |  |  |
| --- | --- | --- |
| **Elemento** | **Cardinalidad** | **Significado** |
| @tipoEmision | 1..1 | Tipo de emisión que se solicita: emisión de pasajero o de extras |
| localizador | 1..1 | Localizador que se necesita para recuperar la reserva y poder así emitir los billetes |

Tabla 8: Respuesta de transacción Emitir: EmitirRS

|  |  |  |
| --- | --- | --- |
| **Elemento** | **Cardinalidad** | **Significado** |
| billetes | 1..1 | Contiene lista de billetes |
| billete | 1..n | Contiene información del billete emitido |
| @numBillete | 1..1 | Identificador único del billete |
| @tipoBillete | 1..1 | Tipo de billete: pasajero o extra |
| @tipoPasajero | 1..1 | Tipo de pasajero del billete: [ADT](#AcronimoADT), [CHD](#AcronimoCHD) o [INF](#AcronimoINF) |
| @estadoBillete | 1..1 | Estado del billete: emitido, cancelado… |
| pasajeros | 1..n | Lista de pasajeros que se especificaron en la transacción de Reserva |

Tabla 9: Petición de transacción CancelaReserva: CancelaReservaRQ

|  |  |  |
| --- | --- | --- |
| **Elemento** | **Cardinalidad** | **Significado** |
| localizador | 1..1 | Localizador que se necesita para recuperar la reserva y poder cancelarla |

Tabla 10: Respuesta de transacción CancelaReserva: CancelaReservaRS

|  |  |  |
| --- | --- | --- |
| **Elemento** | **Cardinalidad** | **Significado** |
| desgloseImporte | 1..1 | Contiene los importes desglosados por tipo de pasajero: [ADT](#AcronimoADT), [CHD](#AcronimoCHD) y [INF](#AcronimoINF) |

Tabla 11: Petición de transacción RecuperaReserva: RecuperaReservaRQ

|  |  |  |
| --- | --- | --- |
| **Elemento** | **Cardinalidad** | **Significado** |
| localizador | 0..1 | Localizador que el proveedor arroja al reservar |
| @nombre | 0..1 | Nombre del cliente que ha reservado |
| @apellidos | 0..1 | Apellidos del cliente que ha reservado |
| @codigoOrigen | 0..1 | Localización de origen del primer tramo de la reserva |
| @codigoDestino | 0..1 | Localización de destino del primer tramo de la reserva |
| @fechaSalida | 0..1 | Fecha de salida del primer tramo de la reserva |
| @fechaReserva | 0..1 | Fecha en la que se realizó la reserva |

Tabla 12: Respuesta de transacción RecuperaReserva: RecuperaReservaRS

|  |  |  |
| --- | --- | --- |
| **Elemento** | **Cardinalidad** | **Significado** |
| pasajeros | 1..n | Lista de pasajeros que se especificaron en la transacción de Reserva |
| billetes | 1..1 | Contiene lista de billetes |
| desgloses | 1..1 | Desgloses obtenidos en la transacción de Valoracion |
| @estadoReserva | 1..1 | Estado en el que se encuentra la reserva: confirmada, cancelada… |

### 1.1.9 Certificación de Amadeus (proceso)

En este apartado el lector puede consultar el flujo de trabajo que Amadeus solicita al desarrollador para completar una funcionalidad que use sus webservices y certificar así el desarrollo de acuerdo a las políticas del [GDS](#AcronimoGDS).

Amadeus proporciona unas guías a alto nivel del ciclo de vida del proyecto ([Figura 13](#Figura13)) en el Kick-off en el que preferentemente tiene que acudir el desarrollador jefe de proyecto y los distintos interesados del mismo.

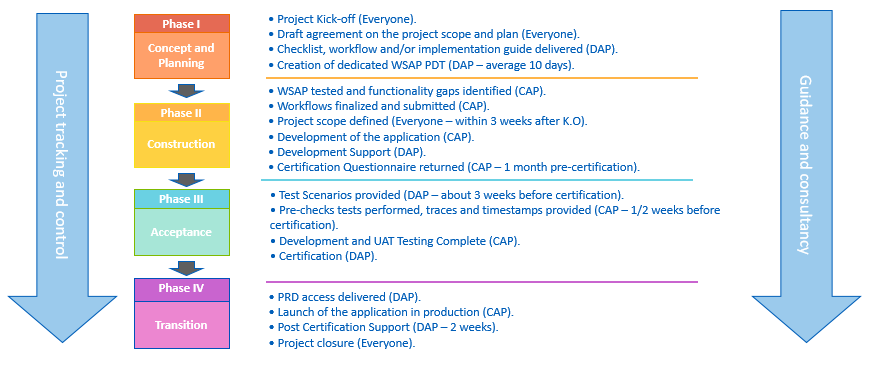


Figura 13: Ciclo de vida del proyecto de Amadeus

Amadeus aconseja seguir un flujo de transacciones que permitirá cumplir con las expectativas de un desarrollo, pero siempre siguiendo el flujo óptimo para el que fue diseñado para usarse en el [GDS](#AcronimoGDS). No tiene por qué seguirse el flujo que se indica en la documentación que Amadeus facilita para un desarrollo. Sin embargo tendrá que comunicarse a los técnicos de Amadeus el cambio y la razón del mismo con respecto al flujo recomendado.

Amadeus proporciona de forma abierta cuál será el checklist que realizará sobre el desarrollo, lo que permite al desarrollador prestar atención a qué criterios han de cumplirse para pasar la certificación. En el capítulo [2.3.1](#_2.3.1_Certificación_de) se pueden ver en profundidad estos criterios.

## 1.2 Definición del problema

En este capítulo se definen las partes existentes de una reserva aérea y la razón por la cual surge la necesidad de poder modificar dichas reservas. Finalmente se introducirá brevemente cuál ha de ser el planteamiento para dar respuesta a la necesidad.

Del conjunto de conceptos que se definirán en el apartado [1.2.1](#_1.2.1_Unidades_que), se enumerarán luego en el capítulo [1.2.3](#_1.2.3_En_qué) cuáles de ellos van a incluirse en la modificación de reservas y porqué.

En el contenido de los siguientes apartados: [1.2.1](#_1.2.1_Unidades_que) y [1.2.2](#_1.2.2_Por_qué) se menciona la existencia, previamente mencionada, del cliente o usuario final. Éste el que toma en última instancia, las decisiones en una reserva y elige el contenido de cada una de las partes de la misma.

### 1.2.1 Unidades que forman una reserva

La información que el cliente o usuario final elige y proporciona para realizar una reserva, y por lo tanto información visible para el desarrollador, puede clasificarse mediante los siguientes conjuntos:

#### Itinerario y tarifa

Cada uno de los vuelos seleccionados por el cliente final, que tendrán un origen y un destino; una compañía aérea que opera y otra compañía, normalmente la misma, que vende el vuelo. Cada uno de los vuelos ha de tener también asociada una fecha: día y hora.

En fases anteriores a la reserva: disponibilidad y valoración, el cliente ya habrá seleccionado la tarifa. Esta tarifa tendrá asociado un precio, una clase, una clase cabina y unas condiciones de tarifa: equipajes incluidos, cargos por tarjeta…

#### Pasajeros

Nombre y apellidos de los pasajeros que viajen en el avión. Dependiendo de la compañía y de los destinos solicitados en fases previas a la reserva, el cliente habrá de facilitar también el documento de identidad y/o pasaporte.

Es posible, dependiendo del destino, que se solicite también documentación para certificar la residencia de un pasajero o la pertenencia de éste a una familia numerosa.

Se conoce también la edad de cada uno de los pasajeros y, por lo tanto, se conoce ya, a qué tipo de pasajero corresponde cada uno de ellos.

Cada pasajero tendrá, desde la primera transacción de disponibilidad, una tarifa asociada. De esta forma es muy habitual que estos tres tipos de pasajero tengan precios y condiciones de tarifa distintas.

En algunas compañías es posible que los bebés [INF](#AcronimoINF) no tengan coste, mientras que en otras no exista diferencia de precio entre adultos [ADT](#AcronimoADT) y niños [CHD](#AcronimoCHD).

#### Forma de pago

El cliente o usuario final elige cómo abonar el importe de la reserva y ha de proporcionar consiguientemente la información necesaria para realizarlo. Sin embargo, Amadeus permite dos tipos de abono: tarjeta (débito o crédito) y “cash”. El pago “cash” evita que el nº de tarjeta y ccv tengan que vincularse a la reserva. De esta forma el cliente abona el importe a la agencia mediante tarjeta y es la agencia quien, a través de una cuenta corriente, abona el importe de la reserva.

No todas las compañías agregadas al [GDS](#AcronimoGDS) permiten este pago “cash”, por lo que queda a expensas del cliente del [GDS](#AcronimoGDS) conocer qué compañías lo permiten, y evitar así errores al reservar.

#### Información del cliente

El cliente final de la reserva ha de proporcionar información de contacto y de carácter general, como dirección en la que reside, correo electrónico y teléfono de contacto. El [GDS](#AcronimoGDS) pondrá a disposición de la compañía aérea esta información con el fin de facilitar la resolución de cualquier incidencia relacionada con la reserva.

#### Extras

Casi todas las compañías aéreas permiten la reserva de asientos, equipaje o suplementos extra. Es habitual que todo extra conlleve, por lo tanto, el abono de importes adicionales sobre el importe o importes de los pasajeros.

Otros datos que forman una reserva pero que el cliente no puede seleccionar:

* Fecha hasta la cual será válida una reserva [LTD](#AcronimoLTD). Si no se ha abonado el importe de la reserva, ésta se cancela.
* Compañía aérea validadora: compañía que ejecuta la emisión del vuelo, confirma que se ha abonado el importe de la reserva e imprime los billetes.
* Comisión de la compañía aérea. Normalmente es un elemento que Amadeus introduce de forma automática.

### 1.2.2 Por qué surge la necesidad de modificar una reserva

Desde el momento en el que un cliente realiza una reserva, existe la posibilidad de que éste no pueda llevarla a término o aprovecharla de la forma en que ha sido planificada. En ocasiones un cliente puede tener la necesidad de cancelar por completo el viaje o en otras simplemente modificar la fecha de uno de los segmentos.

Dependiendo de la compañía/s aéreas que vendan el vuelo, una reserva es, desde el momento en que se vende, desde 100% reembolsable hasta no reembolsable. Muchas agencias acaban corriendo con los gastos de cancelación de muchas de las reservas, para atraer y facilitar así la venta de billetes.

Una alternativa a esta estrategia es la modificación de reservas. Muchas de las modificaciones que se plantean en el siguiente apartado son totalmente gratuitas. Otras conllevan penalizaciones que, puede compensar o no los gastos de cancelación originados y la creación de una nueva reserva con el cambio deseado. La estrategia para afrontar la viabilidad y compensación de cada caso corre a cargo de la agencia y por lo tanto queda fuera del alcance de este proyecto.

### 1.2.3 En qué consiste la modificación de una reserva

Todos los datos mencionados en el capítulo [1.2.1](#_1.2.1_Unidades_que) pueden ser modificados a través del [GDS](#AcronimoGDS) Amadeus una vez que estos han sido guardados en el [PNR](#AcronimoPNR).

Es especialmente sencillo modificar nombres de pasajeros, pasaportes e información de contacto ya que si quisiéramos podríamos hacerlo con apenas dos transacciones: [PNR\_Cancel](#PNRCANCEL) para cancelar uno de los datos y [PNR\_AddMultiElements](#PNRADDMULTIELEMENTS) para añadir el nuevo dato.

El conjunto de elementos: Itinerario y tarifa, representan una mayor complejidad de modificación, al mismo tiempo que implican un mayor beneficio para el cliente. En este proyecto se ha definido como meta la modificación del itinerario y la tarifa de una reserva.

El [GDS](#AcronimoGDS) Amadeus permite la modificación completa de un itinerario. A continuación se ejemplifican los cambios que podrán realizarse:

*Itinerario de la reserva:*

#### Segmento 1

Palma – Madrid con Iberia, número de vuelo 7610 con fecha de salida 2018-10-17T12:15:00 y fecha prevista de llegada 2018-10-17T13:20:00

Clase cabina: Y (turista)

#### Segmento 2

Madrid - Palma con Air Europa, número de vuelo 0085 con fecha de salida 2018-10-25T10:00:00 y fecha prevista de llegada 2018-10-25T11:15:00

Clase cabina: F (primera)

*Ejemplos de modificación:*

* El cliente quiere cambiar la fecha de la ida, para salir de Palma el 18 de Octubre.
* El cliente quiere eliminar la vuelta de la reserva.
* El cliente quiere añadir un nuevo segmento, sin modificar ninguno de los ya existentes.
* El cliente quiere cambiar la clase cabina de la ida de **Y** a **F**.
* El cliente no quiere volar con Iberia en la ida, pero quiere mantener los horarios y la clase cabina.

El primer paso entonces para poder empezar una modificación es conocer qué quiere cambiar el cliente e indicárselo así al [GDS](#AcronimoGDS) para que éste arroje las opciones disponibles que el cliente podrá luego elegir.

En los siguientes apartados se desglosarán, a alto nivel, cuáles son los pasos de una modificación ([1.3.1](#_1.3.1._Idea_básica)) y todo el trabajo que ha de realizarse para hacer cada uno de estos pasos posibles ([1.3.2](#_1.3.2._Qué_tiene)). Además se verá un avance de los resultados que se obtuvieron en la certificación de Amadeus.

## 1.3. Respuesta al problema

Este capítulo servirá para visualizar qué ha de desarrollarse para dar respuesta a la necesidad del cliente Logitravel: modificar reservas aéreas. Para ello se debe primero, analizar y desglosar el trabajo en varios pasos ([1.3.1](#_1.3.1._Idea_básica)). Estos pasos tendrán que adaptarse a las exigencias de Logitravel y a las exigencias del proveedor Amadeus.

En el capítulo [1.3.2](#_1.3.2._Qué_tiene) se detallará el desarrollo que ha sido necesario. Éste abarca cuatro distintas áreas: [API](#AcronimoAPI) de transportes de TravelgateX ([1.3.2.1](#_1.3.2.1._Nuevas_transacciones)), Webservices de Amadeus ([1.3.2.2](#_1.3.2.2_Flujo_de)), integración entre las transacciones de Travelgate y Amadeus ([1.3.2.3](#_1.3.2.3_Integración_a)), y finalmente el formulario de pruebas y certificación ([1.3.2.4](#_1.3.2.4_Formulario_de)).

Por último ([1.3.3](#_1.3.3_Certificación_de)) se introducirá más en detalle el proceso de certificación del desarrollo por parte de Amadeus y se hará un avance de los resultados obtenidos.

### 1.3.1. Idea básica del flujo implementado para modificar reservas

Como se ha adelantado en el apartado [1.2.3](#_1.2.3_En_qué), para poder empezar una modificación, se necesita conocer un conjunto de información que indique qué reserva quiere modificarse y qué partes de ésta quieren modificarse. Pese a que en los capítulos [3](#_Análisis), [4](#_Diseño) y [5](#_Programación) de análisis, diseño y programación, se profundizará en todos los elementos necesarios para empezar, a continuación se realizará un avance de los mismos para pre visualizar ahora a alto nivel qué se necesita:

* Localizador de la reserva para poder recuperarla y reabrir el [PNR](#AcronimoPNR)
* Vuelos reservados y cuáles de ellos quieren modificarse, o en su defecto, qué vuelos quieren añadirse.
* Otras preferencias del cliente como:
* Prefiero mantener la misma compañía
* Prefiero mantener la clase cabina

No todas las reservas serán modificables. Hay una serie de factores, algunos controlables, que nos impedirán realizar la modificación y por lo tanto, rechazarán ésta nada más empezar el flujo. Por consiguiente el cliente será informado de ello.

Un requisito esencial para poder modificar una reserva es la existencia de billetes y, por lo tanto, que se haya realizado la emisión y consiguiente abono del importe de la reserva. Otros factores como rutas o aerolíneas concretas serán un impedimento para modificar una reserva. Existe también la posibilidad que el cambio en la reserva solicitado por el cliente no tenga disponibilidad, es decir, la disponibilidad que se pida no obtenga ningún resultado para una fecha o un itinerario determinado.

La primera fase entonces sería recuperar la reserva y pedir disponibilidad con estos nuevos datos. Si todo va bien y se puede modificar la reserva, se obtendría la disponibilidad de los vuelos y los nuevos requisitos solicitados por el cliente.

De la lista de opciones que se mostrarán al cliente, éste deberá elegir una. De cara al cliente final, ésta será casi la única decisión que deberá tomar. Como paso opcional, el cliente deberá ser capaz de modificar también su forma de pago.

La siguiente fase deberá ser la reserva de esta nueva opción seleccionada por el cliente y, si se ha solicitado, la modificación de la forma de pago. En el apartado [1.3.2.2](#_1.3.2.2_Flujo_de) se introducirá más en detalle qué pasos forman esta reserva.

Finalmente se pasarían a emitir los nuevos billetes. También se detallará la lógica y abanico de casos que pueden darse en ésta la última y quizá más compleja fase de las tres que forman esta primera idea de desarrollo.

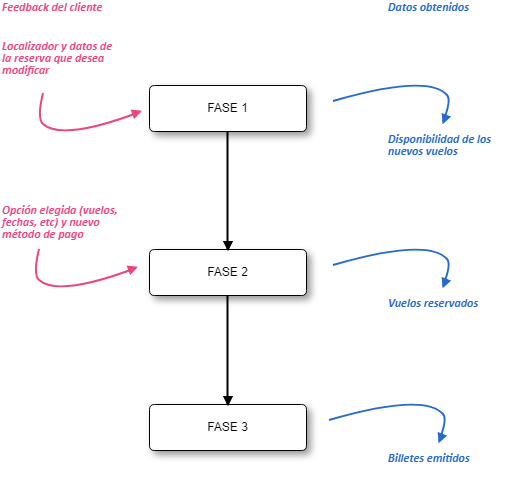
En la siguiente [Figura 14](#Figura14) se esquematiza y resume la idea definida en este capítulo:

Figura 14: Esquema de la aplicación Modificar Reservas

### 1.3.2. Qué tiene que desarrollarse para que una agencia pueda modificar reservas de Amadeus a través de TravelgateX

Este apartado abarca la introducción de las cuatro áreas que se analizarán, diseñarán y desarrollarán en este proyecto. Los dos primeros capítulos [1.3.2.1](#_1.3.2.1._Nuevas_transacciones) y [1.3.2.2](#_1.3.2.2_Flujo_de) introducen los diagramas de flujo de la integración, tanto por la parte del cliente como por la parte del proveedor. En el siguiente apartado [1.3.2.3](#_1.3.2.3_Integración_a) se introducirán las claves de la integración: programación, transformación de información del proveedor, control de errores, tecnología que se usará, formato del código y funciones más destacadas. Finalmente, en el capítulo [1.3.2.4](#_1.3.2.4_Formulario_de) se introducirá brevemente qué clase de formulario interesa tener para realizar pruebas al mismo tiempo que se desarrolla la integración.

### 1.3.2.1. Nuevas transacciones a desarrollar para la API de transportes

En el capítulo [1.1.8](#_1.1.8_Estado_previo) se han destacado las principales transacciones y elementos/atributos que éstas incluyen en su estado previo a este trabajo. Dado que el contenido de la [API](#AcronimoAPI) no permitía un flujo adaptado para la modificación de reservas, surge la necesidad de construir nuevas transacciones, de tal forma que el cliente pueda enviar la información (en una [RQ](#AcronimoRQ)) y la integración pueda responderle (en una [RS](#AcronimoRS)).

El diseño de las transacciones es siempre una lucha entre la adaptación al proveedor y la adaptación al resto de transacciones ya existentes en la [API](#AcronimoAPI) de transportes. Ha de encontrarse un equilibrio en la ecuación de tal forma que sea posible realizar el flujo que el proveedor recomienda, y al mismo tiempo asegurar la continuidad de los elementos clave que la [API](#AcronimoAPI) de transportes ya tiene. Ya que la modificación de reservas puede en un futuro desarrollarse con más proveedores, el flujo ha de ser también lo más genérico posible.

El análisis de todas estas restricciones y pautas, conduce al diseño de las siguientes transacciones y a cambios en algunos de los elementos ya existentes.

#### DisponibilidadModificarReserva ([DMR](#AcronimoDMR)): petición [DMR\_RQ](#AcronimoDMR_RQ) y respuesta [DMR\_RS](#AcronimoDMR_RS)

En esta transacción el cliente proporcionaría el localizador de la reserva y los tramos de la reserva, indicando cuál es la modificación a realizar en cada uno de ellos. Por lo tanto, surge la necesidad de introducir un nuevo concepto en los tramosDisponibilidad ya existentes:

Tabla 13: Elemento tramoDisponibilidad de la petición DMR\_RQ

|  |  |  |
| --- | --- | --- |
| **Elemento** | **Cardinalidad** | **Significado** |
| tramoDisponibilidad | 1..n | Contiene la información necesaria del itinerario para realizar una búsqueda |
| locOrigen | 1..1 | Localizacion del origen |
| locDestino | 1..1 | Localización del destino |
| @fechaSalida | 1..1 | Fecha salida del origen |
| @fechaLlegada | 1..1 | Fecha llegada al destino |
| @actionTramo | 1..1 | Modificación a realizar en el tramo: [N](#ModifN) (ninguna), [KF](#ModifKF) (mantener vuelos y tarifas), [K](#ModifK) (mantener vuelos), [C](#ModifC) (cambiar), [A](#ModifA) (añadir), [R](#ModifR) (quitar) |

Por lo tanto, se aprovecha el tipo tramoDisponibilidad ya existente, añadiéndole un atributo obligatorio para la transacción [DMR](#AcronimoDMR). Este atributo tendría 6 posibles valores:

* N – ninguna – Ninguna modificación al tramo.
* KF – mantener vuelos y tarifas – Mantener los vuelos, las clases y las condiciones de tarifa a la que pertenece el vuelo.
* K – mantener vuelos – Mantener los vuelos, conservando así la compañía/s reservadas.
* C – cambiar – Se mantendría únicamente el origen y destino del tramo, permitiendo así que la compañía del vuelo cambie.
* A – añadir – Si se añade un tramo que no existía al reservar, vendría indicado siempre con esta etiqueta.
* R – quitar/eliminar – Eliminar el tramo de la reserva.

Los datos proporcionados en los tramos y el localizador permitirán realizar la búsqueda de alternativas que el cliente tendrá la capacidad de seleccionar. Los resultados obtenidos en la búsqueda serían presentados al cliente de una forma muy similar a cómo se hace en [DisponiblidadRS](#Tabla2).

En el capítulo [4 Diseño](#_Diseño) se presentan los elementos que forman esta transacción y en capítulo [5 Programación](#_Programación) se describe cada uno de ellos.

#### ReservaModificarReserva ([RMR](#AcronimoRMR)): petición [RMR\_RQ](#AcronimoRMR_RQ) y respuesta [RMR\_RS](#AcronimoRMR_RS)

Una vez que el cliente final seleccione las opciones proporcionadas en la anterior transacción, el cliente tendrá que facilitar principalmente tres elementos:

Desgloses. Construidos a partir de los segmentos y tarifas proporcionados en la respuesta de [DMR](#AcronimoDMR).

* Billetes. Billetes de la reserva original.
* Localizador. El [PNR](#AcronimoPNR) ha de recuperarse y abrirse de nuevo, por lo tanto se necesita tener acceso a él. El cliente deberá proporcionarlo en la petición de esta transacción.

En la respuesta se proporcionaría al cliente, entre otros elementos, la factura que incluiría el nuevo coste de la reserva. Dado que la nueva reserva puede ser más económica que la original, el nuevo coste incluido en la factura podrá ser negativo.

En los apartados de diseño [4.1](#_4.1_Transacciones_API) y [4.2](#_4.2_Diagramas_de), y en capítulo [5 Programación](#_Programación), se extienden los conceptos que formarán la factura incluida en esta transacción.

La gran cantidad de cambios que se le permiten al cliente sobre la reserva original y la variedad de compañías aéreas existentes en el [GDS](#AcronimoGDS), y sus diferentes políticas, complica ésta transacción. De esta forma, ésta transacción y la siguiente ([EMR](#AcronimoEMR)), abren en el flujo con el proveedor una gran cantidad de ramificaciones posibles que deberán realizarse para terminar la modificación. Dado que la reserva ([RMR](#AcronimoRMR)) y emisión ([EMR](#AcronimoEMR)) serán dos transacciones independientes, se necesitarán nodos exclusivamente creados para transferir información de la respuesta de una transacción a la petición de la siguiente.

#### EmitirModificarReserva ([EMR](#AcronimoEMR)): petición [EMR\_RQ](#AcronimoEMR_RQ) y respuesta [EMR\_RS](#AcronimoEMR_RS)

Para realizar la emisión se necesitan principalmente:

* Localizador: de nuevo ha de recuperarse y abrirse el [PNR](#AcronimoPNR), consecuentemente se necesita el localizador de la reserva.
* Billetes. Billetes de la reserva.
* Tipo de emisión @issueType. En la transacción anterior (RMR) se ha determinado qué tipo de emisión ha de realizarse.

En la respuesta se facilitará al cliente la lista de nuevos billetes.

De cara al cliente entonces, se desglosará el flujo en tres transacciones:

* DisponibilidadModificarReserva ([DMR](#AcronimoDMR))
* ReservaModificarReserva ([RMR](#AcronimoRMR))
* EmitirModificarReserva ([EMR](#AcronimoEMR))

En el siguiente capítulo se introducirá cuál será el flujo de transacciones con el proveedor.

### 1.3.2.2 Flujo de transacciones con Amadeus

El flujo a alto nivel que Amadeus recomienda para modificar una reserva se divide también en tres fases:

* Recuperación del [PNR](#AcronimoPNR) y comprobación de posibilidad de modificar reserva.
* Búsqueda de nuevas recomendaciones.
* Actualización del [PNR](#AcronimoPNR) y reemisión del billete o revalidación.

Estas fases encajarían de la siguiente forma con las transacciones introducidas en el capítulo anterior:

#### DisponiblidadModificarReserva

* Recuperación del [PNR](#AcronimoPNR) y comprobación de posibilidad de modificar reserva.
* Búsqueda de nuevas recomendaciones.

#### ReservaModificarReserva

* Actualización del [PNR](#AcronimoPNR).

#### EmitirModificarReserva

* Reemisión del billete o revalidación.

En el capítulo [4.2](#_4.2_Diagramas_de) se detallará cada una de las tres fases establecidas en el plan de proyecto de Amadeus para modificar reservas con sus servicios.

Además de algunos de los webservices vistos es el capítulo [1.1.7](#_1.1.7_Estado_actual), es necesario trabajar con los siguientes webservices:

#### Ticket\_CheckEligibility

Permite comprobar si los billetes de la reserva son candidatos para ser modificados.

#### Ticket\_ATCShopperMasterPricerTravelBoardSearch

Variación del servicio [MasterPricer](#MASTERPRICER) visto en el capítulo [1.1.7](#_1.1.7_Estado_actual) usado para la búsqueda de disponibilidad convencional. Este webservice permite sin embargo la búsqueda de tarifas con las que modificar la reserva.

#### Ticket\_RepricePNRWithBookingClass

Este servicio se usa para calcular el nuevo precio de los pasajeros e itinerario seleccionado y calcula además los costes que suponen los cambios realizados acorde a cada aerolínea.

#### Ticket\_ReissueConfirmedPricing

Confirma los cambios realizados y los escribe en el [PNR](#AcronimoPNR). Una vez confirmados los cambios, el [PNR](#AcronimoPNR) está preparado para que se emitan sus billetes.

Los webservices definidos en este apartado forman parte del Amadeus Ticket Changer ([ATC](#AcronimoATC)) Shopper, diseñados por el [GDS](#AcronimoGDS) para facilitar la modificación de reservas para agencias de viajes online u Online Travel Agency ([OTA](#AcronimoOTA)).

### 1.3.2.3 Integración a desarrollar

Para realizar un flujo completo de modificación, la integración realizaría entonces las tres transacciones presentadas en el capítulo [1.3.2.1](#_1.3.2.1._Nuevas_transacciones) usando las peticiones y respuestas diseñadas e incluidas en la [API](#AcronimoAPI), y realizaría las siguientes operaciones para cada una de ellas:

* El cliente enviará en las peticiones ([DMR\_RQ](#AcronimoDMR_RQ), [RMR\_RQ](#AcronimoRMR_RQ) o [EMR\_RQ](#AcronimoEMR_RQ)) los datos necesarios previamente acordados entre TravelgateX y la agencia Logitravel.
* La integración transforma la petición al conjunto de peticiones que requiere el proveedor para realizar la operación previamente acordada entre TravelgateX y Amadeus.
* A continuación la integración realiza el conjunto de peticiones al proveedor.
* El proveedor procesa la petición que le llega y envía de vuelta como respuesta el producto que el cliente ha pedido.
* De nuevo, la integración transforma los datos obtenidos en las respuestas del proveedor y los coloca en cada uno de los elementos que formarán las respuestas ([DMR\_RS](#AcronimoDMR_RS), [RMR\_RS](#AcronimoRMR_RS) o [EMR\_RS](#AcronimoEMR_RS)).
* Finalmente la integración envía la respuesta al cliente.

La programación del código para modificar reservas en la integración, es la fase del proyecto que más tiempo conlleva, reducir este tiempo dependerá de los bien analizados que hayan sido los flujos con el proveedor y de buen diseño de las transacciones con el cliente.

La integración de Amadeus ha sido ya programada en el lenguaje de programación Visual Basic de la plataforma .NET, por lo tanto, todo el nuevo código deberá programarse también en este lenguaje. En el capítulo [2.2.3](#_2.2.3_Requisitos_funcionales) se enumerarán algunas de las ventajas que implica el uso de este lenguaje en el ámbito del desarrollo de integraciones.

La implementación ha de cumplir además con las pautas y exigencias que cliente y proveedor demanden. La integración ha de controlar también el tiempo de corte que establece el cliente; el control de errores y excepciones para clasificar cada una de las modificaciones fallidas que se produzcan una vez que el desarrollo esté en producción; mantener el funcionamiento del resto de transacciones en producción.

Algunas funcionalidades que habrán de implementarse, usarán funcionalidades ya existentes en la integración de Amadeus. Por este motivo, es especialmente crítico, mantener el buen funcionamiento del código existente y aprovechar al máximo el mismo.

El código a desarrollar ha de ser mantenible y legible para futuras tareas de mantenimiento que puedan llegar a realizar otros desarrolladores que trabajen en la empresa.

### 1.3.2.4 Formulario de pruebas

Otra tarea indispensable para este proyecto es la construcción de un entorno para realizar pruebas durante el desarrollo y la certificación del mismo. La construcción de las peticiones a mano imposibilitaría la completitud del desarrollo a tiempo. Por este motivo es crucial la automatización de esta tarea.

Para cumplir este objetivo se ha de desarrollar un formulario que permita realizar las siguientes peticiones de manera visual, introduciendo los datos en campos de texto y sin la necesidad de montar las peticiones [XML](#AcronimoXML) manualmente de Disponibilidad, Valoración, Reserva, Recuperar Reserva, Cancelar Reserva, Emisión, Void, DMR, RMR, EMR.

El formulario ha de permitir además lanzar las peticiones al proveedor y recibir las respuestas de manera práctica: se han de poder copiar las respuestas del proveedor y ver la lista de tarifas que obtenemos en una disponibilidad de forma gráfica.

Mediante el formulario, un cliente ha de poder completar un flujo completo de reserva y de modificación de reserva. Se han de poder ver los errores y modificar el tiempo de corte de cada una de las transacciones que se lancen.

Todo y que la finalidad del formulario es únicamente facilitar el trabajo al desarrollador de este proyecto, es importante que su uso sea intuitivo y que éste sea reutilizable y fácil de modificar y mantener. Sin embargo, su funcionamiento y requisitos anteriormente enumerados, no se rigen por ninguna de las exigencias de TravelgateX, Logitravel o Amadeus, como así sucede con la integración desarrollada.

### 1.3.3 Certificación de Amadeus (avance de los resultados)

Tras revisar los tests realizados, Amadeus informó al desarrollador de la existencia de una anomalía en el flujo de transacciones que supondrían un warning en la certificación. Amadeus recomendó la modificación de la aplicación desarrollada.

Esto supuso un retraso de unos días en el despliegue de la integración en producción para su uso comercial.

Todo y que la presencia de un warning en una certificación no impide a la agencia el uso de la aplicación desarrollada, se decidió realizar el cambio para tener el flujo de transacciones con el proveedor adecuado y evitar que la aplicación fuese catalogada por Amadeus como defectuosa.

Una vez realizados los cambios, se realizaron de nuevo las pruebas que Amadeus solicitó. Amadeus reviso de nuevo éstas y finalmente informaron el éxito de la certificación.

# Plan del proyecto

En este capítulo se definen los planes que se han realizado para llevar a cabo este trabajo. Los planes se han llevado a cabo en colaboración con las partes interesadas que participan durante el desarrollo de este trabajo y que utilizarán la modificación de reservas una vez que se despliegue en producción.

A lo largo de los apartados de este capítulo se exponen los planes de:

* [Alcance](#_2.1_Alcance_del): donde se define qué ha de realizarse en el proyecto y qué queda fuera de él. Se aporta una visión general del proyecto y se define la lista de entregables.
* [Requisitos](#_2.2_Requisitos_del): en el que se detallará el conjunto de requisitos no funcionales y funcionales del proyecto.
* [Criterios de aceptación](#_2.3_Criterios_de): políticas y criterios de las dos partes interesadas externas para aceptar el proyecto desarrollado por TravelgateX.
* [Restricciones del proyecto](#_2.4_Restricciones_del): se definen las dos circunstancias más restrictivas que van a condicionar la planificación y posterior desarrollo del trabajo.
* [Organización](#_2.5_Organización): sirve para dar un vistazo a la cultura y forma de trabajar de los empleados de TravelgateX encargados del desarrollo de integraciones.
* [Partes interesadas](#_2.6_Partes_interesadas): se enumeran a los implicados en el proyecto y se explica qué relación existe entre ellos.
* [Riesgos](#_2.7_Riesgos): se detallan los dos mayores riesgos del proyecto y el plan de gestión elaborado para controlarlos.
* [Tiempo](#_2.8_Gestión_del): se detalla la duración de las tareas del proyecto y se exponen todas ellas en un diagrama.
* [Comunicaciones](#_2.9_Plan_de): se definen cuáles serán los canales y la forma de comunicación a utilizar entre los diferentes interesados externos e internos.

## 2.1 Alcance del proyecto

La motivación de este proyecto surge de la necesidad de uno de los clientes de TravelgateX, Logitravel, de poder modificar las reservas realizadas con Amadeus. Además, la relación de negocio entre Logitravel y Amadeus, ya consolidada desde hace años, provoca que muchos de los proyectos que Amadeus saca adelante, sean una oportunidad para sus mejores clientes para mejorar y perfeccionar sus relaciones de venta. De esta forma, cuando Amadeus desarrolla un proyecto, los primeros en saberlo e interesarse en él, son las agencias y en este caso, también TravelgateX.

En la planificación establecida por Amadeus y en la que participaron agentes de negocio de Logitravel y el desarrollador encargado de los [GDS](#AcronimoGDS)s en TravelgateX, se definieron las funcionalidades existentes que puede aportar Amadeus para la necesidad de modificar reservas. En la reunión se destacaron cuales iban a ser las fases del desarrollo y las funcionalidades que deberían desarrollarse para cumplir los objetivos de la agencia. También se enumeraron y explicaron el resto de funcionalidades que podían introducirse en el proyecto si así se consideraba. A continuación se detalla cuál es el alcance y el resto de funcionalidades que en un principio quedaron fuera de éste.

Se define que un usuario ha de poder modificar el itinerario reservado mediante la elección de una opción entre las de una lista de disponibles. Para ello se expusieron dos alternativas: modificar el itinerario sabiendo la nueva fecha para la que reservar, o sin saberla. Estas dos opciones son compatibles y pueden confluir, sin embargo han de desarrollarse ambas por separado y con webservices distintos. Se estableció que la primera opción entraba en el alcance del proyecto y la segunda fuera.

Se estableció que para toda compañía aérea que permita la modificación del itinerario, la funcionalidad a desarrollar debería poder ser capaz de la modificación fuera posible. De esta forma, si una serie de aerolíneas necesitan un flujo de transacciones concretas para la modificación y otras, otro flujo, la aplicación ha de poder adaptarse a ambas y completar una modificación con éxito.

Para certificar el correcto y adecuado funcionamiento del desarrollo, Amadeus impone una serie de pruebas predefinidas en su plan de proyecto, que deberá realizar el desarrollador una vez termine la construcción de la aplicación.

Quedó fuera del propósito de la reunión de planificación, las herramientas o funcionalidades extra a desarrollar que son necesarias para llevar a cabo el proyecto. Por lo tanto, en la reunión no se definió la parte del desarrollo de cara al cliente Logitravel.

Mediante la colaboración entre Logitravel y TravelgateX se definieron los activos a desarrollar: nuevas llamadas en la [API](#AcronimoAPI) de transportes para poder modificar una reserva; código que integre los dos flujos (transacciones con Amadeus y transacciones con Logitravel). Una vez desplegado el desarrollo en producción, tal y como se define en el capítulo [7](#_Despliegue), se considera por terminado este proyecto, y por lo tanto el mantenimiento de la integración queda fuera del alcance de este proyecto.

Se incluye dentro del alcance definido internamente por TravelgateX, la creación de un formulario de pruebas que agilice el desarrollo y la certificación de Amadeus. Han de realizarse también las gestiones necesarias para permitir la visualización de estadísticas de reservas modificadas por los clientes una vez que el desarrollo esté funcionando en producción.

### 2.1.1 Entregables

La planificación del alcance y realización de los paquetes de tareas definidos en la [Figura 16](#Figura16), incluida en el capítulo [2.8 Gestión del tiempo](#_2.8_Gestión_del), conlleva la elaboración de los siguientes entregables:

Tabla 14: Listado de entregables definidos en el alcance del plan de proyecto

|  |  |
| --- | --- |
| Amadeus | ENT-01: Escenarios de test realizados para una correcta certificación por parte del [GDS](#AcronimoGDS) Amadeus |
| ENT-02: Diagramas de flujo de la funcionalidad |
| Logitravel | ENT-03: Integración de Amadeus con la funcionalidad modificar reservas |
| ENT-04: Nuevas llamadas en la [API](#AcronimoAPI) de transportes |
| TravelgateX | ENT-05: Formulario de pruebas |
| ENT-06: Documentación de los nuevos elementos incluidos en la [API](#AcronimoAPI) de transportes |
| ENT-07: Mejoras para futuros proyectos con Amadeus |
| ENT-08: Acta de reunión del Kick-Off |
| ENT-09: Conjunto de tareas en el Jira de TravelgateX |
| ENT-10: Listado de requisitos |
| ENT-11: Apuntes de mejora o cambios sobre el proyecto |

## 2.2 Requisitos del proyecto

En el apartado [2.2.1](#_2.2.1_Requisitos_de) de este capítulo se definen los requisitos de proyecto que hacen referencia a las restricciones temporales, colaboración entre interesados, hitos temporales y trabajo como desarrollador en la empresa TravelgateX. En el apartado [2.2.2](#_2.2.2_Requisitos_no) se detalla la tecnología que se usa para la construcción de las transacciones de la [API](#AcronimoAPI), para desarrollar la integración y para construir el formulario de pruebas. Finalmente se detallan a alto nivel los requisitos funcionales de producto en el apartado [2.2.3](#_2.2.3_Requisitos_funcionales).

### 2.2.1 Requisitos de proyecto

Amadeus solamente dará soporte sobre el desarrollo durante el periodo de tiempo estipulado en el plan de proyecto presentado en la reunión Kick-Off, véase [Figura 16](#Figura13). Por este motivo, todas las dudas o conceptos que se desconozcan han de solicitarse durante ese periodo.

Para la realización de la certificación, se contará con la ayuda de otro técnico de Amadeus, proporcionando este las herramientas para completar correctamente dicha certificación.

La colaboración entre los interesados externos y las herramientas a utilizar durante cada periodo del desarrollo, deberá seguir las pautas presentadas en el capítulo [2.9 Plan de gestión de comunicaciones.](#_2.9_Plan_de)

Deberán tenerse conocimientos intermedios de inglés para tratar con el proveedor Amadeus, leer y entender la documentación que éste proporciona, y programar el software de este proyecto.

Está planificado el despliegue de la aplicación en producción para el 5 de Septiembre de 2017. La reunión de cierre del proyecto está concertada para el día 11 del mismo mes. En ella deberá estar presentes el desarrollador de TravelgateX encargado del desarrollo de la aplicación, y el personal técnico y comercial designado por cliente y proveedor que haya seguido el desarrollo del proyecto.

Tal y como se detalla en el plan de proyecto de Amadeus, el desarrollador tendrá a disposición el paquete de Web Services aproximadamente diez días después de la realización del Kick-Off.

### 2.2.2 Requisitos no funcionales

Las nuevas transacciones de la [API](#AcronimoAPI) de transportes, la integración y el formulario de pruebas deberán ser programadas mediante el lenguaje Visual Basic. Dada la implementación de las transacciones y de la integración en este lenguaje, no habrá problemas con la serialización de objetos [XML](#AcronimoXML) ya que el lenguaje incorpora en sus librerías tratamiento de este tipo de objetos.

Todo nuevo contenido que se desarrolle deberá mantener la compatibilidad con el código ya existente, tanto en la [API](#AcronimoAPI) de transportes, como en la integración de Amadeus.

Las llamadas y elementos y atributos que se incluyan deberán ser multilenguaje (español e inglés) y el código deberá estar escrito en inglés.

La integración ha de mantener un control exhaustivo de las excepciones y tipificar correctamente los errores que retorne el proveedor. Además se ha de controlar la apertura y cierre de sesiones en Amadeus. Los errores producidos durante la ejecución de una transacción han de poder visualizarse también en el formulario de pruebas.

Un flujo completo de transacciones: Disponibilidad, Valoración, Reserva, Emitir, DMR, RMR y EMR ha de poder realizarse de manera gráfica (mediante el uso de botones y campos de texto).

### 2.2.3 Requisitos funcionales

El cliente introducirá el tiempo de corte, credenciales y [URL](#AcronimoURL)s antes de realizar las peticiones de cualquier transacción de la [API](#AcronimoAPI) de transportes. También podrá elegir el idioma de las peticiones y respuestas [XML](#AcronimoXML).

El cliente visualizará los errores, warnings y transacciones realizadas con el proveedor, así como el tiempo transcurrido y fecha de cada transacción.

En la transacción “Disponibilidad Modificar Reserva” ([DMR](#AcronimoDMR)), el cliente indicará el itinerario y las modificaciones que desee realizar en él. Deberá indicar también el localizador de la reserva y las preferencias de búsqueda: clase cabina, sólo vuelos directos, compañías, incluir lowcost.

En la respuesta de [DMR](#AcronimoDMR), el cliente ha de poder visualizar los segmentos y tarifas obtenidos en la búsqueda. De los segmentos, el cliente ha de poder visualizar la información de vuelo como fechas, tipo de avión, duración y terminales de entrada y salida. De las tarifas, ha de poder visualizar a qué segmentos hacen referencia y el importe y condiciones de la tarifa.

En la transacción “Reserva Modificar Reserva” ([RMR](#AcronimoRMR)), el cliente indicará de nuevo el localizador de la reserva, así como tarifa elegida, billetes y forma de pago con la que desee abonar el nuevo importe. En la respuesta ha de poder visualizar la factura y el tipo de emisión que se ha de realizar en la siguiente transacción “Emitir Modificar Reserva” ([EMR](#AcronimoEMR)).

En la petición [EMR](#AcronimoEMR), el cliente indicará de nuevo el localizador, billetes y tipo de emisión obtenida en la transacción [RMR](#AcronimoRMR). En la respuesta podrá visualizar los nuevos billetes.

El usuario del formulario de pruebas podrá indicar el tiempo de corte, credenciales y [URL](#AcronimoURL)s. El usuario podrá realizar un flujo completo de forma gráfica o pegando el [XML](#AcronimoXML) montado manualmente.

Para cada una de las transacciones, el usuario ha de poder indicar los campos obligatorios que se solicitan:

* Itinerario y pasajeros en disponibilidad.
* Opciones elegidas para realizar la valoración.
* Información de cliente, pasajeros y forma de pago en reserva.
* Localizadores y billetes en las transacciones de Recuperar Reserva, Cancelar Reserva, Emitir y Void.

En la transacción [DMR](#AcronimoDMR), el usuario ha de poder indicar el localizador de la reserva y los itinerarios y modificación que desea realizar para cada una de ellos.

Para las transacciones [RMR](#AcronimoRMR) y [EMR](#AcronimoEMR) el usuario ha de poder indicar los localizadores y billetes, así como la forma de pago, si desea cambiarla.

## 2.3 Criterios de aceptación

En este capítulo se exponen los criterios de aceptación establecidos en el plan, definidos y acordados entre el desarrollador y el proveedor Amadeus [2.3.1](#_2.3.1_Certificación_de) y entre el desarrollador y la agencia Logitravel [2.3.2](#_2.3.2_Criterios_de).

### 2.3.1 Certificación de Amadeus (criterios de aceptación)

En este capítulo se presenta el checklist de certificación por parte del [GDS](#AcronimoGDS) Amadeus, que deberá cumplir la aplicación desarrollada. En cada uno de los criterios presentados en las tablas [Tabla 15](#Tabla15), [Tabla 16](#Tabla16) y [Tabla 17](#Tabla17), se indica el webservice al aplica, la definición del requisito no funcional y de dicho criterio y cómo será tratado el incumplimiento del mismo.

Tabla 15: Criterios de aceptación generales

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Web Service (s)** | **Requisito no funcional** | **Error o warning** |
| **Criterio** |
| GEN-01 | Todos | Los servicios [XML](#AcronimoXML) implementados no deben estar deprecados | Error |
| servicios [XML](#AcronimoXML) deprecados = 0 |
| GEN-02 | Todos | La aplicación no debe crear bucles infinitos en el host de Amadeus | Error |
| bucles infinitos en el host de Amadeus = 0 |
| GEN-03 | Todos | Ha de implementarse un control de errores adecuado | Error |
| errores sin tratar <= 5 (por transacción) |
| GEN-04 | Todos | El formato de los datos y sus variables deben comprobarse antes de enviarlos a Amadeus | Error |
| formato de datos y variables enviados a Amadeus sin comprobar = 0 |
| GEN-05 | Todos | Deben evitarse transacciones innecesarias o redundantes | Warning |
| transacciones innecesarias o redundantes = 0 |
| GEN-06 | Todos | Deben usarse mensajes estructurados cuando sea posible | Warning |
| mensajes no estructurados <= 1 (por transacción) |
| GEN-07 | Todos | No deben crearse reservas duplicadas | Error |
| reservas duplicadas = 0 |

Tabla 16: Criterios de aceptación de gestión de sesión

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Web Service (s)** | **Requisito no funcional** | **Error o warning** |
| **Criterio** |
| SEM-01 | Todos | Todas las sesiones deben cerrarse adecuadamente en la aplicación | Error |
| sesiones sin cerrar = 0 |
| SEM-02 | Todos | Los tiempos de inactividad o respuesta deben ser tratados por la aplicación | Error |
| milisegundos de inactividad = 0 |
| SEM-03 | Todos | La secuencia de números de sesión debe ser incrementada y adecuadamente insertada en el encabezamiento [SOAP](#AcronimoSOAP) de la petición: 1.0, 1.1, 2.0, 2.1 | Error |
| errores de incremento de índice de sesión = 0 |

Tabla 17: Criterios de aceptación ATC

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Web Service (s)** | **Requisito no funcional** | **Error o warning** |
| **Criterio** |
| ATC-01 | Todos los servicios ATC | La aplicación debe proveer un ticket por pasajero (incluidos bebés) | Error |
| tickets por pasajero >= 1 |
| ATC-02 | ATC Shopper | Las recomendaciones (tarifas) devueltas por [ATC](#AcronimoATC) Shopper deben visualizarse correctamente en la aplicación | Error |
| recomendaciones de [ATC](#AcronimoATC) Shopper no mostradas en la aplicación = 0 |
| ATC-03 | ATC Shopper | El Web Service [Air\_SellFromRecommendation](#AIRSELL) debe usarse solamente sobre tarifas devueltas por el [ATC](#AcronimoATC) Shopper | Error |
| compra de tarifas no [ATC](#AcronimoATC) Shopper con [Air\_SellFromRecommendation](#AIRSELL) = 0 |
| ATC-04 | ATC Shopper | La aplicación debe tratar correctamente los tipos de pasajero que se envíen en las peticiones a Amadeus | Error |
| pasajeros con tipo incorrecto (no Amadeus) = 0 |
| ATC-05 | ATC Shopper | La aplicación debe devolver la lista de aerolíneas que se recibirá por parte de Amadeus en [ATC](#AcronimoATC) [Eligibility](#TICKET_ELIGIBILITY) | Error |
| aerolíneas devueltas por [ATC](#AcronimoATC) [Elegibility](#TICKET_ELIGIBILITY) no mostradas = 0 |
| ATC-06 | ATC Shopper | La aplicación debe recuperar la reserva si ésta se ha hecho a través de Amadeus y eliminar los segmentos originales una vez que estos se modifiquen | Warning |
| segmentos modificados no eliminados = 0 |
| ATC-07 | ATC Shopper | [ATC](#AcronimoATC) [Eligibility](#TICKET_ELIGIBILITY) debe ser sistemáticamente utilizado antes de cualquier servicio adicional de [ATC](#AcronimoATC) Shopper | Warning |
| servicios usados sin hacer previamente la llamada a [ATC](#AcronimoATC) [Eligibility](#TICKET_ELIGIBILITY) = 0 |

### 2.3.2 Criterios de aceptación de la agencia

En este capítulo se presentan los criterios que se acordaron entre la agencia y el desarrollador de TravelgateX para aceptar la aplicación desarrollada. No se requiere ningún tipo de certificación para aceptar o no la aplicación desarrollada, como así ocurre con el [GDS](#AcronimoGDS). Aun así, se espera que los criterios enumerados a continuación se cumplan dentro de unos límites aceptables (dentro de los márgenes que se han ido cumpliendo en todas las aplicaciones desarrolladas para la agencia).

* Control de errores y warnings siguiendo los tipos que ya se utilizan en otras transacciones.
* Control del tiempo de corte.
* Visualización de los resultados obtenidos por parte del proveedor, tratados y sin tratar.
* Visualización de la fecha en la que se ha realizado una petición.

## 2.4 Restricciones del proyecto

En este capítulo se detallan las restricciones del proyecto. Primero se detallan las restricciones de tiempo establecidas por el [GDS](#AcronimoGDS) en el apartado [2.4.1](#_2.4.1_Restricción_de) y después, en el apartado [2.4.2](#_2.4.2_Factores_ambientales:), los factores ambientales.

### 2.4.1 Restricción de tiempo

En este apartado se muestra el plan de proyecto establecido por Amadeus que restringe la duración del proyecto desarrollado por TravelgateX. Se indican los periodos de duración del proyecto y los periodos de soporte técnico preestablecidos por el [GDS](#AcronimoGDS): [Figura 13](#Figura13).

### 2.4.2 Factores ambientales: cultura del Dpto. de integraciones

TravelgateX ha desarrollado y mantiene una gran cantidad de integraciones. Esto supone el mantenimiento y actualización de muchos servicios desarrollados por ingenieros que ya no trabajan en el Dpto. de integraciones. De esta forma, es esencial que dichos desarrollos cumplan con una serie de criterios, que ayuden a su posterior mantenimiento.

Entre los requisitos más destacados que el Dpto. de integraciones propone, se encuentran:

* Desarrollar en base a funcionalidades compartidas entre [API](#AcronimoAPI)s.
* Desarrollar utilizando el vocabulario inglés.
* La integración debe seguir una estructura estandarizada de transacciones: construcción de petición, consulta de tiempo transcurrido, realización de llamada al proveedor, registro de errores durante la llamada y registro de transacción. Finalmente pueden tratarse los datos recibidos en la respuesta de la transacción.
* Desarrollar toda nueva integración en el lenguaje de programación C#.
* Compartir con el Dpto. de integraciones cualquier funcionalidad que se considere útil para toda integración desarrollada.
* Utilizar nomenclatura compartida entre integraciones.
* Realización de *code reviews* de manera quincenal en los que se persigue:
* Mejorar la calidad del código analizado.
* Servir como herramienta a los desarrolladores para aprender cuándo y cómo aplicar técnicas de calidad, consistencia y mantenibilidad.

## 2.5 Organización

En el apartado [2.5.1](#_2.5.1_Cultura_y) de este capítulo, se verán algunas de las consignas más destacadas de la cultura y estilo de TravelgateX, que no han sido mencionadas anteriormente en el apartado [2.4.2](#_2.4.2_Factores_ambientales:). Finalmente, en el apartado [2.5.2](#_2.5.2_Participantes_internos) se indica quien participa internamente en este proyecto.

### 2.5.1 Cultura y estilo de TravelgateX

Además de los requisitos de desarrollo mencionados en [2.4.2](#_2.4.2_Factores_ambientales:), TravelgateX cuando con una serie de consignas que los empleados tienden a seguir, mejorando así la colaboración entre equipos de desarrollo.

* Documentación de cualquier cambio realizado sobre las [API](#AcronimoAPI)s.
* Crear y compartir *posts* de ayuda, solucionadores de problemas, [FAQ](#AcronimoFAQ)s, notas de reunión, etc. en la página web corporativa (interna).
* De forma semanal, realizar retrospectivas en las que participen todos los integrantes de un Dpto.
* Utilización (en el Dpto. de integraciones) de la metodología *scrum* para proyectos de larga duración.
* Utilización de la metodología *kanban* para la gestión de incidencias.
* Realizar semanalmente, reuniones de seguimiento con el cliente.

### 2.5.2 Participantes internos del proyecto

El proyecto de modificación de reservas a través del servicio [ATC](#AcronimoATC) Shopper de Amadeus cuenta con un único desarrollador, integrante del Dpto. de integraciones aéreas de TravelgateX. Él será quien planifique, analice, diseñe y desarrolle todas las fases de este proyecto.

## 2.6 Partes interesadas

En este capítulo se enumeran los interesados internos y los externos, así como la relación que existe entre ellos. Se ahondará en las capas de interesados más bajas, lo que facilitará al lector comprender cómo se ejecutará la comunicación interna y externa entre los diferentes interesados.

### 2.6.1 Interesados internos y externos

Los interesados internos de este proyecto son:

#### Dpto. de integraciones aéreas

Además del desarrollador encargado del proyecto, el resto de integrantes del mismo Dpto. están implicados en la mejora y mantenimiento de las integraciones aéreas. La complejidad e importancia de este desarrollo, mejorará la calidad y capacidad técnica del Dpto.

#### Dpto. de integraciones hoteleras y de ancillaries

El resto de departamentos de integraciones también puede verse enriquecida de cualquier mejora o funcionalidad compartida que surja a partir del desarrollo. Un claro ejemplo es la posibilidad de compartir el formulario de pruebas.

Los interesados externos de este proyecto son:

#### Amadeus

El proveedor de los servicios y productos aéreos va a empezar una nueva clase de producto.

#### Logitravel

La agencia que se usará la aplicación desarrollada.

#### Otros clientes

La incorporación del servicio de modificar reservas puede ser un reclamo para otros clientes que decidan conectarse a TravelgateX.

### 2.6.2 Relación entre interesados

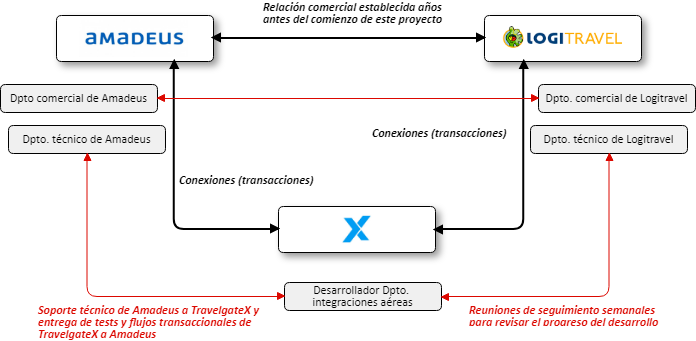
En la [Figura 15](#Figura15) se representa visualmente la relación ente los interesados internos y externos del proyecto.

Figura 15: Relación entre los interesados del proyecto

## 2.7 Riesgos

En este capítulo se enumeran los riesgos más destacados a tener en cuenta durante el ciclo de vida del proyecto y qué medidas deben tomarse para aplacar las incidencias y problemas que puedan surgir durante el mismo.

### 2.7.1 Complejidad del proyecto

La aplicación por desarrollar planificada es uno de los proyectos con mayor complejidad realizados en el departamento de integraciones aéreas. Además, ninguno de los integrantes del equipo tiene más de un año de experiencia con [GDS](#AcronimoGDS)s. Ninguno de los integradores que trabajan en el equipo ha realizado la integración de Amadeus, de tal manera que solamente se cuenta con la experiencia de ir gestionado las incidencias que han ido surgiendo a lo largo de unos 5 meses.

La inexperiencia o falta de capacidad para resolver los conflictos que surjan en el proyecto, puede retrasar la salida a producción del mismo. Dado que la modificación de reservas es un concepto nuevo para el conjunto de integradores de TravelgateX (incluido el Dpto. de integraciones hoteleras), no existe la posibilidad de ayuda de otro integrador basada en la experiencia con conceptos de modificación.

La estrategia que se toma para mitigar este riesgo consiste en la dedicación de más horas de la jornada laboral en el estudio de la documentación que proporciona Amadeus en su extranet, de tal forma que los conceptos básicos del [GDS](#AcronimoGDS) queden claros antes de iniciar el trabajo sobre este proyecto.

### 2.7.2 Planificación temporal

La jornada laboral del integrador no está dedicada exclusivamente a este proyecto. El integrador ha de seguir con la gestión de incidencias de otras integraciones y con el desarrollo de otros pequeños proyectos y mejoras que tienen también una planificación temporal.

Otro factor de riesgo muy importante a controlar es el limitado número de integrantes del proyecto, tanto interna como externamente. Cómo ya se ha citado anteriormente, solo un integrador de TravelgateX lleva el desarrollo del proyecto. Además, sólo un técnico de la agencia está a cargo del proyecto por la parte de cliente y dos técnicos pueden dar soporte desde la parte de Amadeus.

Cualquier incidencia que surja durante el ciclo de vida del proyecto relacionada con estas cuatro personas puede impedir cumplir con los plazos planificados o incluso terminar con el proyecto de forma indeseada.

Por esta razón es importante que todo el trabajo que se vaya realizando, se documente y se mantenga actualizado en las herramientas de la empresa. De esta forma, aunque pueda darse un importante retraso en la salida del proyecto, otro empleado puede hacerse cargo del proyecto sin tener que empezar desde cero.

## 2.8 Gestión del tiempo

Este capítulo contiene la estructura de descomposición del trabajo ([EDT](#AcronimoEDT)) en el apartado [2.8.1](#_2.8.1_Estructura_de); los paquetes de trabajo del proyecto desplegados en las diferentes tablas que se incluyen en el apartado [2.8.2](#_2.8.2_Planificación_de), como fichas de tareas con duración, fecha de inicio y fin, tipo de tarea y descripción de la tarea. En el apartado [2.8.3](#_2.8.3_Diagrama_de) se puede ver de forma gráfica la planificación temporal de estas tareas mediante un diagrama de Gantt.

Ha de tenerse en cuenta que la duración de las tareas y las fechas de inicio y fin no van a cuadrar dado que el integrador encargado del proyecto tiene también otras tareas diarias que desempeñar durante la jornada laboral de 8h.

### 2.8.1 Estructura de descomposición del trabajo (EDT)

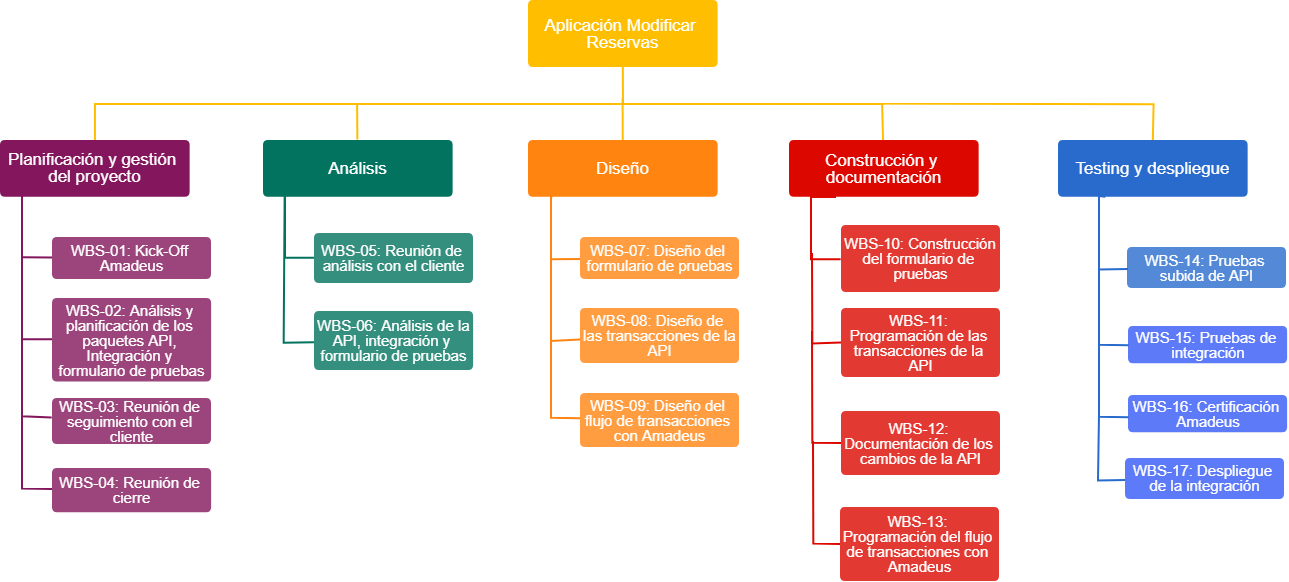
En la [Figura 16](#Figura16)se desglosan en paquetes de tareas, las diferentes áreas del proyecto:

Figura 16: Estructura de descomposición del trabajo (EDT)

### 2.8.2 Planificación de la duración de los paquetes de trabajo

Tabla 18: Paquetes de trabajo

|  |  |
| --- | --- |
| **Kick-Off Amadeus** | |
| **ID** | WBS-01 |
| **Duración** | 3h |
| **Fecha inicio** | 14/03/2017 |
| **Fecha fin** | 14/03/2017 |
| **Entradas** | - |
| **Salidas** | ENT-08: Acta de reunión del Kick-Off |
| **Tipo de tarea** | Trabajo fijo |
| **Descripción** | Reunión entre Amadeus, Logitravel y TravelgateX para presentar y planificar el proyecto [ATC](#AcronimoATC) Shopper para la modificación de reservas de Amadeus. |
| **Análisis y planificación de los paquetes API, Integración y formulario de pruebas** | |
| **ID** | WBS-02 |
| **Duración** | 17h |
| **Fecha inicio** | 24/03/2017 |
| **Fecha fin** | 30/03/2017 |
| **Entradas** | Extranet de Amadeus, Documentación [API](#AcronimoAPI) |
| **Salidas** | ENT-09: Conjunto de tareas en el Jira de TravelgateX |
| **Tipo de tarea** | Trabajo fijo |
| **Descripción** | Analizar y planificar el conjunto de tareas y como estructurarlas en el Jira de la empresa de tal forma que pueda desglosarse el desarrollo de las transacciones de la [API](#AcronimoAPI) de transportes, la integración y el formulario de pruebas |
| **Reunión de seguimiento con el cliente** | |
| **ID** | WBS-03 |
| **Duración** | 25h (25 x 1h) |
| **Fecha inicio** | 14/03/2017 |
| **Fecha fin** | 29/08/2017 |
| **Entradas** | ENT-09: Jiras en progreso y completados |
| **Salidas** | ENT-11: Apuntes de mejora o cambios sobre el proyecto |
| **Tipo de tarea** | Trabajo variable (tarea periódica) |
| **Descripción** | Reunirse semanalmente para mostrar el avance del proyecto y sacar dudas o mejoras planteadas por el cliente |
| **Reunión de cierre** | |
| **ID** | WBS-04 |
| **Duración** | 3h |
| **Fecha inicio** | 11/09/2017 |
| **Fecha fin** | 11/09/2017 |
| **Entradas** | - |
| **Salidas** | ENT-07: Mejoras para futuros proyectos con Amadeus |
| **Tipo de tarea** | Trabajo fijo |
| **Descripción** | Reunión de cierre de proyecto entre Amadeus, Logitravel y TravelgateX para analizar cómo ha ido el proyecto y proponer posibles mejoras para el futuro |
| **Reunión de análisis con el cliente** | |
| **ID** | WBS-05 |
| **Duración** | 3h (3 x 1h) |
| **Fecha inicio** | 27/03/2017 |
| **Fecha fin** | 08/04/2017 |
| **Entradas** | ENT-08: Acta de reunión del Kick-off |
| **Salidas** | Requisitos funcionales (apuntes) |
| **Tipo de tarea** | Trabajo variable (tarea periódica) |
| **Descripción** | Reunirse semanalmente para discutir y analizar el flujo de transacciones a realizar y los requisitos funcionales que debe cumplir la aplicación |
| **Análisis de la API, integración y formulario de pruebas** | |
| **ID** | WBS-06 |
| **Duración** | 40h |
| **Fecha inicio** | 07/04/2017 |
| **Fecha fin** | 20/04/2017 |
| **Entradas** | Apuntes de las reuniones de análisis, Extranet Amadeus |
| **Salidas** | ENT-10: Requisitos funcionales y no funcionales |
| **Tipo de tarea** | Trabajo fijo |
| **Descripción** | Analizar las transacciones a diseñar y construir por parte del cliente y por parte del proveedor y analizar el desarrollo del formulario de pruebas para sacar los requisitos de cada una de estas 3 partes |
| **Diseño del formulario de pruebas** | |
| **ID** | WBS-07 |
| **Duración** | 8h |
| **Fecha inicio** | 10/04/2017 |
| **Fecha fin** | 11/04/2017 |
| **Entradas** | ENT-10: Requisitos del formulario de pruebas (funcionales y no funcionales) |
| **Salidas** | Mockup formulario |
| **Tipo de tarea** | Trabajo fijo |
| **Descripción** | Hacer un mockup del formulario de pruebas |
| **Diseño de las transacciones de la API** | |
| **ID** | WBS-08 |
| **Duración** | 12h |
| **Fecha inicio** | 13/04/2017 |
| **Fecha fin** | 18/04/2017 |
| **Entradas** | Extranet Amadeus, ENT-10: Requisitos (funcionales y no funcionales) |
| **Salidas** | Esquema a alto nivel de las transacciones de modificar reserva |
| **Tipo de tarea** | Trabajo fijo |
| **Descripción** | Diseñar un diagrama a alto nivel de las llamadas, elementos y atributos que formarán parte de las nuevas transacciones de modificar reserva en la [API](#AcronimoAPI) de transportes de TravelgateX |
| **Diseño del flujo de transacciones con Amadeus** | |
| **ID** | WBS-09 |
| **Duración** | 18h |
| **Fecha inicio** | 27/04/2017 |
| **Fecha fin** | 04/05/2017 |
| **Entradas** | Extranet Amadeus, ENT-10: Requisitos (funcionales y no funcionales) |
| **Salidas** | ENT-02: Diagramas de flujo de las transacciones |
| **Tipo de tarea** | Trabajo fijo |
| **Descripción** | Diseñar los diagramas de flujo de las transacciones de modificar reservas de la integración |
| **Construcción del formulario de pruebas** | |
| **ID** | WBS-10 |
| **Duración** | 20h |
| **Fecha inicio** | 11/04/2017 |
| **Fecha fin** | 20/04/2017 |
| **Entradas** | ENT-10: Requisitos del formulario de pruebas (funcionales y no funcionales), Mockup formulario |
| **Salidas** | ENT-05: Formulario de pruebas |
| **Tipo de tarea** | Trabajo fijo |
| **Descripción** | Programar las funcionalidades y construir la interfaz gráfica del formulario |
| **Programación de las transacciones de la API** | |
| **ID** | WBS-11 |
| **Duración** | 10h |
| **Fecha inicio** | 18/04/2017 |
| **Fecha fin** | 21/04/2017 |
| **Entradas** | Esquema a alto nivel de las transacciones de modificar reserva |
| **Salidas** | ENT-04: Clases vb de las transacciones de modificar reserva |
| **Tipo de tarea** | Trabajo fijo |
| **Descripción** | Programar los cambios y nuevas transacciones de modificar reserva |
| **Documentación de los cambios de la API** | |
| **ID** | WBS-12 |
| **Duración** | 6h (6 x 1h) |
| **Fecha inicio** | 24/04/2017 |
| **Fecha fin** | 05/06/2017 |
| **Entradas** | ENT-04: Clases vb de las transacciones de modificar reserva |
| **Salidas** | ENT-11: Documentación de la [API](#AcronimoAPI) en la web corporativa |
| **Tipo de tarea** | Trabajo variable (tarea periódica) |
| **Descripción** | Describir los elementos y atributos que forman las nuevas llamadas y cualquier otro cambio realizado en la [API](#AcronimoAPI) |
| **Programación del flujo de transacciones con Amadeus** | |
| **ID** | WBS-13 |
| **Duración** | 170h |
| **Fecha inicio** | 08/05/2017 |
| **Fecha fin** | 27/06/2017 |
| **Entradas** | Extranet Amadeus, ENT-02: Diagramas de flujo de las transacciones |
| **Salidas** | ENT-03: Integración vb |
| **Tipo de tarea** | Trabajo fijo |
| **Descripción** | Programar la integración |
| **Pruebas subida de API** | |
| **ID** | WBS-14 |
| **Duración** | 6h (3 x 2h) |
| **Fecha inicio** | 01/05/2017 |
| **Fecha fin** | 15/05/2017 |
| **Entradas** | Nuevas versiones de la [API](#AcronimoAPI) |
| **Salidas** | Ficheros [XML](#AcronimoXML) de los tests realizados |
| **Tipo de tarea** | Trabajo variable (tarea periódica) |
| **Descripción** | Comprobar el funcionamiento de los cambios de la [API](#AcronimoAPI) y del resto de transacciones ya existentes a lo largo de las diferentes subidas: entorno de test, entorno de pre-producción y entorno de producción. |
| **Pruebas de integración** | |
| **ID** | WBS-15 |
| **Duración** | 56h |
| **Fecha inicio** | 28/06/2017 |
| **Fecha fin** | 17/08/2017 |
| **Entradas** | ENT-03: Integracion vb, ENT-05: Formulario de pruebas |
| **Salidas** | Mejoras sobre la integración |
| **Tipo de tarea** | Trabajo fijo |
| **Descripción** | Construir pruebas unitarias sobre la integración, encontrar fallos y mejoras a posibles futuros problemas |
| **Certificación Amadeus** | |
| **ID** | WBS-16 |
| **Duración** | 20h |
| **Fecha inicio** | 03/07/2017 |
| **Fecha fin** | 20/07/2017 |
| **Entradas** | ENT-03: Integración vb, ENT-05: Formulario de pruebas |
| **Salidas** | ENT-01: Escenarios de test completados |
| **Tipo de tarea** | Trabajo fijo |
| **Descripción** | Revisar si la integración cumple con el checklist de certificación y realizar los escenarios de test que impone Amadeus para pasar la certificación |
| **Despliegue de la integración** | |
| **ID** | WBS-17 |
| **Duración** | 28h |
| **Fecha inicio** | 24/07/2017 |
| **Fecha fin** | 05/09/2017 |
| **Entradas** | ENT-03: Integración vb |
| **Salidas** | Despliegue del desarrollo en producción |
| **Tipo de tarea** | Trabajo fijo |
| **Descripción** | Realizar pruebas de funcionamiento de la integración en los entornos de test, pre-producción y producción |

### 2.8.3 Diagrama de Gantt

La [Figura 17](#Figura17) presentada a continuación representa el diagrama de Gantt a alto nivel en los que se presentan los 17 paquetes de tareas. En el eje **X** de la representación, se han separado 5 áreas, indicadas en la parte superior y cada una representada con un color. El eje **Y** de la representación sirve para indicar el tiempo de las tareas y de la totalidad del proyecto. Los paquetes de tareas están indicados con su id dentro del [EDT](#AcronimoEDT) y de las fichas de tareas vistas en el capítulo [2.8.2](#_2.8.2_Planificación_de). Las tareas periódicas se han representado con el texto en blanco del ID del paquete de tareas y ocupando el espacio temporal que ocupan dichas tareas periódicas.

En la [Figura 17](#Figura17) hay representados 3 hitos:

* [14/03/2017: KickOff Amadeus (WBS-01)](#Hito1)
* [11/09/2017: Reunión de cierre (WBS-04)](#Hito2)
* [05/09/2017: Despliegue de la integración](#Hito3)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Mes** | Día | Planificación y gestión del proyecto | Análisis | Diseño | Construcción y documentación | Testing y despliegue |
| **Marzo ‘17** | 06 |  |  |  |  |  |
| 13 |  |  | **WBS-01: 14/03/2017 – Kick-Off Amadeus** |  |  |
| 20 | WBS-02 |  |  |  |  |
| 27 |  |  |  |  |  |
| **Abril ‘17** | 03 |  | WBS-06  WBS-05 |  |  |  |
| 10 |  |  |  | WBS-10 |  |
| 17 |  |  | WBS-07  WBS-08 |  |  |
| 24 |  |  | WBS-09 | WBS-11 |  |
| **Mayo ‘17** | 01 |  |  |  | WBS-12 | WBS-14 |
| 08 |  |  |  | WBS-13 |  |
| 15 |  |  |  |  |  |
| 22 |  |  |  |  |  |
| 29 |  |  |  |  |  |
| **Junio ‘17** | 05 | WBS-03 |  |  |  |  |
| 12 |  |  |  |  |  |
| 19 |  |  |  |  |  |
| 26 |  |  |  |  | WBS-16  WBS-15 |
| **Julio ‘17** | 03 |  |  |  |  |  |
| 10 |  |  |  |  |  |
| 17 |  |  |  |  |  |
| 24 |  |  |  |  | WBS-17 |
| 31 |  |  |  |  |  |
| **Agosto ‘17** | 07 |  |  |  |  |  |
| 14 |  |  |  |  |  |
| 21 |  |  |  |  |  |
| 28 |  |  |  |  |  |
| **Septiembre ‘17** | 04 |  |  |  | **WBS-17: 05/09/2017 – Despliegue** |  |
| 11 |  |  |  |  |  |
| 18 |  | **WBS-04: 11/09/2017 – Reunión de cierre** |  |  |  |
| 25 |  |  |  |  |  |

Figura 17: Diagrama de Gantt

## 2.9 Plan de gestión de comunicaciones

En este capítulo se detalla cual ha de ser el plan de comunicación interno y externo. En el apartado [2.9.1](#_2.9.1_Comunicación_interna) se define la situación de los trabajadores de la empresa TravelgateX en la que se desarrolla el proyecto, así como las herramientas de comunicación internas. A continuación, en los capítulos [2.9.2](#_2.9.2_Comunicación_con) y [2.9.3](#_2.9.3_Comunicación_con) se explican las herramientas de comunicación que han de utilizarse durante el desarrollo, con los interesados externos que participan en el proyecto.

### 2.9.1 Comunicación interna

El departamento de integraciones de TravelgateX se separa en 3 áreas:

* Vuelos
* Hoteles
* Ancillaries

Aunque los integradores de cada área no compartan [API](#AcronimoAPI) de desarrollo, muchos detalles, consultas o dudas pueden llegar a resolverse con la colaboración de los integradores que forman las tres áreas. Todos ellos comparten oficina, por lo que en muchas ocasiones la comunicación entre todos es verbal y presencial. Sin embargo, con el fin de evitar continuas interrupciones entre trabajadores, se utiliza la herramienta *Slack* como chat interno y el correo de empresa si el chat no es suficiente.

La comunicación y colaboración entre integradores de las tres áreas, permite a la larga, el intercambio de puestos de trabajo. Con el suficiente conocimiento, un integrador de hoteles puede adaptarse en poco tiempo al trabajo realizado en el área de vuelos y viceversa.

### 2.9.2 Comunicación con Logitravel

La herramienta utilizada normalmente para comunicar el desarrollo de las tareas es *Jira*. El desarrollador expone el avance de un desarrollo en su ticket de *Jira*. A continuación el equipo de soporte traslada el mensaje al ticket conectado con el cliente y éste responde en este ticket. Ésta comunicación en dos pasos evita el exceso de interrupciones innecesarias al desarrollador.

En ocasiones el integrador puede comunicarse directamente con el cliente y viceversa, pero siempre utilizando el correo asociado al área de integraciones aéreas. De esta forma, el resto de integradores que trabajan en la misma área pueden seguir la conversación y el conocimiento aprendido queda adherido al área.

El manager del Dpto. cuenta con un teléfono por el que puede comunicarse también de forma directa con el cliente. Éste ha de utilizarse solamente en caso de urgencia.

### 2.9.3 Comunicación con Amadeus

En el plan de proyecto de Amadeus, presentado en el Kick-Off del proyecto, se presentaron a los dos técnicos que colaboran con el desarrollador de Travelgate. La comunicación entre el desarrollador y los técnicos proporcionados por Amadeus ha de hacerse a través de correo, abriendo siempre hilos de correos correctamente etiquetados.

Amadeus cuenta además con un sistema de consultas abierta a los desarrolladores y comerciales de sus clientes. A través de él también pueden abrirse incidencias o consultas.

# Análisis

En este capítulo se definen los requisitos funcionales y no funcionales de los productos. El apartado [3.1](#_3.1_Requisitos_no) Requisitos no funcionales se separa en tres sub-apartados: [3.1.1](#_3.1.1_Requisitos_no) en el que se detallan los requisitos no funcionales de las transacciones de la [API](#AcronimoAPI); [3.1.2](#_3.1.2_Requisitos_no) en el que se detallan los requisitos no funcionales de la integración; [3.1.3](#_3.1.3_Requisitos_no) en el que se detallan los requisitos no funcionales del formulario de pruebas. Finalmente, en el apartado [3.2](#_3.2_Requisitos_funcionales) se detallan los requisitos funcionales.

## 3.1 Requisitos no funcionales

En este capítulo se han separado los requisitos no funcionales en las tres grandes áreas que separan el proyecto:

* Transacciones de la API de transportes
* Integración
* Formulario de pruebas

### 3.1.1 Requisitos no funcionales de las transacciones de la API de transportes

Tabla 19: Requisitos no funcionales de las transacciones de la [API](#AcronimoAPI) de transportes

|  |  |
| --- | --- |
| RNF-01 | Las peticiones y respuestas, así como elementos y atributos que las forman deberán ser desarrolladas en el lenguaje de programación Visual Basic. |
| RNF-02 | Las nuevas llamadas implementadas han de permitir la compatibilidad entre el lenguaje de programación y la creación de objetos [XML](#AcronimoXML). |
| RNF-03 | Las nuevas llamadas han de permitir la creación en última instancia de archivos .[wsdl](#AcronimoWSDL) y .[xsd](#AcronimoXSD), de tal forma que el sistema de TravelgateX pueda interpretar el contenido creado. |
| RNF-04 | Las llamadas han de ser compatibles con el resto de contenido ya existente en la [API](#AcronimoAPI) de transportes. |
| RNF-05 | Todos los elementos y atributos creados deberán ser correctamente explicados en el código mediante comentarios que los definan. |
| RNF-06 | La nomenclatura de las llamadas y su contenido deberá seguir acorde con el estilo de la [API](#AcronimoAPI) de transportes. |
| RNF-07 | Para permitir la serialización y deserialización de objetos a [XML](#AcronimoXML)s, los elementos creados deberán ser del tipo XmlElement y los atributos del tipo XmlAttribute. |
| RNF-08 | Todas las clases creadas en la [API](#AcronimoAPI) deberán ser públicas. |
| RNF-09 | Todas las clases creadas en la [API](#AcronimoAPI) deberán heredar de las clases básicas de la [API](#AcronimoAPI), Transportation y estas a su vez de Base, conteniendo ya éstas elementos generales. |
| RNF-10 | Las clases deberán ser desarrolladas con un constructor que permita la declaración de objetos vacíos, así como objetos que inicialicen todas sus variables. |
| RNF-11 | Las clases, elementos y atributos que vayan a ser visibles en las transacciones [XML](#AcronimoXML) han de permitir su visualización en español y en inglés. |

### 3.1.2 Requisitos no funcionales de la integración

Tabla 20: Requisitos no funcionales de la integración

|  |  |
| --- | --- |
| RNF-12 | Las nuevas clases en la integración para la modificación de reservas deberán desarrollarse mediante el lenguaje de programación Visual Basic. |
| RNF-13 | La integración ha de controlar las excepciones que se produzcan durante la ejecución. |
| RNF-14 | La integración ha de controlar los errores que devuelva cualquier transacción realizada con el proveedor y tipificar adecuadamente dichos errores para describir su causa. |
| RNF-15 | La integración ha de controlar los errores de servidor y de cliente que se produzcan durante la ejecución de las transacciones. |
| RNF-16 | La integración ha de controlar el tiempo transcurrido desde el comienzo de la transacción hasta su finalización y cortar la ejecución en el tiempo previamente estipulado para dicha transacción. |
| RNF-17 | La integración ha de registrar en la respuesta de la transacción de cliente (ej: DisponibilidadRS) todas las transacciones que se hayan realizado con el proveedor. |
| RNF-18 | La integración ha de abrir una sesión con las credenciales adecuadas y predeterminadas con el proveedor y mantener dicha sesión para todas las transacciones que se realicen, así como cerrar la sesión una vez terminado el flujo de transacciones deseado. |
| RNF-19 | El código desarrollado ha de mantener la compatibilidad con el resto de funcionalidades ya desarrolladas en la integración. |
| RNF-20 | El código desarrollado ha de aprovechar al máximo las funcionalidades desarrolladas en la integración previamente para facilitar su posterior mantenimiento y legibilidad del código. |
| RNF-21 | La integración ha de minimizar el uso de memoria y procesador durante su ejecución, de tal forma que la ejecución de las nuevas llamadas ha de demostrar un uso medio igual al resto de transacciones ya desarrolladas en la integración. |
| RNF-22 | El código desarrollado debe escribirse en inglés para permitir su legibilidad para el resto de integrantes de TravelgateX. |
| RNF-23 | La integración deberá incluir entre un 5 y un 10% de comentarios respecto a la totalidad de nuevo código desarrollado. |
| RNF-24 | La integración debe seguir la estructura estandarizada de transacciones que sigue el equipo de desarrollo de integraciones de TravelgateX: construcción de petición, consulta de tiempo transcurrido, realización de llamada al proveedor, registro de errores durante la llamada y registro de transacción. Finalmente pueden tratarse los datos recibidos en la respuesta de la transacción. |
| RNF-25 | La programación realizada deberá facilitar la compatibilidad con la creación y ejecución de pruebas sobre el proyecto. |

### 3.1.3 Requisitos no funcionales del formulario de pruebas

Tabla 21: Requisitos no funcionales del formulario de pruebas

|  |  |
| --- | --- |
| RNF-26 | El formulario de pruebas deberá desarrollarse mediante el lenguaje de programación Visual Basic. |
| RNF-27 | El formulario ha de permitir el manejo de transacciones (petición respuesta) de forma completamente visual. |
| RNF-28 | El formulario de pruebas ha de controlar y visualizar cualquier error de ejecución arrojado por la integración o durante la ejecución de código del mismo formulario. |
| RNF-29 | Los resultados obtenidos en las transacciones de disponibilidad han de poder visualizarse de manera gráfica en una tabla de opciones. |
| RNF-30 | El formulario no ha de perjudicar ni alterar los resultados obtenidos en ninguna de las transacciones realizadas en él. |
| RNF-31 | El formulario ha de ser compatible con el resto de integraciones de vuelos que existen en TravelgateX. |
| RNF-32 | El formulario no debe usar terminología o tipología del proveedor Amadeus. |
| RNF-33 | El formulario de pruebas ha de poder abrirse mediante un ejecutable. |

## 3.2 Requisitos funcionales

En este capítulo se enumeran los requisitos funcionales distinguiendo aquellos que deberá cumplir la integración y aquellos que deberá cumplir el formulario de pruebas.

En los requisitos funcionales de la integración, se separan aquellos que deberán cumplirse para toda nueva transacción desarrollada, de los requisitos que debe cumplir cada una de las transacciones por separado.

Tabla 22: Requisitos funcionales

|  |  |
| --- | --- |
|  | Requisitos funcionales de la integración |
|  |  |
|  | En todas las transacciones |
| RF-01 | El cliente introduce el tiempo de corte de la transacción. |
| RF-02 | El cliente introduce las credenciales y las [URL](#AcronimoURL)s a las que atacará la transacción. |
| RF-03 | El cliente elige el idioma de las transacciones [XML](#AcronimoXML). |
| RF-04 | El cliente visualiza los errores de la transacción. |
| RF-05 | El cliente visualiza los warnings de la transacción. |
| RF-06 | El cliente visualiza las transacciones del proveedor (petición y respuesta). |
| RF-07 | El cliente visualiza el tiempo transcurrido en la ejecución de la transacción. |
| RF-08 | El cliente visualiza la fecha en la que se realiza la transacción. |
|  |  |
|  | En la transacción [DMR](#AcronimoDMR) |
| RF-09 | El cliente indica los tramos del nuevo itinerario. |
| RF-10 | El cliente indica el tipo de modificación de un tramo. |
| RF-11 | El cliente indica el origen, destino y fechas de un tramo. |
| RF-12 | El cliente indica el localizador de la reserva. |
| RF-13 | El cliente filtra la búsqueda por clase cabina. |
| RF-14 | El cliente filtra la búsqueda por solo vuelos directos (sin escalas). |
| RF-15 | El cliente filtra la búsqueda por compañía. |
| RF-16 | El cliente elige si la búsqueda incluye compañías lowcost. |
| RF-17 | El cliente elige el tipo de viaje que desea realizar: [RT](#AcronimoRT), [OW](#AcronimoOW), [OJ](#AcronimoOJ), [CT](#AcronimoCT). |
| RF-18 | El cliente visualiza los segmentos arrojados en la búsqueda. |
| RF-19 | El cliente visualiza el número de transporte del segmento. |
| RF-20 | El cliente visualiza el origen, destino y fecha del segmento. |
| RF-21 | El cliente visualiza la compañía que vende el vuelo y la compañía que opera el vuelo. |
| RF-22 | El cliente visualiza las terminales de salida y llegada del segmento. |
| RF-23 | El cliente visualiza el tipo de transporte (tipo de avión). |
| RF-24 | El cliente visualiza la duración del segmento. |
| RF-25 | El cliente visualiza si el segmento tiene parada técnica y sus detalles. |
| RF-26 | El cliente visualiza el tipo de tarifa: [RT](#AcronimoRT), [OW](#AcronimoOW), [OJ](#AcronimoOJ), [CT](#AcronimoCT). |
| RF-27 | El cliente visualiza las condiciones de la tarifa. |
| RF-28 | El cliente visualiza el importe de la tarifa desglosada por tipo de pasajero: [ADT](#AcronimoADT), [CHD](#AcronimoCHD) e [INF](#AcronimoINF). |
| RF-29 | El cliente visualiza la moneda del importe. |
| RF-30 | El cliente visualiza las tasas del importe y su tipo. |
| RF-31 | El cliente visualiza los cargos o penalizaciones que deberán abonarse por la modificación de la reserva. |
| RF-32 | El cliente visualiza los segmentos asociados a cada tarifa. |
| RF-33 | El cliente visualiza la compañía validadora de la tarifa. |
| RF-34 | El cliente visualiza la familia de tarifa asociada. |
| RF-35 | El cliente visualiza el tipo y cantidad de equipajes incluidos en la tarifa. |
| RF-36 | El cliente visualiza los elementos extra que incluye la tarifa. |
| RF-37 | El cliente visualiza la clase cabina de cada segmento. |
| RF-38 | El cliente visualiza la clase de cada segmento. |
| RF-39 | El cliente visualiza el número de asientos disponibles por cada clase. |
| RF-40 | El cliente visualiza el tipo de tarifa: pública, privada o negociada. |
| RF-41 | El cliente visualiza los billetes de la reserva. |
| RF-42 | El cliente visualiza la cantidad de pasajeros desglosados por tipo de pasajero: [ADT](#AcronimoADT), [CHD](#AcronimoCHD) e [INF](#AcronimoINF). |
|  |  |
|  | En la transacción [RMR](#AcronimoRMR) |
| RF-43 | El cliente indica la tarifa seleccionada. |
| RF-44 | El cliente indica la forma de pago. |
| RF-45 | El cliente indica los billetes de la reserva. |
| RF-46 | El cliente indica el localizador de la reserva. |
| RF-47 | El cliente indica el Delta Price (Importe diferencial aceptado entre el precio de la tarifa obtenida en [DMR](#AcronimoDMR) y el precio que se obtenga en la [RMR](#AcronimoRMR)) de la reserva. |
| RF-48 | El cliente visualiza los datos de un pasajero guardados en la reserva. |
| RF-49 | El cliente visualiza los billetes de la reserva. |
| RF-50 | El cliente visualiza la factura de la reserva una vez ha sido ya modificada. |
| RF-51 | El cliente visualiza los importes desglosados de la factura, por tipo de pasajero y los cargos de la misma. |
| RF-52 | El cliente visualiza la compañía que cobra la factura. |
| RF-53 | El cliente visualiza el [LTD](#AcronimoLTD). |
| RF-54 | El cliente visualiza el tipo de emisión. |
|  |  |
|  | En la transacción [EMR](#AcronimoEMR) |
| RF-55 | El cliente indica el tipo de emisión. |
| RF-56 | El cliente indica los billetes de la reserva. |
| RF-57 | El cliente indica el localizador de la reserva. |
| RF-58 | El cliente visualiza los nuevos billetes de la reserva. |
|  |  |
|  | Requisitos funcionales del formulario de pruebas |
|  |  |
|  | En todas las transacciones |
| RF-59 | El usuario copia la petición y la respuesta [XML](#AcronimoXML). |
| RF-60 | El usuario indica el tiempo de corte de la transacción. |
| RF-61 | El usuario indica las credenciales y las [URL](#AcronimoURL)s. |
| RF-62 | El usuario elige el entorno del proveedor al que se realizará la transacción: test o producción. |
| RF-63 | El usuario indica el tipo de transacción y pega la petición que posteriormente podrá realizar. |
|  |  |
|  | En la transacción de Disponibilidad |
| RF-64 | El usuario indica el origen, destino y fechas de cada tramo. |
| RF-65 | El usuario añade una compañía de conexión. |
| RF-66 | El usuario elimina una compañía de conexión. |
| RF-67 | El usuario indica el número y la edad de los pasajeros. |
| RF-68 | El usuario realiza la petición de Disponibilidad. |
| RF-69 | El usuario elige el tipo de viaje: [OW](#AcronimoOW), [RT](#AcronimoRT) o [OJ](#AcronimoOJ) |
|  |  |
|  | En la transacción de Valoración |
| RF-70 | El usuario elige las preferencias de Valoración. |
| RF-71 | El usuario elige las opciones devueltas por la Disponibilidad en una tabla. |
| RF-72 | El usuario realiza la petición de Valoración. |
|  |  |
|  | En la transacción de Reserva |
| RF-73 | El usuario indica el nombre, apellidos, email, teléfono, calle, localidad, C.P., país y nacionalidad del cliente titular de la reserva. |
| RF-74 | El usuario indica el tratamiento de un pasajero y su sexo. |
| RF-75 | El usuario indica el nombre, apellidos, fecha de nacimiento, tipo, id y fecha de caducidad del documento de identidad y C.P. del municipio en el que reside. |
| RF-76 | El usuario indica la forma de pago: card o cash. |
| RF-77 | El usuario indica el titular, número, tipo, [CVV](#AcronimoCVV) y fecha de caducidad de la tarjeta. |
| RF-78 | El usuario indica el delta Price. |
| RF-79 | El usuario realiza la petición de Reserva. |
|  |  |
|  | En las transacciones de Recuperar, Cancelar, Emitir y Vaciar billete |
| RF-80 | El usuario indica el localizador de la reserva. |
| RF-81 | El usuario indica el número de un billete. |
| RF-82 | El usuario indica el tipo de emisión. |
| RF-83 | El usuario indica el tipo de billete: normal o extra. |
| RF-84 | El usuario realiza las peticiones Recuperar Reserva, Cancelar Reserva, Emitir billetes y Vaciar Billete. |
|  |  |
|  | En las transacciones de Modificar Reserva |
| RF-85 | El usuario indica el localizador de la reserva. |
| RF-86 | El usuario elige el tipo de viaje: [OW](#AcronimoOW), [RT](#AcronimoRT) o [OJ](#AcronimoOJ). |
| RF-87 | El usuario indica el origen, destino y fechas de cada tramo. |
| RF-88 | El usuario indica la acción a realizar en cada tramo: [N](#ModifN), [KF](#ModifKF), [K](#ModifK), [C](#ModifC), [R](#ModifR) o [A](#ModifA). |

# Diseño

El capítulo de diseño que se presenta a continuación contiene los diseños elaborados a partir de los requisitos extraídos del análisis expuesto en el capítulo [3 Análisis](#_Análisis). Primero, en el apartado [4.1](#_4.1_Transacciones_API), se detalla el diseño de las tres transacciones a desarrollar en la [API](#AcronimoAPI) de transportes para que un cliente pueda modificar una reserva a través de TravelgateX. A continuación, se presentan los diagramas de flujo que deberán ser ejecutados en la integración para modificar reservas con Amadeus.

La representación de los diagramas mostrados en el apartado [4.1](#_4.1_Transacciones_API) no se basa en ningún estándar de diseño. La finalidad de los diagramas es facilitar la posterior programación de las clases que se detallan en el capítulo [5.1.2](#_5.1.2_Descripción_de).

Las representaciones mostradas en el capítulo [4.2](#_4.2_Diagramas_de) siguen el diseño básico de un diagrama de flujo, utilizado habitualmente en el diseño de flujos de ejecución en una aplicación software. Durante este apartado se detalla cada una de las operaciones y condiciones que se representan gráficamente en las figuras.

## 4.1 Transacciones API Transportes

En este capítulo se define el diseño de las clases que tendrán que desarrollarse y las relaciones entre ellas a partir de los requerimientos definidos en el capítulo [3 Análisis](#_Análisis).

#### Del requerimiento [RF-01 al RF-08](#Tabla22_RF01):

Se utilizan las clases TransportationBaseRQ y TransportationBaseRS previamente desarrolladas y disponibles para el desarrollador de TravelgateX en la [API](#AcronimoAPI) de Transportes.

#### Del requerimiento [RF-09 al RF-42](#Tabla22_RF09):

[Figura 18](#Figura18): Diseño de clases de la transacción [DMR](#AcronimoDMR)

#### Del requerimiento [RF-43 al RF-54](#Tabla22_RF43):

[Figura 19](#Figura19): Diseño de clases de la transacción [RMR](#AcronimoRMR)

#### Del requerimiento [RF-55 al RF-58](#Tabla22_RF55):

Figura 20: Diseño de clases de la transacción [EMR](#AcronimoEMR)

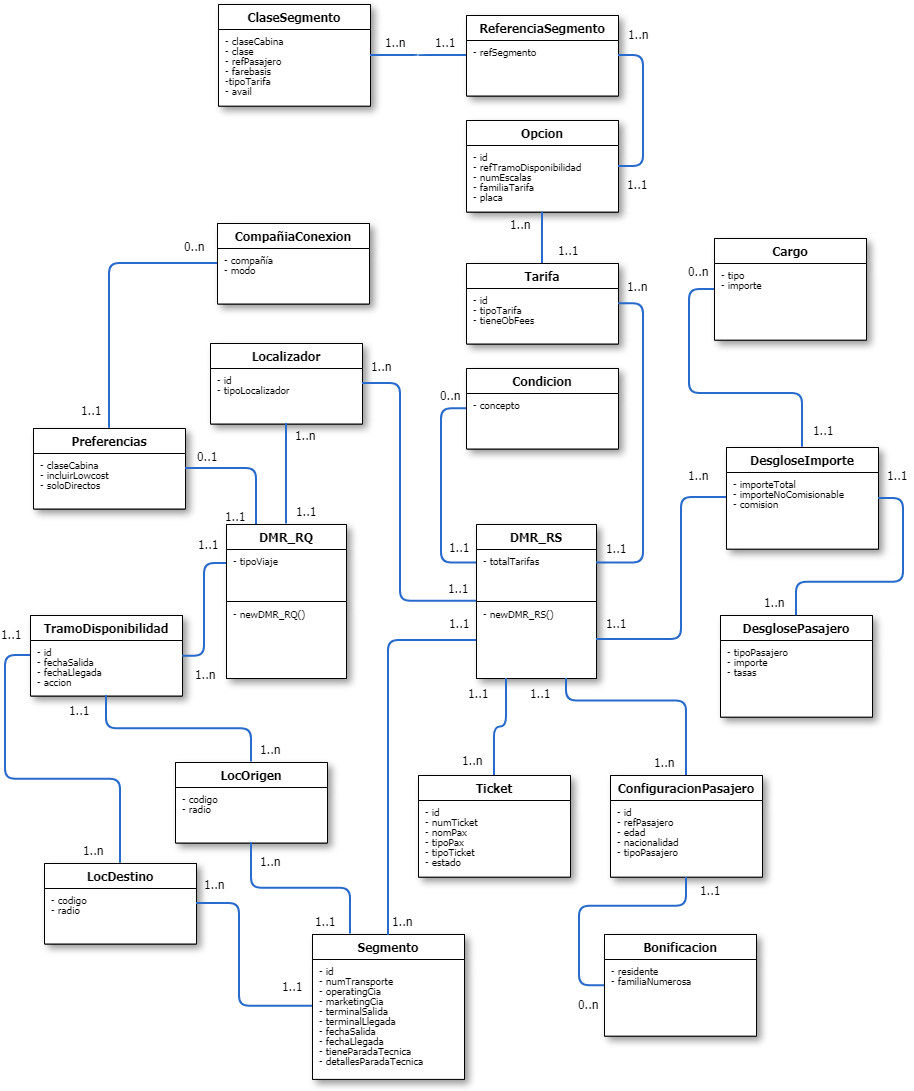


Figura 18: Diseño de las clases de la transacción DMR

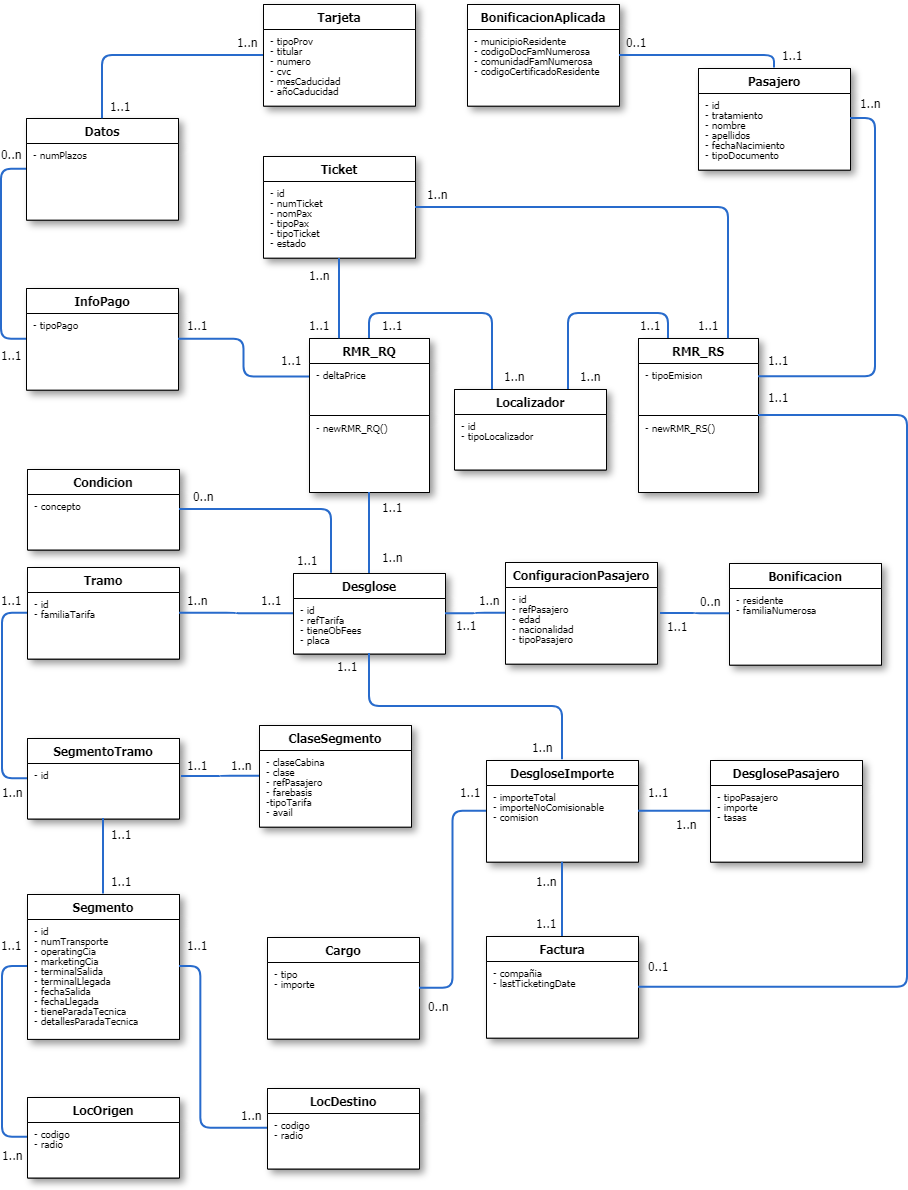


Figura 19: Diseño de clases de la transacción RMR

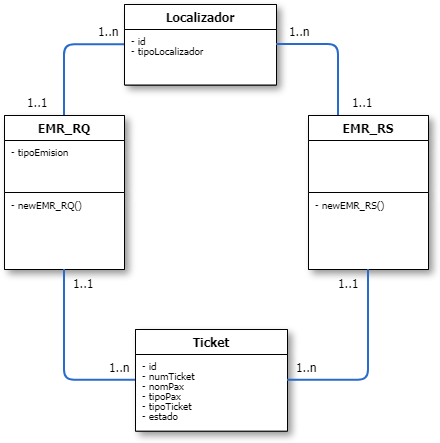


Figura 20: Diseño de clases de la transacción EMR

## 4.2 Diagramas de flujo de transacciones con el GDS

La aplicación basada en el conjunto de Web Services de [ATC](#AcronimoATC) Shopper de Amadeus consta de tres flujos independientes: Disponibilidad o Disponibilidad Modificar Reserva ([DMR](#AcronimoDMR)), Reserva o Reserva Modificar Reserva ([RMR](#AcronimoRMR)) y Emisión o Emitir Modificar Reserva ([EMR](#AcronimoEMR)).

Los diagramas representados en las figuras [Figura 21](#Figura21), Figura 22 y Figura 23, corresponden a la segunda versión diseñada por recomendación de Amadeus. La primera versión presentada a Amadeus, ejecutada alguna de las operaciones con un tipo de sesión inadecuada, lo que hubiera supuesto un warning en la certificación. Tal y como se explica en los diagramas, las partes modificadas con respecto a la primera versión, son aquellas que están resaltadas en color.

Éste cambio de plan supone el rediseño de las transacciones y el flujo de ejecución de ellas y, por lo tanto, algunas de las tareas presentadas en el capítulo [2.8.1](#_2.8.1_Estructura_de) han de sufrir cambios en la planificación temporal.

#### Disponibilidad [DMR](#AcronimoDMR)

La transacción [DMR](#AcronimoDMR) permite obtener las recomendaciones disponibles existentes en [ATC](#AcronimoADT) Shopper [MasterPricer](#TICKET_ATCMP) de acuerdo a los parámetros solicitados por el usuario:

* + Cambio de clase
  + Cambio de aerolínea
  + Cambio de horario
  + Cambio de itinerario

Primero se recupera el [PNR](#AcronimoPNR) y se comprueba que éste contenga tickets emitidos. Si tiene tickets emitidos se comprueba si alguna de ellos es un ticket del tipo extra. Si es así, se recupera la lista de [TSM](#AcronimoTSM)s para comprobar si todos los [TSM](#AcronimoTSM)s son del tipo *financial impact extra*. En el caso que alguno de los [TSM](#AcronimoTSM)s no sea de este tipo, el flujo de ejecución se corta cerrando sesión.

En cualquier otro caso se sigue el flujo abriendo el [TST](#AcronimoTST) para consultar información de los pasajeros y cerrando sesión. El resto de llamadas se realizan con sesión stateless.

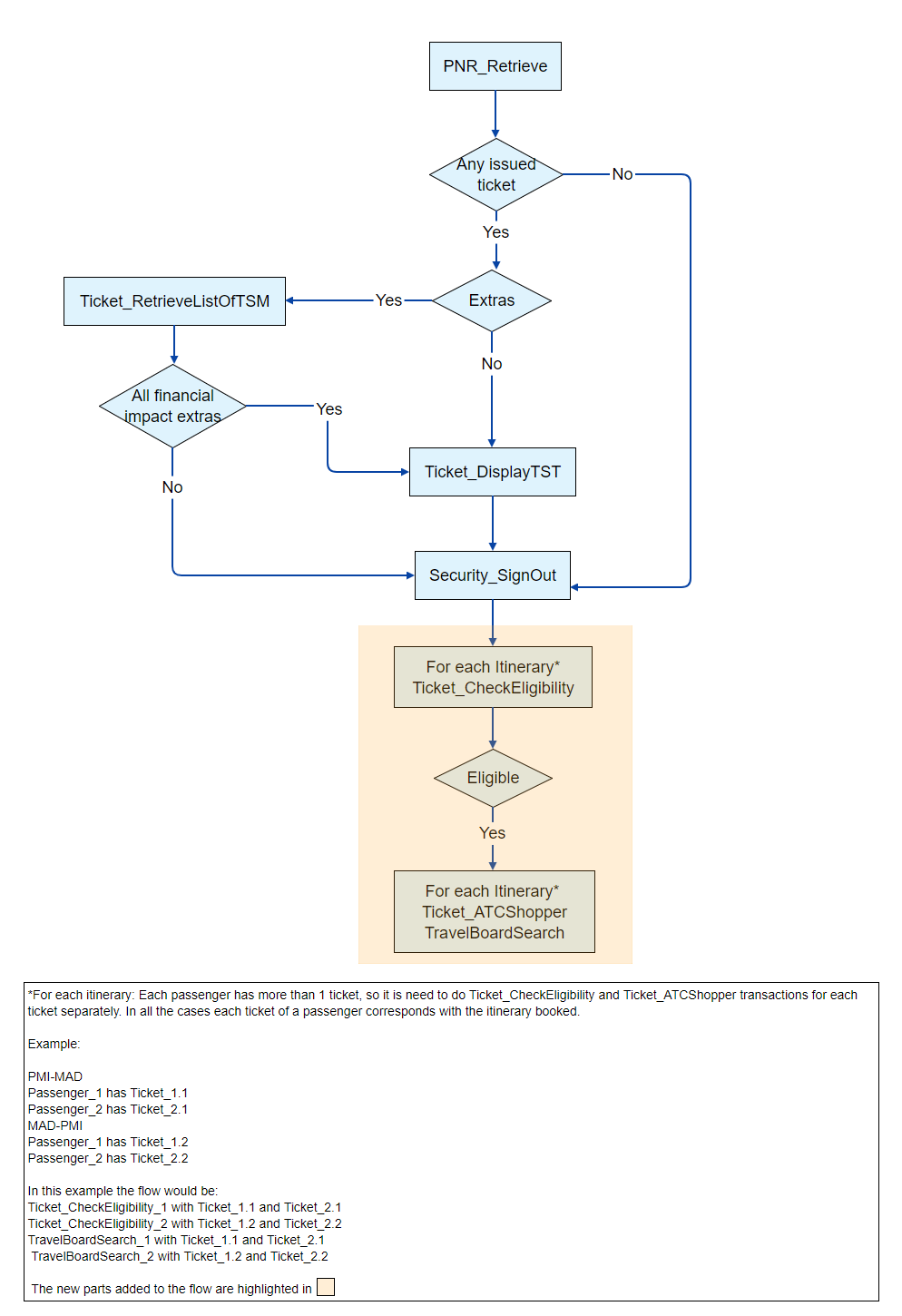
A continuación, para cada itinerario (referencia de la explicación al diagrama), se comprueba si los tickets son modificables con [Ticket\_CheckEligibility](#TICKET_ELIGIBILITY) y en caso afirmativo se realiza la petición a [Ticket\_ATCShopperTravelBoardSearch](#TICKET_ATCMP). Ésta devuelve las recomendaciones que hay disponibles con los criterios que se han solicitado.

Figura 21: Diagrama de flujo de ejecución de la transacción DMR

#### 

Figura 22: Diagrama de flujo de ejecución de la transacción RMR

#### Reserva [RMR](#AcronimoRMR)

La transacción [RMR](#AcronimoRMR) empieza recuperando el [PNR](#AcronimoPNR) y comprobando que cambios se han solicitado en la transacción [DMR](#AcronimoDMR). Se cancelan los segmentos que sean necesarios y se realiza el [Sell](#AIRSELL) de aquellos segmentos solicitados por el usuario.

Si existe más de un ticket asociado a cada pasajero, se recupera y abre de nuevo [PNR](#AcronimoPNR).

Una vez que se han cancelado y añadido todos los segmentos se ejecuta un [Ticket\_RepricePNRWithBookingClass](#TICKET_REPRICE) por cada itinerario de las reserva. En la respuesta se recibirán los importes de penalización, residual value, tasas e importes totales definitivos. Para confirmar se llama a [Ticket\_ReissueConfirmedPricing](#TICKET_REISSUE), de nuevo, una ejecución por itinerario.

A continuación se comprueba si hay cambio de precio con respecto al importe obtenido en [DMR](#AcronimoDMR). Si hay cambio de precio y delta Price no es suficiente, el flujo se detiene cerrando sesión. Si no hay cambio de precio o el Delta Price es suficiente, se reabre el [PNR](#AcronimoPNR).

Ahora se comprueba la existencia de [EMD](#AcronimoEMD)s en el [PNR](#AcronimoPNR) y en ese caso, se recupera la lista de [TSM](#AcronimoTSM)s con la llamada Ticket\_RetrieveListOfTSM.

A continuación hay que realizar los cambios necesarios en la forma de pago dependiendo del tipo de reemisión/revalidación se esté dando. Se cancela la forma de pago mediante [PNR\_Cancel](#PNRCANCEL) y se crea la nueva a través de FOP\_CreateFormOfPayment. Dependiendo del tipo de modificación que se esté realizando, se añaden las formas de pago vieja/nueva y las formas de pago para los [TSM](#AcronimoTSM)s que se hayan podido crear al hacer el [Reprice](#TICKET_REPRICE).

Finalmente se cierra el [PNR](#AcronimoPNR) confirmando los cambios mediante [PNR\_AddMultiElements](#PNRADDMULTIELEMENTS) y cerrando sesión con Security\_SignOut.

En la [Figura 22](#Figura22) se puede apreciar la representación gráfica mediante un diagrama de flujo del flujo de ejecución explicado para [RMR](#AcronimoRMR).

#### Emisión [EMR](#AcronimoEMR)

La transacción [EMR](#AcronimoEMR) empieza recuperando el [PNR](#AcronimoPNR). Se comprueba si hay alguna comisión parcial que haya podido quedar al cancelar algún segmento y manteniendo otros. Si existe la se elimina ([PNR\_Cancel](#PNRCANCEL)) y se crea una nueva comisión vinculada a todos los segmentos existentes en el [PNR](#AcronimoPNR) modificado.

Dependiendo de si es una revalidación, reemisión y de si tiene o no [EMD](#AcronimoEMD)s asociados se sigue un camino u otro en el flujo. Hay en total 5 posibilidades:

1. Revalidación sin [EMD](#AcronimoEMD)
2. Revalidación con [EMD](#AcronimoEMD)
3. Reemisión sin [EMD](#AcronimoEMD)
4. Reemisión con [EMD](#AcronimoEMD) (solo residual value)
5. Reemisión con [EMD](#AcronimoEMD) (penalización y residual value)

Si es un escenario 1, se emiten los tickets con DocIssuance\_IssueTicket. Si es un escenario 2 además de emitir con DocIssuance\_IssueTicket, se realiza la emisión de los [TSM](#AcronimoTSM) de penalización con DocIssuance\_IssueMiscellaneousDocuments. Si en cambio es un escenario 3, se realiza la emisión de cada pasajero con DocIssuance\_IssueTicket por separado, recuperando el [PNR](#AcronimoPNR) con [PNR\_Retrieve](#PNRRETRIEVE) tras cada emisión.

Si es un escenario 4 o un 5 ha de recuperarse el [TST](#AcronimoTST) con Ticket\_DisplayTST para obtener información que necesaria para emitir. Si es un escenario 4 se realiza una llamada a DocIssuanceIssueCombined para cada [TSM](#AcronimoTSM), mientras que si es un escenario 5 se realiza una llamada a DocIssuanceIssueCombined para cada [TST](#AcronimoTST). Tras cada llamada a DocIssuanceIssueCombined se reabre el [PNR](#AcronimoPNR) con [PNR\_Retrieve](#PNRRETRIEVE) si todavía faltan [TSM](#AcronimoTSM)/[TST](#AcronimoTST) por emitir. Para los escenarios 4 y 5 se usan las mismas llamadas con diferentes parámetros en las peticiones.

Finalmente se recupera de nuevo el [PNR](#AcronimoPNR) para obtener números de billete nuevos y se cierra sesión con Security\_SignOut.

La [Figura 23](#Figura23) es la representación gráfica del flujo que acaba de explicarse.

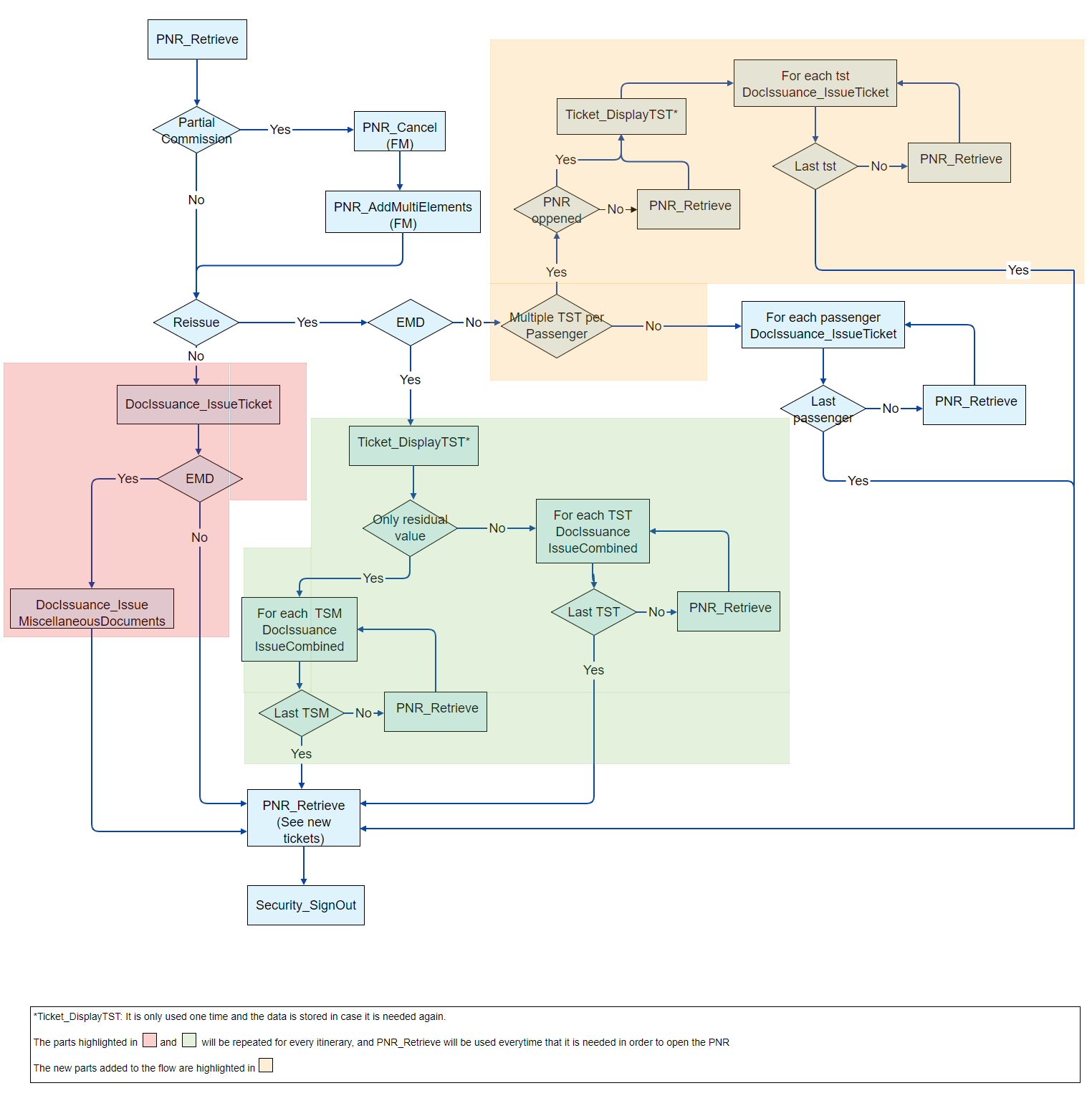


Figura 23: Diagrama de flujo de ejecución de la transacción EMR

## 4.3 Mockups del formulario de pruebas

En base a los requisitos funcionales del formulario de pruebas extraídos en la fase de análisis, se han diseñado los mockups a alto nivel representados en las figuras Figura 24, Figura 25, Figura 26 y Figura 27.

En la parte superior del formulario el usuario puede establecer el timeout (tiempo de corte de la transacción), la configuración y a qué entorno (test o producción) apuntarán las llamadas.

En la parte derecha el usuario puede copiar y pegar las peticiones [RQ](#AcronimoRQ) y respuestas [RS](#AcronimoRS) [XML](#AcronimoXML). Justo debajo el usuario puede visualizar los errores y warnings que se hayan dado durante la ejecución de la última transacción ejecutada en el formulario.

Más abajo, en la parte derecha, el usuario puede visualizar y seleccionar las opciones obtenidas en las transacciones de disponibilidad.

En la [Figura 24](#Figura24) se representa la pestaña de “Disponibilidad-Valoración” en la que el usuario cuenta con todos los parámetros para realizar las transacciones de Disponibilidad y Valoración.

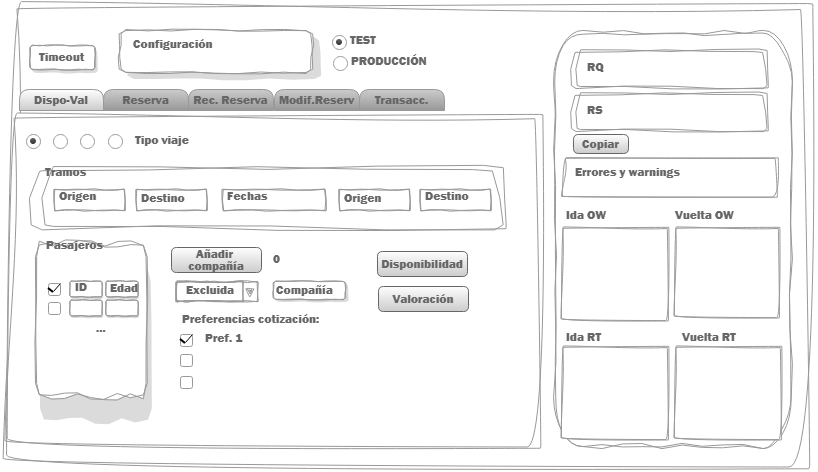


Figura 24: Mockup de la pestaña Disponibilidad-Valoración

En la [Figura 25](#Figura25) se representa la pestaña de “Reserva” en la que el usuario cuenta con todos los parámetros para realizar la transacción de Reserva.

En la [Figura 26](#Figura26) se representa la pestaña de “Recuperar Reserva” en la que el usuario cuenta con todos los parámetros para realizar las transacciones de Recuperar Reserva, Cancelar Reserva, Emitir y Void.

En la [Figura 27](#Figura27) se representa la pestaña de “Modificar Reserva” en la que el usuario cuenta con todos los parámetros para realizar las transacciones de [DMR](#AcronimoDMR), [RMR](#AcronimoRMR) y [EMR](#AcronimoEMR).

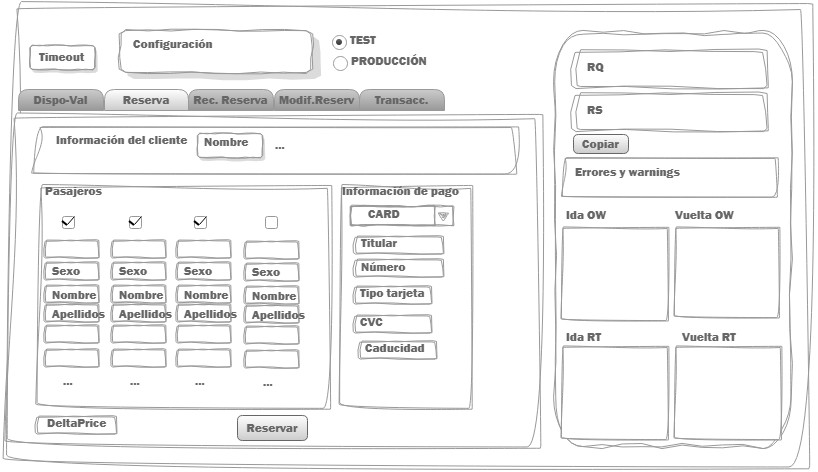
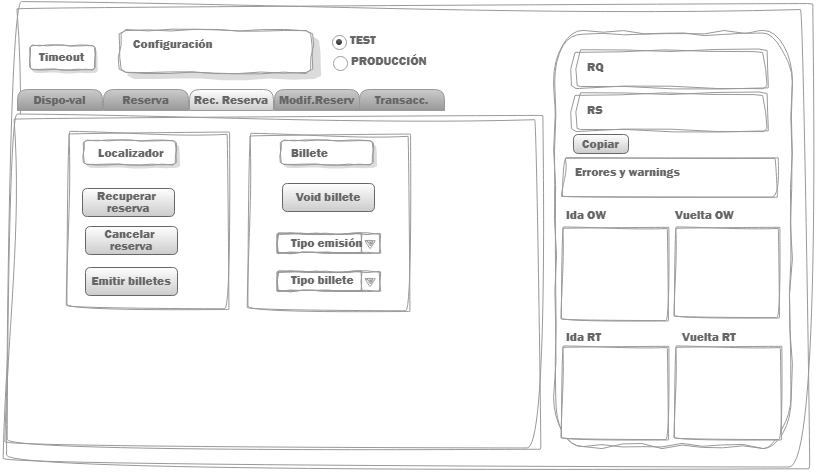


Figura 26: Mockup de la pestaña Recuperar Reserva

Figura 25: Mockup de la pestaña Reserva

## 

Figura 27: Mockup de la pestaña Modificar Reserva

# Programación

Este capítulo sirve como guía de información de la programación de los entregables a desarrollar:

* Transacciones [API](#AcronimoAPI) Transportes (véase en el apartado [5.1](#_5.1_Transacciones_de))
* Integración (véase en el apartado [5.2](#_5.2_Integración))
* Formulario de pruebas (véase en el apartado [5.3](#_5.3_Formulario_de))

Se adjuntan como anexo los ficheros correspondientes a estos tres paquetes de software desarrollados.

La solución de la integración contiene también el proyecto de tests unitarios (véase en el aparatado [5.4](#_5.4_Programación_de)).

## 5.1 Transacciones de la API de Transportes

En el apartado [5.1.1](#_5.1.1_Lenguaje/tecnología) de este capítulo, se indica el lenguaje de programación y librerías utilizadas para desarrollar las nuevas transacciones de la [API](#AcronimoAPI) de Transportes tal y como se han representado en el diseño expuesto en el capítulo [4.1](#_4.1_Transacciones_API) de esta memoria, y en base a los requisitos funcionales y no funcionales enumerados en el capítulo [3 Análisis](#_Análisis).

A continuación, en el apartado [5.2](#_5.2_Integración) se cita el código programado y se enumeran los ficheros desarrollados que se adjuntarán como anexo a esta memoria.

En el apartado [5.3](#_5.3_Formulario_de) se muestra el formulario de pruebas desarrollado y un ejemplo de uso. Finalmente, en el apartado [5.4](#_5.4_Programación_de) contenido dentro de este capítulo, se explican los escenarios de test desarrollados y su importancia para la mantenibilidad del código.

### 5.1.1 Lenguaje/tecnología

El lenguaje de programación con el que se programan las nuevas transacciones de la [API](#AcronimoAPI) de Transportes ya citadas en anteriores capítulos: [DMR](#AcronimoDMR), [RMR](#AcronimoRMR) y [EMR](#AcronimoEMR), así como los cambios necesarios introducidos en los elementos ya existentes en otras transacciones, es Visual Basic.

Se ha escogido este lenguaje porque el resto de transacciones que ya incluye la [API](#AcronimoAPI) está desarrollado con este mismo lenguaje. Sin embargo, una alternativa válida para desarrollar la misma [API](#AcronimoAPI) desde 0 en otro lenguaje puede ser perfectamente C#. Ofrece las mismas funcionalidades que ya se han utilizado para desarrollar la [API](#AcronimoAPI) de Transportes y además permite crear los ficheros necesarios para subirse en los frontales del servicio [WSDL](#AcronimoWSDL) que existe en el sistema de TravelgateX.

Visual Basic (o C#) es necesario para que el cliente pueda atacar a las integraciones por .dll a través del entorno de test que tiene TravelgateX para que éste realice pruebas antes de subir los cambios de la [API](#AcronimoAPI) en el entorno de pre-producción o producción.

Al ser un lenguaje del paradigma de programación “Programación Orientada a Objetos” ([POO](#AcronimoPOO)), se construyen las clases necesarias para la creación de las nuevas llamadas. Todas las clases, elementos y atributos que se detallan en el apartado siguiente ([5.1.2](#_5.1.2_Descripción_de)) son públicos.

Las tres llamadas desarrolladas son clases serializables, por lo tanto no se pueden heredar. Además las tres deben heredar de las clases padre TransportationBaseRQ para las peticiones y TransportationBaseRS para las respuestas.

Los elementos que incluyen las clases son del tipo XmlElementMultiLanguage y se indican los ElementName para cada idioma (español e inglés). Cuando un elemento tenga el mismo nombre para todos los idiomas con los que se vaya a poder utilizar la [API](#AcronimoAPI), es importante definirlos con el tipo XmlElement.

Los atributos son del tipo XmlAttributeMultiLanguage y, al igual que los elementos, se indica el nombre del atributo en los dos idiomas.

Los elementos que sean raíz de una transacción [XML](#AcronimoXML) deben definirse del tipo XmlRootMultiLanguage.

Para la programación de las clases, elementos y atributos, son suficientes las librerías del sistema (Microsoft) [XML](#AcronimoXML) y librerías internas de la [API](#AcronimoAPI) de Transportes, ya que se van instanciar clases que deben ser heredadas por las nuevas peticiones y respuestas desarrolladas.

### 5.1.2 Descripción de las transacciones programadas

En este capítulo se detallan los elementos y atributos que contienen las nuevas llamadas de la [API](#AcronimoAPI) de Transportes que se usarán en la integración de Amadeus para modificar reservas. Los elementos y atributos expuestos en las tablas [Tabla 23](#Tabla23), [Tabla 24](#Tabla24) y [Tabla 25](#Tabla25), no representan el código en si (programado en Visual Basic), sino la definición (cardinalidad y descripción) de cada uno de ellos. Éste formato es que se ha utilizado para documentar los cambios realizado en la [API](#AcronimoAPI) y que, como se ha establecido en el alcance del proyecto, es uno de los entregables que se han de presentar en TravelgateX.

En esta memoria se describe aproximadamente el 60% de los elementos que se han desarrollado para las nuevas transacciones [DMR](#AcronimoDMR), [RMR](#AcronimoRMR) y [EMR](#AcronimoEMR), los más importantes de cada transacción. El código de cada una de las transacciones se adjunta como anexo. En el código, cada uno de los elementos cuenta con una descripción /\*comentario\*/.

Tabla 23: Elementos y atributos más importantes de DMR

|  |  |  |  |
| --- | --- | --- | --- |
| **Elemento** | **Card.** | **Tipo** | **Significado** |
| DisponibilidadModificarReservaRQ | 1..1 | obj. | Nodo raíz |
| @tipoViaje | 1..1 | enum | Tipo de viaje: ([OW](#AcronimoOW), [RT](#AcronimoRT), [OJ](#AcronimoOJ) o [CT](#AcronimoCT)) |
| Localizador | 1..1 | obj. | Contiene información del localizador de la reserva |
| Localizador/@id | 1..1 | string | Identificador del localizador |
| Localizador/@tipoLocalizador | 1..1 | enum | Tipo de localizador: (PROVEEDOR, UNIVERSAL, EMISION, TRANSPORTISTA, REEMSOLSO, SERVICIO, CHECKIN) |
| Preferencias | 1..1 | obj. | Contiene las preferencias de la búsqueda |
| Preferencias/@claseCabina | 1..1 | enum | Clase cabina preferida para la búsqueda: (N, Y, C, F, CAMAROTE, YP) |
| Preferencias/@incluirLowCost | 1..1 | bool | Si es true, se buscarán vuelos de compañías lowcost |
| Preferencias/@soloDirectos | 1..1 | bool | Si es true, se buscarán solo vuelos directos (sin escalas) |
| Preferencias/CompaniasConexion | 0..1 | obj. | Incluye una lista de compañías |
| Preferencias/CompaniasConexion  /CompaniaConexion | 1..n | obj. | Compañía incluida o excluida de la búsqueda |
| Preferencias/CompaniasConexion  /CompaniaConexion/@cia | 1..1 | string | Código [IATA](#AcronimoIATA) compañía |
| Preferencias/CompaniasConexion  /CompaniaConexion/@modo | 1..1 | enum | Modo de filtro: (INCLUDED, EXCLUDED) |
| TramosDisponibilidad | 1..1 | obj. | Incluye una lista de tramos |
| TramosDisponibilidad/TramoDisponibilidad | 1..n | obj. | Incluye la información de un tramo de la búsqueda |
| TramosDisponibilidad/TramoDisponibilidad  /@id | 1..1 | int | Identificador único del tramo |
| TramosDisponibilidad/TramoDisponibilidad  /@fechaSalida | 1..1 | date | Fecha de salida |
| TramosDisponibilidad/TramoDisponibilidad  /@fechaLlegada | 1..1 | date | Fecha de llegada |
| TramosDisponibilidad/TramoDisponibilidad  /@horaSalida | 0..1 | string | Hora de salida |
| TramosDisponibilidad/TramoDisponibilidad  /@horaLlegada | 0..1 | string | Hora de llegada |
| TramosDisponibilidad/TramoDisponibilidad  /@accion | 1..1 | enum | Modificación a realizar al tramo: ([N](#ModifN), [K](#ModifK), [KF](#ModifKF), [R](#ModifR), [C](#ModifC), [A](#ModifA)) |
| TramosDisponibilidad/TramoDisponibilidad  /LocOrigen | 1..1 | obj. | Incluye la información de la localización de partida del vuelo |
| TramosDisponibilidad/TramoDisponibilidad  /LocOrigen/@codigo | 1..1 | string | Código [IATA](#AcronimoIATA) del aeropuerto o ciudad |
| TramosDisponibilidad/TramoDisponibilidad  /LocOrigen/@radio | 0..1 | int | Radio de búsqueda respecto a la localización |
| TramosDisponibilidad/TramoDisponibilidad  /LocDestino | 1..1 | obj. | Incluye la información de la localización de destino del vuelo |
| DisponibilidadModificarReservaRS | 1..1 | obj. | Nodo raíz |
| Transportes | 1..1 | obj. | Incluye información de segmentos y tarifas obtenidos en la búsqueda |
| Transportes/@totalTarifas | 1..1 | int | Número de tarifas obtenidas en la búsqueda |
| Transportes/Tarifas | 1..1 | obj. | Contiene una lista de tarifas |
| Transportes/Tarifas/Tarifa | 1..n | obj. | Contiene información de la tarifa |
| Transportes/Tarifas/Tarifa/@id | 1..1 | int | Identificador único de tarifa |
| Transportes/Tarifas/Tarifa/@tipoTarifa | 1..1 | enum | Tipo de tarifa: ([OW](#AcronimoOW), [RT](#AcronimoRT), [OJ](#AcronimoOJ) o [CT](#AcronimoCT)) |
| Transportes/Tarifas/Tarifa/@tieneObFees | 1..1 | bool | Si es true, la tarifa tiene cargos por pago con tarjeta |
| Transportes/Tarifas/Tarifa/Condiciones | 0..1 | obj. | Contiene una lista de condiciones |
| Transportes/Tarifas/Tarifa/Condiciones  /Condicion | 1..n | obj. | Condición de tarifa |
| Transportes/Tarifas/Tarifa/Condiciones  /Condicion/ConceptoTipificado | 1..1 | obj. | Información de la condición |
| Transportes/Tarifas/Tarifa/Opciones | 1..1 | obj. | Contiene una lista de opciones |
| Transportes/Tarifas/Tarifa/Opciones/Opcion | 1..n | obj. | Alternativas de tramos que incluye la tarifa |
| Transportes/Tarifas/Tarifa/Opciones/Opcion  /@id | 1..1 | int | Identificador único de opción |
| Transportes/Tarifas/Tarifa/Opciones/Opcion  /@refTramoDisponibilidad | 1..1 | int | Referencia al identificador único del tramoDisponibilidad de la petición [DMR](#AcronimoDMR) |
| Transportes/Tarifas/Tarifa/Opciones/Opcion  /@numEscalas | 1..1 | int | Cantidad de escalas del tramo |
| Transportes/Tarifas/Tarifa/Opciones/Opcion  /@placa | 1..1 | string | Compañía validadora de la reserva |
| Transportes/Tarifas/Tarifa/Opciones/Opcion  /@familiaTarifa | 0..1 | string | Familia de la tarifa que proporciona la compañía que vende el vuelo |
| Transportes/Tarifas/Tarifa/Opciones/Opcion  /ReferenciasSegmentos | 1..1 | obj. | Contiene una lista de referencias segmento |
| Transportes/Tarifas/Tarifa/Opciones/Opcion  /ReferenciasSegmentos/ReferenciaSegmento | 1..n | obj. | Contiene la referencia al identificador único de un segmento de la lista de segmentos obtenidos en la búsqueda e información aplicada a dicho segmento |
| Transportes/Tarifas/Tarifa/Opciones/Opcion  /ReferenciasSegmentos/ReferenciaSegmento  /@refSegmento | 1..1 | int | Referencia al identificador único de segmento |
| Transportes/Tarifas/Tarifa/Opciones/Opcion  /ReferenciasSegmentos/ReferenciaSegmento  /ClasesSegmento | 1..1 | obj. | Contiene una lista de clasesSegmento |
| Transportes/Tarifas/Tarifa/Opciones/Opcion  /ReferenciasSegmentos/ReferenciaSegmento  /ClasesSegmento/ClaseSegmento | 1..n | obj. | Contiene información aplicada al segmento y a un pasajero (hay una claseSegmento por pasajero) |
| Transportes/Tarifas/Tarifa/Opciones/Opcion  /ReferenciasSegmentos/ReferenciaSegmento  /ClasesSegmento/ClaseSegmento  /@claseCabina | 1..1 | enum | Clase cabina del vuelo |
| Transportes/Tarifas/Tarifa/Opciones/Opcion  /ReferenciasSegmentos/ReferenciaSegmento  /ClasesSegmento/ClaseSegmento  /@clase | 1..1 | string | Clase del vuelo |
| Transportes/Tarifas/Tarifa/Opciones/Opcion  /ReferenciasSegmentos/ReferenciaSegmento  /ClasesSegmento/ClaseSegmento  /@refPasajero | 1..1 | int | Referencia al identificador úncio del pasajero (dentro de ConfiguracionPasjero) |
| Transportes/Tarifas/Tarifa/Opciones/Opcion  /ReferenciasSegmentos/ReferenciaSegmento  /ClasesSegmento/ClaseSegmento  /@fareBasis | 0..1 | string | Código de la tarifa aplicado por el [GDS](#AcronimoGDS) o compañía validadora |
| Transportes/Tarifas/Tarifa/Opciones/Opcion  /ReferenciasSegmentos/ReferenciaSegmento  /ClasesSegmento/ClaseSegmento  /@fareType | 1..1 | enum | Tipo de tarifa: (PUB, PRI, NEGO, CORP) |
| Transportes/Tarifas/Tarifa/Opciones/Opcion  /ReferenciasSegmentos/ReferenciaSegmento  /ClasesSegmento/ClaseSegmento  /@avail | 1..1 | int | Cantidad de asientos disponibles para la ClaseSegmento |
| Transportes/Tarifas/Tarifa/DesgloseImporte | 1..1 | obj. | Contiene información del precio de la tarifa |
| Transportes/Tarifas/Tarifa/DesgloseImporte  /@moneda | 1..1 | string | Moneda de pago |
| Transportes/Tarifas/Tarifa/DesgloseImporte  /@importeTotal | 1..1 | dec | Importe total de la tarifa |
| Transportes/Tarifas/Tarifa/DesgloseImporte  /@importeNoComisionable | 1..1 | dec | Importe no comisionable de la tarifa |
| Transportes/Tarifas/Tarifa/DesgloseImporte  /@comision | 1..1 | dec | Importe de la comisión |
| Transportes/Tarifas/Tarifa/DesgloseImporte  /Cargos | 0..1 | obj. | Contiene una lista de cargos |
| Transportes/Tarifas/Tarifa/DesgloseImporte  /Cargos/Cargo | 1..n | obj. | Importes extra no incluidos en el importe total de la tarifa |
| Transportes/Tarifas/Tarifa/DesgloseImporte  /Cargos/Cargo/@tipo | 1..1 | string | Tipo de cargo (información del cargo) |
| Transportes/Tarifas/Tarifa/DesgloseImporte  /Cargos/Cargo/@importe | 1..1 | dec | Importe del cargo |
| Transportes/Tarifas/Tarifa/DesgloseImporte  /DesglosesPasajeros | 1..1 | obj. | Contiene una lista de DesglosePasajero |
| Transportes/Tarifas/Tarifa/DesgloseImporte  /DesglosesPasajeros/DesglosePasajero | 1..n | obj. | Contiene los importes desglosados por tipo de pasajero |
| Transportes/Tarifas/Tarifa/DesgloseImporte  /DesglosesPasajeros/DesglosePasajero  /@tipoPax | 1..1 | enum | Tipo de pasajero: ([ADT](#AcronimoADT), [CHD](#AcronimoCHD), [INF](#AcronimoINF)) |
| Transportes/Tarifas/Tarifa/DesgloseImporte  /DesglosesPasajeros/DesglosePasajero  /@importe | 1..1 | dec | Importe del tipo de pasajero |
| Transportes/Tarifas/Tarifa/DesgloseImporte  /DesglosesPasajeros/DesglosePasajero  /@tasas | 1..1 | dec | Tasas del tipo de pasajero |
| Transportes/Tarifas/Tarifa  /ConfiguracionesPasajeros | 1..1 | obj. | Contiene la lista de ConfiguracionPasajero |
| Transportes/Tarifas/Tarifa  /ConfiguracionesPasajeros  / ConfiguracionPasajeros | 1..n | obj. | Información del pasajero |
| Transportes/Tarifas/Tarifa  /ConfiguracionesPasajeros  /ConfiguracionPasajero/@id | 1..1 | int | Identificador único del pasajero |
| Transportes/Tarifas/Tarifa  /ConfiguracionesPasajeros  /ConfiguracionPasajero/@refPasajero | 1..1 | int | Referencia única del pasajero (se usa para referenciar las maletas incluidas por pasajero) |
| Transportes/Tarifas/Tarifa  /ConfiguracionesPasajeros  /ConfiguracionPasajero/@edad | 0..1 | int | Edad del pasajero |
| Transportes/Tarifas/Tarifa  /ConfiguracionesPasajeros  /ConfiguracionPasajero/@nacionalidad | 0..1 | string | Nacionalidad del pasajero |
| Transportes/Tarifas/Tarifa  /ConfiguracionesPasajeros  /ConfiguracionPasajero/@tipoPax | 1..1 | enum | Tipo del pasajero |
| Transportes/Tarifas/Tarifa  /ConfiguracionesPasajeros  /ConfiguracionPasajero/Bonificaciones | 0..1 | obj. | Bonificaciones aplicadas al pasajero |
| Transportes/Tarifas/Tarifa  /ConfiguracionesPasajeros  /ConfiguracionPasajero/Bonificaciones  /@residente | 0..1 | enum | Tipo de residente: (BP, BI, CE, RE…) |
| Transportes/Tarifas/Tarifa  /ConfiguracionesPasajeros  /ConfiguracionPasajero/Bonificaciones  /@familiaNumerosa | 0..1 | enum | Tipo de familia numerosa: (F1, F2) |
| Transportes/Segmentos | 1..1 | obj. | Contiene una lista de segmentos |
| Transportes/Segmentos/Segmento | 1..n | obj. | Contiene información del vuelo |
| Transportes/Segmentos/Segmento/@id | 1..1 | int | Identificador único del vuelo |
| Transportes/Segmentos/Segmento  /@numTransporte | 1..1 | string | Número de vuelo (p.e. IB98950) |
| Transportes/Segmentos/Segmento  /@operatingCia | 1..1 | string | Aerolínea que opera el vuelo |
| Transportes/Segmentos/Segmento  /@marketingCia | 1..1 | string | Aerolínea que vende el vuelo |
| Transportes/Segmentos/Segmento  /@terminalSalida | 0..1 | string | Terminal de salida |
| Transportes/Segmentos/Segmento  /@terminalLlegada | 0..1 | string | Terminal de llegada |
| Transportes/Segmentos/Segmento  /@fechaSalida | 1..1 | date | Fecha de salida |
| Transportes/Segmentos/Segmento  /@fechaLlegada | 1..1 | date | Fecha de llegada |
| Transportes/Segmentos/Segmento  /@tieneParadaTecnica | 1..1 | bool | Si es true, el vuelo tiene una o más paradas técnicas |
| Transportes/Segmentos/Segmento  /ParadaTecnicaDetalles | 0..1 | obj. | Detalles de las paradas técnicas |
| Transportes/Segmentos/Segmento  /LocOrigen | 1..1 | obj. | Contiene información de la localización de partida del vuelo |
| Transportes/Segmentos/Segmento  /LocDestino | 1..1 | obj. | Contiene información de la localización de llegada del vuelo |
| Transportes/Tickets | 1..1 | obj. | Contiene una lista de tickets (billetes) |
| Transportes/Tickets/Ticket | 1..n | obj. | Contiene información del ticket |
| Transportes/Tickets/Ticket/@id | 1..1 | int | Identificador único del ticket |
| Transportes/Tickets/Ticket/@numTicket | 1..1 | string | Número del ticket |
| Transportes/Tickets/Ticket/@nomPax | 0..1 | string | Nombre del pasajero asociado al ticket |
| Transportes/Tickets/Ticket/@tipoPax | 1..1 | enum | Tipo de pasajero asociado al ticket |
| Transportes/Tickets/Ticket/@tipoTicket | 1..1 | enum | Tipo de ticket: (eTicket, Extra) |
| Transportes/Tickets/Ticket/@estado | 1..1 | enum | Estado en el que se encuentra el ticket: (Open, Confirmed, Voided, Refunded) |
| Transportes/Localizadores | 1..n | obj. | Contiene información del localizador de la reserva |

Tabla 24: Elementos y atributos más importantes de RMR

|  |  |  |  |
| --- | --- | --- | --- |
| **Elemento** | **Card.** | **Tipo** | **Significado** |
| ReservaModificarReservaRQ | 1..1 | obj. | Nodo raíz |
| @deltaPrice | 1..1 | dec | Importe diferencial aceptado entre el precio de la tarifa obtenida en [DMR](#AcronimoDMR) y el precio que se obtenga en la [RMR](#AcronimoRMR) |
| Desgloses | 1..1 | obj. | Contiene una lista de desgloses |
| Desgloses/Desglose | 1..n | obj. | Contiene información de los segmentos y la tarifa |
| Desgloses/Desglose/@id | 1..1 | int | Identificador único del desglose |
| Desgloses/Desglose/@refTarifa | 1..1 | int | Referencia al identificador único de la tarifa de [DMR](#AcronimoDMR) |
| Desgloses/Desglose/@tieneObFees | 1..1 | bool | Si es true, la tarifa tiene cargos por pago con tarjeta |
| Desgloses/Desglose/@placa | 1..1 | string | Compañía validadora de la reserva |
| Desgloses/Desglose/Condiciones | 0..1 | obj. | Contiene una lista de condiciones |
| Desgloses/Desglose/Tramos | 1..1 | obj. | Contiene una lista de tramos |
| Desgloses/Desglose/Tramos/Tramo | 1..n | obj. | Contiene información de lo opción seleccionada en [DMR](#AcronimoDMR) |
| Desgloses/Desglose/Tramos/Tramo/@id | 1..1 | int | Identificador único de tramo |
| Desgloses/Desglose/Tramos/Tramo  /@familiaTarifa | 0..1 | string | Familia de la tarifa que proporciona la compañía que vende el vuelo |
| Desgloses/Desglose/Tramos/Tramo  /SegmentosTramo | 1..1 | obj. | Contiene una lista de segmentoTramo |
| Desgloses/Desglose/Tramos/Tramo  /SegmentosTramo/SegmentoTramo | 1..n | obj. | Contiene información de los vuelos y las clases asociadas a cada uno |
| Desgloses/Desglose/Tramos/Tramo  /SegmentosTramo/SegmentoTramo/@id | 1..1 | int | Identificador único de segmentoTramo |
| Desgloses/Desglose/Tramos/Tramo  /SegmentosTramo/SegmentoTramo  /Segmento | 1..1 | obj. | Contiene información del vuelo |
| Desgloses/Desglose/Tramos/Tramo  /ClasesSegmento | 1..1 | obj. | Contiene una lista de clases segmento |
| Transportes/Tarifas/Tarifa/DesgloseImporte | 1..1 | obj. | Contiene información del precio de la tarifa |
| Desgloses/Desglose/ConfiguracionesPasajero | 1..1 | obj. | Contiene una lista de configuraciones pasajero |
| InfoPago | 0..1 | obj. | Información de pago |
| InfoPago/@tipoPago | 1..1 | string | Tipo de pago: (CARD, CASH) |
| InfoPago/DatosPago | 0..1 |  | Contiene una lista de datos de pago |
| InfoPago/DatosPago/DatoPago | 1..n |  | Contiene información del método de pago |
| InfoPago/DatosPago/DatoPago/@numPlazos | 1..1 | int | Número de plazos para realizar el pago |
| InfoPago/DatosPago/DatoPago/Tarjeta | 1..1 |  | Información de la tarjeta |
| InfoPago/DatosPago/DatoPago/Tarjeta  /@tipoProv | 1..1 | string | Tipo de proveedor (p.e. Master Card o Visa) |
| InfoPago/DatosPago/DatoPago/Tarjeta  /@titular | 1..1 | string | Titular de la tarjeta |
| InfoPago/DatosPago/DatoPago/Tarjeta  /@numero | 1..1 | int | Número de la tarjeta |
| InfoPago/DatosPago/DatoPago/Tarjeta  /@cvc | 0..1 | int | Código de seguridad |
| InfoPago/DatosPago/DatoPago/Tarjeta  /@mesCaducidad | 1..1 | int | Mes de caducidad |
| InfoPago/DatosPago/DatoPago/Tarjeta  /@añoCaducidad | 1..1 | int | Año de caducidad |
| Tickets | 1..1 | obj. | Contiene una lista de tickets |
| Localizador | 1..1 | obj. | Contiene información del localizador de la reserva |
| ReservaModificarReservaRS | 1..1 | obj. | Nodo raíz |
| @tipoEmision | 1..1 | string | Tipo de emisión que ha de realizarse en [EMR](#AcronimoEMR) |
| Localizadores | 1..n | obj. | Contiene información del localizador de la reserva |
| Tickets | 1..1 | obj. | Contiene una lista de tickets |
| Pasajeros | 1..1 | obj. | Contiene una lista de pasajeros |
| Pasajeros/Pasajero | 1..n | obj. | Contiene información del pasajero |
| Pasajeros/Pasajero/@id | 1..1 | int | Identificador único del pasajero |
| Pasajeros/Pasajero/@tratamiento | 1..1 | enum | Tratamiento del pasajero (MR, MRS ,[INF](#AcronimoINF), [CHD](#AcronimoCHD)) |
| Pasajeros/Pasajero/@nombre | 1..1 | string | Nombre |
| Pasajeros/Pasajero/@apellidos | 1..1 | string | Apellidos |
| Pasajeros/Pasajero/@fechaNacimiento | 1..1 | date | Fecha de nacimiento |
| Pasajeros/Pasajero/@tipoDocumento | 0..1 | enum | Tipo de documentación de identificación: (NATIONAL\_ID, PASSPORT, RESIDENT\_ID, FOREIGN\_PASSPORT, BIRTH\_NOTIFICATION) |
| Pasajeros/Pasajero/@idDocumento | 0..1 | string | Número del documento de identificación |
| Pasajeros/Pasajero/@caducidadDocumento | 0..1 | date | Fecha de caducidad del documento de identificación |
| Pasajeros/Pasajero/@nacionalidad | 0..1 | string | Nacionalidad del pasajero |
| Pasajeros/Pasajero/@sexo | 0..1 | string | Sexo del pasajero |
| Pasajeros/Pasajero/ BonificacionesAplicadas | 0..1 | obj. | Bonificaciones aplicadas al pasajero |
| Pasajeros/Pasajero/BonificacionesAplicadas  /@municipioResidente | 0..1 | string | Municipio del residente |
| Pasajeros/Pasajero/BonificacionesAplicadas  /@codigoDocumentoFamiliaNumerosa | 0..1 | string | Documento de familia numerosa |
| Pasajeros/Pasajero/BonificacionesAplicadas  /@comunidadFamiliaNumerosa | 0..1 | string | Comunidad de familia numerosa |
| Pasajeros/Pasajero/BonificacionesAplicadas  /@codigoCertificadoResidente | 0..1 | string | Código de certificado de residente |
| Factura | 1..1 | obj. | Factura de la reserva |
| Factura/@compania | 1..1 | string | Compañía validadora de la reserva |
| Factura/@lastTicketingDate | 1..1 | date | Fecha del último checkin |
| Factura/DesgloseImporte | 1..1 | obj. | Contiene información del precio de la tarifa |

Tabla 25: Elementos y atributos más importantes de EMR

|  |  |  |  |
| --- | --- | --- | --- |
| **Elemento** | **Card.** | **Tipo** | **Significado** |
| EmitirModificarReservaRQ | 1..1 | obj. | Nodo raíz |
| @tipoEmision | 1..1 | string | Tipo de emisión que ha de realizarse en [EMR](#AcronimoEMR) |
| @localizador | 1..1 | string | Localizador de la reserva |
| Tickets | 1..1 | obj. | Contiene una lista de billetes |
| EmitirModificarReservaRS | 1..1 | obj. | Nodo raíz |
| Tickets | 1..1 | obj. | Contiene una lista de billetes |
| Localizadores | 1..n | obj. | Contiene información del localizador de la reserva |

## 5.2 Integración

En este apartado se explica el lenguaje y la tecnología utilizada para programar la integración y se destacan las funcionalidades desarrolladas en la integración más destacadas. Para ello apartado se separa en dos sub-apartados: [5.2.1 Lenguaje/tecnología](#_5.2.1_Lenguaje/tecnología) y [5.2.2 Funcionalidades destacadas.](#_5.2.2_Funcionalidades_destacadas)

### 5.2.1 Lenguaje/tecnología

El lenguaje de programación con el que se programa la integración es Visual Basic. Se ha escogido este lenguaje porque la integración ya estaba desarrollada en este lenguaje.

Visual Basic (o C#) es necesario para que el cliente pueda atacar a las integraciones por .dll a través del entorno de test que tiene TravelgateX para que éste realice pruebas antes de subir los cambios de la integración en el entorno de pre-producción o producción.

### 5.2.2 Funcionalidades destacadas

A continuación se presentan las funcionalidades más destacadas de la Modificación de Reservas.

#### TimeoutManager

Es la funcionalidad que controla el tiempo de ejecución de la transacción. Se declara indicando el tiempo de corte (timeout) especificado en el atributo @timeout de la petición.

* GetRemainingMilliseconds: devuelve el tiempo que falta para llegar al timeout.
* GetEllapsedMilliseconds: devuelve el tiempo que ha pasado desde que ha empezado la ejecución de la transacción.

#### ConnectionManager

Es la funcionalidad que se encarga de crear el cliente [SOAP](#AcronimoSOAP) y utilizar dicho cliente para realizar la petición al proveedor. Contiene todas las llamadas de Amadeus desarrolladas en la integración. Se han desarrollado derivados de esta funcionalidad para para paquete de Web Services de Amadeus (p.e. ConnectionManagerBook para todas las llamadas del paquete de Web Services de tarificación y reserva de recomendaciones de Amadeus).

#### BookingRetrieve

Clase que monta toda la información que se recibe al recuperar la reserva.

#### Eligibility

Clase que se encarga de construir la petición para la llamada [Ticket\_CheckEligibility](#TICKET_ELIGIBILITY) y procesar y parsear la respuesta recibida. Si un ticket no es modificable, se añadiría un error de “Ticket no modificable” en la respuesta de [DMR](#AcronimoDMR).

#### Travelboard

Clase que se encarga de construir la petición para la llamada [Ticket\_ATCShopperMasterPricer TravelBoardSearch](#TICKET_ATCMP) y procesar y parsear la respuesta recibida. Monta todas las tarifas y los segmentos a partir de los datos obtenidos en la respuesta.

#### BookingMake

Clase que monta toda la información para realizar la reserva de segmentos y pasajeros. Además se utiliza para montar las facturas de la misma a partir de los datos obtenidos

#### RepricePNR

Clase que se encarga de construir la petición para la llamada [Ticket\_RepricePNR WithBookingClass](#TICKET_REPRICE) y procesar y parsear (transformar) la respuesta recibida.

#### ReissueConfirmedPricing

Clase que se encarga de construir la petición para la llamada [Ticket\_ReissueConfirmed PricingTicketInfo](#TICKET_REISSUE) y procesar y parsear la respuesta recibida.

#### GetTipoEmision

Funcionalidad que calcula el tipo de emisión que ha de realizarse en la transacción [EMR](#AcronimoEMR) a partir de:

* repriceType: tipo de reprice que Amadeus indica
* isThereEMD: Si el [PNR](#AcronimoPNR) tiene cargos [EMD](#AcronimoEMD) o no
* tipoEMD: tipo de [EMD](#AcronimoEMD) que hay en el [PNR](#AcronimoPNR)
* formOfPayment: forma de pago con el que se abone el importe de la modificación de la reserva
* addictionalCollection: la modificación tiene importe adicional

Se obtendrá uno de los siguientes tipos de emisión:

* Revalidate without [EMD](#AcronimoEMD)
* Revalidate with [EMD](#AcronimoEMD) without additional collection
* Reissue without [EMD](#AcronimoEMD) with additional collection
* Reissue with [EMD](#AcronimoEMD) only penalti
* Reissue with [EMD](#AcronimoEMD) only residual value without additional collection
* Reissue with [EMD](#AcronimoEMD) only residual value
* Reissue with [EMD](#AcronimoEMD) with penalty and residual value
* Reissue with [EMD](#AcronimoEMD) with penalty and residual value without additional collection

#### DoEndTransactionAndRetrieve

Confirma los cambios realizados en el [PNR](#AcronimoPNR) y lo reabre.

#### DoIgnoreAndRetrieve

Ignora los cambios realizados en el [PNR](#AcronimoPNR) y lo reabre.

#### ATC\_IssueTicket

Realiza la revalidación o reemisión de los billetes dependiendo del tipo de emisión obtenida del atributo @tipoEmision de la petición [EMR](#AcronimoEMR).

#### ATC\_IssueCombined

Realiza la reemisión de los billetes dependiendo de si la modificación de [RMR](#AcronimoRMR) ha generado un valor residual o una penalización por modificación.

## 5.3 Formulario de pruebas

En el sub-apartado [5.3.1](#_5.3.1_Lenguaje/tecnología) se describe el lenguaje y tecnología utilizada para programar el formulario de pruebas que se muestra con un ejemplo de uso a través de un vídeo en el apartado [5.3.2](#_5.3.2_Ejemplo_de).

### 5.3.1 Lenguaje/tecnología

El lenguaje de programación con el que se programa el formulario de pruebas es Visual Basic. Se ha escogido este lenguaje por dos razones: es el lenguaje que tenemos que utilizar en las transacciones de la [API](#AcronimoAPI) y en la integración y consecuentemente se tiene más práctica y soltura. Otra razón de peso es la ligera mayor velocidad con la que se programa al carecer de puntos y coma, se tienen que utilizar menos paréntesis y apenas se necesitan abrir o cerrar llaves.

### 5.3.2 Ejemplo de uso (vídeo)

Anexo a esta memoria se puede encontrar el software del formulario de pruebas y un vídeo mostrando su uso.

## 5.4 Programación de tests unitarios

En este apartado se destaca la importancia de crear tests unitarios para la integración ([5.4.1](#_5.4.1_Importancia)) y qué lenguaje de programación es el elegido para programar dichos tests. En el sub-apartado [5.4.2](#_5.4.2_Lenguaje/tecnología) también se enumeran y se describen algunas de las funcionalidades que proporcionan algunas librerías del lenguaje seleccionado.

### 5.4.1 Importancia

En el proyecto de test, se incluyen tests que comprueban el funcionamiento de transacciones o funcionalidades unitarias dentro de la integración. Casi todas las transacciones que tiene el [GDS](#AcronimoGDS) tienen su clase de test con uno o más tests unitarios.

Estos tests no realizan la llamada al proveedor en ningún entorno, sino que es el desarrollador el que especifica las transacciones que quiere testear. Así, por ejemplo, si se quiere comprobar que en unas determinadas condiciones, una reserva funciona correctamente, se puede hacer que el test se ejecute simulando el comportamiento real de la reserva, pero sin realizar la transacción. Esto se consigue sobrescribiendo las funciones que contactan con el proveedor asignándose siempre la misma respuesta (el string). Es muy útil, por ejemplo, cuando una reserva ha fallado y no se puede reproducir en producción. Al pasarle al test, las trazas del error, se reproducirá el comportamiento de la integración en las mismas condiciones pero sin hacer la llamada de verdad.

Cada test tiene al menos un string con la respuesta de alguna de las llamadas de la transacción, lo que conlleva que una clase de test puede llegar a tener una enorme cantidad de texto que llega incluso a ralentizar el PC con el que el desarrollador ejecute los tests. Este es uno de los motivos por los cuales es conveniente seguir un formato estándar al crear cada test. Otra razón de peso es la fácil legibilidad que implica que todos los tests en cualquier [GDS](#AcronimoGDS) sigan el mismo formato.

Pese a que los tests ayudan a que no se fastidie la integración al mantenerla, éstos también conllevan en ocasiones un mantenimiento. Ya que gran parte de los tests comprueban el funcionamiento de una transacción completa, cuando el flujo de una transacción cambia, entonces los tests también tendrán que cambiarse. Hay muchas otras causas que pueden provocar que tenga que actualizarse un test. Pese a todo, cuando un test falla, lo más probable es que se deba a algún cambio en la integración que el desarrollador haya hecho mal.

### 5.4.2 Lenguaje/tecnología

El lenguaje de programación con el que se programa el formulario de pruebas es C#.

#### Moq

Librería que permite la utilización de “mocks”, que no son más que objetos simulados que imitan el comportamiento de objetos reales. Se pueden comportamientos dependiendo de los parámetros que se le pasen al objeto “mock” e incluso ejecutar acciones complejas accediendo al mismo parámetro proporcionado al método simulado.

*Ejemplo de uso en la integración:*

* Creación de un mock ConnectionManager
* Setup de las llamadas de ConnectionManager para que devuelvan el objeto que deseemos (p.e. callPNR\_Retrieve que devuelva el [PNR\_Reply](#PNRRETRIEVE) que se indique en el return).
* Creación de un mock [DMR](#AcronimoDMR) para llamar al método process que ejecuta la transacción.

# Pruebas

En el apartado [6.1](#_6.1_Certificación_Amadeus) de este capítulo se describen las pruebas que se han realizado para certificar el desarrollo por parte de Amadeus. A continuación, en el apartado 6.2, se detallan algunas de las pruebas que se realizaron para comprobar el buen funcionamiento de las transacciones y detectar errores no controlados anteriormente durante el desarrollo.

## 6.1 Certificación Amadeus

La certificación del desarrollo [ATC](#AcronimoATC) Shopper constó de tres fases. Primero el desarrollador realiza las pruebas sobre los escenarios que Amadeus indica. A continuación se presentan los diagramas de flujo y los resultados de las pruebas. Finalmente Amadeus comprueba los resultados y pasa el checklist de los criterios de aceptación.

### 6.1.1 Escenarios a probar y presentar

Amadeus describe al desarrollador los tests que han de realizarse obligatoriamente para efectuar la certificación de la aplicación.

Todos y cada uno de los tests que se realizaron durante la certificación se hicieron sobre el entorno de producción, con reservas reales, de tal forma que se pudiera demostrar con seguridad, el correcto funcionamiento de la aplicación. Solo una vez realizadas las pruebas y efectuadas con éxito, Amadeus procede a revisar dichas pruebas y repasar los criterios de aceptación de las tablas [15](#Tabla15), [16](#Tabla16) y [17](#Tabla17) pre-acordados.

Estos son los escenarios que habían de probarse:

1. Residual value donde la aerolínea no permite emisión del [EMD](#AcronimoEMD)

Test recomendado con American Airlines en itinerario MIA-LAX-MIA, cambiando la fecha de la ida y la clase.

1. Revalidación sin [EMD](#AcronimoEMD)

Test recomendado con Finnair en itinerario HEL-ARN-HEL, cambiando la fecha de la ida y manteniendo la clase.

1. Revalidación con [EMD](#AcronimoEMD)

Test recomendado con Finnair en itinerario HEL-ARN-HEL, cambiando la fecha de la ida y la clase.

1. Reemisión con [EMD](#AcronimoEMD) de penalización

Test recomendado con Air Europa en itinerario MAD-AGP-MAD, cambiando la fecha de la ida y manteniendo la clase.

1. Reemisión sin [EMD](#AcronimoEMD) de penalización

Test recomendado con Iberia en itinerario MAD-LHR-MAD, cambiando la fecha de la ida y manteniendo la clase.

1. Residual value, sin penalización y sin additional collection

Test recomendado con Air France en itinerario MAD-SIN-MAD, cambiando el itinerario y la fecha de la vuelta y cambiando la clase.

1. Residual value, con penalización y sin additional collection

Test recomendado con Avianca en itinerario MAD-MED-MAD, cambiando la fecha de la ida y cambiando la clase.

1. Residual value, con penalización y con additional collection

Test recomendado con Avianca en itinerario PPT-MAD-PPT, cambiando la MAD por AGP para cualquier fecha.

1. Residual value, sin penalización y con additional collection

Test recomendado con Air France en itinerario PPT-MAD-PPT, cambiando la MAD por AGP para cualquier fecha.

Para algunos de los casos, se precisó de la intervención de Amadeus y de las aerolíneas Iberia y Air Europa ya que, en un momento dado del flujo, una aerolínea ha de cambiar el estado de un billete para completar el test.

Con el fin de comprobar que los tests realizados por el desarrollador en TravelgateX sean correctos, Amadeus precisa de los localizadores de las reservas que hayan sido modificadas. De esta forma, gracias a que las reservas y los billetes quedan guardados en el [GDS](#AcronimoGDS), el técnico puede repasar todos los cambios que hayan sido sobre el [PNR](#AcronimoPNR).

### 6.1.2 Primer intento

Tras revisar los tests realizados, Amadeus informó al desarrollador de la existencia de una anomalía en el flujo de transacciones que supondrían un warning en la certificación. El problema se encontrada en el uso de sesiones continuas (en Amadeus *stateful*) en llamadas en las que no eran necesarias.

Amadeus recomendó la modificación de los flujos para adaptar algunas de las transacciones al uso de peticiones sin sesión (en Amadeus *stateless*). Dicha modificación supondría la modificación de los flujos de las transacciones [DMR](#AcronimoDMR) y [RMR](#AcronimoRMR), lo que implica por lo tanto un rediseño de los mismos y la programación de los cambios en la integración.

La presencia de un warning en una certificación, no impide a la agencia el uso de la aplicación desarrollada en el entorno de producción (desplegar la aplicación para el uso comercial). Sin embargo, se incluiría la aplicación dentro de una lista de “defectuosos” para tener constancia de una anomalía en el software.

### 6.1.2 Segundo intento

El desarrollador de TravelgateX, juntamente con Logitravel, decidieron retrasar el despliegue de la aplicación en producción para solucionar el warning obtenido en el primer intento. Esto supuso solamente un retraso de 5 días. En la [Figura 28](#Figura28) se puede ver en el diagrama de Gantt la diferencia temporal que supuso la corrección del warning.

De todas formas, se mantuvo la reunión de cierre del proyecto para la fecha planificada.

Una vez implementados los cambios, se realizaron de nuevo los tests sobre los 9 escenarios presentados en el apartado [6.1.1](#_6.1.1_Escenarios_a) y se modificaron los diagramas de flujo para cerrar sesión con Signout\_Session antes de todas las llamadas *stateless*.

Al cabo de unos días Amadeus comunicó a los responsables del desarrollo en TravelgateX y Logitravel, el éxito de la certificación, por lo que se podía empezar a preparar el despliegue de la integración en producción (capítulo 7).

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Mes** | Día | Planificación y gestión del proyecto | Análisis | Diseño | Construcción y documentación | Testing y despliegue |
| **Marzo ‘17** | 06 |  |  |  |  |  |
| 13 |  |  | **WBS-01: 14/03/2017 – Kick-Off Amadeus** |  |  |
| 20 | WBS-02 |  |  |  |  |
| 27 |  |  |  |  |  |
| **Abril ‘17** | 03 |  | WBS-06  WBS-05 |  |  |  |
| 10 |  |  |  | WBS-10 |  |
| 17 |  |  | WBS-07  WBS-08 |  |  |
| 24 |  |  | WBS-09 | WBS-11 |  |
| **Mayo ‘17** | 01 |  |  |  | WBS-12 | WBS-14 |
| 08 |  |  |  | WBS-13 |  |
| 15 |  |  |  |  |  |
| 22 |  |  |  |  |  |
| 29 |  |  |  |  |  |
| **Junio ‘17** | 05 | WBS-03 |  |  |  |  |
| 12 |  |  |  |  |  |
| 19 |  |  |  |  |  |
| 26 |  |  |  |  | WBS-16  WBS-15 |
| **Julio ‘17** | 03 |  |  |  |  |  |
| 10 |  |  |  |  |  |
| 17 |  |  |  |  |  |
| 24 |  |  |  |  | WBS-17 |
| 31 |  |  |  |  |  |
| **Agosto ‘17** | 07 |  |  |  |  |  |
| 14 |  |  |  |  |  |
| 21 |  |  |  |  |  |
| 28 |  |  |  |  |  |
| **Septiembre ‘17** | 04 |  |  |  | **Retraso de 5 días (Certificación)** |  |
| 11 |  |  |  | **WBS-17: 12/09/2017 – Despliegue** |  |
| 18 |  | **WBS-04: 11/09/2017 – Reunión de cierre** |  |  |  |
| 25 |  |  |  |  |  |

Figura 28: Diagrama de Gantt (desviaciones temporales)

## 6.2 Fase interna de pruebas

Antes de empezar a preparar el despliegue de la integración en producción, es recomendable la realización de pruebas sobre todos los flujos de la integración, incluidos los 3 nuevos. Esta tarea es crucial para la puesta en producción del desarrollo, puesto que a partir de entonces, clientes reales pueden usar la aplicación.

Los tests realizados ayudan también a detectar errores que se han pasado por alto en la certificación. Esta fase sirve también para limpiar y mejorar la efectividad, rendimiento y legibilidad del código desarrollado, de cara a la puesta en producción de la aplicación.

Esta fase es más sencilla con la presencia de tests unitarios ([5.4](#_5.4_Programación_de)) previos al desarrollo. Si se ha modificado incorrectamente alguna parte de la integración que tuviera un test unitario, es muy probable que este falle, permitiendo así ver el error antes de poner nada en producción.

Pese a que la integración tenga tests unitarios, es importante comprobar cada uno de los flujos de transacciones correctamente y en el entorno de producción de Amadeus. Algunas de las pruebas que es recomendable realizar y que pueden ayudar a encontrar errores pasados por alto por el desarrollador y por la certificación de Amadeus son los siguientes:

* Múltiples pasajeros de distintas edades
* Itinerarios con escalas
* Pago con tarjeta
* Eliminar y añadir tramos a la reserva (en la certificación no se prueba esta funcionalidad que sin embargo si de desarrolló y estará en producción)
* Provocar errores a propósito para mejorar la tipificación de los mismos

# Despliegue

Todas las líneas de integraciones de TravelgateX tienen programadas automáticamente una serie de pruebas en el entorno de test de cada proveedor, una vez que se realiza la subida de un cambio en un repositorio git remoto.

De esta forma cuando un de

## Referencias bibliográficas

1. SiteMinder, *Sistema de distribución global: cómo dar sentido a tu gestión de ingresos*, SiteMinder, 2017.
2. Sabre. (s.f.). En Wikipedia. Recuperado el 09 de Junio de 2018 de <https://es.wikipedia.org/wiki/Sabre>
3. Amadeus IT Group. (s.f.). En Wikipedia. Recuperado el 09 de Juno de 2018 de <https://es.wikipedia.org/wiki/Amadeus_IT_Group>
4. Galileo GDS. (s.f.). En Wikipedia. Recuperado el 09 de Junio de 2018 de <https://en.wikipedia.org/wiki/Galileo_GDS>
5. M. Strauss, The Difference between CRS and GDS in travel industry, Travel Industry Blog, 2013.