



**Universitat de les  
Illes Balears**

Escola Politècnica Superior

**Memòria del Treball de Fi de Grau**

**Desarrollo de un programa que permita la  
modificación de reservas en un GDS (Global  
Distribution System)**

Xavier Solano Trujillo

**Grau de Edificació Enginyeria Informàtica Telemàtica**

Any acadèmic 2017-18

DNI de l'alumne: 43181627R

Treball tutelat per Antoni Lluís Mesquida  
Departament de

S'autoritza la Universitat a incloure aquest treball en el Repositori Institucional per a la seva consulta en accés obert i difusió en línia, amb finalitats exclusivament acadèmiques i d'investigació	Autor	Tutor
	Sí X	No X

Paraules clau del treball:

GDS, Amadeus, Logitravel, TravelgateX, integración, reserva



# SUMARIO

Lista de tablas.....	iv
Lista de figuras .....	v
Lista de acrónimos.....	vi
Resumen.....	vii
Introducción.....	1
1.1 Contextualización.....	1
1.1.1 TravelgateX.....	1
1.1.2 Logitravel.....	2
1.1.3 Problemática: ¿por qué surge la necesidad de modificar una reserva? .....	3
1.1.4 Aplacar la necesidad.....	3
1.1.5 Objetivos principales y secundarios y equipo del proyecto.....	4
1.1.6 Funcionamiento de una integración .....	4
1.1.7 Principales transacciones en una integración de vuelos .....	5
1.1.8 Relación entre proveedores, TravelgateX y Logitravel .....	8
1.1.9 GDS: Global Distribution System.....	9
1.1.10 Conceptos y terminología del GDS Amadeus .....	10
1.1.11 Web Services y WSDL.....	11
1.1.12 Estado actual de los webservices de Amadeus.....	172
1.1.13 Estado previo de la API de transportes de TravelgateX.....	13
1.2 Definición del problema .....	13
1.2.1 Unidades que forman una reserva.....	13
1.2.2 En qué consiste la modificación de una reserva.....	15
1.3. Respuesta al problema .....	16
1.3.1. Idea básica del flujo implementado para modificar reservas.....	16
1.3.2. Productos a desarrollarse.....	17
Plan del proyecto .....	22
2.1 Alcance del proyecto.....	22
2.1.1 Reunión inicial (Kick-Off) .....	23
2.1.2 Requisitos .....	24
2.1.3 Estructura de descomposición del trabajo (EDT).....	27
2.1.4 Entregables.....	27
2.1.5 Criterios de aceptación .....	28
2.2 Cronograma.....	30
2.2.1 Resumen del cronograma.....	30
2.2.2 Planificación de la duración de los paquetes de trabajo.....	30
2.2.3 Diagrama de Gantt.....	33
2.3 Partes interesadas .....	35
2.3.1 Interesados internos.....	35
2.3.2 Interesados externos .....	35

2.3.3 Relación entre interesados.....	36
2.4 Plan de gestión de comunicaciones .....	36
2.4.1 Comunicación interna .....	36
2.4.2 Comunicación con Logitravel.....	36
2.4.3 Comunicación con Amadeus.....	37
2.5 Riesgos .....	37
2.5.1 Complejidad del proyecto.....	37
2.5.2 Planificación temporal.....	38
Análisis.....	39
3.1 Requisitos no funcionales .....	39
3.1.1 Requisitos no funcionales de las transacciones de la API de transportes .....	39
3.1.2 Requisitos no funcionales de la integración .....	40
3.1.3 Requisitos no funcionales del formulario de pruebas.....	40
3.2 Requisitos funcionales .....	41
3.2.1 Requisitos funcionales de las transacciones de la API de transportes.....	41
3.2.2 Requisitos funcionales del formulario de pruebas.....	42
Diseño .....	44
4.1 Modelo de datos de las transacciones.....	44
4.2 Diagramas de flujo de transacciones con el GDS.....	48
4.3 Mockups del formulario de pruebas .....	52
Construcción.....	56
5.1 Transacciones de la API de Transportes.....	56
5.1.1 Lenguaje y tecnología utilizadas .....	56
5.1.2 Descripción de las transacciones programadas .....	57
5.2 Integración .....	57
5.2.1 Lenguaje y tecnología utilizadas .....	57
5.2.2 Funcionalidades destacadas.....	57
5.3 Formulario de pruebas.....	59
5.3.1 Lenguaje/tecnología.....	60
5.3.2 Ejemplo de uso (vídeo).....	60
5.4 Programación de tests unitarios.....	60
5.4.1 Importancia.....	60
5.4.2 Lenguaje y tecnología utilizadas .....	60
Pruebas .....	62
6.1 Certificación por parte de Amadeus.....	62
6.1.1 Escenarios a probar y presentar .....	62
6.1.2 Primer intento.....	63
6.1.3 Segundo intento.....	64
6.2 Fase interna de pruebas .....	66
Despliegue .....	67
7.1 Procedimientos y entornos de despliegue.....	67

7.2 Instalación de la integración en el entorno de test.....	68
7.3 Instalación de la integración en los entornos de pre-producción y producción.....	68
Conclusiones .....	69
Anexo.....	71
Referencias bibliográficas .....	81

## **Lista de tablas**

- Tabla 1: Petición de transacción Disponibilidad: DisponibilidadRQ  
Tabla 2: Respuesta de transacción Disponibilidad: DisponibilidadRS  
Tabla 3: Petición de transacción Valoracion: ValoracionRQ  
Tabla 4: Respuesta de transacción Valoracion: ValoracionRS  
Tabla 5: Petición de transacción Reserva: ReservaRQ  
Tabla 6: Respuesta de transacción Reserva: ReservaRS  
Tabla 7: Petición de transacción Emitir: EmitirRQ  
Tabla 8: Respuesta de transacción Emitir: EmitirRS  
Tabla 9: Petición de transacción CancelaReserva: CancelaReservaRQ  
Tabla 10: Respuesta de transacción CancelaReserva: CancelaReservaRS  
Tabla 11: Petición de transacción RecuperaReserva: RecuperaReservaRQ  
Tabla 12: Respuesta de transacción RecuperaReserva: RecuperaReservaRS  
Tabla 13: Elemento tramoDisponibilidad de la petición DMR\_RQ  
Tabla 14: Listado de entregables definidos en el alcance del plan de proyecto  
Tabla 15: Criterios de aceptación generales  
Tabla 16: Criterios de aceptación de gestión de sesión  
Tabla 17: Criterios de aceptación ATC  
Tabla 18: Paquetes de trabajo  
Tabla 19: Requisitos no funcionales de las transacciones de la API de transportes  
Tabla 20: Requisitos no funcionales de la integración  
Tabla 21: Requisitos no funcionales del formulario de pruebas  
Tabla 22: Requisitos funcionales de las transacciones de la API de transportes  
Tabla 23: Requisitos funcionales del formulario de pruebas  
Tabla 24: Elementos y atributos más importantes de DMR  
Tabla 25: Elementos y atributos más importantes de RMR  
Tabla 26: Elementos y atributos más importantes de EMR

## **Lista de figuras**

- Figura 1: Representación visual simple de una integración
- Figura 2: Representación visual de una integración con usuario y producto
- Figura 3: Formulario web de Disponibilidad
- Figura 4: Formulario web de Valoración
- Figura 5: Formulario web de Valoración una vez seleccionadas las opciones
- Figura 6: Representación visual de una integración con TravelgateX y cliente/proveedor
- Figura 7: One Way Trip
- Figura 8: Round Trip
- Figura 9: Circle Trip
- Figura 10: Open Jaw simple en el destino
- Figura 11: Open Jaw simple en el origen
- Figura 12: Open Jaw doble
- Figura 13: Esquema de la aplicación Modificar Reservas
- Figura 14: Ciclo de vida del proyecto de Amadeus
- Figura 15: Estructura de descomposición del trabajo (EDT)
- Figura 16: Diagrama de Gantt
- Figura 17: Relación entre los interesados del proyecto
- Figura 18: Modelo de datos de la transacción DMR
- Figura 19: Modelo de datos de la transacción RMR
- Figura 20: Modelo de datos de la transacción EMR
- Figura 21: Diagrama de flujo de ejecución de la transacción DMR
- Figura 22: Diagrama de flujo de ejecución de la transacción RMR
- Figura 23: Diagrama de flujo de ejecución de la transacción EMR
- Figura 24: Mockup de la pestaña Disponibilidad-Valoración
- Figura 25: Mockup de la pestaña Reserva
- Figura 26: Mockup de la pestaña Recuperar Reserva
- Figura 27: Mockup de la pestaña Modificar Reserva
- Figura 28: Diagrama de Gantt (desviaciones temporales)

## **Lista de acrónimos**

ADT: *Adult passenger*  
API: *Application Programming Interface*  
ATC: *Amadeus Ticket Changer*  
CHD: *Child passenger*  
CRS: *Customer Reservations System*  
CT: *Circle Trip*  
CVV: *Card Verification Value*  
DMR: DisponibilidadModificarReserva  
DMR\_RQ: Petición de la transacción DisponibilidadModificarReserva  
DMR\_RS: Respuesta de la transacción DisponibilidadModificarReserva  
EDT: Estructura de Descomposición del Trabajo  
EMD: *Electronic Miscellaneous Document*  
EMR: EmitirModificarReserva  
EMR\_RQ: Petición de la transacción EmitirModificarReserva  
EMR\_RS: Respuesta de la transacción EmitirModificarReserva  
FAQ: *Frequently Asked Questions*  
GDS: *Global Distribution System*  
IATA: *International Air Transport Association*  
IBM: *International Business Machines*  
INF: *Infant passenger*  
LTD: *Last Ticketing Date*  
OJ: *Open Jaw*  
OTA: *Online Travel Agency*  
OW: *One Way Trip*  
PNR: *Passenger Name Record*  
POO: Programación Orientada a Objetos  
RMR: ReservaModificarReserva  
RMR\_RQ: Petición de la transacción ReservaModificarReserva  
RMR\_RS: Respuesta de la transacción ReservaModificarReserva  
RQ: Petición o *request* de una transacción  
RS: Respuesta o *response* de una transacción  
RT: *Round Trip*  
SABRE: *Semi Automatic Business Environment Research*  
SOAP: *Simple Access Object Protocol*  
TSM: *Transitional Stored Miscellaneous*  
TST: *Transitional Stored Ticket*  
URL: *Uniform Resource Locator*  
W3C: *World Wide Web Consortium*  
WSDL: *Web Services Description Language*  
XML: *Extensible Markup Language*  
XSD: *XML Schema Definition*

## Resumen

En el ámbito turístico existen muchas alternativas para realizar la compra online de billetes aéreos. Hoy en día, muchas aerolíneas ya tienen su propia página web con el fin de ofrecer a sus clientes su stock de vuelos. Sin embargo, la mayoría de clientes [1] realiza la búsqueda de vuelos a través de metabuscadores [2] que acaban derivando en agencias de viajes online.

Tantas alternativas ofrecen al consumidor una forma fácil de buscar y encontrar vuelos baratos. Sin embargo existen muy pocas o casi ninguna alternativa para alterar los billetes comprados. Además dado el gran abanico de posibilidades en cuanto a aerolíneas, cambiar un billete se convierte en un proceso complejo, ya que cada aerolínea tiene sus propias reglas.

De esta forma, una agencia que venda billetes de muchas aerolíneas, necesita conocer las reglas de cancelación o modificación de los billetes de todas ellas, aplicando un proceso diferente para cada una cuando un cliente decide modificar su vuelo.

Es por esto que casi ninguna agencia ofrece un servicio de modificación de billetes. Este proyecto pretende ofrecer a la agencia online Logitravel un servicio para unificar las reglas de modificación de las múltiples aerolíneas que vende y brindar al cliente la posibilidad de realizar múltiples modificaciones en un billete. Para ello, la mejor alternativa es utilizar los servicios de uno de los sistemas de venta de billetes más consolidado hoy en día: Amadeus [3], con el que una agencia tiene acceso a billetes de casi 500 aerolíneas diferentes.

TravelgateX es la empresa encargada por Logitravel para desarrollar este servicio. Ambas empresas mantienen una relación de negocio business to business [4], unificando el acceso a reservas aéreas de múltiples aerolíneas, agregadores o consolidadores [5] como Travelfusion, en un único formato: API Transportes. Para ofrecer la posibilidad de modificar reservas a Logitravel, se ha desarrollado una funcionalidad que procese los servicios de Amadeus y además los integre con el resto de servicios que ofrece TravelgateX en la API.

Para poder llevar a cabo el desarrollo, se ha realizado una planificación del proyecto juntamente con los implicados en el mismo: Amadeus y Logitravel. La complejidad que supone la alteración de billetes aéreos, implica que el desarrollador deba entender el contexto en el que se enmarcan los procedimientos de reservas aéreas y el conocimiento en profundidad de los servicios previamente construidos en TravelgateX para este ámbito.

El resultado ofrecido es la aplicación para modificar reservas 100% funcional desplegada en el entorno de producción al que puede acceder Logitravel.

Pese a la complejidad del proyecto y la limitada participación en un proyecto tecnológico de esta envergadura, cabe destacar la satisfacción de Logitravel con el producto resultante y el potencial beneficio que se espera de un producto escaso en el mercado.

La mayor lección aprendida ha sido lo beneficiosa que resulta la colaboración continua entre desarrollador y el cliente. Dicha colaboración ha permitido que el producto desarrollado cumpla con las expectativas de Logitravel y además sin sufrir desviaciones temporales imprevistas.

## INTRODUCCIÓN

En este capítulo se introduce el contexto y el entorno en el que se enmarca este proyecto (1.1) y posteriormente, una vez que se ha dado a conocer que conceptos se mueven en este entorno, se define el problema (1.2) y cómo dar respuesta al mismo (1.3).

### 1.1 Contextualización

En esta sección se detalla primero qué hace TravelgateX (1.1.1) y qué hace Logitravel (1.1.2), para luego poder introducir la necesidad que surge (1.1.3) y explicar cómo aplacarla (1.1.4). Posteriormente, se enumeran los objetivos a cumplir en este proyecto (1.1.5). A continuación, se empiezan a explicar con más detalle los conceptos que envuelven al proyecto:

- ¿Qué es una integración? (véase apartado 1.1.6)
- Principales transacciones de una integración de vuelos (véase apartado 1.1.7)
- Relación entre proveedores, TravelgateX y Logitravel (véase apartado 1.1.8)
- GDS: Global Distribution System [6] (véase apartado 1.1.9)
- Conceptos y terminología del GDS Amadeus [3] (véase apartado 1.1.10)
- Web Services y WSDL (véase apartado 1.1.11)
- Estado actual de los Web Services de Amadeus (véase apartado 1.1.12)
- Estado previo de la API de Transportes de TravelgateX (véase apartado 1.1.13)

#### 1.1.1 TravelgateX

En este apartado se introduce cual es el marco de trabajo en la empresa TravelgateX para este proyecto y se detallan algunos de los aspectos claves del departamento de integraciones.

Entre otras cosas, TravelgateX se dedica a conectar los servicios turísticos de proveedores con clientes. Los principales servicios que conecta son reservas aéreas, hoteleras, actividades, alquiler de coches, trenes o ferries. Todos estos productos son provistos por múltiples proveedores mediante servicios que poco o nada se parecen entre sí. Es por ello que Logitravel,

principal cliente de TravelgateX, se conecta con TravelgateX para unificar todos los servicios en uno solo.

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 APIs.
- Compartir con el Dpto. de integraciones cualquier funcionalidad que se considere útil para toda integración que se desarrolle en el futuro.
- Utilizar nomenclatura compartida entre integraciones.
- Realización de *code reviews* [7] 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.
- Documentación de cualquier cambio realizado sobre las APIs.
- Crear y compartir *posts* de ayuda, solucionadores de problemas, FAQs, 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* [8] para proyectos de larga duración.
- Utilización de la metodología *kanban* [9] para la gestión de incidencias.
- Realizar semanalmente, reuniones de seguimiento con el cliente.

### 1.1.2 Logitravel

Logitravel es una agencia de viajes online o OTA, de las siglas *Online Travel Agency*, que vende productos turísticos de diferentes ámbitos. Gran parte de ellos los obtiene a partir de la unificación de servicios de múltiples proveedores que ofrece TravelgateX. De esta forma, la relación de negocio ya consolidada entre estas dos empresas, permite la puesta en marcha de proyectos que impliquen un beneficio mutuo.

Pese a que TravelgateX y Logitravel son empresas distintas e independientes, durante muchos años se han confundido sus papeles. TravelgateX es para Logitravel la plataforma que integra los servicios de múltiples proveedor a través de un único formato que Logitravel pueda entender y vender así los productos al cliente final. De esta forma, TravelgateX juega un papel de intermediario entre proveedores y clientes, mientras que Logitravel es quien vende finalmente el producto.

En el ámbito de reservas aéreas, Logitravel cuenta con un sistema que integra todos los productos que obtiene a partir de las integraciones de TravelgateX. Este sistema, apodado

agregador, conecta todos los proveedores de TravelgateX para que su venta en la página web de Logitravel pueda ofrecer múltiples alternativas (aerolíneas y precios) en un solo proceso.

A lo largo de las diferentes secciones de este capítulo se dan más detalles sobre el rol que juega Logitravel en este proyecto y qué clase de colaboración existe entre el desarrollador de TravelgateX y la agencia.

### 1.1.3 Problemática: ¿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 vuelos.

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.

Sin embargo, la modificación de un billete aéreo es un proceso complejo dada la enorme cantidad de aerolíneas que existen, cada una de ellas aplicando sus propias reglas de modificación.

Uno de los proveedores de TravelgateX que puede ofrecer una unificación de estas reglas es el GDS Amadeus [3]. Un GDS ofrece en tiempo real las tarifas de las compañías áreas con las que tienen acuerdos de distribución. A través de su red, los clientes de los GDSs pueden efectuar consultas, reservas y ventas de billetes. Más adelante, en el apartado 1.1.9 se expande esta definición.

La motivación de este proyecto surge entonces de la necesidad de uno de los clientes de TravelgateX: Logitravel, de poder modificar las reservas realizadas por sus clientes. Además, la relación de negocio entre Logitravel, TravelgateX y Amadeus, ya consolidada desde hace años, provoca que muchos de los proyectos que Amadeus saca adelante, sean una oportunidad para sus mejores clientes de 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.

### 1.1.4 Aplacar la necesidad

El GDS Amadeus es una de las pocas empresas que ofrecen funcionalidades con las que modificar una reserva aérea. Además Amadeus cuenta con más de 20 años de experiencia en la industria del turismo aéreo y hotelero, así como el trato con agencias de viajes online OTAs. Por lo tanto, la mejor solución para dar respuesta al problema planteado es desarrollar una aplicación de modificar reservas mediante los servicios que ofrece Amadeus.

A continuación se introducen a alto nivel, los paquetes de trabajo que deben desarrollarse para aplacar la necesidad mediante la aplicación Modificación de Reservas:

- Servicios que proporcionen a Logitravel la capacidad de modificar una reserva. Dichos servicios deberán seguir el formato ya consolidado en TravelgateX, por lo tanto, la agencia deberá ser capaz de adaptar con facilidad sus servicios a la aplicación desarrollada en este proyecto.
- Funcionalidades que transformen los servicios proporcionados por el proveedor Amadeus a los servicios que utiliza Logitravel en el agregador de vuelos.

Una vez contextualizado el proyecto con todos los conceptos que se necesitan entender, se expandirá la descripción de la necesidad y describirá en qué consiste una modificación en la sección 1.2. Después, en la sección 1.3 se detallará cómo dar respuesta a la necesidad.

### 1.1.5 Objetivos principales y secundarios y equipo del proyecto

El principal objetivo de este proyecto es ofrecer a Logitravel la capacidad de modificar las reservas que hagan sus clientes a través de la integración Amadeus.

Para cumplir con este objetivo, es necesario el desarrollo de los siguientes paquetes de trabajo introducidos a alto nivel, y que coinciden con los objetivos secundarios planificados:

- Nuevas operaciones que permitan conectar el proveedor Amadeus con el cliente Logitravel para establecer un flujo de modificación de reservas.
- Funcionalidad que integre los formatos y tecnologías dispares de Amadeus y Logitravel.
- Aplicación que permita la realización de pruebas sobre las operaciones desarrolladas.

El equipo del proyecto en TravelgateX que se ocupa del cumplimiento de estos objetivos está formado por un único desarrollador. Las responsabilidades para llevar a cabo todos los paquetes de trabajo de este proyecto recaen entonces sobre autor de esta memoria. Sin embargo, otras tareas secundarias como instalar los productos desarrollados en los diferentes entornos de TravelgateX, recaen sobre el equipo de desarrollo y operaciones (DEVOPS) de TravelgateX.

### 1.1.6 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 y respuesta RS. En el contexto de este trabajo, los sistemas que van a conectarse son dos servicios web: el GDS 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 en una 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, se sitúan conceptualmente a los participantes, las transacciones (RQ y RS) y la integración, en un diagrama.

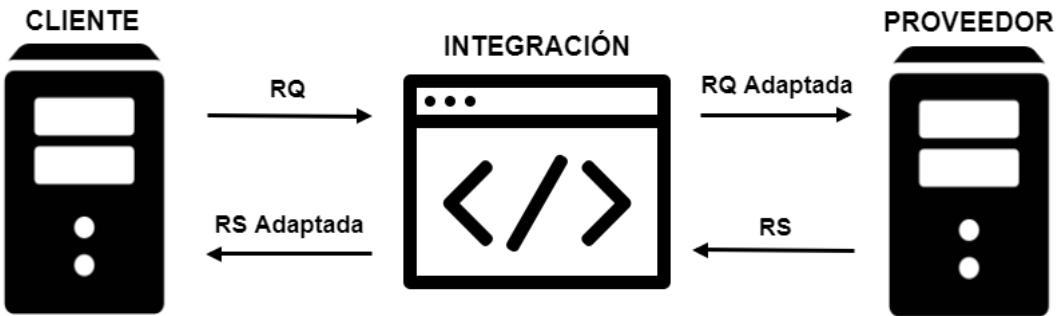


Figura 1: Representación visual simple de una integración

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* [10].

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

- Dirección URL del servicio web.
- Usuario y contraseña, o información que permita a un agente identificarse.
- Protocolo criptográfico de mutuo acuerdo para la comunicación segura a través de internet.

### 1.1.7 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 explica 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, sino que pasará antes siempre por un servicio web.

En la Figura 2 presentada a continuación, se añaden los nuevos participantes mencionados hasta ahora en este apartado: usuario, sitio web, base de datos y producto.

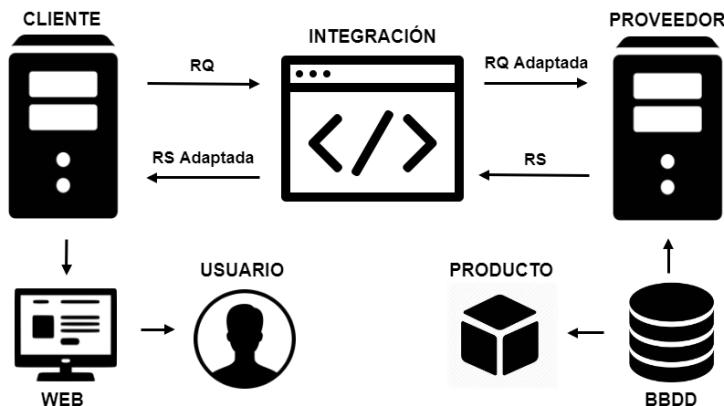


Figura 2: Representación visual de una integración con usuario y producto

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:

### 1. Disponibilidad (*Availability*)

Se solicita al proveedor qué productos están disponibles. La consulta a realizar dependerá de unos criterios predeterminados de búsqueda que el cliente escoga. 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 [1.1.6](#), el proveedor procesa la petición y responde enviando, si están disponibles, los productos solicitados. La información de dichos productos se transforma y se envía en el 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 indicados 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.

### 2. Valoración (Cotización/*Valuation/Quote*)

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

El propósito 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, como lo es en la disponibilidad.

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

### 3. Reserva (Confirmación/Booking)

Se realiza la compra del producto elegido. El pago del mismo al proveedor suele realizarse, sin embargo, a través de otra transacción realizada posteriormente denominada Emisión o *Ticketing*.

Para comprender mejor estas transacciones, se presentan a continuación a través de los habituales pasos que sigue un cliente en un sitio web de reservas:

#### 1. 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, se realiza una transacción de disponibilidad.

Figura 3: Formulario web de Disponibilidad

#### 2. Valoración

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

Desde 40,60 € por persona				VUELOS A LA CARTA
Ida Mallorca - Barcelona		Vuelta Barcelona - Mallorca		
<input type="radio"/>	07:00 Mallorca	→ 07:50 DIRECTO Barcelona	34,61 € <small>9 personas</small>	12:50 Barcelona → 13:50 DIRECTO Mallorca 5,99 €
<input type="radio"/>	07:00 Mallorca	→ 07:55 DIRECTO Barcelona	39,99 € <small>1 persona</small>	14:30 Barcelona → 15:30 DIRECTO Mallorca 5,99 € <small>2 personas</small>
<input type="radio"/>	08:05 Mallorca	→ 09:00 DIRECTO Barcelona	39,99 € <small>1 persona</small>	08:00 Barcelona → 09:00 DIRECTO Mallorca 22,99 € <small>1 persona</small>

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](#).

Vuelos seleccionados			
	Salida	Llegada	Clase
→ Ida 08/04/2018 <b>AirEuropa</b>	UX6006 Mallorca (PMI) Air Europa 07:00	Barcelona (BCN) 07:50	Turista
← Vuelta 11/04/2018 <b>RYANAIR</b>	FR2896 Barcelona (BCN) Ryanair 12:50	Mallorca (PMI) 13:50	Turista Standard Fare
			Precio Base: <b>40,60 €</b> por persona
			¿Aún no está listo para comprar? <b>Bloquear precio por 9 € / persona</b>

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

### 3. 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 apartado [1.1.8](#), 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 de la sección [1.1](#) restante.

#### 1.1.8 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, de más de 20 integraciones de vuelos, entre ellas integraciones con [webservices](#) de compañías aéreas como Iberia, Vueling, American Airlines o integraciones con sistemas [GDS](#) como Galileo [11], Sabre [12] 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](#) que utilice TravelgateX.

TravelgateX cuenta con una [API](#) 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](#). 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.

En la Figura 6 se incorporan el proveedor (Amadeus), la agencia cliente (Logitravel) y la empresa que integra Logitravel y Amadeus (TravelgateX) al esquema básico de integración.

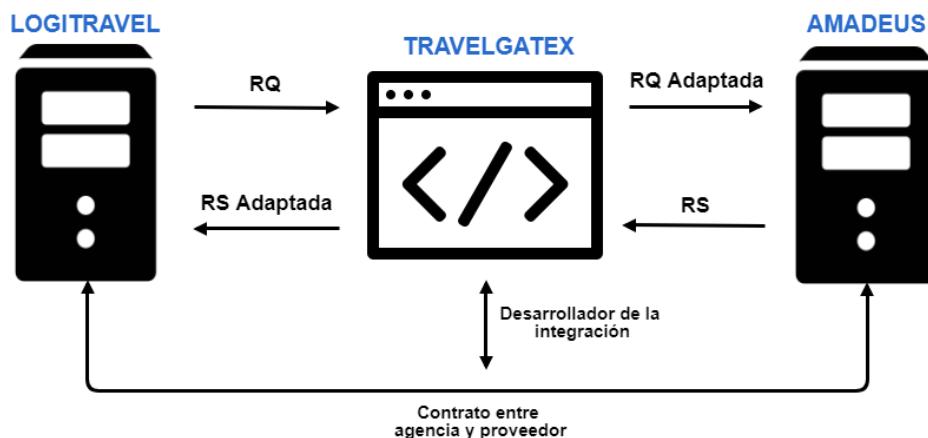


Figura 6: Representación visual de una integración con TravelgateX y cliente/proveedor

### 1.1.9 GDS: Global Distribution System

Desde un punto de vista de negocio, un GDS 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 proporciona en directo, información sobre el producto, como su disponibilidad y precio.

David Chestler, vicepresidente ejecutivo de SiteMinder en América, compara un GDS 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" [13].

Para entender mejor qué es un GDS, 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 [3] [11] [12] [14].

- 1957 es el año en el que IBM y American Airlines empiezan a trabajar juntos en la creación de lo que se conoce como CRS, un sistema semi-automatizado que llamaron SABRE.
- La versión final del proyecto SABRE, 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 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 CRSS para ser incluidos en sus bases de datos.
- En 1971 United Airlines crearon un nuevo CRS llamado Apollo.

- Con el tiempo, algunas agencias de viajes recibían comisiones por parte de los CRSs 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, se encargaba también de la distribución.
- SABRE en los 90' y más tarde el GDS 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 GDSs.

### 1.1.10 Conceptos y terminología del GDS Amadeus

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

Ya que cada GDS usa una terminología propia, para acotar los conceptos definidos a continuación, se toman solamente las denominaciones bautizadas y utilizadas por Amadeus.

Algunos conceptos, aunque fueron acuñados por los GDSs, 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)

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

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

#### Viaje de ida y vuelta (*Round Trip* o RT)

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

#### Viaje circular (*Circle Trip* o CT)

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

#### *Open Jaw simple (OJ 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](#)).

#### *Open Jaw simple (OJ 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](#)).

#### *Open Jaw doble (OJ 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](#)).

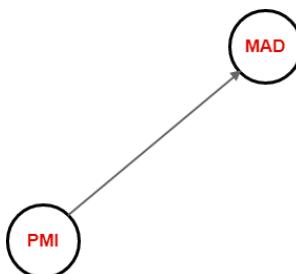


Figura 7: One Way Trip

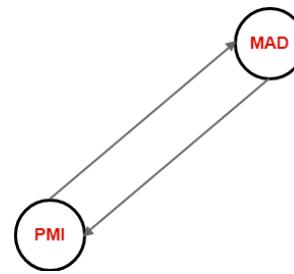


Figura 8: Round Trip

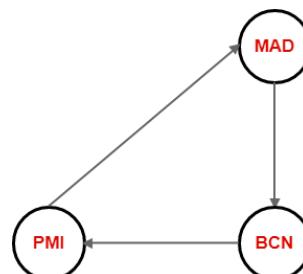


Figura 9: Circle Trip

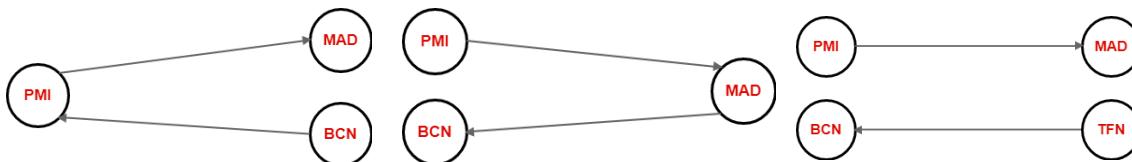


Figura 10: Open Jaw simple en el destino

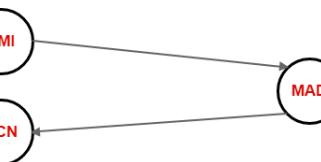


Figura 11: Open Jaw simple en el origen

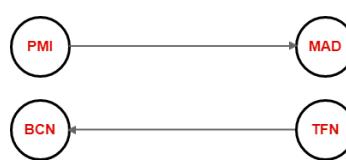


Figura 12: Open Jaw doble

### ***Passenger Name Record (PNR)***

El PNR 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, de forma automática, información adicional para constatar que el PNR se ha creado correctamente.

Hay 7 elementos obligatorios que deben añadirse al PNR:

- 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). Si no se ha abonado el importe de la reserva, ésta se cancela.
- Compañía aérea validadora (*validating carrier*).
- Comisión de la compañía aérea. Normalmente es un elemento que Amadeus introduce de forma automática.

Un PNR 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 introducidos los campos obligatorios en el PNR 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. De esta forma se puede seguir introduciendo información o modificarse información ya existente.

### **Cabina**

Comúnmente llamada también clase cabina. Existen distintas cabinas para diferenciar las ventajas en los servicios que vende la compañía un segmento determinado. 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 por 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 distingue tres tipos de pasajeros, separándolos a través de un rango de edades:

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

## Billete

Identificador asociado a un pasajero o a un elemento extra, como una maleta o un asiento, y que su emisión está controlada por 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 de la reserva.

## 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.

### 1.1.11 Web Services y WSDL

En este apartado se explica de forma breve qué es un *webservice* y para qué sirve un WSDL [15]. Este apartado guarda una estrecha relación con el apartado 1.1.6 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 es un protocolo basado en XML [16] 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 qué tipos de datos necesitan las aplicaciones que interoperarán para utilizarlo.

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

### 1.1.12 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 ofrecen 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. Para cada tarifa se indican los segmentos que pueden elegirse.

#### Informative Pricing (y variante Best)

Permite la cotización de las tarifas devueltas en MasterPricer Search. Existen tres versiones del *webservice*, cada una de ellas sirve con un propósito:

- Cotizar con PNR: 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: Es la opción más utilizada puesto que es necesaria para cotizar un vuelo obtenido en la disponibilidad y que luego ha de reservarse. Ya que no existe aún el PNR, la cantidad de pasajeros, forma de pago e itinerario, tiene que proporcionarse en la petición.
- Cotizar refrescando la disponibilidad (Informative Best Pricing): Antes de cotizar, el GDS vuelve a lanzar una consulta a MasterPricer para obtener así vuelos más baratos en una clase distinta, o en la misma clase si estos precios han cambiado.

#### Air Sell

*Webservice* diseñado para comprar las tarifas obtenidas en MasterPricer. Es una llamada necesaria y obligatoria que sirve como paso previo a la inclusión de los vuelos en el PNR. Una vez ejecutada la llamada, Amadeus pre-reserva la tarifa y elimina la plaza libre de la clase.

#### PNR Add Multi Elements

*Webservice* esencial para añadir cualquier información de la reserva en el PNR. Además de la introducción de datos, este *webservice* permite realizar el *commit* de los cambios realizados en el PNR.

### **PNR Retrieve**

Permite recuperar el PNR a través, por ejemplo, del localizador que Amadeus proporciona al crear la reserva.

### **PNR Cancel**

Permite eliminar elementos guardados en el PNR.

### **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.

## **1.1.13 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 de transportes de TravelgateX. De esta forma el lector puede conocer los principales elementos de cada transacción para entender así su estructura a alto nivel.

Dado que se pretende presentar las transacciones de forma introductoria, la mayoría de elementos y atributos descritos en las siguientes tablas no se corresponden con los que realmente existen en la API, puesto que su descripción sería demasiado extensa para el propósito que se busca en este apartado. El contenido de estas tablas, puede consultarse en el anexo (Tabla 1 – Tabla 12).

Las filas sombreadas de la tabla representan los elementos o atributos incluidos en la raíz de las peticiones/resuestas, es decir, en la Tabla 1, el elemento tramosDisponibilidad es uno de los elementos que cuelgan de la raíz de elementos de la petición DisponibilidadRQ.

## **1.2 Definición del problema**

En esta sección se definen las partes existentes de una reserva aérea y cuál ha de ser el planteamiento para dar respuesta a la necesidad planteada por Logitravel de modificar reservas.

### **1.2.1 Unidades que forman una reserva**

En este apartado se alude la existencia de cliente o usuario final. No debe confundirse dicho cliente con la agencia con la que se integra (Logitravel). Este usuario es el que toma en última instancia las decisiones en una reserva y elige el contenido de cada una de las partes de la misma.

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, etc.

### **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: ADT, CHD o INF.

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 no tengan coste, mientras que en otras no exista diferencia de precio entre adultos ADT y niños CHD.

### **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 CVV 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 permiten este pago “cash”, por lo que queda a expensas del cliente del GDS 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, tal como: dirección en la que reside, correo electrónico y teléfono de contacto. El GDS 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 cargos adicionales sobre el importe o importes de los pasajeros.

### **Otros datos que forman una reserva, pero que el cliente no puede modificar:**

- Fecha hasta la cual será válida una reserva o LTD. 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 En qué consiste la modificación de una reserva

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

Es especialmente sencillo modificar nombres de pasajeros, pasaportes e información de contacto ya que si quisiéramos podríamos hacerlo con apenas dos transacciones: [PNR Cancel](#) para cancelar uno de los datos y [PNR AddMultiElements](#) 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](#) 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](#) 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 y todo el trabajo que ha de realizarse para hacer cada uno de estos pasos posibles.

## 1.3. Respuesta al problema

Esta sección 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). Estos pasos tendrán que adaptarse a las exigencias de Logitravel y a las exigencias del proveedor Amadeus.

Por último, en el apartado 1.3.2 se detallan los tres paquetes de trabajo necesarios para desarrollar la aplicación.

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

Como se ha adelantado en el apartado 1.2.2, 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, 4 y 5 de análisis, diseño y programación, se profundizará en todos los elementos necesarios, 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
- 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” o “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.

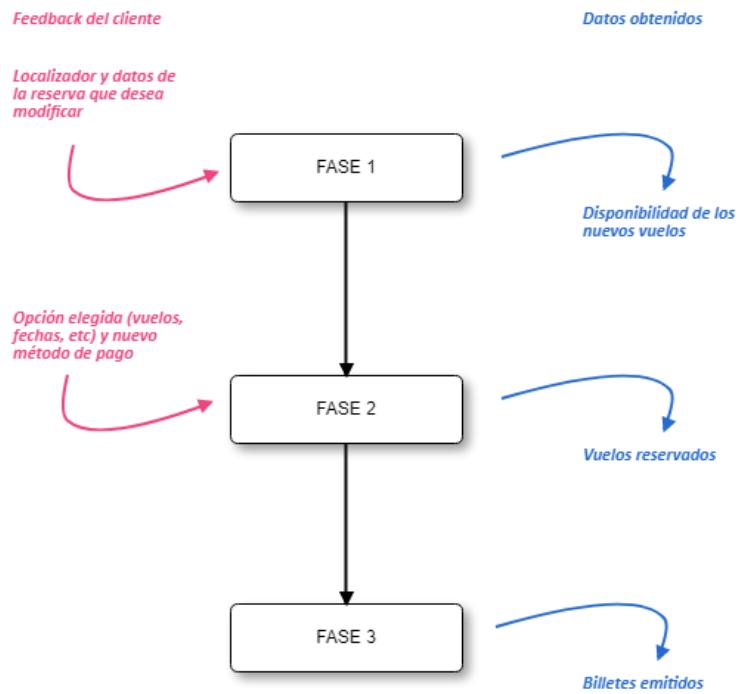
Un requisito esencial para poder modificar una reserva es la existencia de billetes y, por lo tanto, que se haya realizado la emisión y consiguiente abono del importe de la reserva. Otros factores como rutas o aerolíneas concretas serán un impedimento para modificar una reserva. Existe también la posibilidad 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 será 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 sub-apartado 1.3.2.1 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 quizás más compleja fase de las tres que forman esta primera idea de desarrollo.

En la siguiente [Figura 13](#) se esquematiza y resume la idea definida en este capítulo:



[Figura 13: Esquema de la aplicación Modificar Reservas](#)

### 1.3.2. Productos a desarrollarse

Este apartado abarca la introducción de las tres áreas que se analizarán, diseñarán y construyeran en este proyecto. El primer sub-apartado [1.3.2.1](#) introduce el flujo que formará la modificación de reservar por la parte del cliente. A continuación, en los sub-apartados [1.3.2.2](#) y [1.3.2.3](#) se introducen las transacciones que formarán el flujo de operaciones de Amadeus y las claves de la integración que transforman el flujo del cliente con el flujo del proveedor. Finalmente, en el sub-apartado [1.3.2.4](#) 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 apartado [1.1.13](#) 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](#) 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](#)) y la integración pueda responderle (en una [RS](#)).

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](#) 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](#) 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 tres transacciones y realizar algunos cambios en algunos de los elementos ya existentes en la [API](#).

## 1. DisponibilidadModificarReserva (DMR): petición DMR\_RQ y respuesta DMR\_RS

En esta transacción el cliente proporcionaría el localizador y los tramos de la reserva, indicando cuál es la modificación a realizar en cada uno de estos tramos. Por lo tanto, surge la necesidad de introducir un nuevo concepto en los tramosDisponibilidad ya existentes (elemento sombreado). Para ello, se aprovecha el tipo tramoDisponibilidad ya existente, añadiéndole un atributo obligatorio para la transacción DMR:

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	Localización 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: <u>N</u> (ninguna), <u>KF</u> (mantener vuelos y tarifas), <u>K</u> (mantener vuelos), <u>C</u> (cambiar), <u>A</u> (añadir), <u>R</u> (quitar)

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 o la fecha 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 DisponibilidadRS.

En el capítulo 4 Diseño se presentan los elementos que forman esta transacción y en capítulo 5 Construcción se describe cada uno de ellos.

## 2. ReservaModificarReserva (RMR): petición RMR\_RQ y respuesta RMR\_RS

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

Desgloses. Construidos a partir de los segmentos y tarifas proporcionados en la respuesta de DMR.

- Billetes. Billetes de la reserva original.
- Localizador. El PNR 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 las secciones de diseño 4.1 y 4.2, y en capítulo 5 Construcción, 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, y sus diferentes políticas, complica ésta transacción. De esta forma, ésta transacción y la siguiente (EMR), 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) y emisión (EMR) serán dos transacciones independientes, se necesitarán nodos exclusivamente creados para transferir información de la respuesta de una transacción a la petición de la siguiente.

### 3. EmitirModificarReserva (EMR): petición EMR\_RQ y respuesta EMR\_RS

Para realizar la emisión se necesita principalmente:

- Localizador. De nuevo ha de recuperarse y abrirse el PNR, 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)
- ReservaModificarReserva (RMR)
- EmitirModificarReserva (EMR)

En el siguiente sub-apartado se introduce cuál ha de ser 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 y comprobación de posibilidad de modificar reserva.
- Búsqueda de nuevas recomendaciones.
- Actualización del PNR y reemisión del billete o revalidación.

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

#### DisponibilidadModificarReserva (DMR)

- Recuperación del PNR y comprobación de posibilidad de modificar reserva.

- Búsqueda de nuevas recomendaciones.

#### **ReservaModificarReserva (RMR)**

- Actualización del PNR.

#### **EmitirModificarReserva (EMR)**

- Reemisión del billete o revalidación.

En la sección [4.2](#) 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 apartado [1.1.12](#), 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 visto en el capítulo [1.1.12](#) 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. Una vez confirmados los cambios, el PNR está preparado para que se emitan los billetes.

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

### **1.3.2.3 Integración a desarrollar**

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

- El cliente enviará en las peticiones (DMR\_RQ, RMR\_RQ o EMR\_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, RMR RS o EMR RS).
- Finalmente la integración envía la respuesta al cliente.

La integración de Amadeus ha sido ya programada en el lenguaje de programación Visual Basic [17] de la plataforma .NET [18], por lo tanto, todo el nuevo código deberá programarse también en este lenguaje. En el sub-apartado 2.1.2.3 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 por parte de Amadeus. La construcción de las peticiones a mano imposibilitaría la completitud del desarrollo en un término de tiempo aceptable. 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 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 se obtengan 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 sí sucede con la integración.

## PLAN DEL PROYECTO

En este capítulo se define la planificación que se realizó para llevar a cabo el proyecto. Ésta se ha llevado a cabo en colaboración con las partes interesadas que han participado durante el desarrollo y que utilizarán la modificación de reservas una vez que se despliegue en producción.

A lo largo de las secciones de este capítulo se exponen los planes de:

- Alcance: 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 definen los paquetes de tareas y la lista de entregables. Además se enumeran los requisitos del proyecto y cuáles serán a alto nivel los requisitos del producto y los criterios de aceptación por parte de proveedor y cliente.
  - Requisitos: en el que se detallarán los requisitos del proyecto y el conjunto de requisitos no funcionales y funcionales del producto.
  - Criterios de aceptación: políticas y criterios de las dos partes interesadas externas para aceptar el proyecto desarrollado por TravelgateX.
- Tiempo: se detalla la duración de las tareas del proyecto y se exponen todas ellas en un diagrama.
- Partes interesadas: se enumeran a los implicados en el proyecto y se explica qué relación existe entre ellos y el proyecto.
- Comunicaciones: se definen cuáles serán los canales y la forma de comunicación a utilizar entre los diferentes interesados externos e internos.
- Riesgos: se detallan los mayores riesgos del proyecto y el plan de gestión elaborado para controlarlos.

### 2.1 Alcance del proyecto

Esta sección está desglosada en los siguientes apartados:

- Celebración de la reunión inicial del proyecto *Kick-Off*, en el que se explica el ciclo de vida del proyecto propuesto por Amadeus y el alcance del producto planificado en la reunión (véase en apartado 2.1.1).
- Enumeración de los requisitos del proyecto y requisitos funcionales y no funcionales del producto a alto nivel (véase en apartado 2.1.2).

- Descripción de la estructura de descomposición del trabajo (EDT) (véase en apartado 2.1.3).
- Enumeración de los entregables del proyecto (véase en apartado 2.1.4).
- Descripción detallada de los criterios de aceptación de Amadeus y de Logitravel (véase en apartado 2.1.5).

### 2.1.1 Reunión inicial (Kick-Off)

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.

Amadeus proporciona unas guías a alto nivel del ciclo de vida del proyecto (Figura 14) en el *Kick-off* (reunión de arranque del proyecto) en el que preferiblemente ha de acudir el desarrollador jefe de la aplicación y los distintos interesados del proyecto.

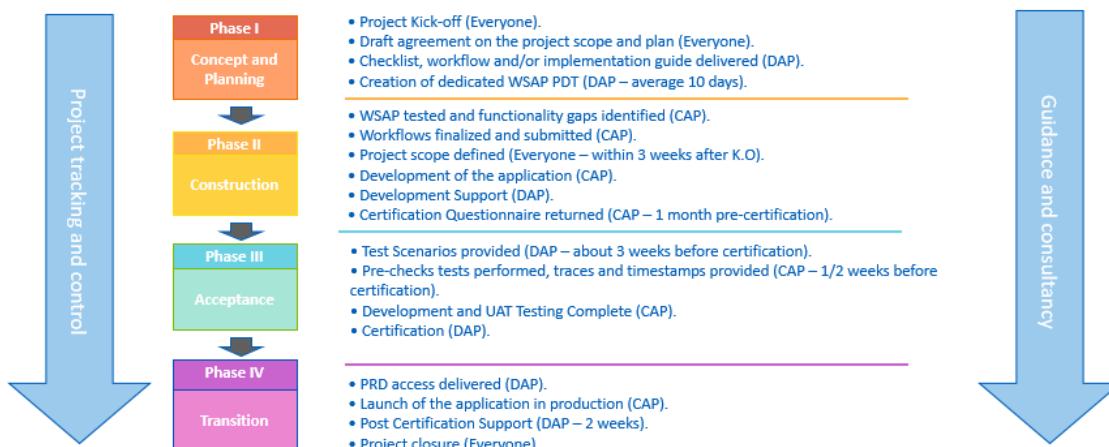


Figura 14: Ciclo de vida del proyecto de Amadeus

Amadeus aconseja seguir un flujo de transacciones que permitirá cumplir con las expectativas de un desarrollo, siguiendo el flujo óptimo para el que fue diseñado el servicio del GDS. 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 inicialmente.

Amadeus proporciona de forma abierta cuál será el *checklist* que realizará sobre la aplicación, lo que permite al desarrollador prestar atención a qué criterios han de cumplirse para pasar la certificación. En el apartado 2.5.1 se pueden ver en profundidad estos criterios.

En la planificación establecida por Amadeus, y en la que participaron agentes de negocio de Logitravel y el desarrollador encargado de los GDSs 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 que la modificación de la reserva 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 ni la construcción de ningún formulario de pruebas.
- Mediante la colaboración entre Logitravel y TravelgateX se definieron los activos a desarrollar:
  - Nuevas llamadas en la API de transportes para poder modificar una reserva
  - Código que integre los dos flujos (transacciones con Amadeus y transacciones con Logitravel).
- 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.
- Una vez desplegado el desarrollo en producción, se considera por terminado este proyecto, y por lo tanto el mantenimiento de la integración queda fuera del alcance planificado.

### 2.1.2 Requisitos

En el sub-apartado 2.1.2.1 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. Se detallan a alto nivel los requisitos funcionales de producto en el apartado 2.1.2.2. Finalmente en el sub-apartado 2.1.2.3 se detalla la tecnología que se usa para la construcción de las transacciones de la API, para desarrollar la integración y para construir el formulario de pruebas.

Los requisitos funcionales y no funcionales enumerados en los sub-apartados 2.1.2.2 y 2.1.2.3 están definidos a alto nivel, de tal manera que constituyan las bases para el acta de constitución. No será hasta el capítulo 3 que se detallen los requisitos definitivos, resultado de un análisis posterior y exhaustivo.

### **2.1.2.1 Requisitos del proyecto**

Los requisitos del proyecto mostrados a continuación se muestran por orden de prioridad:

- 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 deberán 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.
- 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 14](#). 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 éste las herramientas para completar correctamente dicha certificación.
- La integración debe seguir una estructura estandarizada de transacciones (política 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.
- 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*.
- 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.
- La colaboración entre los interesados externos y las herramientas a utilizar durante cada periodo del desarrollo, deberán seguir las pautas presentadas en la sección [2.4: Plan de gestión de comunicaciones](#).
- 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.
- Las funcionalidades desarrolladas deberán compartirse siempre que sea posible con el resto de [APIs](#) usadas en el departamento de integraciones. Además, deberá comunicarse cualquier funcionalidad que se considere útil para el resto de departamentos.
- Desarrollar utilizando nomenclatura compartida entre integraciones.

### **2.1.2.2 Requisitos funcionales**

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

- 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), el cliente indicará el itinerario y las modificaciones que deseé realizar en él. Deberá indicar también el localizador de la reserva y las preferencias de búsqueda: clase cabina, sólo vuelos directos, compañías o incluir *lowcost*.
- En la respuesta de DMR, 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 cada una.
- En la transacción “Reserva Modificar Reserva” (RMR), el cliente indicará de nuevo el localizador de la reserva, así como tarifa elegida, billetes y forma de pago con la que deseé 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).
- En la petición EMR, el cliente indicará de nuevo el localizador, billetes y tipo de emisión obtenida en la transacción RMR. En la respuesta podrá visualizar los nuevos billetes.
- El usuario del formulario de pruebas podrá indicar el tiempo de corte, credenciales y URLs. El usuario podrá realizar un flujo completo de forma gráfica o pegando el XML 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, el usuario ha de poder indicar el localizador de la reserva y los itinerarios y modificación que desea realizar para cada uno de ellos.
- Para las transacciones RMR y EMR, el usuario ha de poder indicar los localizadores y billetes, así como la forma de pago, si desea cambiarla.

#### **2.1.2.3 Requisitos no funcionales**

- Las nuevas transacciones de la API de transportes, la integración y el formulario de pruebas deberán ser programadas mediante el lenguaje Visual Basic [17]. Dada la implementación de las transacciones y de la integración en este lenguaje, no habrá problemas con la serialización [19] de objetos XML, 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 de transportes, como en la integración de Amadeus.

- Las llamadas, elementos y atributos que se incluyan en la API 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) en el formulario de pruebas.

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

En la Figura 15 se desglosan en un EDT, mediante paquetes de tareas, las diferentes áreas del proyecto:

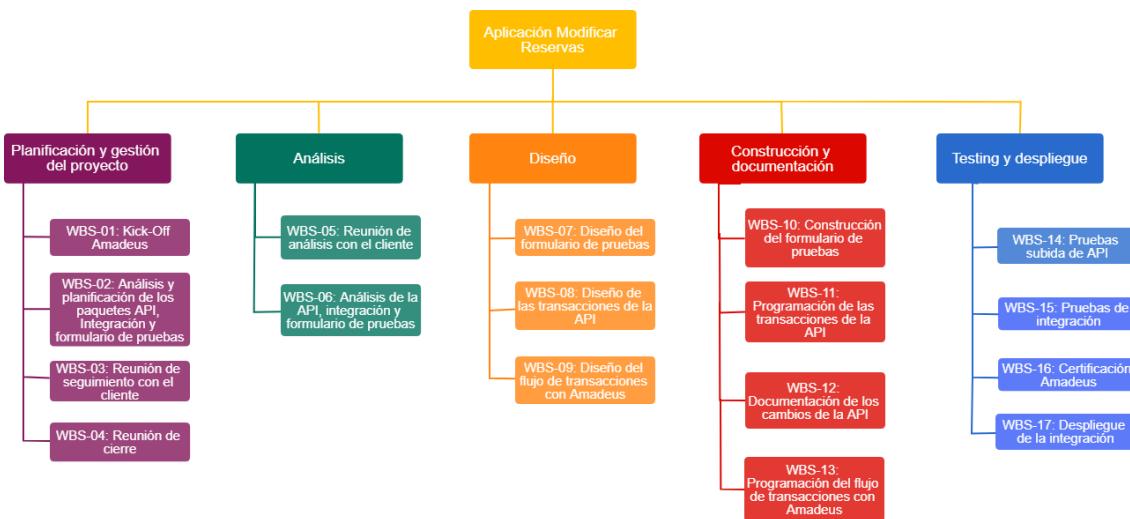


Figura 15: Estructura de descomposición del trabajo (EDT)

### 2.1.4 Entregables

En la Tabla 14 se enumeran los entregables definidos en el alcance del proyecto.

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 <u>GDS Amadeus</u>
	ENT-02: Diagramas de flujo de la aplicación
Logitravel	ENT-03: Integración de Amadeus que incluya la aplicación “modificar reservas”
TravelgateX	ENT-04: Nuevas llamadas en la <u>API</u> de transportes
TravelgateX	ENT-05: Formulario de pruebas
	ENT-06: Documentación de los nuevos elementos incluidos en la <u>API</u> 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

## 2.1.5 Criterios de aceptación

En este apartado se exponen los criterios de aceptación definidos y acordados entre el desarrollador y el proveedor Amadeus en el sub-apartado 2.1.5.1 y entre el desarrollador y Logitravel en el sub-apartado 2.1.5.2.

### 2.1.5.1 Certificación de Amadeus

En este sub-apartado se presenta el *checklist* de certificación por parte del GDS Amadeus, que deberá cumplir la aplicación desarrollada. En cada uno de los criterios presentados en las tablas Tabla 15, Tabla 16 y Tabla 17, se indica el webservice al que 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 implementados no deben estar deprecados		Error
		servicios XML deprecados = 0		
GEN-02	Todos	La aplicación no debe crear bucles infinitos en el host de Amadeus		Error
		bucle infinito 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 <u>SOAP</u> 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
		Criterion	
ATC-01	Todos los servicios ATC	La aplicación debe proveer un ticket por pasajero (incluidos bebés) tickets por pasajero >= 1	Error
ATC-02	ATC Shopper	Las recomendaciones (tarifas) devueltas por ATC Shopper deben visualizarse correctamente en la aplicación recomendaciones de ATC Shopper no mostradas en la aplicación = 0	Error
ATC-03	ATC Shopper	El Web Service Air SellFromRecommendation debe usarse solamente sobre tarifas devueltas por el ATC Shopper compra de tarifas no ATC Shopper con Air SellFromRecommendation = 0	Error
ATC-04	ATC Shopper	La aplicación debe tratar correctamente los tipos de pasajero que se envíen en las peticiones a Amadeus pasajeros con tipo incorrecto (no Amadeus) = 0	Error
ATC-05	ATC Shopper	La aplicación debe devolver la lista de aerolíneas que se recibirá por parte de Amadeus en ATC Eligibility aerolíneas devueltas por ATC Eligibility no mostradas = 0	Error
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 segmentos modificados no eliminados = 0	Warning
ATC-07	ATC Shopper	ATC Eligibility debe ser sistemáticamente utilizado antes de cualquier servicio adicional de ATC Shopper servicios usados sin hacer previamente la llamada a ATC Eligibility = 0	Warning

#### 2.1.5.2 Criterios de aceptación de Logitravel

En este sub-apartado 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. 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 manteniendo la tipología que haya venido utilizándose en otras transacciones previamente desarrolladas en TravelgateX.
- 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.

## 2.2 Cronograma

En esta sección, primero se resumen las principales fechas previstas del proyecto en el apartado 2.2.1. A continuación, los paquetes de trabajo del proyecto desplegados en las diferentes tablas que se incluyen en el apartado 2.2.2, como fichas de tareas con duración, fecha de inicio y fin, tipo de tarea y descripción de la tarea. En el apartado 2.2.3 se puede ver de forma gráfica la planificación temporal de estas tareas mediante un diagrama de Gantt [20].

### 2.2.1 Resumen del cronograma

#### **Inicio previsto:**

La reunión Kick-off de planificación entre TravelgateX, Amadeus y Logitravel fue programada para el 14 de marzo de 2017.

#### **Final previsto:**

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 deberán 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.

#### **Duración prevista:**

La duración del proyecto planificada es por lo tanto de algo menos de 6 meses.

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

En la Tabla 18 se detalla cada uno de los paquetes de trabajo mediante las fechas de inicio y fin, su duración y tipo (fija o periódica). También se enumeran las entradas y salidas de cada una de estas tareas, así como una pequeña descripción del trabajo a realizar en cada una de ellas.

Tabla 18: Paquetes de trabajo

<b>Kick-Off Amadeus</b>	
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 <u>ATC Shopper</u> para la modificación de reservas de Amadeus.
<b>Análisis y planificación de los paquetes API, Integración y formulario de pruebas</b>	
ID	WBS-02
Duración	17h
Fecha inicio	24/03/2017
Fecha fin	30/03/2017
Entradas	Extranet de Amadeus, Documentación API
Salidas	ENT-09: Conjunto de tareas en el Jira de TravelgateX
Tipo de tarea	Trabajo fijo

<b>Descripción</b>	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 de transportes, la integración y el formulario de pruebas
<b>Reunión de seguimiento con el cliente</b>	
<b>ID</b>	WBS-03
<b>Duración</b>	25h (25 x 1h)
<b>Fecha inicio</b>	14/03/2017
<b>Fecha fin</b>	29/08/2017
<b>Entradas</b>	ENT-09: Jiras en progreso y completados
<b>Salidas</b>	ENT-11: Apuntes de mejora o cambios sobre el proyecto
<b>Tipo de tarea</b>	Trabajo variable (tarea periódica)
<b>Descripción</b>	Reunirse semanalmente para mostrar el avance del proyecto y sacar dudas o mejoras planteadas por el cliente
<b>Reunión de cierre</b>	
<b>ID</b>	WBS-04
<b>Duración</b>	3h
<b>Fecha inicio</b>	11/09/2017
<b>Fecha fin</b>	11/09/2017
<b>Entradas</b>	-
<b>Salidas</b>	ENT-07: Mejoras para futuros proyectos con Amadeus
<b>Tipo de tarea</b>	Trabajo fijo
<b>Descripción</b>	Reunión de cierre de proyecto entre Amadeus, Logitravel y TravelgateX para analizar cómo ha ido el proyecto y proponer posibles mejoras para el futuro
<b>Reunión de análisis con el cliente</b>	
<b>ID</b>	WBS-05
<b>Duración</b>	3h (3 x 1h)
<b>Fecha inicio</b>	27/03/2017
<b>Fecha fin</b>	08/04/2017
<b>Entradas</b>	ENT-08: Acta de reunión del Kick-off
<b>Salidas</b>	Requisitos funcionales (apuntes)
<b>Tipo de tarea</b>	Trabajo variable (tarea periódica)
<b>Descripción</b>	Reunirse semanalmente para discutir y analizar el flujo de transacciones a realizar y los requisitos funcionales que debe cumplir la aplicación
<b>Análisis de la API, integración y formulario de pruebas</b>	
<b>ID</b>	WBS-06
<b>Duración</b>	40h
<b>Fecha inicio</b>	07/04/2017
<b>Fecha fin</b>	20/04/2017
<b>Entradas</b>	Apuntes de las reuniones de análisis, Extranet Amadeus
<b>Salidas</b>	ENT-10: Requisitos funcionales y no funcionales
<b>Tipo de tarea</b>	Trabajo fijo
<b>Descripción</b>	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
<b>Diseño del formulario de pruebas</b>	
<b>ID</b>	WBS-07
<b>Duración</b>	8h
<b>Fecha inicio</b>	10/04/2017
<b>Fecha fin</b>	11/04/2017
<b>Entradas</b>	ENT-10: Requisitos del formulario de pruebas (funcionales y no funcionales)
<b>Salidas</b>	Mockup formulario
<b>Tipo de tarea</b>	Trabajo fijo
<b>Descripción</b>	Hacer un mockup del formulario de pruebas
<b>Diseño de las transacciones de la API</b>	
<b>ID</b>	WBS-08
<b>Duración</b>	12h
<b>Fecha inicio</b>	13/04/2017
<b>Fecha fin</b>	18/04/2017
<b>Entradas</b>	Extranet Amadeus, ENT-10: Requisitos (funcionales y no funcionales)
<b>Salidas</b>	Esquema a alto nivel de las transacciones de modificar reserva
<b>Tipo de tarea</b>	Trabajo fijo

<b>Descripción</b>	Diseñar un diagrama a alto nivel de las llamadas, elementos y atributos que formarán parte de las nuevas transacciones de modificar reserva en la <u>API</u> de transportes de TravelgateX
<b>Diseño del flujo de transacciones con Amadeus</b>	
<b>ID</b>	WBS-09
<b>Duración</b>	18h
<b>Fecha inicio</b>	27/04/2017
<b>Fecha fin</b>	04/05/2017
<b>Entradas</b>	Extranet Amadeus, ENT-10: Requisitos (funcionales y no funcionales)
<b>Salidas</b>	ENT-02: Diagramas de flujo de las transacciones
<b>Tipo de tarea</b>	Trabajo fijo
<b>Descripción</b>	Diseñar los diagramas de flujo de las transacciones de modificar reservas de la integración
<b>Construcción del formulario de pruebas</b>	
<b>ID</b>	WBS-10
<b>Duración</b>	20h
<b>Fecha inicio</b>	11/04/2017
<b>Fecha fin</b>	20/04/2017
<b>Entradas</b>	ENT-10: Requisitos del formulario de pruebas (funcionales y no funcionales), Mockup formulario
<b>Salidas</b>	ENT-05: Formulario de pruebas
<b>Tipo de tarea</b>	Trabajo fijo
<b>Descripción</b>	Programar las funcionalidades y construir la interfaz gráfica del formulario
<b>Programación de las transacciones de la API</b>	
<b>ID</b>	WBS-11
<b>Duración</b>	10h
<b>Fecha inicio</b>	18/04/2017
<b>Fecha fin</b>	21/04/2017
<b>Entradas</b>	Esquema a alto nivel de las transacciones de modificar reserva
<b>Salidas</b>	ENT-04: Clases vb de las transacciones de modificar reserva
<b>Tipo de tarea</b>	Trabajo fijo
<b>Descripción</b>	Programar los cambios y nuevas transacciones de modificar reserva
<b>Documentación de los cambios de la API</b>	
<b>ID</b>	WBS-12
<b>Duración</b>	6h (6 x 1h)
<b>Fecha inicio</b>	24/04/2017
<b>Fecha fin</b>	05/06/2017
<b>Entradas</b>	ENT-04: Clases vb de las transacciones de modificar reserva
<b>Salidas</b>	ENT-11: Documentación de la <u>API</u> en la web corporativa
<b>Tipo de tarea</b>	Trabajo variable (tarea periódica)
<b>Descripción</b>	Describir los elementos y atributos que forman las nuevas llamadas y cualquier otro cambio realizado en la <u>API</u>
<b>Programación del flujo de transacciones con Amadeus</b>	
<b>ID</b>	WBS-13
<b>Duración</b>	170h
<b>Fecha inicio</b>	08/05/2017
<b>Fecha fin</b>	27/06/2017
<b>Entradas</b>	Extranet Amadeus, ENT-02: Diagramas de flujo de las transacciones
<b>Salidas</b>	ENT-03: Integración vb
<b>Tipo de tarea</b>	Trabajo fijo
<b>Descripción</b>	Programar la integración
<b>Pruebas subida de API</b>	
<b>ID</b>	WBS-14
<b>Duración</b>	6h (3 x 2h)
<b>Fecha inicio</b>	01/05/2017
<b>Fecha fin</b>	15/05/2017
<b>Entradas</b>	Nuevas versiones de la <u>API</u>
<b>Salidas</b>	Ficheros XML de los tests realizados
<b>Tipo de tarea</b>	Trabajo variable (tarea periódica)
<b>Descripción</b>	Comprobar el funcionamiento de los cambios de la <u>API</u> y del resto de transacciones ya existentes a lo largo de las diferentes subidas: entorno de test, entorno de preproducción 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

### 2.2.3 Diagrama de Gantt

La Figura 16 mostrada a continuación representa el diagrama de Gantt [20] en los que se incluyen los 17 paquetes de tareas. En el eje X se desglosa en las 5 áreas con las que se ha separado el proyecto. Cada una de ellas se indica en la parte superior y se distingue del resto mediante un color. El eje Y de la representación sirve para indicar la duración de las tareas que componen el proyecto. Cada tarea tiene asociada el id asignado durante la elaboración del EDT.

Las tareas fijas (no periódicas) se representan con el ID en fuente negra sobre color, mientras que las periódicas con el ID en fuente blanca sobre color. Cada una de ellas ocupa el espacio temporal en el diagrama de acuerdo a la duración de las mismas.

En la Figura 16 hay representados mediante un punto de color negro los 3 principales hitos del proyecto:

- 14/03/2017: KickOff Amadeus (WBS-01)
- 11/09/2017: Reunión de cierre (WBS-04)
- 05/09/2017: Despliegue de la integración

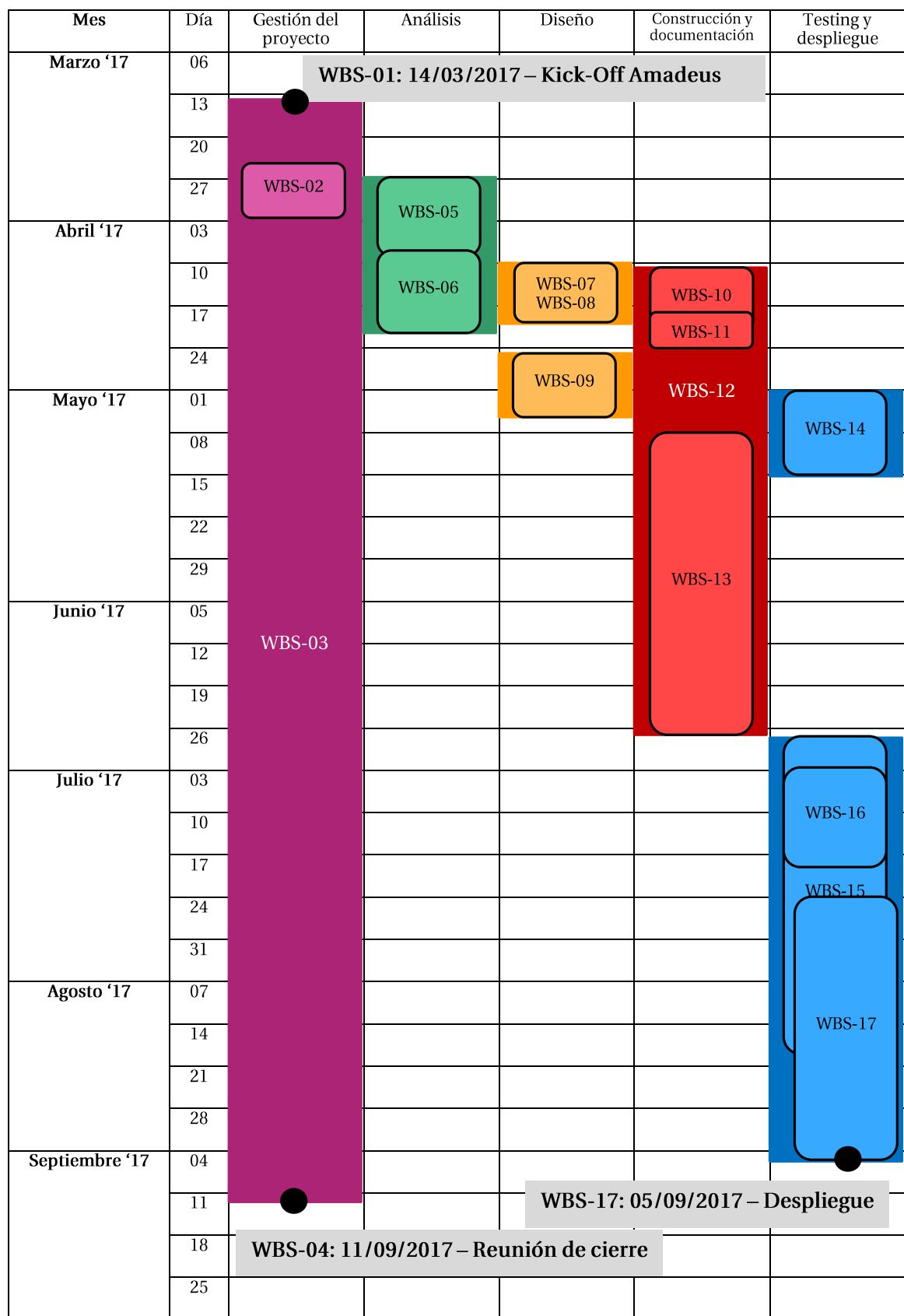


Figura 16: Diagrama de Gantt

## 2.3 Partes interesadas

En esta sección se enumeran los interesados internos y 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.

### 2.3.1 Interesados internos

Los interesados internos de este proyecto son:

#### Dpto. de integraciones aéreas

El proyecto de modificación de reservas a través del servicio [ATC Shopper](#) de Amadeus cuenta con un único desarrollador, integrante del Dpto. de integraciones aéreas de TravelgateX. Él es el encargado de planificar, analizar, diseñar y construir todas las fases de este proyecto.

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 pueden verse enriquecidos de cualquier mejora o funcionalidad compartida que surja a partir del desarrollo. Un claro ejemplo es la posibilidad de compartir el formulario de pruebas.

### 2.3.2 Interesados externos

Los interesados externos de este proyecto son:

#### Amadeus

El proveedor va a empezar a vender una nueva clase de producto.

#### Logitravel

La agencia que 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.

### 2.3.3 Relación entre interesados

En la [Figura 17](#) se representa visualmente la relación entre los interesados internos y externos del proyecto.

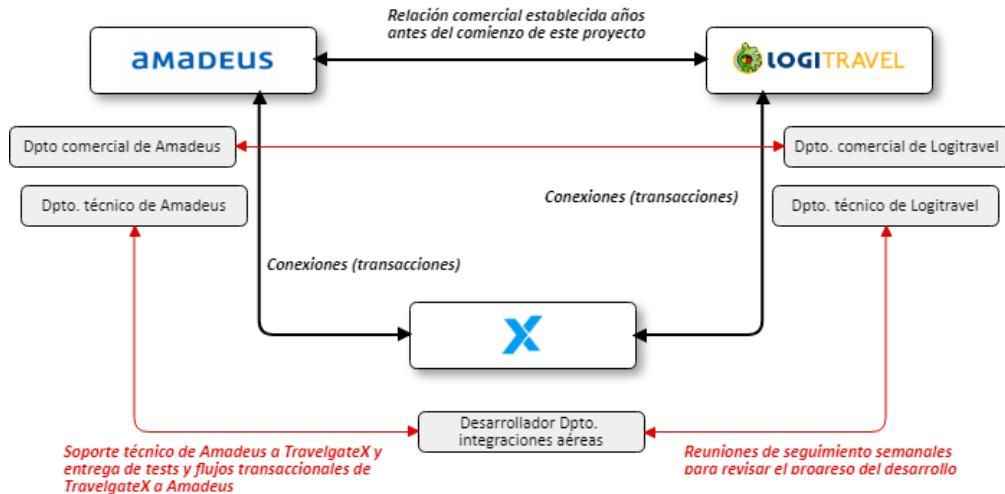


Figura 17: Relación entre los interesados del proyecto

## 2.4 Plan de gestión de comunicaciones

En esta sección se detalla cual ha de ser el plan de comunicación, interno y externo, entre los participantes del proyecto. Incluidas en la sección se encuentra el apartado [2.4.1](#) en el que se define la situación de los trabajadores de la empresa TravelgateX, así como las herramientas de comunicación internas con las que trabajan sus empleados. A continuación, en los apartados [2.4.2](#) y [2.4.3](#) se explican las herramientas y cómo utilizarlas para mantener la comunicación con los interesados externos.

### 2.4.1 Comunicación interna

El departamento de integraciones de TravelgateX se separa en 3 áreas: vuelos, hoteles y *ancillaries*. Aunque ninguna de estas tres áreas comparte API de desarrollo, muchos detalles, consultas o dudas, pueden llegar a resolverse con la colaboración directa de sus integrantes. 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* [21] 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 que se realice en el área de vuelos y viceversa.

### 2.4.2 Comunicación con Logitravel

La herramienta utilizada normalmente para comunicar el desarrollo de las tareas es *Jira* [22]. 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 mismo *ticket*. Esta 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 el mismo departamento pueden seguir la conversación y el conocimiento aprendido queda adherido al área.

El desarrollador de este proyecto 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.

### 2.4.3 Comunicación con Amadeus

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 abierto a los desarrolladores y al personal comercial de sus clientes. A través de él también pueden abrirse incidencias.

## 2.5 Riesgos

En esta sección 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.

Cada uno de los riesgos se desglosan en: riesgo, descripción y estrategia para mitigarlo.

### 2.5.1 Complejidad del proyecto

#### Riesgo

La aplicación por desarrollar planificada es uno de los proyectos con mayor complejidad realizados en el departamento de integraciones aéreas.

#### Descripción

Ninguno de los integrantes del equipo tiene más de un año de experiencia con GDSs. 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 gestionando 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.

#### Estrategia

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

## 2.5.2 Planificación temporal

### Riesgo

La jornada laboral del integrador no está dedicada exclusivamente a este proyecto.

### Descripción

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.

### Estrategia

La estrategia para mitigar este riesgo es la división de la jornada laboral en horas, y dedicar siempre 5 horas al desarrollo del proyecto y 3 horas a gestión de incidencias y mejoras. Puesto que ninguna de las incidencias resueltas los jueves va a desplegarse en producción los viernes, otra posible pauta es dedicar el 100% de las horas de los jueves al proyecto.

### Riesgo

Limitado número de integrantes del proyecto, tanto interna como externamente.

### Descripción

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 el lado de Amadeus.

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

### Estrategia

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.

## ANÁLISIS

En este capítulo se definen los requisitos funcionales y no funcionales de los productos tras una fase exhaustiva de análisis. En la sección [3.1](#) se detallan los requisitos no funcionales y en la sección [3.2](#) los requisitos funcionales.

### 3.1 Requisitos no funcionales

Esta sección se separa en tres apartados:

- Requisitos no funcionales de las nuevas transacciones de la [API](#) de transportes ([3.1.1](#))
- Requisitos no funcionales de la integración ([3.1.2](#))
- Requisitos no funcionales del formulario de pruebas ([3.1.3](#))

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

La [Tabla 19](#) contiene la lista de requisitos no funcionales de las nuevas transacciones de la [API](#) de transportes.

Tabla 19: Requisitos no funcionales de las transacciones de la [API](#) 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 <a href="#">XML</a> .
RNF-03	Las nuevas llamadas han de permitir la creación en última instancia de archivos <a href="#">.wsdl</a> y <a href="#">.xsd</a> , 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 <a href="#">API</a> 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 <a href="#">API</a> de transportes.
RNF-07	Para permitir la serialización y deserialización de objetos a <a href="#">XMLs</a> , los elementos creados deberán ser del tipo <a href="#">XmlElement</a> y los atributos del tipo <a href="#">XmlAttribute</a> .
RNF-08	Todas las clases creadas en la <a href="#">API</a> deberán ser públicas.
RNF-09	Todas las clases creadas en la <a href="#">API</a> deberán heredar de las clases básicas de la <a href="#">API</a> , <a href="#">Transportation</a> y estas a su vez de <a href="#">Base</a> , 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 han de permitir su visualización en español y en inglés.
--------	--

### 3.1.2 Requisitos no funcionales de la integración

La Tabla 20 contiene la lista de 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

La Tabla 21 contiene la lista de 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 esta sección se enumeran los requisitos funcionales, distinguiendo aquellos que deberá cumplir la integración, en el apartado [3.2.1](#), de aquellos que deberá cumplir el formulario de pruebas, en el apartado [3.2.2](#).

Ha de tenerse en cuenta que no existen requisitos funcionales de la integración, puesto que esta se construye a partir de los requisitos funcionales de la [API](#) de transportes, es decir, la integración se encarga simplemente de traducir el formato Amadeus al formato TravelgateX y viceversa, siguiendo los requisitos presentados en [3.2.1](#).

### 3.2.1 Requisitos funcionales de las transacciones de la API de transportes

En la [Tabla 22](#) se enumeran los requisitos funcionales de las nuevas transacciones de la [API](#) de transportes.

Tabla 22: Requisitos funcionales de las transacciones de la [API](#) de transportes

	Requisitos funcionales de las transacciones de la API de transportes
	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 URLs a las que atacará la transacción.
RF-03	El cliente elige el idioma de las transacciones XML.
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
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, OW, OJ, CT.
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, OW, OJ, CT.
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, CHD e INF.
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: <u>ADT</u> , <u>CHD</u> e <u>INF</u> .
	En la transacción <u>RMR</u>
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 <u>DMR</u> y el precio que se obtenga en la <u>RMR</u> ) 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 <u>LTD</u> .
RF-54	El cliente visualiza el tipo de emisión.
	En la transacción <u>EMR</u>
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.

### 3.2.2 Requisitos funcionales del formulario de pruebas

En la Tabla 23 se enumeran los requisitos funcionales del formulario de pruebas.

Tabla 23: Requisitos funcionales del formulario de pruebas

	Requisitos funcionales del formulario de pruebas
	En todas las transacciones
RF-59	El usuario copia la petición y la respuesta <u>XML</u> .
RF-60	El usuario indica el tiempo de corte de la transacción.
RF-61	El usuario indica las credenciales y las <u>URLs</u> .
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: <u>OW</u> , <u>RT</u> o <u>OJ</u>

	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, <u>CVV</u> 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: <u>OW</u> , <u>RT</u> o <u>OJ</u> .
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: <u>N</u> , <u>KE</u> , <u>K</u> , <u>C</u> , <u>R</u> o <u>A</u> .

Este capítulo contiene los diseños elaborados a partir de los requisitos extraídos del análisis. Primero, en la sección [4.1](#), se detallan los modelos de datos de las tres transacciones a desarrollar en la API de Transportes para que un Logitravel pueda modificar una reserva a través de TravelgateX.

A continuación, en la sección [4.2](#), se presentan los diagramas de flujo que deberán ser ejecutados en la integración para modificar reservas con Amadeus.

Finalmente, en la sección [4.3](#) se detallan los *mockups* diseñados para las principales funcionalidades del formulario de pruebas.

## 4.1 Modelo de datos de las transacciones

En esta sección se define el modelo de datos que tendrá que construirse y las relaciones entre los datos a partir de los requisitos definidos en el capítulo [3 Análisis](#).

La representación de los diagramas mostrados en esta sección 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 apartado [5.1.2](#).

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

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

Se utilizan las clases *TransportationBaseRQ* y *TransportationBaseRS* previamente desarrolladas y disponibles para el desarrollador de TravelgateX en la API de Transportes. No es necesaria la elaboración del modelo de datos de estas clases puesto que ya están construidas y disponibles para su utilización por parte del desarrollador.

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

Figura 18: Modelo de datos de la transacción Disponibilidad Modificar Reserva ([DMR](#)).

La petición [DMR\\_RQ](#) consta de conjunto de tramos (*TramoDisponibilidad*), Localizadores y una serie de Preferencias. Cada *TramoDisponibilidad* consta de una o más Localizaciones de Origen y Destino. En las Preferencias se pueden añadir CompañíasConexión.

La respuesta [DMR\\_RS](#) consta de una lista de Localizadores, una lista de Segmentos y una lista de Tarifas, así como los *Tickets* de la reserva. Cada Tarifa consta de los siguientes conjuntos:

- Condición

- Opción: contiene una lista de ReferenciaSegmento (cada una de estas contiene una lista de ClasesSegmento)
- DesgloseImporte: contiene dos listas, una para los Cargos y otra para los DesglosePasajero
- ConfiguraciónPasajero: contiene una lista de Bonificaciones

Cada segmento consta de una o más Localizaciones de Origen y Destino.

#### **Del requisito RF-43 al RF-54:**

Figura 19: Modelo de datos de la transacción Reserva Modificar Reserva (RMR).

La petición RMR\_RQ contiene los siguientes conjuntos de elementos:

- Desglose: cada desglose está formado por un conjunto de Tramos, Condiciones, DesgloseImporte y ConfiguracionesPasajeros.
- *Ticket*
- Localizador
- InfoPago: formado por un conjunto de Datos y éste a su vez por un conjunto de Tarjetas.

Un Tramo contiene una lista de SegmentoTramo. Cada SegmentoTramo contiene un Segmento y una lista de ClasesSegmento.

La respuesta RMR\_RS por su parte contiene una lista de Localizadores y *Tickets* (igual que la petición) y una lista de Pasajeros, así como la Factura. Una factura constituye de un conjunto de DesgloseImportes y un conjunto de Cargos.

#### **Del requisito RF-55 al RF-58:**

Figura 20: Modelo de datos de la transacción Emitir Modificar Reserva (EMR).

La petición EMR\_RQ y la respuesta EMR\_RS de esta transacción contienen ambas una lista de Localizadores y una lista de *Tickets*.

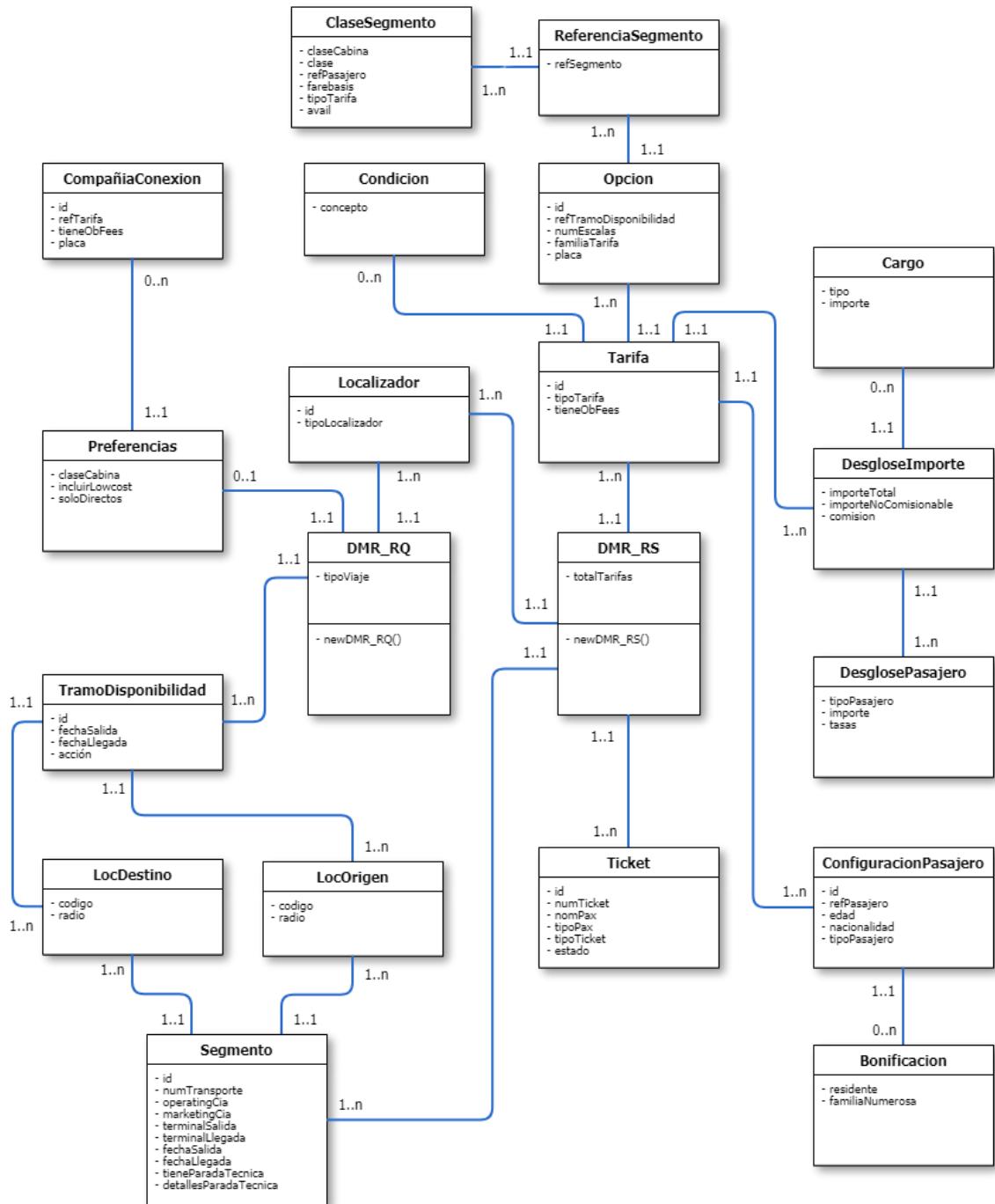


Figura 18: Modelo de datos de la transacción DMR

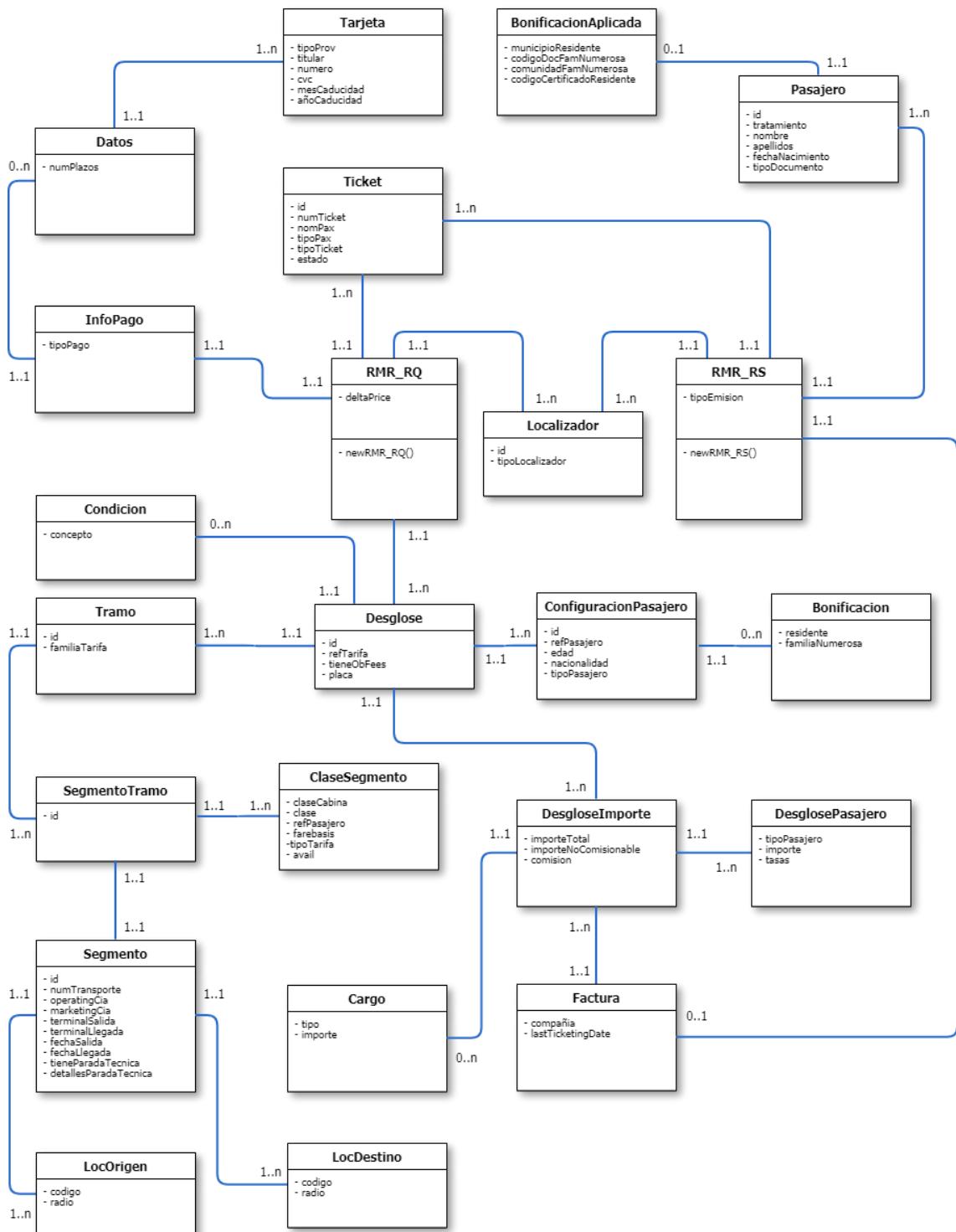


Figura 19: Modelo de datos de la transacción RMR

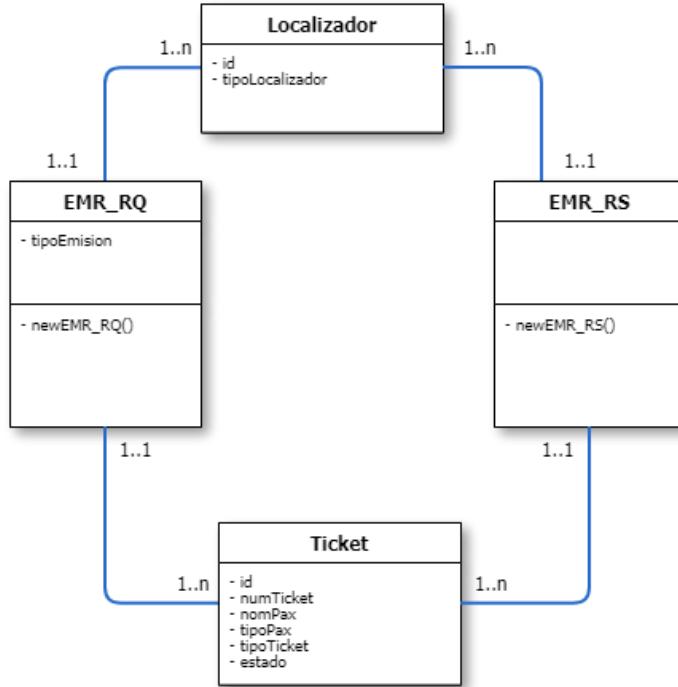


Figura 20: Modelo de datos de la transacción EMR

## 4.2 Diagramas de flujo de transacciones con el GDS

Las representaciones mostradas en esta sección 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.

Los diagramas representados en las figuras Figura 21, Figura 22 y Figura 23, corresponden a la segunda versión diseñada por recomendación de Amadeus. La primera versión presentada a Amadeus, ejecutaba alguna de las operaciones con un tipo de sesión inadecuada, lo que hubiera supuesto un *warning* en la certificación.

### Disponibilidad DMR

La transacción DMR permite obtener las recomendaciones disponibles existentes en ATC Shopper MasterPricer de acuerdo a los parámetros solicitados por el usuario, que son los siguientes:

- Cambio de clase
- Cambio de aerolínea
- Cambio de horario
- Cambio de itinerario

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

En cualquier otro caso se sigue el flujo abriendo el TST para consultar información de los pasajeros y cerrando sesión. El resto de llamadas se realizan con sesión *stateless*.

A continuación, para cada itinerario, se comprueba si los tickets son modificables con Ticket\_CheckEligibility y en caso afirmativo se realiza la petición a Ticket\_ATCShopperTravelBoardSearch. Ésta devuelve las recomendaciones que hay disponibles con los criterios que se han solicitado.

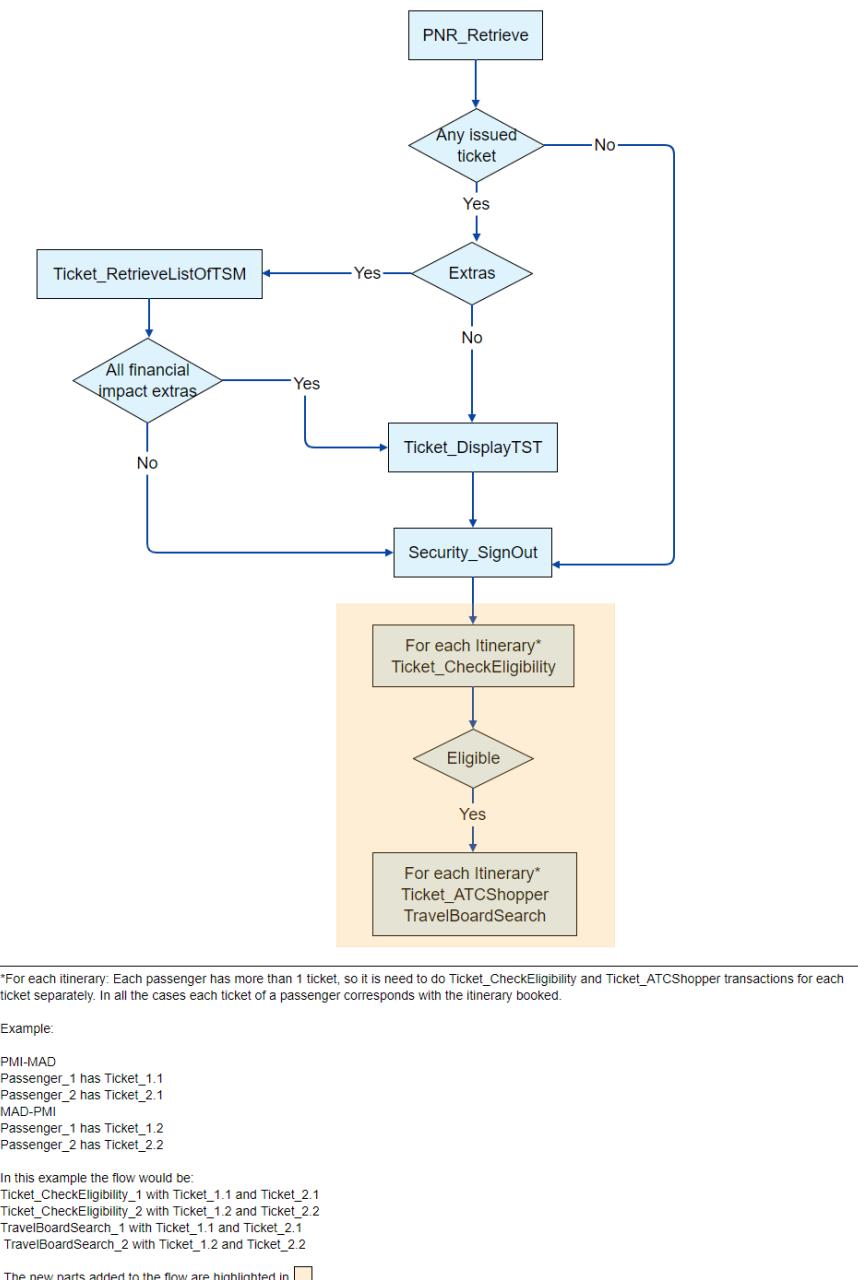


Figura 21: Diagrama de flujo de ejecución de la transacción DMR

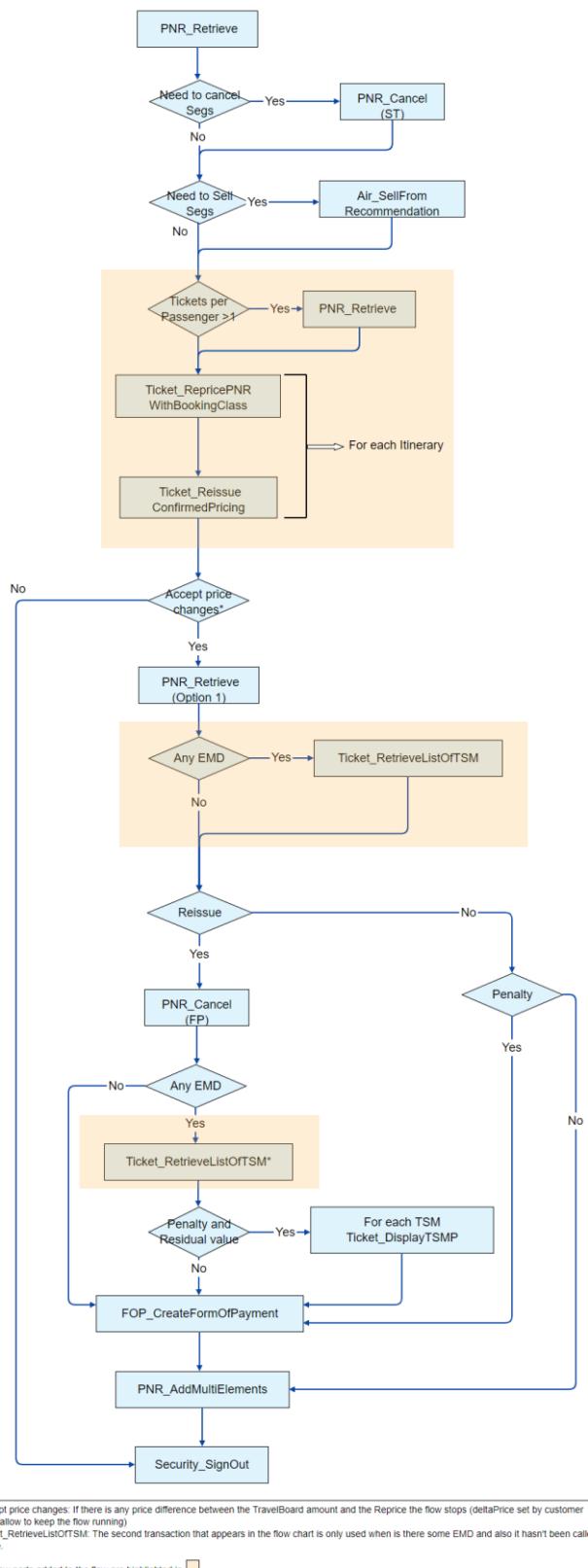


Figura 22: Diagrama de flujo de ejecución de la transacción RMR

### **Reserva RMR**

La transacción RMR empieza recuperando el PNR y comprobando que cambios se han solicitado en la transacción DMR. Se cancelan los segmentos que sean necesarios y se realiza el Sell de aquellos segmentos solicitados por el usuario.

Si existe más de un ticket asociado a algún pasajero, se recupera y se reabre el PNR.

Una vez que se han cancelado y añadido todos los segmentos, se ejecuta un Ticket\_RepricePNRWithBookingClass por cada itinerario de la reserva. En la respuesta se recibirán los importes de penalización, residual value, tasas e importes totales definitivos. Para confirmar se llama a Ticket\_ReissueConfirmedPricing.

A continuación se comprueba si hay cambio de precio con respecto al importe obtenido en DMR. Si hay cambio de precio y el *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.

Ahora se comprueba la existencia de EMDs en el PNR y en el caso de que hubiera alguno, se recupera la lista de TSMs con la llamada Ticket\_RetrieveListOfTSM.

A continuación hay que realizar los cambios necesarios en la forma de pago dependiendo del tipo de reemisión/revalidación que se esté dando. Se cancela la forma de pago mediante PNR\_Cancel 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 TSMs que se hayan podido crear al hacer el Reprice.

Finalmente se cierra el PNR confirmando los cambios mediante PNR\_AddMultiElements y cerrando sesión con Security\_SignOut.

En la Figura 22 se puede apreciar la representación gráfica mediante un diagrama de flujo del flujo de ejecución explicado para RMR.

### **Emisión EMR**

La transacción EMR empieza recuperando el PNR. Se comprueba si hay alguna comisión parcial que haya podido quedar al cancelar algún segmento. Si existe alguna comisión, ésta se elimina (PNR\_Cancel) y se crea una nueva vinculada a todos los segmentos existentes en el PNR modificado.

Dependiendo de si es una revalidación, reemisión y de si tiene o no EMDs asociados, se sigue un camino u otro en el flujo. Hay en total 5 posibilidades:

1. Revalidación sin EMD
2. Revalidación con EMD
3. Reemisión sin EMD
4. Reemisión con EMD (sólo *residual value*)
5. Reemisión con EMD (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 de penalización con DocIssuance\_IssueMiscellaneousDocuments. Si es un escenario 3, se realiza la emisión de cada pasajero con DocIssuance\_IssueTicket por separado, recuperando el PNR con PNR\_Retrieve tras cada emisión.

Si es un escenario 4 o un escenario 5, ha de recuperarse el TST con Ticket\_DisplayTST para obtener la información que se necesitará para emitir. Si es un escenario 4 se realiza una llamada a DocIssuanceIssueCombined para cada TSM, mientras que si es un escenario 5 se realiza una llamada a DocIssuanceIssueCombined para cada TST. Tras cada llamada a DocIssuanceIssueCombined se reabre el PNR con PNR\_Retrieve si todavía faltan TSM/TST 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 para obtener los nuevos números de billete y finalmente se cierra sesión con Security.SignOut.

La Figura 23 es la representación gráfica del flujo que acaba de explicarse.

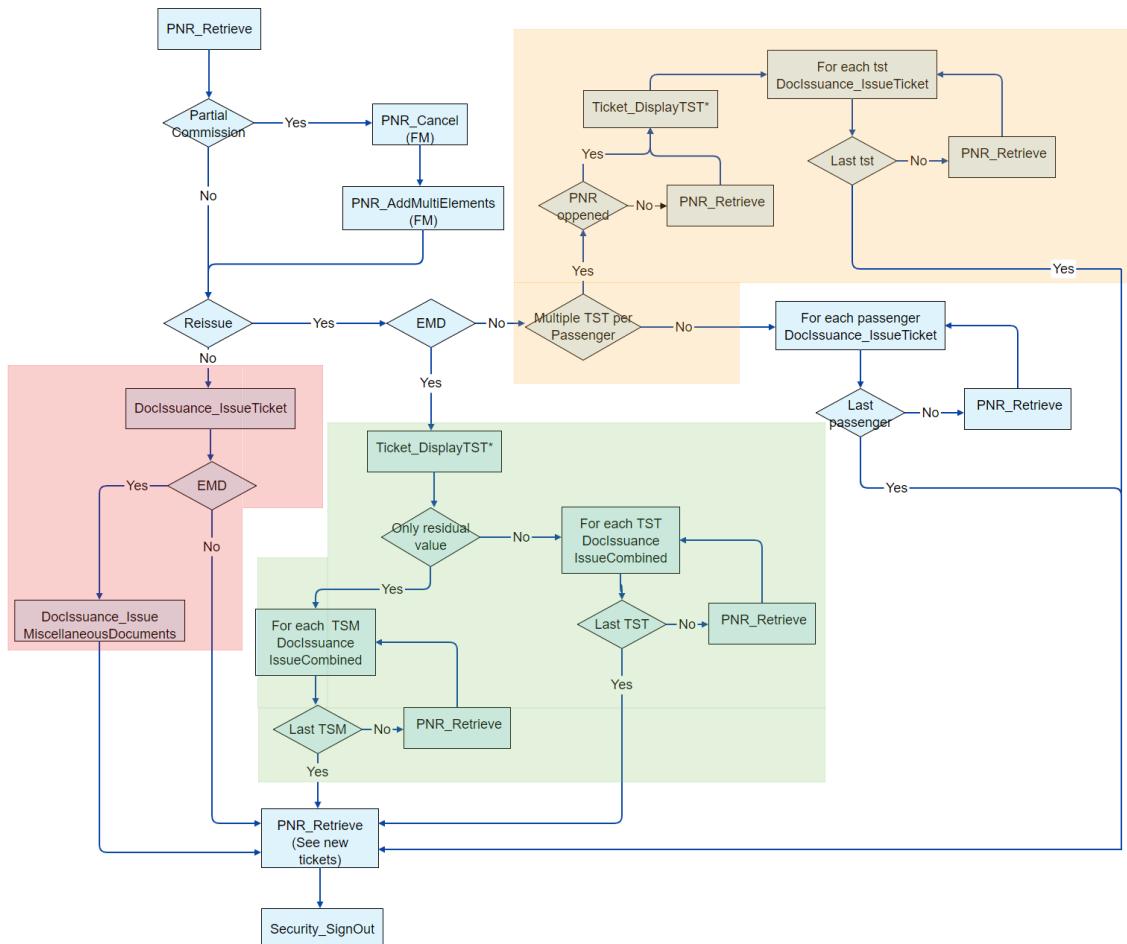


Figura 23: Diagrama de flujo de ejecución de la transacción EMR

#### 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 y respuestas RS XML. 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 (Disponibilidad o DMR).

Los datos presentados hasta ahora están disponibles en cualquiera de las pestañas del formulario.

La Figura 24 es el *mockup* de la pestaña de “Disponibilidad-Valoración” en la que el usuario cuenta con todos los parámetros para realizar las transacciones de Disponibilidad y Valoración. Una vez seleccionado el tipo de viaje, los tramos y los pasajeros, el usuario podrá proceder a realizar la petición de Disponibilidad. Las opciones obtenidas se pintarán en los cuatro cuadros de la parte inferior derecha. El usuario entonces podrá seleccionar las opciones y pulsar el botón de Valoración para cotizar la tarifa.

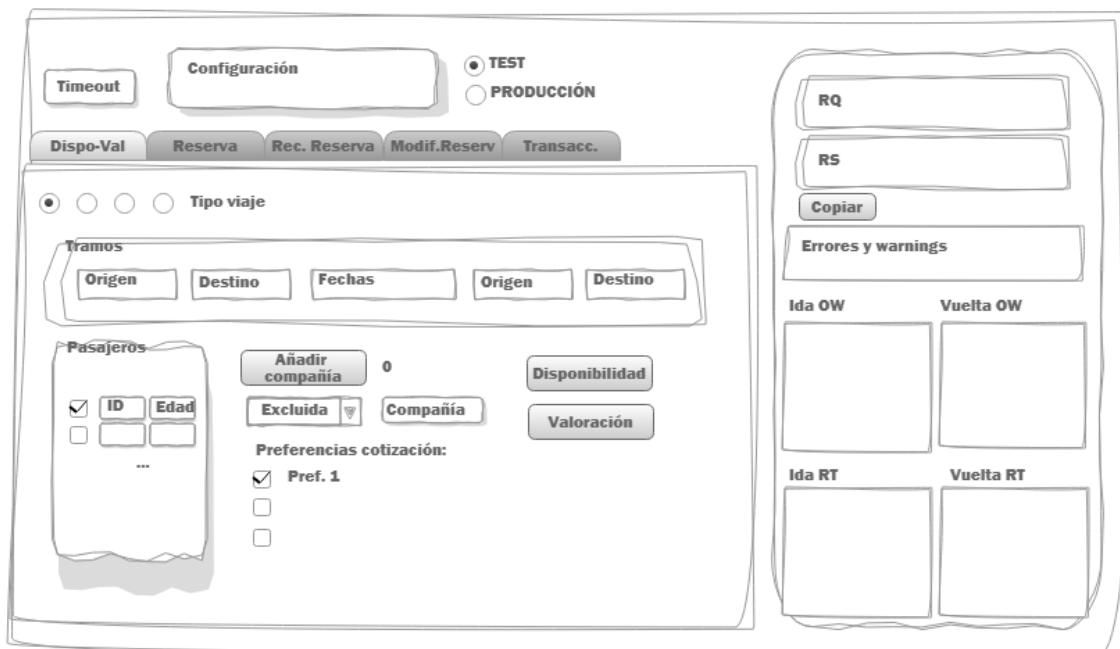


Figura 24: Mockup de la pestaña Disponibilidad-Valoración

En la Figura 25 se representa la pestaña de “Reserva”, en la que el usuario cuenta con todos los parámetros para realizar la transacción de Reserva. Para poder reservar, el usuario seleccionaría mediante los *checkbox* de la parte superior, los pasajeros de la reserva y rellenaría los datos para cada uno de ellos a través de los campos de texto de cada columna. Una vez introducidos los datos del cliente y la forma de pago podrá proceder a pulsar el botón de Reservar para realizar la reserva.

The mockup shows the 'Reserva' tab selected. At the top, there are configuration options for 'TEST' and 'PRODUCCIÓN'. Below the tabs, there are buttons for 'Dispo-Val', 'Reserva', 'Rec. Reserva', 'Modif.Reserv', and 'Transacc.'. The main area has sections for 'Información del cliente' (with a 'Nombre' field) and 'Pasajeros' (with a grid for entering passenger details like Sex, Name, Surname, and three additional fields). To the right, there's a 'Información de pago' section with fields for 'CARD', 'Titular', 'Número', 'Tipo tarjeta', 'CVC', and 'Caducidad'. On the far right, there's a panel for 'RQ' and 'RS' with a 'Copiar' button, and an 'Errores y warnings' section. Below these are four large empty boxes labeled 'Ida OW', 'Vuelta OW', 'Ida RT', and 'Vuelta RT'.

Figura 25: Mockup de la pestaña Reserva

En la Figura 26 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. Para ello, el usuario simplemente tiene que introducir el localizador en el campo de texto indicado y pulsar uno de los botones presentados en esta pestaña.

The mockup shows the 'Recuperar Reserva' tab selected. At the top, there are configuration options for 'TEST' and 'PRODUCCIÓN'. Below the tabs, there are buttons for 'Dispo-Val', 'Reserva', 'Rec. Reserva', 'Modif.Reserv', and 'Transacc.'. The main area has two columns of buttons: one for 'Localizador' (with 'Recuperar reserva', 'Cancelar reserva', and 'Emitir billetes') and another for 'Billete' (with 'Billete', 'Void billete', 'Tipo emisión', and 'Tipo billete'). To the right, there's a panel for 'RQ' and 'RS' with a 'Copiar' button, and an 'Errores y warnings' section. Below these are four large empty boxes labeled 'Ida OW', 'Vuelta OW', 'Ida RT', and 'Vuelta RT'.

Figura 26: Mockup de la pestaña Recuperar Reserva

En la Figura 27 se representa la pestaña de “Modificar Reserva” en la que el usuario cuenta con todos los parámetros para realizar las transacciones de DMR, RMR y EMR. El procedimiento para realizar la transacción DMR es muy similar al de Disponibilidad. El usuario escogería un tipo de viaje, los tramos y la acción a realizar para cada uno de ellos. Las opciones se mostrarían de nuevo en los cuatro cuadros de la parte inferior derecha.

Tras seleccionar las opciones mostradas en los cuadros de la esquina inferior derecha, el usuario puede pulsar el botón Reservar para ejecutar la transacción RMR. Finalmente, el usuario puede realizar la transacción EMR pulsando el botón Emitir.

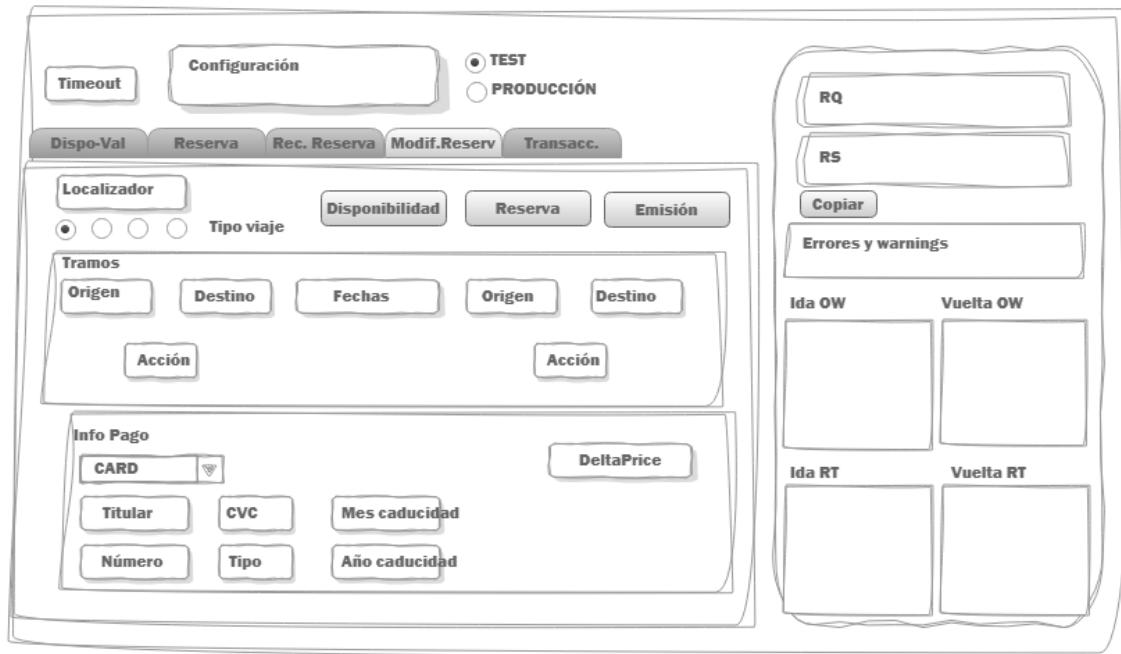


Figura 27: Mockup de la pestaña Modificar Reserva

## CONSTRUCCIÓN

Este capítulo muestra como se ha llevado a cabo la construcción de los siguientes entregables:

- Transacciones API Transportes (véase en la sección [5.1](#))
- Integración (véase en la sección [5.2](#))
- Formulario de pruebas (véase en la sección [5.3](#))
- Tests unitarios incluidos en la solución de la integración (véase en la sección [5.4](#)).

### 5.1 Transacciones de la API de Transportes

En el apartado [5.1.1](#) de esta sección, se indica el lenguaje de programación y librerías utilizadas para desarrollar las nuevas transacciones de la API de Transportes tal y como se han representado en el diseño expuesto en la sección [4.1](#), y en base a los requisitos funcionales y no funcionales enumerados en el capítulo [3 Análisis](#). Después, en el apartado [5.1.2](#), se describen al detalle las transacciones programadas.

#### 5.1.1 Lenguaje y tecnología utilizadas

El lenguaje de programación con el que se programan las nuevas transacciones de la API de Transportes, así como los cambios necesarios introducidos en los elementos ya existentes en otras transacciones, es Visual Basic [17].

Se ha escogido este lenguaje porque el resto de transacciones que ya incluye la API está desarrollado con este mismo lenguaje. Sin embargo, una alternativa válida para desarrollar la misma API desde cero en otro lenguaje hubiera podido ser C# [23]. Ofrece las mismas funcionalidades que ya se han utilizado para desarrollar la API de Transportes y además permite crear los ficheros necesarios para subirse en los frontales del servicio WSDL que existe en el sistema de TravelgateX.

Visual Basic, C#, o cualquier otro lenguaje de Microsoft compatible, es estrictamente necesario para que el cliente pueda acceder a las integraciones a través de la instalación de .dll's [24].

Al ser un lenguaje del paradigma de programación “Programación Orientada a Objetos” (POO), 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](#)) son públicos.

Las tres llamadas desarrolladas son clases serializables [19], es decir no pueden ser heredadas por ninguna subclase. Además las tres clases desarrolladas 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 [25] y se indican los ElementName [26] 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](#), es importante definirlos con el tipo XmlElement [25].

Los atributos son del tipo XmlAttributeMultiLanguage [27] 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](#) deben definirse del tipo XmlRootMultiLanguage [28].

Para la programación de las clases, elementos y atributos, son suficientes las librerías del sistema (Microsoft) [XML](#) y librerías internas de la [API](#) 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 apartado se detallan los elementos y atributos que contienen las nuevas llamadas de la [API](#) de Transportes que se usarán en la integración de Amadeus para modificar reservas. Los elementos y atributos expuestos en las tablas [Tabla 24](#), [Tabla 25](#) y [Tabla 26](#), no representan el código en si (programado en Visual Basic), sino la definición (cardinalidad, tipo y descripción) de cada uno de ellos. Este formato es el que se ha utilizado para documentar los cambios realizados en la [API](#) y que, como se ha establecido en el alcance del proyecto, es uno de los entregables que han de presentarse en TravelgateX.

En este documento se describe aproximadamente el 60% de los elementos que se han desarrollado para las nuevas transacciones [DMR](#), [RMR](#) y [EMR](#), los más importantes de cada transacción.

[Tabla 24: Elementos y atributos más importantes de DMR \(véase en Anexo A.13\)](#)

[Tabla 25: Elementos y atributos más importantes de RMR \(véase en Anexo A.14\)](#)

[Tabla 26: Elementos y atributos más importantes de EMR \(véase en Anexo A.15\)](#)

## 5.2 Integración

En esta sección se explica el lenguaje y la tecnología utilizada para programar la integración (apartado [5.2.1](#)) y se destacan sus funcionalidades más importantes (apartado [5.2.2](#)).

### 5.2.1 Lenguaje y tecnología utilizadas

El lenguaje de programación con el que se programa la integración es Visual Basic [17]. Se ha escogido este lenguaje porque la integración (el resto de transacciones) ya estaba desarrollada en este lenguaje.

Visual Basic (o C# [23]) es necesario para que el cliente pueda atacar a las integraciones por .dll [24] 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*.
- GetElapsedMilliseconds: 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 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 cada uno de los paquetes de *Web Services* de Amadeus (ej.: ConnectionManagerBook para todas las llamadas del paquete de *Web Services* de tarificación y reserva de recomendaciones de Amadeus).

## **BookingRetrieve**

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

## **Eligibility**

Funcionalidad que se encarga de construir la petición para la llamada Ticket\_CheckEligibility y procesar y parsear [29] la respuesta recibida. Si un ticket no es modificable, se añadiría un error de "Ticket no modificable" en la respuesta de DMR.

## **Travelboard**

Funcionalidad que se encarga de construir la petición para la llamada Ticket\_ATCShopperMasterPricer\_TravelBoardSearch y procesar y parsear (transformar al formato TravelgateX) la respuesta recibida. Monta todas las tarifas y los segmentos a partir de los datos obtenidos en la respuesta.

## **BookingMake**

Funcionalidad 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**

Funcionalidad que se encarga de construir la petición para la llamada Ticket\_RepricePNR\_WithBookingClass y procesar y parsear la respuesta recibida.

## **ReissueConfirmedPricing**

Funcionalidad que se encarga de construir la petición para la llamada Ticket\_ReissueConfirmedPricingTicketInfo y procesar y parsear la respuesta recibida.

## **GetTipoEmision**

Funcionalidad que calcula el tipo de emisión que ha de realizarse en la transacción EMR a partir de:

- repriceType: tipo de *reprice* que Amadeus indique

- isThereEMD: si el PNR tiene cargos EMD o no
- tipoEMD: tipo de EMD que hay en el PNR
- formOfPayment: forma de pago con el que se abona el importe de la modificación de la reserva
- addditionalCollection: si la modificación tiene importe adicional o no

Se obtendrá uno de los siguientes tipos de emisión:

- Revalidate without EMD
- Revalidate with EMD without additional collection
- Reissue without EMD with additional collection
- Reissue with EMD only penalty
- Reissue with EMD only residual value without additional collection
- Reissue with EMD only residual value
- Reissue with EMD with penalty and residual value
- Reissue with EMD with penalty and residual value without additional collection

#### **DoEndTransactionAndRetrieve**

Confirma los cambios realizados en el PNR y lo reabre.

#### **DoIgnoreAndRetrieve**

Ignora los cambios realizados en el PNR 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.

#### **ATC\_IssueCombined**

Realiza la reemisión de los billetes dependiendo de si la modificación de RMR ha generado un valor residual o una penalización por modificación.

### **5.3 Formulario de pruebas**

En el apartado 5.3.1 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.

### 5.3.1 Lenguaje/tecnología

El lenguaje de programación con el que se programa el formulario de pruebas es Visual Basic [17]. Se ha escogido este lenguaje por dos razones: es el lenguaje que tenemos que utilizar en las transacciones de la [API](#) y en la integración y consecuentemente se tiene más práctica y soltura. Otra razón de peso es la mayor velocidad con la que se programa, al carecer éste de puntos y coma, utilizar menos paréntesis y necesitar apenas 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 un ejemplo de uso.

## 5.4 Programación de tests unitarios

En esta sección se destaca la importancia de crear tests unitarios para la integración ([5.4.1](#)) y qué lenguaje de programación es el elegido para programar dichos tests. En el apartado [5.4.2](#) 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](#) 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 (*un string*). Es muy útil, por ejemplo, cuando una reserva ha fallado y no se puede reproducir en producción, ya que supondría un coste. Al pasarle al test, las trazas del error, se reproducirá el comportamiento de la integración en las mismas condiciones en las que se produjo un error, 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](#) sigan el mismo formato.

Pese a que los tests ayudan a mantener el funcionamiento de la integración al hacer cambios sobre esta, los test 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 y tecnología utilizadas

El lenguaje de programación con el que se programa el formulario de pruebas es C# [23].

## **Moq** [30]

Librería que permite la utilización de *mocks*, que no son más que objetos simulados que imitan el comportamiento de objetos reales. Se pueden simular 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 *mockConnectionManager*
- *Setup* de las llamadas de *ConnectionManager* para que devuelvan el objeto que deseemos (p.e. *callPNR\_Retrieve* que devuelva el PNR Reply que se indique en el *return*).
- Creación de un *mockDMR* para llamar al método *process* que ejecuta la transacción.

## PRUEBAS

En la sección [6.1](#) de este capítulo se describen las pruebas que se han realizado para certificar el desarrollo por parte de Amadeus. A continuación, en la sección [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 por parte de Amadeus

En el apartado [6.1.1](#) de esta sección, se presentan los escenarios a probar y presentar en la certificación por parte de Amadeus. En los apartados [6.1.2](#) y [6.1.3](#) se detallan los acontecimientos sucedidos en los dos intentos de certificación.

La certificación del desarrollo [ATC Shopper](#) consta de tres fases:

- Primero el desarrollador ha de realizar las pruebas sobre los escenarios que Amadeus haya indicado ([6.1.1](#)).
- A continuación, se facilitan a Amadeus los diagramas de flujo ([4.2](#)) y los resultados de las pruebas ([6.1.1](#)).
- Finalmente, Amadeus comprueba los resultados y pasa el *checklist* de los criterios de aceptación ([2.1.5](#)).

#### 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 realizan durante la certificación se hacen sobre el entorno de producción (con reservas reales), de tal forma que se pueda 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](#), [16](#) y [17](#) pre-acordados.

Estos son los escenarios que han de probarse (las ciudades y aeropuertos citados a continuación están indicados mediante el estándar [IATA](#)):

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

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

2. Revalidación sin EMD

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

3. Revalidación con EMD

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

4. Reemisión con EMD de penalización

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

5. Reemisión sin EMD de penalización

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

6. *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.

7. *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.

8. *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.

9. *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 alguno de estos casos, se precisa la intervención de Amadeus y de las aerolíneas Iberia y Air Europa. En un momento dado del flujo, las aerolíneas han 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, el técnico puede repasar todos los cambios efectuados en el PNR y la fecha exacta de todos ellos.

### 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 corrección iba a suponer la modificación de los flujos de las transacciones DMR y RMR, lo que implicaría el rediseño de dichos flujos y la programación de los cambios necesarios 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 su uso comercial). Sin embargo, Amadeus iba a incluir la aplicación dentro de una lista de “defectuosas” para tener constancia de una anomalía en el software desarrollado y presentado.

### 6.1.3 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 el *planning* temporal. En la [Figura 28](#), 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](#) 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.

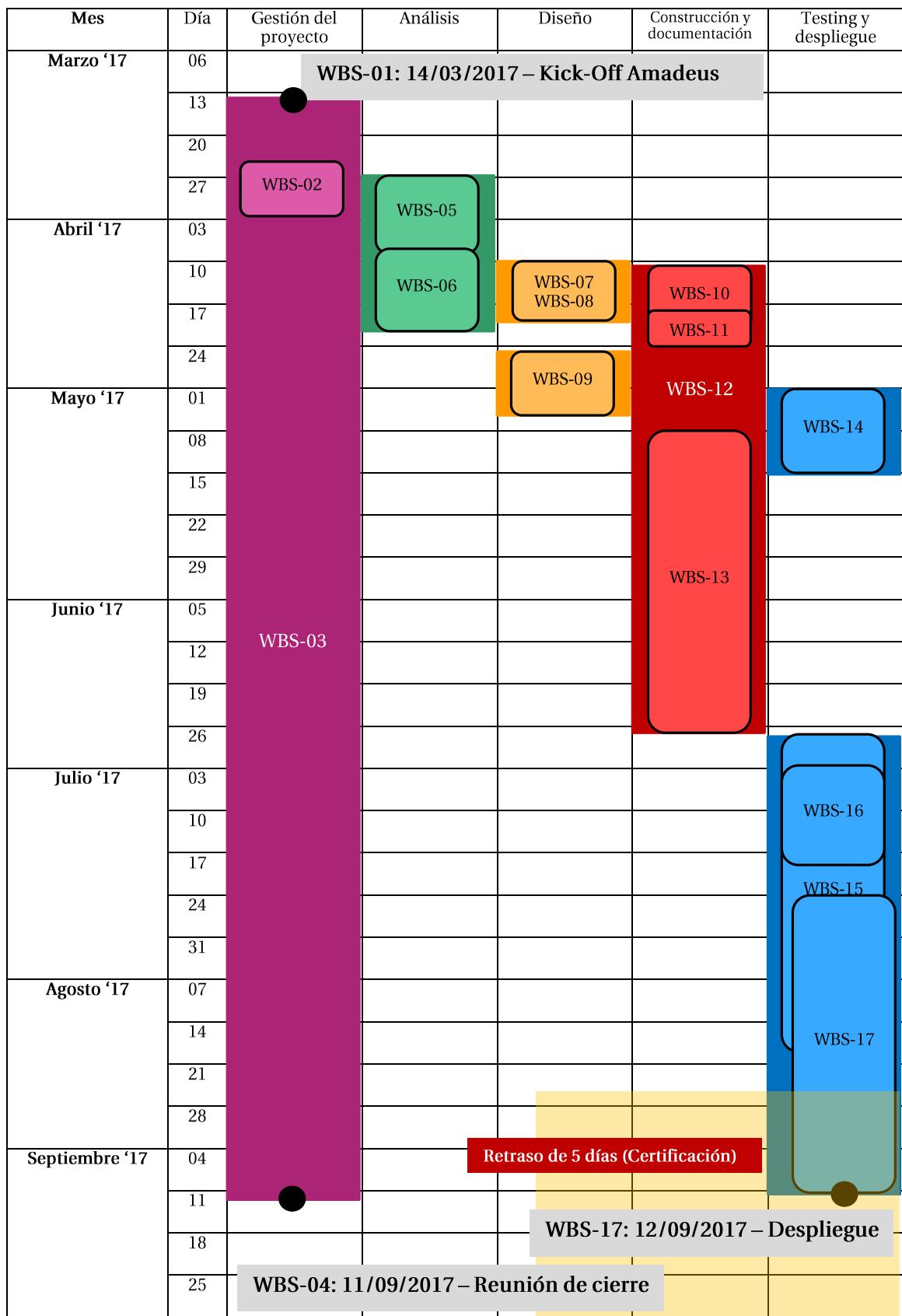


Figura 28: Diagrama de Gantt (desviaciones temporales)

## 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, dado que a partir de entonces, cualquier cliente final podría usar la nueva 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.

Esta fase es más sencilla con la presencia de los tests unitarios ([5.4](#)) que ya incluía la integración previamente al desarrollo (el resto de transacciones que ya había en la integración de Amadeus). 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:

- Flujos completos con múltiples pasajeros de distintas edades
- Flujos con itinerarios que tengan 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

En este capítulo se presenta cuál ha de ser el procedimiento para realizar el despliegue de cualquier integración en producción. En este proyecto, una vez en producción, el cliente tiene acceso a la aplicación desarrollada para modificar reservas a través de Amadeus.

En la sección [7.1](#) de este capítulo se describen los procedimientos y entornos más destacados a tener en cuenta para completar el despliegue de la aplicación. En la sección [7.2](#) se detalla cual es la secuencia de pasos que se realiza para subir una aplicación al entorno de pruebas. Finalmente, en la sección [7.3](#) se explica que ha de hacerse para subir la aplicación al entorno de pre-producción y, posteriormente, al entorno de producción y quién debe realizar estas subidas.

### 7.1 Procedimientos y entornos de despliegue

Todas las líneas de integraciones de TravelgateX cuentan con un paquete de tests unitarios con los que se pueden probar los flujos de transacciones básicos: Disponibilidad, Valoración y Reserva. Normalmente, y si el proveedor lo permite, se realizan todas las pruebas sobre su entorno de test. Por lo tanto, pese a no ser una metodología 100% fiable, sirve para descartar errores graves cometidos durante el desarrollo.

Antes de poder subir la integración en cualquiera de los tres entornos mencionados, es necesario que las pruebas sobre los tests unitarios tengan un porcentaje de errores por debajo del 5%. Además, ninguno de los errores contenidos dentro de este 5% pueden ser errores de sistema tales como:

- Acceso a un objeto vacío
- Acceso a un elemento inexistente
- Errores de serialización y deserialización [19] de [XML](#)
- Sobreuso de memoria o cpu

Otro aspecto a tener en cuenta para realizar el despliegue del desarrollo, es la existencia de tres diferentes entornos por los que debe instalarse la integración antes de poder considerarse desplegada.

Los entornos de despliegue de la aplicación son:

- Entorno de test o entorno de pruebas
- Entorno de pre-producción

- Entorno de producción

Los pasos a seguir y el equipo de TravelgateX que se encarga de ellos, se detalla en las siguientes secciones [7.2](#) y [7.3](#).

## 7.2 Instalación de la integración en el entorno de test

El primero de los entornos por el que tiene que instalarse la integración y comprobar su funcionamiento es el entorno de test. Ningún cliente puede realizar pruebas sobre este entorno, siendo así interno de la empresa, para la realización de pruebas sin una excesiva carga de memoria y procesamiento. Esto se debe a que todas las transacciones que se lancen sobre este entorno se realizan íntegramente a través de una sola máquina servidor.

El procedimiento para realizar la subida de la integración a este entorno es el más sencillo de los tres. Está formado por los siguientes pasos:

- Compilación del paquete *solution* de la integración Amadeus. Como resultado se obtiene el fichero .dll [24].
- Creación de un usuario para realizar la subida de la dll al servidor.
- Subida de la dll al servidor.

Una vez cargada la dll en el servidor de pruebas, pueden empezar a realizarse los tests atacando a la [URL](#) en la que está alojado el servidor, a través de las credenciales internas correspondientes a este entorno. Es importante comprobar entonces el correcto funcionamiento de todas las transacciones de la integración a través de los tests unitarios mencionados en el apartado [7.1](#).

## 7.3 Instalación de la integración en los entornos de pre-producción y producción

Una vez comprobado el funcionamiento de la integración en el entorno de test, el equipo de DEVOPS (*Development and Operations team* in TravelgateX) es el encargado de instalar la integración en el entorno de pre-producción. Esto significa que la integración será instalada en servidores de *google* y *azure* (el número de servidores dedicados a cada integración dependerá de la carga de peticiones que se realicen sobre ésta).

El desarrollador de la integración es el encargado de nuevo de comprobar que la instalación sobre el entorno se ha completado con éxito, realizando de nuevo las mismas pruebas que ya se han probado y completado antes sobre el entorno de test.

Una vez realizada la instalación, el cliente ya puede lanzar peticiones sobre este entorno. De esta forma, mientras que se procede a la instalación de la integración sobre el entorno de producción, el cliente puede comprobar también que se obtienen las respuestas adecuados en este pre-entorno real.

Finalmente, y una vez comprobada la correcta instalación sobre el entorno de pre-producción, se realiza la instalación sobre producción.

Cada día se realizan más de un millón de peticiones de disponibilidad a la integración de Amadeus en el entorno de producción. Por esta razón es crítico que se realice de inmediato, una vez instalada la integración, una monitorización de las peticiones de forma exhaustiva. Si se detecta alguna anomalía o el porcentaje de errores se eleva, se procedería al *rollback* de toda la instalación a la última versión estable de la integración.

## CONCLUSIONES

Este documento ha servido para describir con detalle la elaboración de una aplicación para modificar reservas aéreas a través del *Web Service* complejo de un tercero, como es Amadeus. Se ha detallado el contexto en el que está envuelto el desarrollo de la integración y el plan que se ha elaborado para llevarlo a cabo. Se han incluido también las fases de Análisis, Diseño y Construcción, así como las Pruebas y los entornos y procedimientos necesarios para el despliegue de la aplicación en la empresa en la que se ha construido.

Ha de destacarse que, durante la elaboración del trabajo, se ha buscado facilitar la posterior descripción y elaboración de esta memoria, permitiendo así que muchas de las fases del trabajo se vean plasmadas en este documento de manera descriptiva e instructiva. De esta forma, se ha buscado que el lector entienda muchos de los conceptos que envuelven este trabajo casi como las entendió el desarrollador del mismo.

A nivel personal, este trabajo y la elaboración de la memoria han servido para entender y capacitar al desarrollador a comprender la importancia de buenas metodologías de trabajo para realizar un proyecto de gran envergadura. Pese a no ser estrictamente necesario, la elaboración de esta aplicación es preferible siempre en un entorno profesional como el de TravelgateX. En una empresa especialista en la construcción de integraciones para múltiples clientes se cuenta ya con herramientas y personas capaces de realizar y ayudar a completar con éxito trabajos dedicados a grandes empresas y que implican la concurrencia de muchas transacciones.

Ha resultado satisfactorio el trabajo comunicativo y colaborativo entre el desarrollador de la aplicación y los interesados externos de ésta. Esto ha mejorado la capacidad e incrementado exponencialmente la experiencia del desarrollador para trabajar conjuntamente con expertos en distintas áreas.

En cuanto al producto desarrollado, cabe destacar la satisfacción del cliente durante el proyecto y la facilidad con la que se ha adaptado el producto final a las aplicaciones ya existentes en el entorno de TravelgateX. Un aspecto a mejorar para futuros proyectos ha sido la delicada situación de complejidad-tiempo-personas que se ha presentado durante el ciclo de vida del proyecto. Siendo un proyecto complejo técnicamente y el plazo de tiempo con el que se esperaba cumplir, ha faltado quizás la seguridad de tener más personas a cargo del departamento para cubrir en caso de necesidad la suplencia del desarrollador.

El producto desarrollado ha servido para mejorar y expandir el abanico de servicios de la agencia de viajes para quien se ha elaborado el mismo. Pese a no ser un producto base en la suite de productos y servicios de una agencia, es un buen añadido para la misma ya que ofrece una posibilidad que pocas agencias proveen y puede evitar, a la larga, la pérdida de beneficios o acumulación de cancelaciones costosas a compañías aéreas. Aun así, tras la elaboración de esta memoria, está todavía por ver la importancia de este producto y el beneficio real que implica una vez que éste se use al 100% de su capacidad. Es por ello que desde un principio lo que se ha buscado con la elaboración de esta aplicación es destacarse en el mercado como pionero de un servicio y no tanto el beneficio monetario que implique su uso en producción.

La mayor lección aprendida resultante del proyecto es la facilidad que implica la colaboración constante entre cliente y desarrollador. Las fases de análisis y diseño y la resultante

construcción de la aplicación fueron prácticamente pasos marcados y perfectamente señalados. Con esto se quiere destacar la poca desviación que se dio en el alcance y el tiempo del proyecto, siendo siempre estos dos parámetros muy inestables en trabajos tecnológicos como el descrito en este documento.

Se ha aprendido la importancia que tiene la realización de pruebas sobre todas las partes de la aplicación durante todas las fases del desarrollo. De esta forma, se ha conseguido que la salida a producción no haya sido tan preponderante y crítica, dada la enorme cantidad de transacciones que se ejecutan en el sistema de reservas del cliente.

Ha aumentado la confianza entre proveedor y cliente con la empresa TravelgateX puesto que el desarrollo se ha completado con éxito y sin incidencias, y en menos tiempo del esperado.

# CAPÍTULO



## ANEXO

A.1 (Tabla 1: Petición de transacción Disponibilidad: DisponibilidadRQ)

Elemento o Atributo (@)	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 <i>lowcost</i> en la búsqueda
@tipoViaje	1..1	Tipo de viaje de la búsqueda: <u>OW</u> , <u>RT</u> , <u>OJ</u> o <u>CT</u>

A.2 (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	Localización 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: <u>ADT</u> , <u>CHD</u> y <u>INF</u>
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: <u>ADT</u> , <u>CHD</u> y <u>INF</u>
@tieneCargosPorTarjeta	1..1	Indica si la tarifa implica el abono de cargos adicionales si se paga con tarjeta

A.3 (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 Disponibilidad
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: <u>ADT</u> , <u>CHD</u> y <u>INF</u>
configuracionesPasajeros	1..n	Cantidad de pasajeros desglosados por tipo de pasajero: <u>ADT</u> , <u>CHD</u> y <u>INF</u>

A.4 (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

A.5 (Tabla 5: Petición de transacción Reserva: ReservaRQ)

Elemento	Cardinalidad	Significado
pasajeros	1..1	Contiene una lista de pasajeros
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

A.6 (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: <u>ADT</u> , <u>CHD</u> y <u>INF</u>
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

A.7 (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
-------------	------	---

A.8 (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: <u>ADT</u> , <u>CHD</u> o <u>INF</u>
@estadoBillete	1..1	Estado del billete: emitido, cancelado...
pasajeros	1..n	Lista de pasajeros que se especificaron en la transacción de Reserva

A.9 (Tabla 9: Petición de transacción CancelaReserva: CancelaReservaRQ)

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

A.10 (Tabla 10: Respuesta de transacción CancelaReserva: CancelaReservaRS)

Elemento	Cardinalidad	Significado
desgloseImporte	1..1	Contiene los importes desglosados por tipo de pasajero: <u>ADT</u> , <u>CHD</u> y <u>INF</u>

A.11 (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

A.12 (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...

A.13 (Tabla 24: 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, RT, OI o CT)
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 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, K, KF, R, C, A)
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 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, RT, OI o CT)
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 <u>DMR</u>
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 único 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 <u>GDS</u> 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	1..1	string	Moneda de pago

/@moneda			
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: ( <u>ADT</u> , <u>CHD</u> , <u>INF</u> )
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 (ej.: 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

A.14 (Tabla 25: 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 y el precio que se obtenga en la RMR
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 <u>DMR</u>
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 la opción seleccionada en <u>DMR</u>
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 (ej.: 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 <u>EMR</u>
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, CHD)
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

A.15 (Tabla 26: 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 <u>EMR</u>
@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



## REFERENCIAS BIBLIOGRÁFICAS

- [1] Hosteltur, *Las OTA todavía le ganan a Google en las reservas de viajes, por un 13%*, Hosteltur, 2017  
[https://www.hosteltur.com/123908\\_ota-todavia-le-ganan-google-reservas-viajes-13.html](https://www.hosteltur.com/123908_ota-todavia-le-ganan-google-reservas-viajes-13.html)
- [2] W. Meng, C. Yu, *Advanced Metasearch Engine Technology*, Morgan & Claypool, 2010  
<https://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6812792&queryText=metasearch&newsearch=true>
- [3] Amadeus IT Group. (s.f.). En Wikipedia. Recuperado el 09 de Junio de 2018 de:  
[https://es.wikipedia.org/wiki/Amadeus\\_IT\\_Group](https://es.wikipedia.org/wiki/Amadeus_IT_Group)
- [4] Human Level, *Business to business (B2B): negocio entre empresas*, Human Level, 2017  
<https://www.humanlevel.com/diccionario-marketing-online/business-to-business-b2b>
- [5] ¿qué es un consolidador? (s.f.). En Dvuelos. Recuperado el 30 de Junio de 2018 de:  
<http://www.dvuelos.com/que-es-un-consolidador/>
- [6] Global Distribution System (GDS) (s.f.). En BusinessDictionary. Recuperado el 30 de Junio de 2018 de <http://www.businessdictionary.com/definition/Global-Distribution-System-GDS.html>
- [7] A. Bosu, J. C. Carver, J. Orbeck, C. Chockley, *Process Aspects and Social Dynamics of Contemporary Code Review: Insights from Open Source Development and Industrial Practice at Microsoft*, IEEE Transactions on Software Engineering, 2016  
<https://ieeexplore.ieee.org/document/7484733/>
- [8] What is Scrum? (s.f.). En Scrum.org. Recuperado el 30 de Junio de 2018 de:  
<https://www.scrum.org/resources/what-is-scrum>
- [9] What is Kanban? (s.f.). En Planview LeanKit. Recuperado el 30 de Junio de 2018 de:  
<https://leankit.com/learn/kanban/what-is-kanban/>

- [10] What is Web Service Endpoint? - Definition & Concept. (s.f.). En Study.com. Recuperado el 30 de Junio de 2018 de: <https://study.com/academy/lesson/what-is-web-service-endpoint-definition-concept.html>
- [11] Galileo GDS (s.f.). En Wikipedia. Recuperado el 09 de Junio de 2018 de: [https://en.wikipedia.org/wiki/Galileo\\_GDS](https://en.wikipedia.org/wiki/Galileo_GDS)
- [12] Sabre (s.f.). En Wikipedia. Recuperado el 09 de Junio de 2018 de: <https://es.wikipedia.org/wiki/Sabre>
- [13] SiteMinder, *Sistema de distribución global: cómo dar sentido a tu gestión de ingresos*, SiteMinder, 2017.  
<https://www.siteminder.com/es/r-es/marketing-es/sistema-de-distribucion-global-como-dar-sentido-a-tu-gestion-de-ingresos/>
- [14] M. Strauss, The Difference between CRS and GDS in travel industry, Travel Industry Blog, 2013.  
<https://www.travel-industry-blog.com/gds/the-difference-between-crs-and-gds-in-the-travel-industry/>
- [15] XML WSDL (s.f.). En w3schools. Recuperado el 30 de Junio de 2018 de: [https://www.w3schools.com/Xml/xml\\_wsdl.asp](https://www.w3schools.com/Xml/xml_wsdl.asp)
- [16] Introduction to XML (s.f.). En w3schools. Recuperado el 30 de Junio de 2018 de: [https://www.w3schools.com/xml/xml\\_whatis.asp](https://www.w3schools.com/xml/xml_whatis.asp)
- [17] Guía de Visual Basic (s.f.). En Docs Microsoft. Recuperado el 30 de Junio de 2018 de: <https://docs.microsoft.com/es-es/dotnet/visual-basic/>
- [18] What is .NET? (s.f.). En Microsoft. Recuperado el 30 de Junio de 2018 de: <https://www.microsoft.com/net/learn/what-is-dotnet>
- [19] Serialización (C# y Visual Basic) (s.f.). En msdn Microsoft. Recuperado el 30 de Junio de 2018 de: <https://msdn.microsoft.com/es-es/library/ms233843%28v=vs.120%29.aspx>
- [20] ¿Qué es un diagrama de Gantt y para qué sirve? (s.f.). En OBS Business School Universitat de Barcelona. Recuperado el 30 de Junio de 2018 de: <https://www.obs-edu.com/es/blog-project-management/diagramas-de-gantt/que-es-un-diagrama-de-gantt-y-para-que-sirve>
- [21] Slack. (2018). Slack (Versión 3.2.0) [Software]. Descargado de: <https://slack.com/intl/es-es/downloads/windows>
- [22] Atlassian. (2018). Jira (Versión 7.1) [Software]. Descargado de: <https://es.atlassian.com/software/jira>
- [23] Guía de C# (s.f.). En Docs Microsoft. Recuperado el 30 de Junio de 2018 de: <https://docs.microsoft.com/es-es/dotnet/csharp/>
- [24] ¿Qué es un archivo DLL? (s.f.). En Support.Microsoft. Recuperado el 19 de Junio de 2018 de: <https://support.microsoft.com/es-es/help/815065/what-is-a-dll>
- [25] Clase XmlElementAttribute (s.f.). En msdn Microsoft. Recuperado el 30 de Junio de 2018 de: [https://msdn.microsoft.com/es-es/library/system.xml.serialization.xmlelementattribute\(v=vs.110\).aspx](https://msdn.microsoft.com/es-es/library/system.xml.serialization.xmlelementattribute(v=vs.110).aspx)

- [26] Propiedad Binding.ElementName (s.f.). En msdn Microsoft. Recuperado el 30 de Junio de 2018 de: [https://msdn.microsoft.com/es-es/library/system.windows.data.binding.elementname\(v=vs.110\).aspx](https://msdn.microsoft.com/es-es/library/system.windows.data.binding.elementname(v=vs.110).aspx)
- [27] Propiedad XmlAttributes.XmlArray (s.f.). En msdn Microsoft. Recuperado el 30 de Junio de 2018 de: [https://msdn.microsoft.com/es-es/library/system.xml.serialization.xmlattributes.xmlarray\(v=vs.110\).aspx](https://msdn.microsoft.com/es-es/library/system.xml.serialization.xmlattributes.xmlarray(v=vs.110).aspx)
- [28] Propiedad XmlAttributes.XmlRoot (s.f.). En msdn Microsoft. Recuperado el 30 de Junio de 2018 de: [https://msdn.microsoft.com/es-es/library/system.xml.serialization.xmlattributes.xmlroot\(v=vs.110\).aspx](https://msdn.microsoft.com/es-es/library/system.xml.serialization.xmlattributes.xmlroot(v=vs.110).aspx)
- [29] Parse (s.f.). En TechTerms. Recuperado el 30 de Junio de 2018 de:  
<https://techterms.com/definition/parse>
- [30] Realizar UnitTest en C# con Moq (s.f.). En msdn Microsoft. Recuperado el 30 de Junio de 2018 de: <https://social.msdn.microsoft.com/Forums/es-ES/74383ba0-db9b-4176-8d57-945b0511b99a/realizar-unittest-en-c-con-moq?forum=vcsees>