在线分销产品
国内自动出票服务用户手册（非订单）

目录

1. 概述

国内自动出票服务(AirDemandTicket)为用户提供了集运价校验，运价存储，EI 存储，FP 存储，出票，票面显示等多功能为一体的自动出票服务。

对应主机指令 PAT:A、ETDZ 等指令。

2. 版本

主要内容	国内自动出票服务（非订单）
参考文档	
创建部门	

批准单		
版本号：		
批准人	批准人签字	批准日期

文档控制

版本	修改日期	修改内容描述	作者	批准人	批准日期
V1.0.0	2013.05.13	创建	姚俊安		
V1.0.1	2013.10.21	修改，添加回滚参数	姚俊安		
V1.0.2	2013.11.21	添加对大客户编码的说明	马萍		

3. 业务范围

国内运价计算、存储、出票。
支持无运价 PNR 运价计算存储出票操作和含有运价的 PNR 价格校验和自动出票操作。

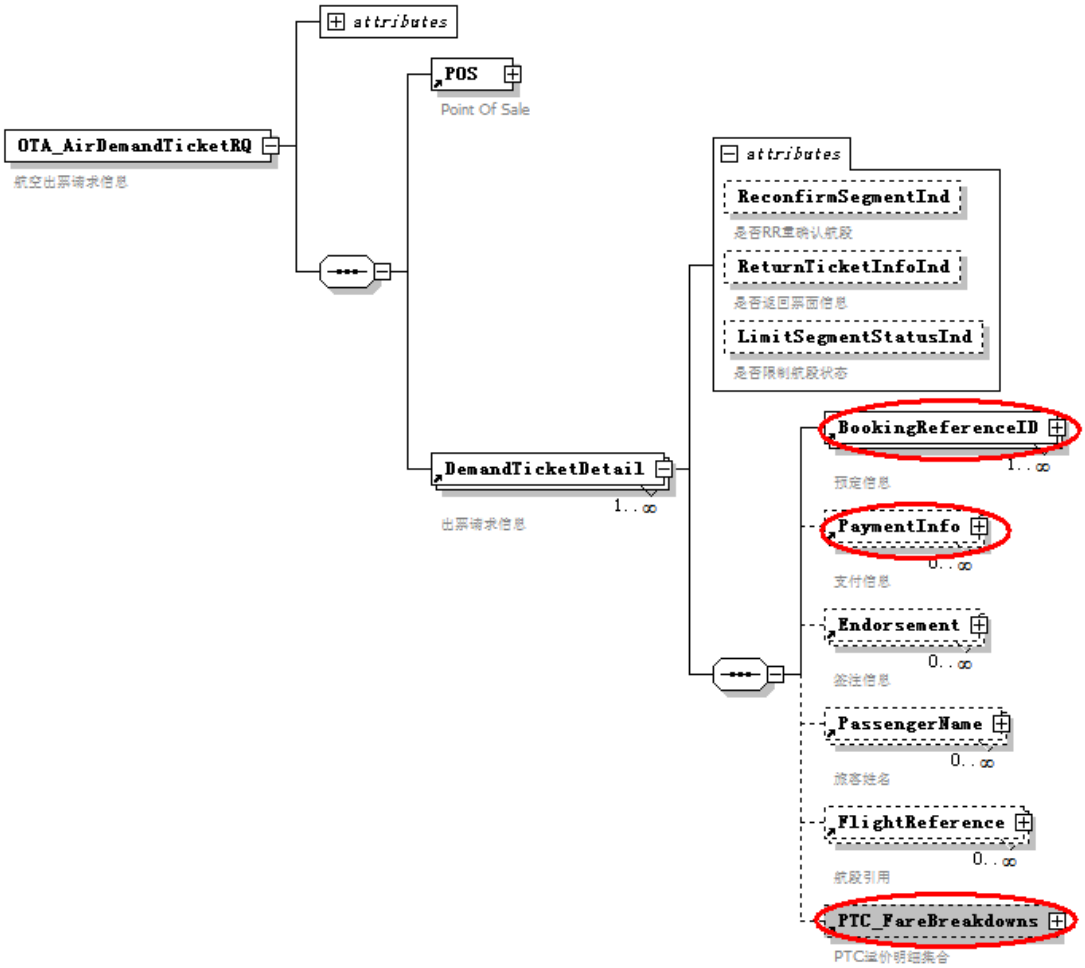
4. 功能范围

- 支持包含多种旅客类型（成人，包含伤残军人，因公带伤警察）或者儿童（不包含无人陪伴的儿童），暂不支持婴儿和其他类型旅客）的 PNR 出票。比如 PNR 中包含 2 名旅客，可以全部为成人，或者全部儿童，或者 1 名成人，1 名儿童；
- 支持 PNR 中的全部旅客或者部分旅客出票。比如 PNR 中有 3 个旅客，可以指定 3 人全部出票，或者第 1 名旅客出票，或者第 1，2 名旅客出票，或者第 2. 3 名旅客出票出票。
- 支持出票航段状态校验。可以限制只有 RR、HK、DK、TK、KL、KK 状态的航段才允许出票。
- 支持用户输入 EI 信息。如果 PNR 中没有 EI，则将用户输入的 EI 写入 PNR；如果 PNR 中有 EI，则忽略用户输入的 EI。
- 支持支付方式 FP 为信用卡类型 CC 输入。
- 支持大客户编码。

5. 接口说明

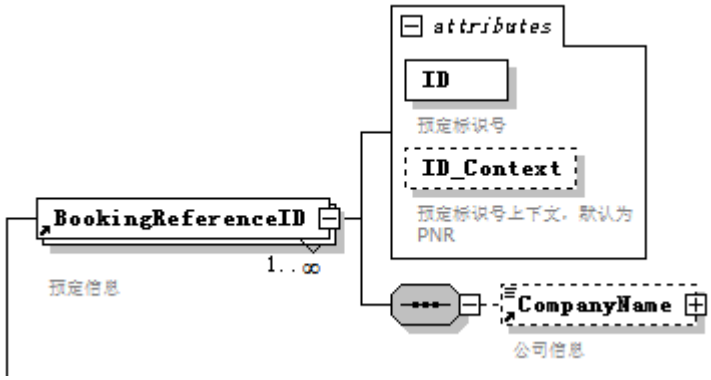
5. 1. 接口结构

5.1.1. 请求结构



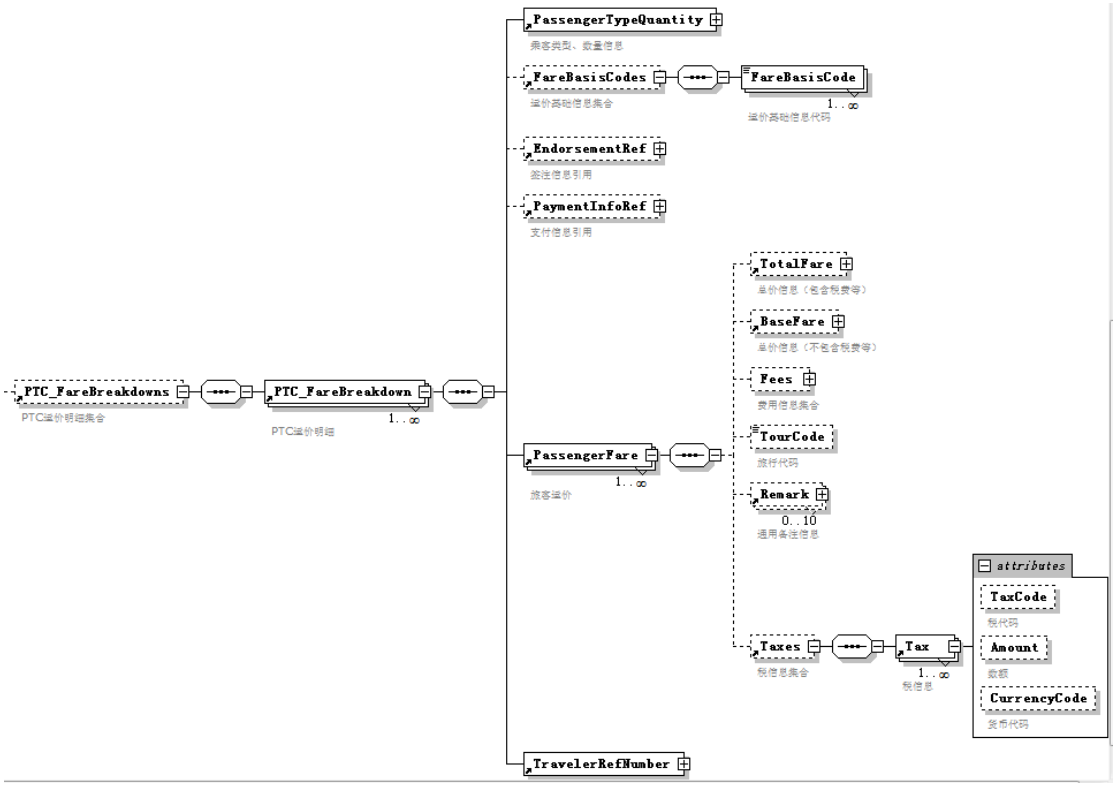
其中标识红色椭圆线的节点结构如下：

a. BookingReferenceID 节点：



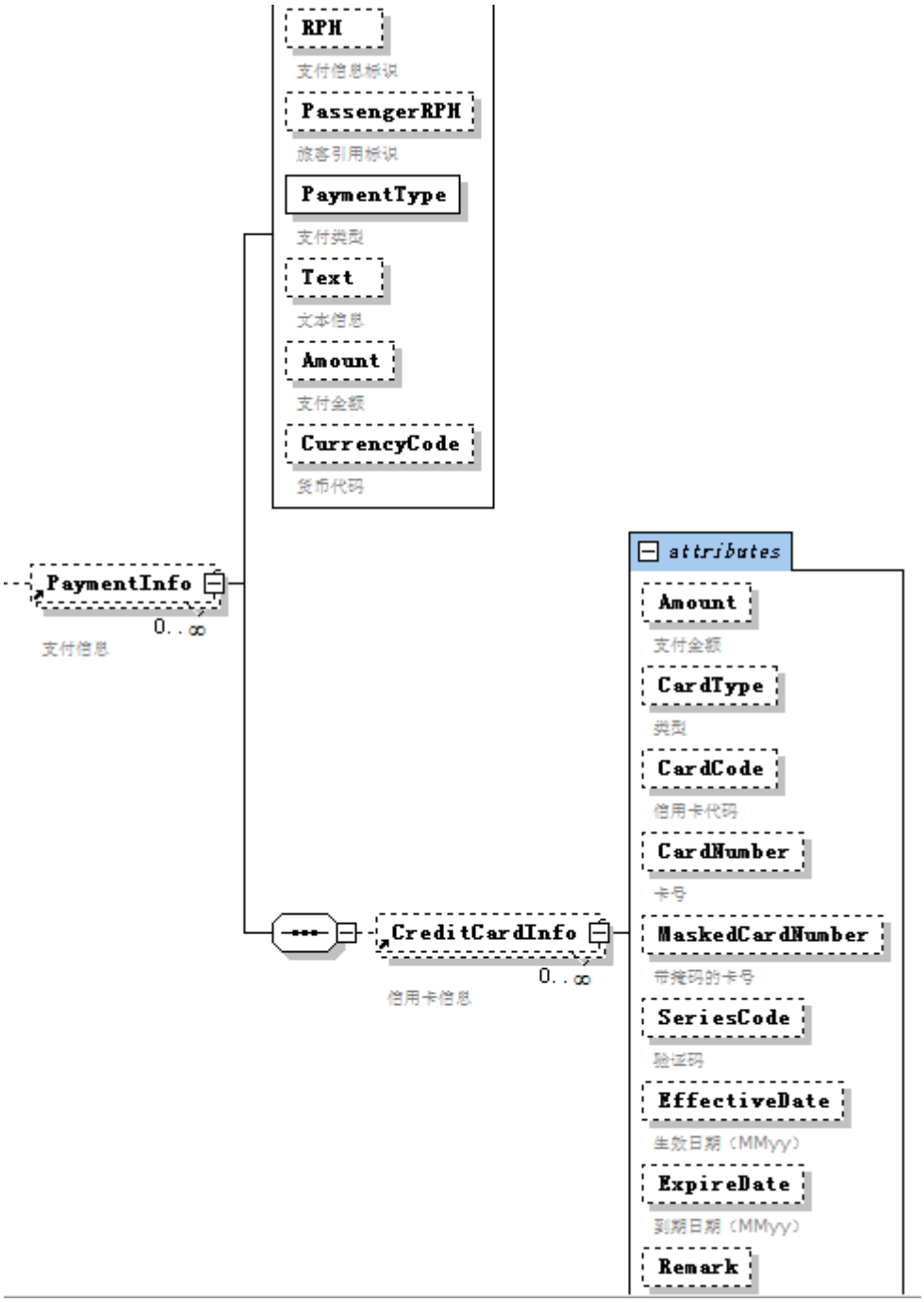
ID 存储 PNR 号，ID_Context 用来标识 ID 存储的是 PNR（PNR）。

b. PTC_FareBreakdowns 节点

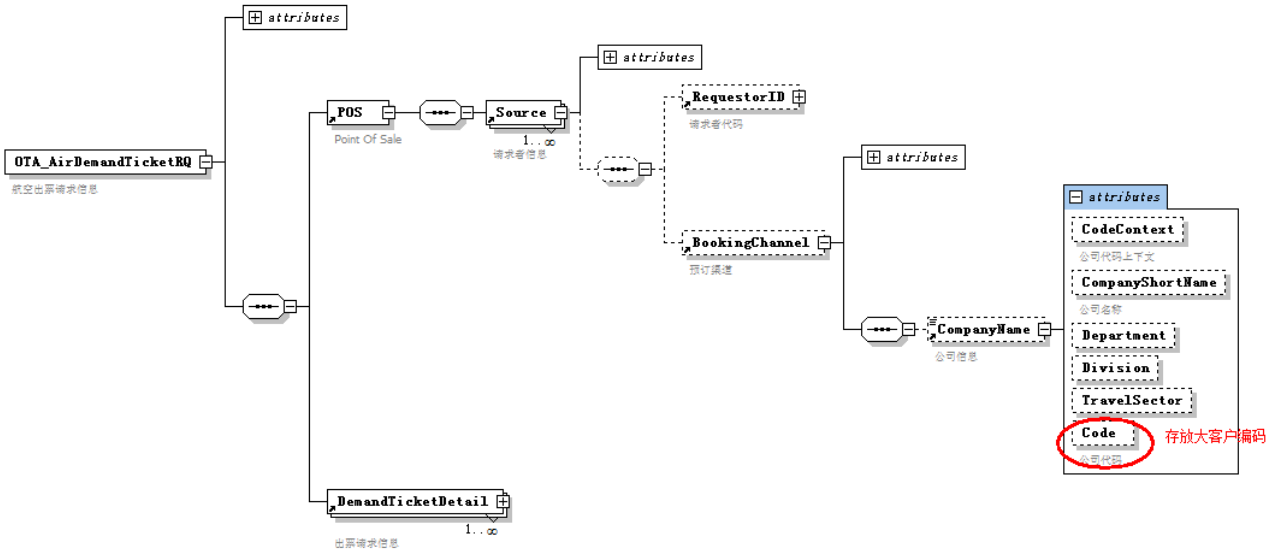


PTC_FareBreakdown 可有多个，表示多个可用的运价，需要标识运价适用的旅客类型，运价基础，票面价，税费信息。

c. PaymentInfo 节点：

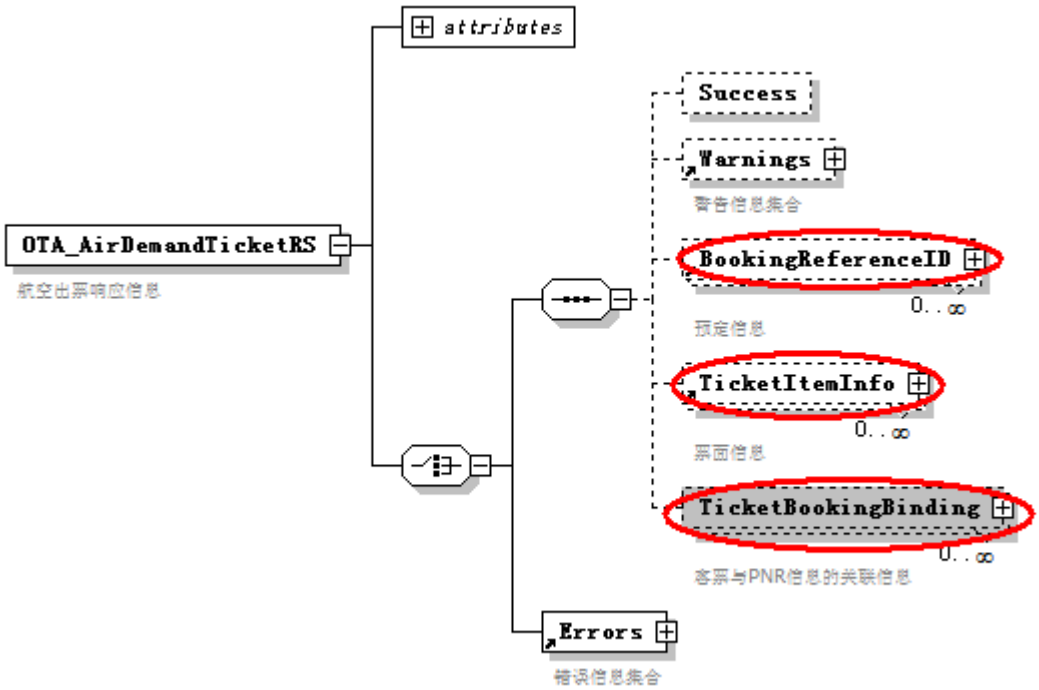


PaymentInfo 节点主要用于输入支付方式为信用卡的情况，若为现金，则可以不输，默认为是现金。PaymentInfo 只需要一个，为 PNR 的所有待出票旅客添加相同的支付信息。



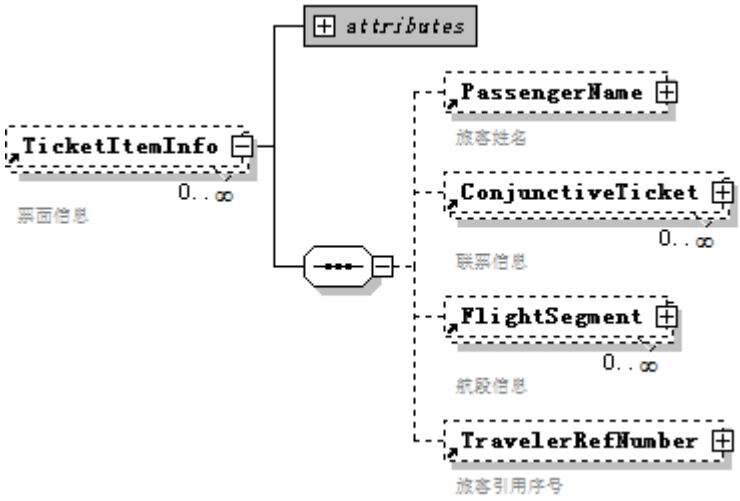
节点 Code，存储客户的大客户编码，用户大客户编码的运价计算、存储。

5.1.2. 响应结构



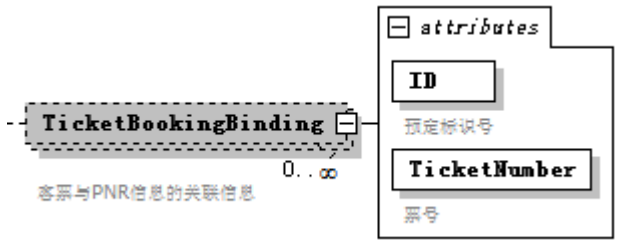
其中标识红色椭圆线的节点结构如下：

a. TicketItmeInfo 节点



TicketItmeInfo 该节点显示了票面上的信息，只有请求节点中 ReturnTicketInfoInd=true 时，票面信息才会在出票成功后显示。

b. TicketBookingBinding 节点



TicketBookingBinding 该节点标识了票号所属的 PNR 号。

5.2. 请求参数说明

元素名称	说明	Rep	类型	默认值	备注
OTA_AirDemandTicketRQ		1			
POS	零售商信息	1			出票 Office 号和打票机号
Source		1			
@PseudoCityCode	出票 Office 号		string		
@OtherID	打票机号	1	int		
BookingChannel	预订渠道	1			

CompanyName	公司信息	1			
@Code	公司代码	1	string		存放大客户编码
@ReturnTicketInfoInd	是否返回票面信息	1	boolean	false	如果为 true，则返回票面信息；false，则不返回
@LimitSegmentStatusInd	是否限制航段状态	1	boolean	false	如果为 true，则限制出票的航段状态；false，则不限制
@RollbackInd	出票失败，pnr 是否回滚到出票前的内容	1	boolean	false	如果为 true，则回滚；false，则不回滚
BookingReferenceID	PNR 信息	1..n			
@ID	PNR	1	string		
@ID_Context	ID 标识	1	enum	PNR	PNR:ID 是 PNR 号；
PassengerName	旅客信息	0..n			如果不填，则默认对该 pnr 的所有旅客进行出票。如果填写了，则对指定的旅客进行出票。
@PassengerRefNumber	旅客序号	0..1	int		旅客在 pnr 中的序号。如果不填写，则不验证输入的序号。
@PassengerTypeCode	旅客类型	0..1	enum	ADT	出票旅客类型，ADT-成人，CHD-儿童，INF-婴儿，JC-因类型。公带伤警察，GM-伤残军人.默认是成人。如果是非成人，请填写对应的旅客
Surname	旅客姓名	1	string		旅客姓名
PaymentInfo	支付信息	0..1			所有旅客输入相同的支付信息，因此实际调用的时候，

				这个节点只需要输入一个。 不填写默认为 CASH.
@PaymentType	支付类型	0..1	enum	
@CurrencyCode	货币代码	0..1	string	
@Amount	支付金额	0..1	decimal	
@CardType	类型	0..1	string	
@CardNumber	卡号	0..1	string	
@SeriesCode	验证码	0..1	string	
@ExpireDate	到期日期	0..1	string	格式：MMyy
Endorsement	签注信息	0..1		所有旅客输入相同的签注信息，因此实际调用的时候，这个节点只需要输入一个。根据出票的航段公司要求填写。当 pata 后，pnr 中没有 EI 项，则会写该 EI 到 pnr 中。
@Info	签注信息	1	string	
PTC_FareBreakdowns	运价信息	0..n		当 pnr 中有运价，则不用填写运价信息。
PTC_FareBreakdown				
FareBasisCodes	运价基础信息	0..1		
FareBasisCode	运价基础信息	0..1	string	
PassengerFare	旅客运价	1		
BaseFare	总价信息	1		不包含税费
@Amount	总价	1	float	不包含税费
Taxes	税费信息	0..1		
Tax	税费代码	1..n	string	税费代码

@TaxCode	税费代码	1	string	税费代码
@Amount	金额	1	float	金额
@CurrencyCode	货币代码	1	string	货币代码

5.3. 响应参数说明

元素名称	说明	Rep	类型	默认值	备注
TES_AirfareAutoTicketRS		1			
Errors	错误信息	0..1			
Error		0..1			
@Code	错误代码	1			
@Type	错误类型	1			
@ShortText	错误详细信息	1			
Warnings		0..1			
Warning		0..1			
@Type	警告类型	1			
@ShortText	警告详细信息	1			
@Code	警告代码	1			
Success	出票成功	0..1			自动出票是否成功，成功返回 OK
TicketItemInfo	票面信息	0..n			
@TicketNumber	票号	1	string		
@TotalAmount	票面总金额	1	decimal		
@CurrencyCode	货币代码	1	string		
@PaymentType	支付方式	1	PayType		

@Endorsement	签注信息	1	string	
PassengerName	旅客信息	1		
@PassengerTypeCode	旅客类型	1	PassengerType	
Surname	旅客姓名	1	string	
FlightSegment	航段信息			
@DepartureDateTime	出发日期和时间		string	
@FlightNumber	航班号		string	不含航空公司代码
@TicketStatus	客票状态		string	
@ResBookDesigCode	舱位		string	
DepartureAirport	出发机场信息			
@LocationCode	机场代码		string	
ArrivalAirport	到达机场信息			
@LocationCode	机场代码		string	
MarketingAirline	市场方航空公司信息			
@Code	航空公司两字码			

6. 错误说明

6.1. http 协议错误

由于接口为 HTTP 接口，HTTP 协议会返回 403 错误：

http 错误代码	说明
403	用户名、密码、IP 错误或无权限

6.2. 接口错误

<Errors>节点记录了错误信息。错误信息分为两类：业务错误、系统错误。详细见下表：

系统错误

错误代码	错误描述	说明
-303	Inner service error!	内部服务错误
-1	Biz Error	业务异常

业务错误

错误代码	错误描述	说明
-402	input error	输入错误
-10040	action code error	航段中的行动代码错误
-10000	response is empty	返回信息为空
-400	format of xml string is not valid	xml 内容格式错误
-1	Biz Error	业务异常
-10070	pnr has been cancelled already	pnr 已被取消
-10020	passenger not found	没有找到旅客信息
-10050	there is no fare information found	找不到运价信息
-51032	fail to deal with the endorsement information	处理签注信息出错
-51041	fail to deal with the payment information	处理支付信息出错
-51060	fail to demand the ticket	出票失败
-51061	fail to demand the ticket partly	部分出票失败
-10080	the ticket information is not found	找不到客票信息
-790	format of ticket number is not valid	票号格式错误
-51080	the ticket is not eTicket type	非电子客票
-303	inner service error	内部服务异常
-51001	there is no carrier with the	未找到婴儿的携带者

	infant	
-51020	already exsit fare information	运价信息已存在
-51040	alreay exsit payment information	已经存在支付信息
-51033	alreay exsit endorsement information	已经存在签注信息
-51070	ticket has been demanded	已出票
-304	sending commands too fast, try again later	访问速度过快,请稍后再试
-9999	unknown error	未知错误
-51050	no available ticket number	票号不可用
-51031	endorsement information is not found	未找到签注信息
-50011	status of printer is not normal	打票机状态错误
-51100	please issue the ticket manually	需要手工出票
-51010	document information is not found	缺少证件信息
-50001	need airline's authorization	需要航空公司授权
-51090	demand ticket timeout	出票超时
-50038	ROLL_BACK__ERROR	回滚异常！
-50039	ROLL_BACK__FAILD	回滚失败！

7. 参考样例

7. 1. 多人单航段 PNR，只对部分人出票

输入样例(OTA_AirDemandTicketRQ.xml)：

在请求样例中，列出了待出票 PNR 号，待出票的旅客信息，需要写入 PNR 的 EI 信息，FP 信息，以及运价信息，以及三个标志位。

EI 信息只有在 PNR 中没有相关 EI 的时候才会写入，如果 PNR 中存在 EI，则忽略该输入参数。

FP 信息只有在 PNR 中没有相关 FP 的时候才会写入，如果 PNR 中存在 FP，则忽略该输入参数。

输入运价信息，且 PNR 中有运价，则校验输入的运价信息与 PNR 中的是否一致，一致：出票；不一致：抛出不一致的异常

输入运价信息，且 PNR 中没有运价，则校验输入的运价信息与 PATA 的结果是否一致，一致：将 PATA 的运价写入 PNR；不一致：抛出不一致的异常

没有输入运价信息，且 PNR 中有运价，直接出票

没有输入运价信息，且 PNR 中没有运价，抛出不存在运价信息的异常

三个标志位：ReconfirmSegmentInd 是否做 RR，默认为 false：不做 RR

ReturnTicketInfoInd 是否显示票面 ，默认为 false：不显示票面信息

LimitSegmentStatusInd 是否限制出票航段状态为 HK,KK,DK,TK,KL,RR，默认为 false：不限制出票的航段状态

```
<OTA_AirDemandTicketRQ>
    <POS>
        <!-- PseudoCityCode = Office 号 -->
        <!-- OtherID = 打票机号 -->
        <Source PseudoCityCode='BJS187' OtherID='60'></Source>
    </POS>
    <!-- 多个 pnr 则多个 DemandTicketDetail 节点 -->
    <!-- ReconfirmSegmentInd = 是否做 RR -->
    <!-- ReturnTicketInfoInd = 是否显示票面 -->
    <!-- LimitSegmentStatusInd = 是否限制出票航段状态为 HK,KK,DK,TK,KL,RR -->
<DemandTicketDetail ReconfirmSegmentInd='true'
        ReturnTicketInfoInd='true' LimitSegmentStatusInd='true'>
        <!--ID = PNR 号 -->
        <BookingReferenceID ID='JSSYBQ' ID_Context='PNR'>
        </BookingReferenceID>
        <!-- EI 备注项 -->
        <Endorsement Info='不得退改签'></Endorsement>
        <!-- FP 支付信息 支付信息不输，则默认是现金支付 CASH-->
        <!-- PaymentType = 支付方式，CREDIT_CARD 为信用卡，CASH 为现金，默认现金方式-->
        <!-- CurrencyCode = 货币代码-->
        <PaymentInfo PaymentType="CREDIT_CARD" CurrencyCode="CNY" >
        <!-- CreditCardInfo = 信用卡信息
```

```
        Amount = 支付金额，CardCode = 信用卡类型，CardNumber = 信用卡号  SeriesCode = 验证码 ExpireDate = 信用卡有效期 -->
        <CreditCardInfo Amount="1520.00" CardCode="TP" CardNumber="123456789012345" SeriesCode="103698" ExpireDate="3015">
        </CreditCardInfo>
        </PaymentInfo>
        <!-- 旅客信息，若多个旅客则多个 PassengerName 节点  -->
        <PassengerName PassengerRefNumber="1"
            PassengerTypeCode="ADT">
            <Surname>WU/XIAO</Surname>
        </PassengerName>
        <!-- 旅客信息，若多个旅客则多个 PassengerName 节点 -->
        <!-- PassengerRefNumber = 旅客在 PNR 中的序号  PassengerTypeCode = 旅客类型 Surname= 旅客姓名-->
        <PassengerName PassengerRefNumber="2"
            PassengerTypeCode="ADT">
            <Surname>ZHENG/XIAO</Surname>
        </PassengerName>
        <PTC_FareBreakdowns>
        <!-- PTC_FareBreakdown = 运价信息 多个运价，则多个 PTC_FareBreakdown 节点  -->
            <PTC_FareBreakdown>
                <!-- PassengerTypeQuantity Code='旅客类型'  -->
                <PassengerTypeQuantity Code='ADT' />
                <!-- 运价基础 -->
                <FareBasisCodes>
                    <FareBasisCode>Y+Y</FareBasisCode>
                </FareBasisCodes>
                <PassengerFare>
                    <!-- 总价=税费+票面价 -->
                    <TotalFare Amount='1520.00' CurrencyCode='CNY' />
                    <!-- 票面价 -->
```

```
                        <BaseFare Amount='1500.00' CurrencyCode='CNY' />
                        <Taxes>
                            <!-- <Tax TaxCode='税费类别' Amount='金额' CurrencyCode='币种' /> -->
                            <Tax TaxCode='CN' Amount='0.00' CurrencyCode='CNY' />
                            <Tax TaxCode='YQ' Amount='20.00' CurrencyCode='CNY' />
                        </Taxes>
                    </PassengerFare>
                </PTC_FareBreakdown>
            </PTC_FareBreakdowns>
        </DemandTicketDetail>
</OTA_AirDemandTicketRQ>
```

对应输出(OTA_AirDemandTicketRS.xml)：

```
 <OTA_AirDemandTicketRS>
    <Success>success</Success>
    <BookingReferenceID ID="JSSYBQ" ID_Context="PNR"/>
    <TicketItemInfo TicketNumber="781-3250419585" TotalAmount="1520.0" CurrencyCode="CNY" PaymentType="CC">
        <PassengerName PassengerTypeCode="ADT">
            <Surname>ZHOU/XIAO</Surname>
        </PassengerName>
        <FlightSegment DepartureDateTime="2013-08-14T07:30:00" TicketStatus="OPEN FOR USE">
            <OperatingAirline Code="" FlightNumber="155"/>
            <DepartureAirport LocationCode="SHA"/>
            <ArrivalAirport LocationCode="PVG"/>
            <MarketingAirline Code="CA"/>
        </FlightSegment>
    </TicketItemInfo>
    <TicketItemInfo TicketNumber="781-3250419586" TotalAmount="1520.0" CurrencyCode="CNY" PaymentType="CC">
```

```
            <PassengerName PassengerTypeCode="ADT">
                <Surname>WU/XIAO</Surname>
            </PassengerName>
            <FlightSegment DepartureDateTime="2013-08-14T07:30:00" TicketStatus="OPEN FOR USE">
                <OperatingAirline Code="" FlightNumber="155"/>
                <DepartureAirport LocationCode="SHA"/>
                <ArrivalAirport LocationCode="PVG"/>
                <MarketingAirline Code="CA"/>
            </FlightSegment>
 </TicketItemInfo>
  <TicketBookingBinding ID="JSSYBQ" TicketNumber="781-3250419585"/>
  <TicketBookingBinding ID="JSSYBQ" TicketNumber="781-3250419586"/>
</OTA_AirDemandTicketRS>
```

7.2. 出票设置回滚标识

RQ

```
<OTA_AirDemandTicketRQ>
    <POS>
        <!-- PseudoCityCode = Office 号 -->
        <!-- OtherID = 打票机号 -->
        <Source PseudoCityCode="BJS187" OtherID="62"></Source>
    </POS>
    <!-- 多个 pnr 则多个 DemandTicketDetail 节点 -->
    <!-- ReconfirmSegmentInd = 是否做 RR -->
    <!-- ReturnTicketInfoInd = 是否显示票面 -->
    <!-- LimitSegmentStatusInd = 是否限制出票航段状态为 HK,KK,DK,TK,KL,RR -->
    <DemandTicketDetail ReconfirmSegmentInd="true"
        ReturnTicketInfoInd="true" LimitSegmentStatusInd="true" RollbackInd="true">
```

```
<!--ID = PNR 号 -->
<!-- ID_Context = 订单号或者 PNR 的标识 -->

<BookingReferenceID ID="JY30E0" ID_Context="PNR" />
<!-- EI 备注项 -->
<Endorsement Info="不得退改签"></Endorsement>
<!-- FP 支付信息 支付信息不输，则默认是现金支付 CASH -->
<!-- PaymentType = 支付方式，CREDIT_CARD 为信用卡，CASH 为现金，默认现金方式 -->
<!-- CurrencyCode = 货币代码 -->
<PaymentInfo PaymentType="CREDIT_CARD" CurrencyCode="CNY">
    <!-- CreditCardInfo = 信用卡信息 Amount = 支付金额，CardCode = 信用卡类型，CardNumber
        = 信用卡号 SeriesCode = 验证码 ExpireDate = 信用卡有效期 -->
    <CreditCardInfo Amount="1680.00" CardCode="TP"
        CardNumber="123456789012345" SeriesCode="103698" ExpireDate="3015">
    </CreditCardInfo>
</PaymentInfo>
<!-- 旅客信息，若多个旅客则多个 PassengerName 节点 -->
<!-- PassengerRefNumber = 旅客在 PNR 中的序号 PassengerTypeCode = 旅客类型 Surname=
    旅客姓名 -->
<PTC_FareBreakdowns>
    <!-- PTC_FareBreakdown = 运价信息 多个运价，则多个 PTC_FareBreakdown 节点 -->
    <PTC_FareBreakdown>
        <!-- PassengerTypeQuantity Code="旅客类型" -->
        <PassengerTypeQuantity Code="ADT" />
        <!-- 运价基础 -->
        <FareBasisCodes>
            <FareBasisCode>RT/Y+RT/Y</FareBasisCode>
        </FareBasisCodes>
        <PassengerFare>
            <!-- 总价=税费+票面价 -->
```

```
                        <TotalFare Amount="2180.00" CurrencyCode="CNY" />
                        <!-- 票面价 -->
                        <BaseFare Amount="1800.00" CurrencyCode="CNY" />
                        <Taxes>
                            <!-- <Tax TaxCode="税费类别" Amount="金额" CurrencyCode="币种" /> -->
                            <Tax TaxCode="CN" Amount="100.00" CurrencyCode="CNY" />
                            <Tax TaxCode="YQ" Amount="280.00" CurrencyCode="CNY" />
                        </Taxes>
                    </PassengerFare>
                </PTC_FareBreakdown>
            </PTC_FareBreakdowns>
        </DemandTicketDetail>
</OTA_AirDemandTicketRQ>
```

RS

```
<OTA_AirDemandTicketRS>
        <Success>success</Success>
        <BookingReferenceID ID="131021113203056853x"
            ID_Context="ORDER" />
        <BookingReferenceID ID="JY30E0" ID_Context="PNR" />
        <TicketItemInfo TicketNumber="781-1774660992"
            TotalAmount="2180.0" CurrencyCode="CNY" PaymentType="CREDIT_CARD"
            Endorsement="不得退改签">
            <PassengerName PassengerTypeCode="ADT">
                <Surname>ZHOU/XIAO</Surname>
            </PassengerName>
            <FlightSegment DepartureDateTime="2013-11-20T07:00:00"
                TicketStatus="OPEN FOR USE">
                <OperatingAirline Code="" FlightNumber="5138" />
```

```
                <DepartureAirport LocationCode="PEK" />
                <ArrivalAirport LocationCode="SHA" />
                <MarketingAirline Code="MU" />
            </FlightSegment>
            <FlightSegment DepartureDateTime="2013-11-21T08:30:00"
                TicketStatus="OPEN FOR USE">
                <OperatingAirline Code="" FlightNumber="5145" />
                <DepartureAirport LocationCode="SHA" />
                <ArrivalAirport LocationCode="PEK" />
                <MarketingAirline Code="MU" />
            </FlightSegment>
        </TicketItemInfo>
        <TicketBookingBinding ID="JY30E0" TicketNumber="781-1774660992" />
</OTA_AirDemandTicketRS>
```

8. 接口调用

国内自动出票服务提供基于 XML 文档规范的标准 HTTP 协议接口，通过在标准 HTTP 通道上传输 XML 格式的请求和数据来完成服务调用。服务的输入输出定义在标准的 XSD 文档中，用户根据自有系统的实现环境结合 XSD 文档进行数据到对象间的转换。在线分销提供服务接口 URL,用户使用 HTTP 协议按照服务定义正确构造请求输入，正确解析数据输出实现服务的调用。

8.1. 接口申请

请与您所在城市的中国航信分支机构联系，资质审查通过后签署试用协议，即可获得试用接口。

8.2. 接口认证

授权用户调用试用接口时需要使用已分配的账号和密码、并使用申请时的 IP。如果已变化 IP 请联系业务人员申请变更 IP。

8.3. 接口地址

接口描述位置：http://espeed.travelsky.com/develop/xml/AirDemandTicket?xsd

接口试用地址：http://espeed.travelsky.com/develop/xml/AirDemandTicket

8.4. JAVA 调用示例

Java 调用 http 接口示例如下：

其中该调用示例程序只是为了展示如何接入使用

```
import java.io.BufferedReader;

import java.io.ByteArrayOutputStream;

import java.io.InputStream;

import java.io.InputStreamReader;

import java.util.zip.GZIPInputStream;

import java.util.zip.GZIPOutputStream;


import org.apache.commons.httpclient.Credentials;

import org.apache.commons.httpclient.DefaultHttpMethodRetryHandler;

import org.apache.commons.httpclient.HttpClient;

import org.apache.commons.httpclient.HttpState;

import org.apache.commons.httpclient.HttpStatus;

import org.apache.commons.httpclient.UsernamePasswordCredentials;

import org.apache.commons.httpclient.auth.AuthScope;

import org.apache.commons.httpclient.methods.ByteArrayRequestEntity;

import org.apache.commons.httpclient.methods.PostMethod;

import org.apache.commons.httpclient.methods.RequestEntity;
```

```
import org.apache.commons.httpclient.params.HttpMethodParams;


/**
 * http 客户端调用示例(数据压缩版)
 */
public class HttpClientDemo {


    public static void main(String[] args) {


        //实际使用时，请设置正确的变量值
        //用户名
        String username = "用户名";
        //密码
        String pwd = "密码";
        //服务地址
        String serviceUrl = "服务地址";
        //请求 XML
        String reqStr = "请求 xml";


        // 构造 HttpClient 的实例
        HttpClient httpClient = new HttpClient();
        //调用验证信息
        HttpState state = new HttpState();
        Credentials credentials = new UsernamePasswordCredentials(username, pwd);
        state.setCredentials(AuthScope.ANY, credentials);
        httpClient.setState(state);


        // 创建 POST 方法的实例
        PostMethod postMethod = new PostMethod(serviceUrl);
        // 使用系统提供的默认的恢复策略
```

```
postMethod.getParams().setParameter(HttpMethodParams.RETRY_HANDLER,
        new DefaultHttpMethodRetryHandler());
try {
    // 请求参数的数据压缩
    ByteArrayOutputStream out = new ByteArrayOutputStream();
    GZIPOutputStream gzip = new GZIPOutputStream(out);
    if (reqStr != null && !"".equals(reqStr)) {
        gzip.write(reqStr.getBytes());
    }
    gzip.close();
    RequestEntity requestEntity = new ByteArrayRequestEntity(out
            .toByteArray());
    postMethod.setRequestEntity(requestEntity);
    postMethod.addRequestHeader("Content-Type",
            "text/html;charset=UTF-8");
    postMethod.addRequestHeader("accept-encoding", "gzip");
    postMethod.addRequestHeader("content-encoding", "gzip");

    // 执行 getMethod
    int statusCode = httpClient.executeMethod(postMethod);
    if (statusCode != HttpStatus.SC_OK) {
        throw new Exception("Invoke Get Method Failed, HttpStatus = "
                + statusCode);
    }

    // 返回结果的数据解压
    InputStream is = postMethod.getResponseBodyAsStream();
    BufferedReader br = new BufferedReader(new InputStreamReader(
            new GZIPInputStream(is)));
```

```
                StringBuffer respStr = new StringBuffer();
                String line = null;
                while ((line = br.readLine()) != null) {
                    respStr.append(line);
                }

                // 打印结果
                System.out.println(respStr.toString());

                out.flush();
                out.close();
                is.close();
            } catch (Exception e) {
                // 发生致命的异常，可能是协议不对或者返回的内容有问题
                e.printStackTrace();
            } finally {
                // 释放连接
                postMethod.releaseConnection();
            }

        }
}
```

8.5. NET 调用示例

.Net 调用 http 接口示例如下：

其中该调用示例程序只是为了展示如何接入使用

```
(1)建立一个 cs 文件,与服务器通信的类：
// 定义 xml 与服务器的通信
class OTA_XMLHTTP
```

```
{
        string user;        //用户名
        string pwd;     //密码
         string url;   //服务地址
        public OTA_XMLHTTP(string _user,string _pwd,string _url)
        {
            user = _user;
            pwd = _pwd;
            url = _url;
        }
        // 获取服务器返回的字符串
        public string GetResponse(string requestXml)
        {
            string xmlString = "";     //要返回的 xml 字符串
            bool a= requestXml.Contains("<?xml version="1.0" encoding="UTF-8"?>");
            requestXml = requestXml.Replace("<?xml version="1.0" encoding="UTF-8"?>\n", "");
            try
            {
                //制定服务器地址
                HttpWebRequest request = (HttpWebRequest)HttpWebRequest.Create(url);
                request.Method = WebRequestMethods.Http.Post; //设定 http 的传递方式
                request.ContentType = "application/x-www-form-urlencoded";
               //设定 http 的 header
               if (user != null && pwd != null)
               {
                    string user_pwd = user + ":" + pwd;
                    byte[] authBytes = Encoding.UTF8.GetBytes(user_pwd.ToCharArray());
                    request.Headers.Add("Authorization", "Basic " + Convert.ToBase64String(authBytes));
               request.Headers.Add("Content-Encoding", "gzip");
               request.Headers.Add("Accept-Encoding", "gzip");
```

```
                }
                //将数据写入流中
                string para = "request";
                ASCIIEncoding encoding = new ASCIIEncoding();
                byte[] data = encoding.GetBytes(para + " = " + requestXml);
                request.ContentLength = data.Length;
                Stream stream = request.GetRequestStream();
                stream.Write(data, 0, data.Length);
                stream.Close();
                //建立获取 http 返回的 response
                HttpWebResponse response = (HttpWebResponse)request.GetResponse();
                //读取 http 返回的字符串
            Stream responseStream = response.GetResponseStream();
            if (response.ContentEncoding.ToLower().Contains("gzip"))
                    responseStream = new GZipStream(responseStream, CompressionMode.Decompress);
            else if (response.ContentEncoding.ToLower().Contains("deflate"))
                    responseStream = new DeflateStream(responseStream, CompressionMode.Decompress);
            StreamReader streamReader = new StreamReader(responseStream, Encoding.UTF8);
        xmlString = streamReader.ReadToEnd();
                streamReader.Close();
            }
            catch (WebException e)
            {
                xmlString = e.Message;
            }
        return xmlString;
    }
}
```

（2）调用上面的文件

调用方法：

```
string user="用户名";
string pwd="密码";
string url="服务地址";
OTA_XMLHTTP xmlHttp = new OTA_XMLHTTP(user, pwd, url);
PonseXmlBox.Text = xmlHttp.GetResponse(requestXml);
```